Gaze Speedup: Eye Gaze Assisted Gesture Typing in Virtual Reality

Maozheng Zhao
mazhao@cs.stonybrook.edu
Department of Computer Science,
Stony Brook University
Stony Brook, New York, USA

Alec Pierce
alecmp@meta.com
Reality Labs Research, Meta Inc
Redmond, Washington, USA

Ran Tan
rantan@meta.com
Reality Labs Research, Meta Inc
Redmond, Washington, USA

Ting Zhang
tingzhang@meta.com
Reality Labs Research, Meta Inc
Redmond, Washington, USA

Tianyi Wang
tianyiwang@meta.com
Reality Labs Research, Meta Inc
Redmond, Washington, USA

Tanya R. Jonker
tanya.jonker@meta.com
Reality Labs Research, Meta Inc
Redmond, Washington, USA

Hrvoje Benko
benko@meta.com
Reality Labs Research, Meta Inc
Redmond, Washington, USA

Aakar Gupta
aakar.hci@gmail.com
Reality Labs Research, Meta Inc
Redmond, Washington, USA

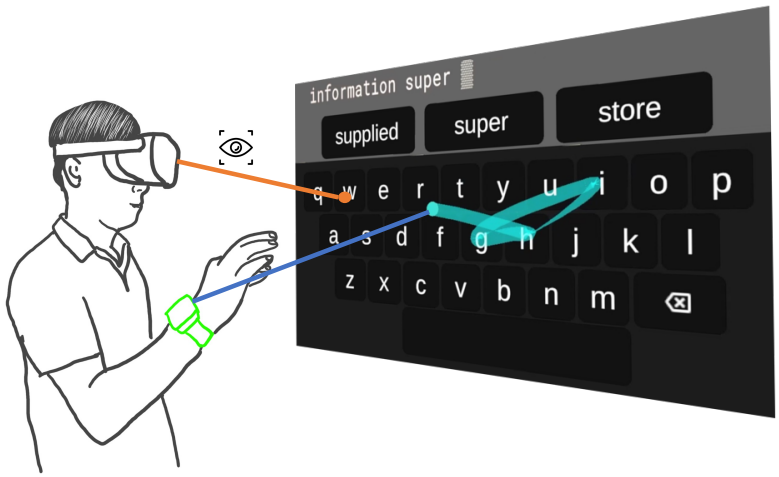


Figure 1: A user is doing gaze-assisted gesture typing using a wrist wearable in virtual reality.

ABSTRACT

Mid-air text input in augmented or virtual reality (AR/VR) is an open problem. One proposed solution is gesture typing where the user performs a gesture trace over the keyboard. However, this requires the user to move their hands precisely and continuously, potentially causing arm fatigue. With eye tracking available on AR/VR devices, multiple works have proposed gaze-driven gesture typing techniques. However, such techniques require the explicit use of gaze which are prone to Midas touch problems, conflicting with other gaze activities in the same moment. In this work, the user is not made aware that their gaze is being used to improve the interaction, making the use of gaze completely implicit. We observed that a user's implicit gaze fixation location during gesture typing is usually the gesture cursor's target location if the gesture cursor is moving toward it. Based on this observation, we propose the Speedup method in which we speed up the gesture cursor toward the user's gaze fixation location, the speedup rate depends on how well the gesture cursor's moving direction aligns with the gaze fixation. To reduce the overshooting near the target in the Speedup method, we further proposed the Gaussian Speedup method in which the speedup rate is dynamically reduced with a Gaussian function when the gesture cursor gets nearer to the gaze fixation.

IUI '23, March 27–31, 2023, Sydney, NSW, Australia

ACM ISBN 979-8-4007-0106-1/23/03.
https://doi.org/10.1145/3581641.3584072

Using a wrist IMU as input, a 12-person study demonstrated that the Speedup method and Gaussian Speedup method reduced users' hand movement by 30% and 22% respectively without any loss of typing speed or accuracy.

CCS CONCEPTS

• **Human-centered computing → Virtual reality**; **Gestural input**.

KEYWORDS

gesture input, implicit eye gaze, virtual reality

ACM Reference Format:
Maozheng Zhao, Alec Pierce, Ran Tan, Ting Zhang, Tianyi Wang, Tanya R. Jonker, Hrvoje Benko, and Aakar Gupta. 2023. Gaze Speedup: Eye Gaze Assisted Gesture Typing in Virtual Reality. In *28th International Conference on Intelligent User Interfaces (IUI '23), March 27–31, 2023, Sydney, NSW, Australia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3581641.3584072

1 INTRODUCTION

Mid-air text input in augmented or virtual reality (AR/VR) is an open problem. The ideal method needs to support good speed and accuracy, have low physical demand, particularly low arm fatigue, be familiar to the user to reduce initial friction of use, and should use a sensing method that does not encumber the user's hands. One potential solution is gesture typing [20, 21] where the user performs a gesture trace over the Qwerty keyboard using hand motion. Gesture typing on smartphones is both familiar and reasonably fast. While it may still be familiar and fast in mid-air, it would require long, precise hand movements, leading to arm fatigue over time. This would be especially true for longer words. With eye tracking available on AR/VR devices, prior works have proposed gaze-driven gesture typing techniques [24, 35]. However, such explicit control using eye gaze has the Midas touch problem [16] and it prevents eyes from observing the environment naturally and thus can easily cause frustration and mental fatigue.

Gaze, however, can offer useful implicit information even if not used for explicit control. We observed that a user's implicit gaze fixation location during gesture typing is usually the gesture cursor's target location if the gesture cursor is moving toward it. Based on this observation, we proposed to use the *alignment* between the wrist cursor's moving direction and the direction toward the gaze fixation as the indication of how likely the gaze fixation location is the target. We proposed the Speedup method which speeds up the wrist cursor toward the gaze fixation location based on the extent of the *alignment*. To enhance the control precision for the Speedup method, we proposed the Gaussian Speedup method which gradually reduces the speedup rate with a Gaussian function when the wrist cursor moves closer to the gaze fixation. In our methods, the user is not made aware that their gaze is being used to improve the interaction, making the use of gaze completely implicit. From the user's perspective, the gesture typing trace is controlled by their hand motion.

We use a wrist-worn inertial motion unit (IMU) as the input device. Smart wrist devices are fairly commonplace, do not encumber users' hands as controllers do, and are suitable for all-day-wearable AR use. Unlike vision-based hand tracking, wrist IMU-based tracking also does not require cameras or high computation power making them ideal for hand motion tracking for lightweight AR glasses. Figure 1 shows a user is doing gaze-assisted gesture typing using a wrist wearable in virtual reality.

We conducted a study with 12 participants to compare our two implicit gaze assistance methods, Speedup and Gaussian Speedup, with the Wrist-Only baseline. Our results demonstrate that the Speedup method significantly reduced participant hand movement by 30% and the Gaussian Speedup method significantly reduced it by 22% without loss of input speed or accuracy.

Our contributions in this paper are:

- We proposed a reliable way of utilizing the implicit eye gaze during gesture typing in VR. That is using the *alignment* between the wrist cursor's moving direction and the direction toward the gaze fixation as the likelihood of the gaze fixation location being the target.
- Two novel implicit gaze-assistance methods for mid-air gesture typing, the Speedup method and the Gaussian Speedup method.
- Validation of the two methods through a user study that shows a significant reduction in the users' required hand movements, without loss in speed and accuracy.

2 RELATED WORK

We divide the related work into typing in AR/VR, single-handed typing using wrist, typing using gaze, and multimodal gaze methods.

2.1 Typing in AR/VR

Typing in AR/VR can be classified into typing with encumbered hands or typing with a free hand. *Encumbered* encompasses techniques where the user interacts with externally grounded devices, such as a physical keyboard [19], as well as techniques where the user's hands are significantly encumbered by controllers or other devices. Current commercial VR devices use controllers for text entry where a ray cast from the controllers is used to select keys on a *qwerty* keyboard [14, 45]. Vulture [29] tracks a single hand using fiducial markers and performs mid-air gesture typing on a distant, large-screen keyboard. Prior work has investigated dedicated handheld devices for VR typing such as Twiddler [2], 9-key keypads [8], smartphones [18], and bimanual touchpads with hover detection [41]. Speicher et al. [43] evaluated multiple techniques based on current commercial controllers including raycasted pointing, direct tapping, controller as gamepad, and found raycasted pointing as the fastest with 15.4 words-per-minute (WPM). Other studies have investigated *non-qwerty* layouts with controllers including circular [8, 53] and cubic [50] layouts. Multiple glove-based or optical tracking-based techniques have been proposed to map a keyboard layout onto the hand/fingers [8, 17, 34, 36], reporting speeds in the range of 5 to 10 WPM.

Multiple techniques have also been proposed for typing in AR/VR which do not require controllers or a physical keyboard. Yu et al. [52] investigated head-based text entry on head-mounted displays. But the constant head movement is not comfortable for users and is not suitable for AR. PalmType [46] supports typing on the

palm by pointing fingers on the keyboard projected on the palm in smart wearable displays. HoldBoard [1] used a smartwatch as the text input device for smart glasses. SwipeZone [10] uses a side touchpad for swipe input for smart eyewear. Our work is in the space of freehand mid-air typing in AR/VR using a single hand.

2.2 Single-handed typing using wrist

Multiple wrist wearables explore text-entry on the smartwatch using the second hand [9, 12, 51]. WrisText [7] is a one-handed text entry technique using whirl gestures. There also are works that investigate typing using rings or finger-mounted devices. RotoSwype [11] uses an IMU ring's pitch and roll to perform a gesture typing trace on a VR headset. TipText [49] and BiTipText [48] use miniature touch input surfaces mounted on the index finger to input key taps. TypeAnywhere [54] uses the commercial TapStrap finger bracelets to type anywhere. Shapeshifter [6] enables gesture typing in VR using a force-based thimble. In this paper, we use a wrist IMU to control a cursor on a virtual keyboard in VR.

2.3 Typing using gaze

There are mainly two kinds of typing methods that use gaze: the dwell-based method and the dwell-free method. Dwelling the gaze on each letter to input the letter is intuitive but requires a waiting time which is not efficient [27, 39]. Less dwell time will cause the Midas touch problem [31]. There are works that explored adjustable or cascading dwell-time keyboards to reduce the dwell time by leveraging the user's input rhythm [28] or language models [33].

Dwell-free methods rely on gaze gestures to input at the character level or word level. EyeK [38] used an in-out-in gesture on a key to type a letter. EyeWrite [47] entered letters using special gaze strokes. pEYEWrite [15] used the sections of a pie menu to represent different sets of letters. Users input a letter by crossing the gaze over the border of a section. Dasher [44] zooms in a letter when the user is gazing at it, and the letter gets selected after a threshold is reached. Morimoto et al's context-switching methods [32] confirm a key by moving the gaze to other parts of the screen.

There are methods that support word-level gaze input. Filteryedping [35] requires the users to look at all the letters in the word and then look at a button to generate a list of candidate words. EyeSwipe [24] used the path of the gaze going through all the letters in a word. The user indicates the start and end letters by a reverse cross gesture.

All the above methods use gaze explicitly. Our methods are the first instance of implicit gaze-assisted typing where the user can freely move their gaze and the users are not even aware that their gaze is being used for input.

2.4 Multimodal gaze methods

There are works combining gaze and manual input for target selection [4, 5, 23]. Previous work showed that multimodal text input could be faster than dwell-based text input. Hansen et al. [13] showed the gaze+dwell text input method can input faster than the dwell-only method. Meena et al. [30] showed that the gaze+switch can input faster than the dwell-only method. TAGSwipe [22] combined gaze and touch gestures to reduce the gaze gestures and dwell time. It outperforms the dwell-based method and the swipe-based method.

3 GAZE ASSISTED GESTURE TYPING

3.1 The likelihood of the gaze fixation location being the target

Using an initial prototype driven by the wrist IMU, we observed that during gesture typing, the user's eye gaze usually moves ahead of the cursor to observe the location of the next key and to guide their hand motion toward that direction. This behavior results in a gaze fixation on the next key. This gaze fixation serves as a key component in our implicit gaze assistance methods. Put simply, we speed up the cursor movement in the direction of the gaze fixation location.

However, users do not always fixate on the next key while drawing the gesture trace. They may look at multiple subsequent keys to scan the whole path or they may divert their attention outside the keyboard, e.g. toward the word suggestions or the phrase box, or something else in the wider scene. Therefore, our use of the gaze fixation signal needs to be proportional to how likely the fixation is the next target.

To this end, we first disregard any fixations outside the keyboard region. For the fixations within the keyboard, we look at the movement direction of the cursor. If the cursor is moving toward the location of the gaze fixation, we consider it likely that the fixation location is the next key. Consequently, we propose to use the *alignment* between the wrist cursor's moving direction and the direction toward the gaze fixation as the likelihood of the fixation location being the target. The *alignment* is represented by the *cosine* of the angle between the two directions. When the two directions align well, the cosine of the angle is close to 1, and the gaze fixation is more likely to be the next key. When the two directions do not align well, cosine the of the angle is close to 0 or even negative, and the gaze fixation location is less likely to be the target.

3.2 Speedup: Speeding up the cursor toward the gaze fixation

In our first gaze-assisted method, *Speedup*, we speed up the wrist cursor in the direction toward the gaze fixation, and the speed perpendicular to this direction keeps unchanged. Thus, higher alignment between the wrist cursor's movement direction vector and the gaze fixation direction vector results in higher cursor speedup. This effectively reduces hand movement when the cursor is moving toward the gaze fixation. We now describe the algorithm behind *Speedup*.

The IMU-based wrist cursor's location updates at a rate of 50 Hz. Let the coordinate of the *wrist cursor* on the keyboard at timestamp t be $W_t = (x_t, y_t)$. In our speedup method, we did not directly use the wrist cursor for gesture input. We used the *speedup wrist cursor* whose location is modified from the *wrist cursor* based on the gaze fixation location. Let the coordinate of the speedup wrist cursor on the keyboard at timestamp t be $S_t = (x_t', y_t')$. At the beginning timestamp 0, the speedup wrist cursor is at the same location as the wrist cursor, namely $S_0 = W_0$. The coordinate of the speedup wrist cursor starts to be different from the coordinate of the wrist

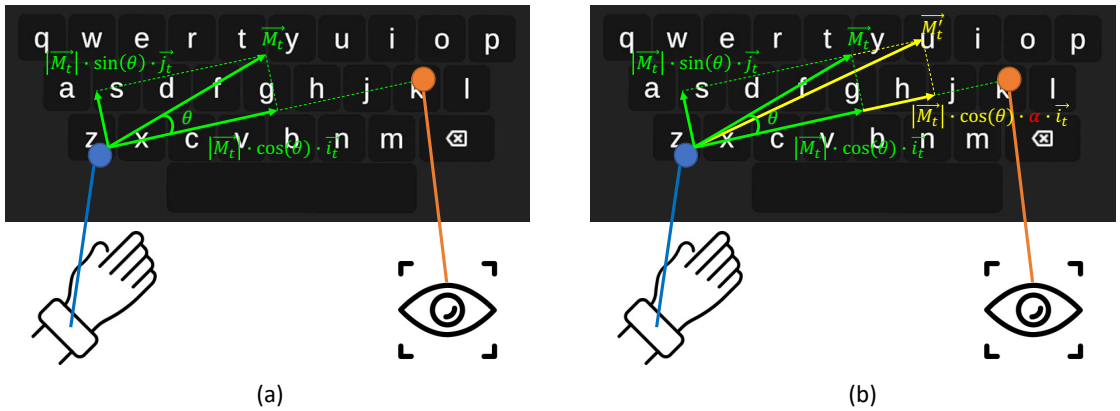


Figure 2: Vector decomposition of the moving offset $\overrightarrow{M_t}$ and $\overrightarrow{M'_t}$ into the direction $\hat{i}_t$ which is the direction toward the gaze fixation and the direction $\hat{j}_t$ which is the direction perpendicular to $\hat{i}_t$. (a) The decomposition of the wristband moving offset $\overrightarrow{M_t}$ into direction $\hat{i}_t$ and direction $\hat{j}_t$. (b) The decomposition of the new moving offset with gaze speedup $\overrightarrow{M'_t}$ into direction $\hat{i}_t$ and direction $\hat{j}_t$. The only difference of $\overrightarrow{M'_t}$ from $\overrightarrow{M_t}$ is that its component on the direction $\hat{i}_t$ is increased by α, where $\alpha \geq 1$.

cursor when the first gaze fixation is detected. After the first gaze fixation is detected, the coordinate of the speedup wrist cursor at the current timestamp S_t is computed as follows.

Let the wrist cursor coordinate at the last timestamp be $W_{t-1} = (x_{t-1}, y_{t-1})$. Then the cursor's moving offset from the last timestamp is $\overrightarrow{M_t} = W_t - W_{t-1}$. For every frame, we get the gaze fixation state F_t (a bool value) from our gaze fixation detection function described in Section 3.4. $F_t == true$ means the gaze is in a fixation state at timestamp t, otherwise, it's in a saccade state. If $F_{t-1} == false$ and $F_t == true$, the gaze is switching from a saccade state to a fixation state, and a new fixation state starts from timestamp t. Once a new fixation state starts, we create a new direction $\hat{i}_t$ from the current speedup wrist cursor location S_t to the new gaze fixation location G_t.

$$\hat{i}_t = \frac{G_t - S_t}{|G_t - S_t|} \tag{1}$$

$\hat{i}_t$ is a unit vector, it's length is 1. If $F_{t-1} == true$ and $F_t == true$, the gaze at timestamp t remains the same fixation state from the last timestamp. If $F_{t-1} == false$ and $F_t == false$, the gaze remains in the same saccade state from the last timestamp.

If there is a gaze fixation at the current timestamp t, namely $F_t = true$, and this gaze fixation state started from timestamp $t - n$ and remained in the fixation state to the current timestamp. When the gaze state switched from saccade to fixation at timestamp $t - n$ ($F_{t-n-1} = false$ and $F_{t-n} = true$), the direction from the speedup wrist cursor S_{t-n} to the gaze fixation location G_{t-n} is computed as

$$\hat{i}_{t-n} = \frac{G_{t-n} - S_{t-n}}{|G_{t-n} - S_{t-n}|}$$

. This direction is used as the direction toward the fixation until the next gaze state switching from saccade to fixation is detected. The gaze state switching from saccade to fixation is the trigger to compute a new direction $\hat{i}$. Even when the gaze is in a saccade state, we continue using the gaze fixation direction $\hat{i}$ from the last state switching (from saccade to fixation) to speed up the trace cursor.

The moving offset of the wrist cursor $\overrightarrow{M_t}$ at timestamp t can be decomposed into direction $\hat{i}_t$ and the direction $\hat{j}_t$ which is perpendicular to $\hat{i}_t$. Let θ be the angle between the wrist cursor's moving vector $\overrightarrow{M_t}$ and $\hat{i}_t$ which is the direction from the wrist cursor to the gaze fixation, $0° \leq \theta \leq 180°$. When $0° \leq \theta \leq 90°$, $\overrightarrow{M_t}$ can be represented as

$$\overrightarrow{M_t} = \left|\overrightarrow{M_t}\right| \cdot \cos(\theta) \cdot \hat{i}_t + \left|\overrightarrow{M_t}\right| \cdot \sin(\theta) \cdot \hat{j}_t \tag{2}$$

as shown in Figure 2 (a). In this way, we decompose $\overrightarrow{M_t}$ into two components on the two directions $\hat{i}_t$ and $\hat{j}_t$. The component in the direction $\hat{i}_t$ is the moving offset toward the gaze fixation while the other component is the moving offset perpendicular to $\hat{i}_t$. The value of $\cos(\theta)$ represents the *alignment* between $\overrightarrow{M_t}$ and $\hat{i}_t$. The smaller θ is, the two vectors are better aligned, and the larger $\cos(\theta)$ is.

To speed up the cursor toward the gaze fixation direction, we increase the component $\left|\overrightarrow{M_t}\right| \cdot \cos(\theta) \cdot \hat{i}_t$ by a constant α ($\alpha \geq 1$) (Figure 2 (b)), the new moving offset with gaze speedup is represented as

$$\overrightarrow{M'}_t = \left|\overrightarrow{M_t}\right| \cdot \cos(\theta) \cdot \alpha \cdot \hat{i}_t + \left|\overrightarrow{M_t}\right| \cdot \sin(\theta) \cdot \hat{j}_t \tag{3}$$

as shown in Figure 2 (b). The speedup wrist cursor location is computed as follows:

$$S_t = S_{t-1} + \overrightarrow{M'}_t \tag{4}$$

We only use the above speed-up process if $0° \leq \theta \leq 90°$. When $90° < \theta \leq 180°$, $\overrightarrow{M_t}$ keeps unchanged, namely $\overrightarrow{M'}_t = \overrightarrow{M_t}$. Because when $90° < \theta \leq 180°$, the speedup wrist cursor is moving away from the gaze fixation. In this situation, the implicit eye gaze location is not the target of the wrist cursor, we will not use the gaze

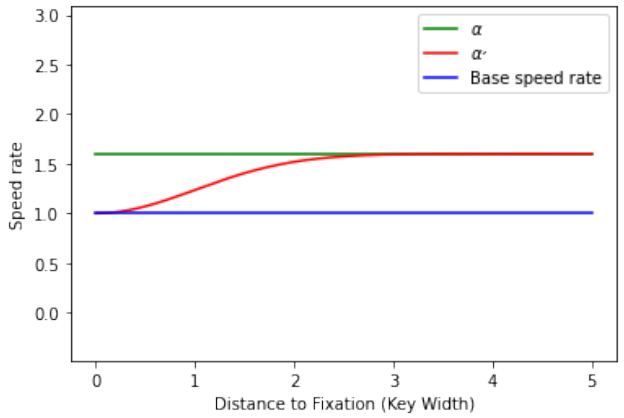


Figure 3: The value of Gaussian speedup rate α', constant speedup rate $\alpha = 1.6$, and base speedup rate 1 changing with the distance d between the speedup wrist cursor and the gaze fixation. The unit of the x axis is the width of a key.

location for speedup. And $\overrightarrow{M_t}$ also keeps unchanged $\overrightarrow{M'_t} = \overrightarrow{M_t}$ before the first gaze fixation is detected because there is no direction $\hat{i}_t$ toward the gaze fixation at this time.

For $0° \leq \theta \leq 90°$, when θ is close to 0°, $\cos(\theta)$ is close to 1. And when θ is close to 90°, $\cos(\theta)$ is close to 0. This means when the moving offset $\overrightarrow{M_t}$ aligns with the direction to fixation $\hat{i}_t$ (θ is close to 0), the overall speedup effect $\cos(\theta) \cdot \alpha$ is close to α. If the two directions are not aligned (θ is close to 90°), the overall speedup effect $\cos(\theta) \cdot \alpha$ is close to 0. After decomposing the moving offset $\overrightarrow{M_t}$ into $\hat{i}_t$ and $\hat{j}_t$ directions, the value of $\cos(\theta)$ becomes an indication of the alignment between moving offset $\overrightarrow{M_t}$ and the direction to fixation $\hat{i}_t$. We use this alignment to automatically adjust the overall speedup effect $\cos(\theta) \cdot \alpha$ on the fly. We call this method of computing the S_t as the *Speedup* method.

The value of α affects input speed, control precision, and user comfort. A larger α leads to a higher speedup rate but low control precision around the target. Besides, the sudden speed change during input due to the large alpha values causes discomfort to users. A small α has less speedup effect but higher control precision, and less discomfort for users. On the other hand, if there is an optimal α value, multiple factors can affect the optimal value, such as the size of the virtual keyboard, the distance between the eyes and the virtual keyboard, the base speed of the cursor, the accuracy of the eye tracker, the smoothing filter for the eye tracking data, and so on. In our implementation, we selected the α value based on a highly iterative and designer-led approach. We tried different alpha values ranging from 1 to 5 on a few examples and found that $\alpha = 1.6$ does not cause obvious discomfort and its control precision is acceptable, at the same time it can utilize the benefits of the speedup.

3.3 Gaussian Speedup

In the process of using the *Speedup* method, we observed that it has a tendency to overshoot the target because of the cursor's speedup. To reduce the overshooting, we propose our second method - *Gaussian Speedup*, which reduces the speedup rate when the wrist cursor gets nearer to the gaze fixation. We use a Gaussian smoothing function to gradually reduce the speedup rate with the distance between the cursor and the gaze fixation. This enables users to have both precise control near the gaze fixation and a higher speed away from the gaze fixation.

To this end, we replace the constant speedup rate α with α', which is a function of the distance d between the speedup wrist cursor and the gaze fixation at the last timestamp $t-1$.

$$\alpha' = 1 + (\alpha - 1) \cdot (1 - exp(-\frac{d^2}{2\sigma^2})) \tag{5}$$

where

$$d = |S_{t-1} - G_{t-1}| \tag{6}$$

.

As d becomes smaller, α' becomes smaller as per the Gaussian function. When d is 0, $\alpha' = 1$. When d is infinite, $\alpha' = \alpha$. In our implementation, we used $\sigma = 1$, where its unit is the width of a key. We chose the value by experimenting with a small number of trials. The value of σ decides the diameter of the area where the user can have precise control. A larger σ leads to larger areas of precise control and less speedup. We used the same $\alpha = 1.6$ as the Speedup method. Figure 3 shows the value of α' and α and the base speed rate changing with distance d. We can see that as d becomes smaller, α' gradually changes from α to the base speed rate 1.

Algorithm 1 shows the pseudo-code for computing the speedup wrist cursor location S_t at timestamp t for the Gaussian speedup method. The pseudo-code of the Speedup method is similar to Algorithm 1, the only difference is $\alpha' = \alpha$ in step 3 for the speedup method.

3.4 Gaze fixation detection

We followed the method in [37] to detect gaze fixation. If the moving speed of the gaze is higher than a threshold for a certain time, the gaze will be considered as fixation, otherwise, it's a saccade. In our implementation, a fixation is detected if the gaze speed is lower than 23.47° per second for more than 5 timestamps. The timestamp update rate is 50 Hz in our Unity application. The latency of gaze fixation detection is 5/50 second = 0.1 second.

Although the above method works well to detect gaze fixation, the gaze may move slowly in the fixation state. During the fixation state, if the gaze moves away from where the fixation was first detected for a distance larger than the width of the key, we will consider a new fixation is detected. This is useful for our gaze-assisted gesture typing method because we used the first gaze sample location of each fixation state as the location for that fixation state.

4 EXPERIMENT

We carried out a user study to evaluate the performance of our proposed methods. We evaluated three methods, *Wrist-only*: the baseline wrist IMU-driven gesture typing technique which does not use gaze-assistance, the *Speedup*, and *Gaussian Speedup* methods we described earlier. Before the study, our hypotheses (H1 to H5) are as follows.

Algorithm 1 Compute the speedup wrist cursor location using the Gaussian speedup method

Require: At timestamp 0, the speedup wrist cursor location S_0 equals the wrist IMU's cursor location W_0.
The wrist IMU's cursor location at the current timestamp $W_t = (x_t, y_t)$ and the previous timestamp $W_{t-1} = (x_{t-1}, y_{t-1})$.
The speedup wrist cursor location at previous timestamp $S_{t-1} = (x_{t-1}', y_{t-1}')$.
The gaze fixation location at the current timestamp $G_t = (x_t'', y_t'')$ and the previous timestamp $G_{t-1} = (x_{t-1}'', y_{t-1}'')$.
The gaze fixation state at the current timestamp F_t and the previous timestamp F_{t-1}. F_t is a bool value representing whether the gaze is in a fixation state.
Speedup direction $\hat{i}_t$ and its orthogonal direction $\hat{j}_t$ if they exist, they do not exist before the first gaze fixation is detected for a trace.

Compute the wristband movement offset vector from the last timestamp to this timestamp $\overrightarrow{M_t} = W_t - W_{t-1}$.
Compute distance $d = |S_{t-1} - G_{t-1}|$ between the speedup wrist cursor and the gaze fixation at the last timestamp $t-1$.
Update the speedup rate $\alpha' = 1 + (\alpha - 1) \cdot (1 - exp(-\frac{d^2}{2\sigma^2})))$.
if $\hat{i}_t$ exists **then**
  The angle between $\overrightarrow{M_t}$ and $\hat{i}_t$ is θ and $0° \le \theta \le 180°$
  $\hat{j}_t$ is the direction perpendicular to $\hat{i}_t$ so that the $\overrightarrow{M_t} = \left|\overrightarrow{M_t}\right| \cdot \cos(\theta) \cdot \hat{i}_t + \left|\overrightarrow{M_t}\right| \cdot \sin(\theta) \cdot \hat{j}_t$
  if $0° \le \theta \le 90°$ **then**
    Compute the moving offset of trace cursor with speedup $\overrightarrow{M'}_t = \left|\overrightarrow{M_t}\right| \cdot \cos(\theta) \cdot \alpha' \cdot \hat{i}_t + \left|\overrightarrow{M_t}\right| \cdot \sin(\theta) \cdot \hat{j}_t$
  else
    $\overrightarrow{M'}_t = \overrightarrow{M_t}$
  end if
else
  $\overrightarrow{M'}_t = \overrightarrow{M_t}$
end if
Compute the speedup wrist cursor location at the current timestamp $S_t = S_{t-1} + \overrightarrow{M'}_t$
if A new gaze fixation is detected, namely $F_{t-1} = false$ and $F_t == true$ **then**
  Compute the direction from the trace cursor to the gaze fixation $\hat{i}_t = \frac{G_t - S_t}{|G_t - S_t|}$
else
  $\hat{i}_t = \hat{i}_{t-1}$
end if
return The speedup wrist cursor location at the current timestamp S_t

- H1: The alignment $\cos(\theta)$ in Equation (3) is a reliable indication of how likely the gaze fixation location is the target location of the wrist cursor.
- H2: The speedup method reduces hand movement.
- H3: The Gaussian-speedup method reduces hand movement.
- H4: The speedup method increases input speed.

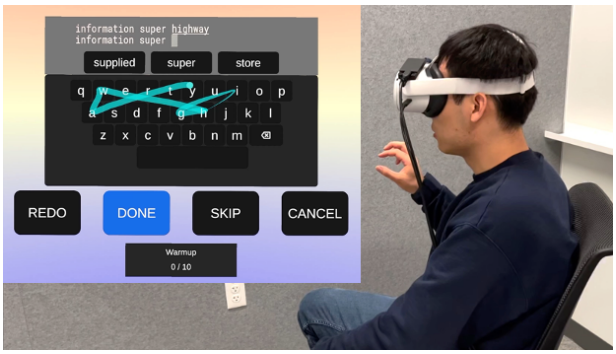


Figure 4: A participant is typing during the study. The subfigure on the top left is what the user saw in the VR headset.

- H5: The Gaussian-speedup method increases input speed.

4.1 Participants

The study has 12 participants (3 male, 9 female, age: 23-42, mean = 30.41, sd = 5.85). The self-reported median familiarity (1: not familiar, 10: very familiar) with gesture typing and AR/VR devices are 4 and 5.5 respectively.

4.2 Task and Apparatus

The gesture typing application was developed in Unity using the SHARK2 decoder algorithm [20]. The task is the standard phrase transcription task with phrases sourced from Mackenzie et al's phrase dataset [26]. We used a Meta Quest 2 VR headset with an eye tracker inserted inside the headset (Figure 4). The eye tracker is specifically designed for the Meta Quest 2 for research purposes. It can be fixed before the lenses in the headset and it directly connects to the laptop by wires. The Unity game engine collects its data. The sampling rate of the eye tracker is 50 Hz. The gaze signal is filtered by a 1€ filter [3] with parameters $fcmin = 1$ and $beta = 1$. At the beginning of the study, users first calibrated the gaze. We recorded the $P75$ accuracy and precision for each user, which means 75% of gaze samples are more accurate or precise than the accuracy or precision values. For the 12 participants in the study, the mean (standard deviation) of the $P75$ gaze tracking accuracy and precision are 1.31° (0.43°) and 0.65° (0.26°).

We used a wristband with an IMU sensor to control the cursor and detect hand gestures. Since inferring distance using an IMU has drift problems, we used an arm motion model that provided the user's wrist position using a combination of biomechanical constraints and wrist IMU data [40, 42], stabilizing the output wrist position. The user's wrist position was then translated into the cursor position on the keyboard. A smaller range of hand movements mapped to the keyboard would require more precision from the user, while a larger range would require less precision. We ensured an optimal translation that balanced the required precision with the required amount of hand movement. The sampling rate of the IMU wristband is 50 Hz. For every $1cm$ the wrist moves in the physical world, the cursor moves $0.119m$ on the virtual keyboard. In the virtual scene, the users' viewpoint is 3 meters away from the keyboard. The keyboard is $3.82m$ wide and $1.5625m$ tall. In this study,

we used a large keyboard so that the gaze tracking is relatively more accurate on the keyboard.

4.3 Study design

We used a within-subjects design where participants did all three methods counterbalanced using a Latin square design across the 12 participants. Before the typing tasks, we first did eye gaze calibration for each user. Note that participants were informed that eye tracking is part of the data collection, but were not told that gaze would be used as an input signal. For each method, participants practiced 5 phrases before starting the study. In each method, participants typed 14 phrases with a break in between (after 7 phrases). We term the first 7 phrases as block 1 and the next 7 as block 2. While the 7 sentences in each block remain the same across users, their order within each block is randomly shuffled. In total there were 12 user × 3 methods × 14 phrases = 504 trials.

4.4 Results

We used multiple metrics to evaluate different aspects of the input methods. There are three types of metrics.

- Type 1: Metrics to test the hypotheses, including hand movement in Section 4.4.1 and overall input speed in Section 4.4.2. Those metrics are used to test whether the hypotheses (H1 to H5) are supported by the study results.
- Type 2: Metrics about input experience, including error rate, backspace usage, suggestion usage, gesture duration, and effect of block in Section 4.4.3 to Section 4.4.7. Although those metrics are not the direct goals to be improved by the proposed methods, they are still important metrics reflecting different aspects of users' input experience. They are also good guardrail metrics because at least the proposed methods should not be detrimental to those metrics.
- Type 3: Subjective metrics including mental demand, physical demand, temporal demand and effort in Section 4.4.8.

4.4.1 Hand movement. To evaluate how much hand movement was reduced by the proposed methods, we recorded the trace length per word for both the original wrist IMU cursor (*Wrist Trace Length*) and the speedup cursor (*Final Trace Length*). For *Speedup* and *Gaussian Speedup*, users only saw the speedup cursor. Wrist Trace Length indicates how much the hand actually moved. The Final Trace Length is the length of the trace that the users finally saw on the keyboard.

Figure 5 shows the mean (95% confidence interval) two quantities for the three methods. Mean (95% CI) Wrist Trace Length per word are as follows: *Wrist-Only*: 1252.2 ([1217.5, 1286.8]), *Speedup*: 874.7 ([853.5, 895.8]), *Gaussian Speedup*: 980.2 ([941.1, 1019.3]). A rm-ANOVA showed a significant effect of method on the Wrist Trace Length ($F_{2,22} = 485.999$, $p < 0.001$, $\eta^2 = 0.978$). Pairwise comparisons with Bonferroni correction yielded significant differences between all three pairs: wrist-only vs. speedup (p<0.001), wrist-only vs. Gaussian speedup (p<0.001), and speedup vs. Gaussian Speedup (p<0.001).

Our *Speedup* method significantly reduced hand movement by 30.15% relative to the *Wrist-Only* baseline. The Gaussian speedup method reduced it by 21.72%. Thus H2 and H3 are directly supported by those results. H1 is also supported by those results. Because H2 and H3 depend on H1. H2 and H3 would not be supported by the results if H1 does not hold.

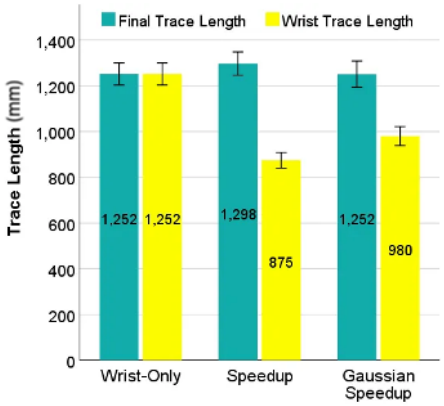


Figure 5: The average (95% confidence interval) trace lengths (in millimeters) per word of the original wrist cursor and the final trace for the three methods. For context, each key length was 97mm.

Figure 6 shows an example Wrist trace and Final trace for inputting the word "able" using both methods. Based on our translation of wrist IMU data to keyboard position, *Speedup* saves 1.2cm of actual hand movement per word and *Gaussian Speedup* saves 0.85cm, which is substantial. This implies an even more sizeable reduction in hand movement over a phrase or a larger piece of text.

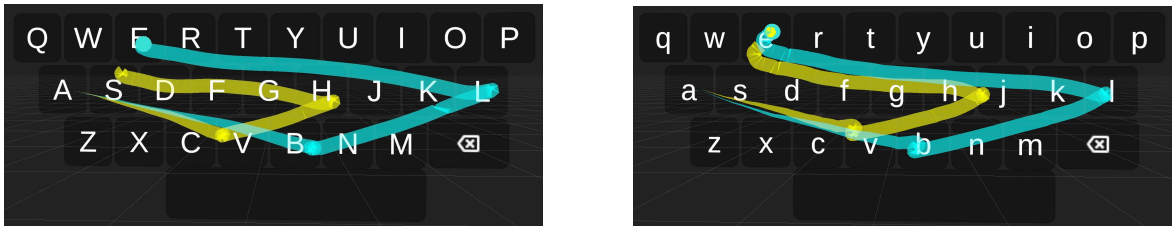
A rm-ANOVA also showed a significant main effect of method on Final Trace Length ($F_{2,22} = 5.668$, $p < 0.05$, $\eta^2 = 0.340$). Pairwise comparisons with Bonferroni correction yielded a significant difference between wrist-only and speedup (p<0.01).

Figure 7 shows the scatter plot of the two trace lengths, with each dot representing a phrase. We can see that the gaze-assisted methods consistently reduced the hand movement for every phrase (All *Speedup* or *Gaussian Speedup* dots are below the diagonal). Further, the wrist trace length seems to be an almost linear function of the final trace length. Thus the hand movement reduction becomes linearly larger as the word becomes longer. This linear relationship also implies that we can predict the amount of hand movement reductions for various lengths of text. H1 is also supported by the consistent hand movement reduction across different phrases in Figure 7.

4.4.2 Overall input speed. We used words per minute (WPM) [25] as the metric for overall input speed:

$$WPM = \frac{|L-1|}{T} \times \frac{1}{5} \tag{7}$$

where L is the length of the typed text in characters including spaces and T is the time from the start of the first trace to the end of the last trace. Figure 8 (a) shows the WPM across all trials. The mean (95% CI) WPM were as follows: *Wrist-Only*: 16.4 ([13.7, 19.1]), *Speedup*: 17.6 ([15.0, 20.1]), *Gaussian Speedup*: 17.1 ([14.7, 19.4]). No significant effects were found. Although the two proposed methods slightly increased the input speeds, the increases are not significant.



(a) Speedup method

(b) Gaussian Speedup method

Figure 6: The final trace (blue) and the original wristband trace (yellow) for inputting the word "able" using the speedup method in (a) and the Gaussian speedup method in (b).

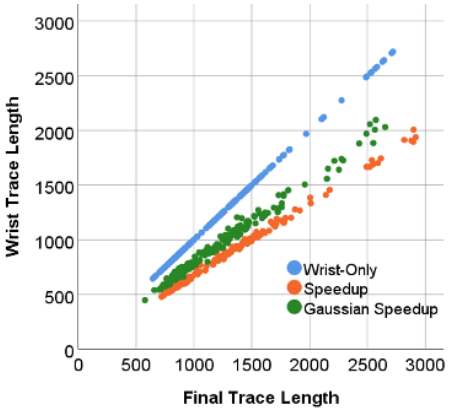


Figure 7: Scatter plot of wrist trace length versus the final trace length for all phrases of the three methods.

Thus H4 and H5 are not supported by the results. The possible causes for the insignificant increases are that users' input speed is not only affected by the cursor speed but also by users' familiarity with the keyboard layout, speed of spelling, rest time between words, and so on. We did a power analysis (tail(s) = one, effect size ds = 0.8, α err prob = 0.05, power (1-β error prob) = 0.8) which shows that at least 12 participants are needed for the t-test of two dependent means. The 12 participants in this study are sufficient to test H4 and H5, more participants may lead to more informative results.

4.4.3 Error rate. To evaluate the accuracy of users' input, we used word error rate (WER) defined as follows:

$$e = \frac{MWD(T, I)}{WordsCount(I)} \times 100\% \tag{8}$$

where $MWD(T, I)$ is the minimum word distance. It is the minimum number of single-character edits (insertions, deletions, or substitutions) required to change the transcribed phrase I to the target phrase T.

As shown in Figure 8 (b), the mean (95% CI) WER was as follows: *Wrist-Only*: 0.98% ([.02%, 1.94%]), *Speedup*: 1.01% ([0.04%, 1.98%]), *Gaussian Speedup*: 0.71% ([−0.03%, 1.46%]). No significant effects were found.

4.4.4 Backspace usage. To evaluate how often users use backspace to correct their input, we used the backspace to words ratio defined as follows:

$$BR = \frac{N_b}{WordsCount(I)} \tag{9}$$

where N_b is the number of times backspace was used in a phrase. The mean (95% CI) backspace usage was as follows: *Wrist-Only*: 7.9% ([4.3%, 11.4%]), *Speedup*: 6.2% ([4.0%, 8.4%]) , *Gaussian Speedup*: 6.1% ([3.3%, 8.9%]), as shown in Figure 8 (c). No significant effects were found.

4.4.5 Suggestion usage. To evaluate how often users select suggestions to correct their input, we used suggestion to words ratio defined as follows:

$$SR = \frac{N_s}{WordsCount(I)} \tag{10}$$

where N_s is the number of times suggestions were selected in a phrase. The mean (95% CI) suggestion to words ratio was as follows: *Wrist-Only*: 5.9% ([4.2%, 7.7%]), *Speedup*: 5.0% ([3.7%, 6.2%]), *Gaussian Speedup*: 6.6% ([4.1%, 9.1%]), as shown in Figure 8 (d). No significant effects were found.

4.4.6 Gesture duration. Figure 9a shows the time duration of each trace with and without gaze-assisted speedup. Any time when the gaze was used to speed up the trace even by a small amount was counted towards the duration with speedup. The mean (95% CI) duration per phrase in seconds was as follows: *Wrist-Only*: 11.9 ([9.5, 14.4]), *Speedup*: 10.8 ([9.1, 12.5]), and *Gaussian Speedup*: 10.5 ([8.2, 12.9]). The mean (95% CI) trace duration with speedup was as follows: *Speedup*: 10.1 ([8.5, 11.8]), *Gaussian Speedup*: 9.9, ([7.6, 12.2]). Thus, 93.5% and 94.2% of the time, the cursor was speeding up in some form. This high percentage is because the speedup algorithm continues to use the prior gaze fixation information until a new gaze fixation is encountered. Thus, even if the user's gaze is in a saccade mode, if there has been a previous gaze fixation on the keyboard, the speedup continues to happen using that prior information.

4.4.7 Effect of Block. To analyze the effect of block along with the method, we also ran 2-way rm-ANOVA tests on the above quantities. However, we found no interaction effects to suggest that the effect of the typing method may have changed over the course of the two blocks. Expectedly, there were main effects of block on the user speed ($F(1, 11) = 20.072$, $p < 0.005$, $\eta^2 = .646$). As seen in Figure 9b, the effect is fairly consistent for all three methods.

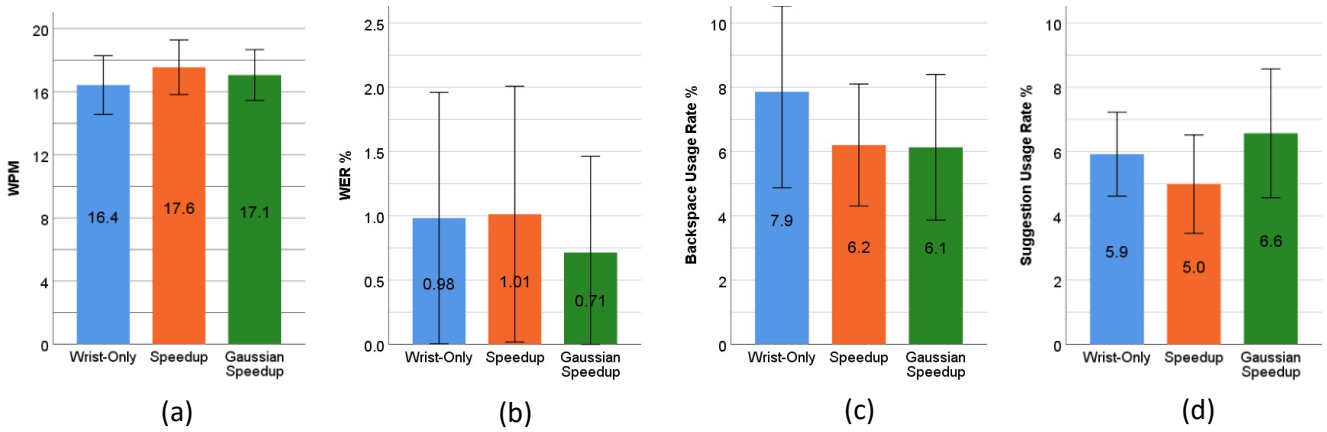
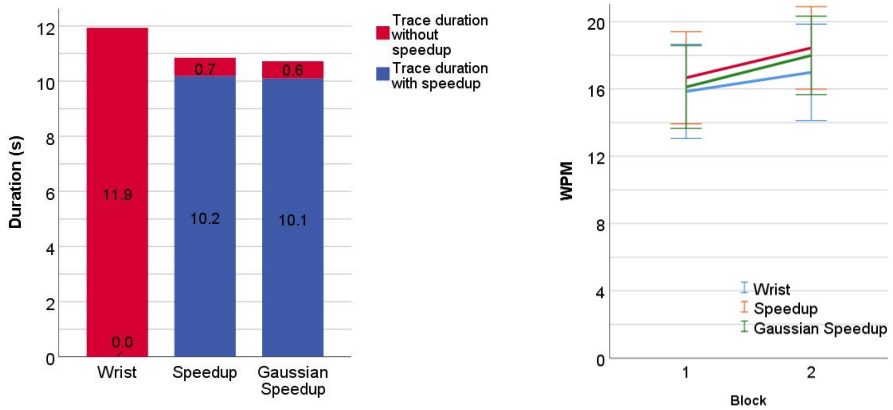


Figure 8: WPM, error rates, backspace rates, and suggestion rates for the three methods.



(a) Gesture duration per phrase **(b) WPM by Block**

Figure 9: (a) Gesture duration, (b) WPM by Block

4.4.8 Subjective ratings. At the end of the study, we asked users to rate different methods on mental demand, physical demand, temporal demand, and effort (1: least demanding, 10: most demanding). The results are shown in Figure 10. This shows that the implicit assistance did not have any negative impact on the task's mental demand or effort for the user. The physical demand scores are similar (although the variance in Wrist-Only is much higher). Longer passages of text may start leading to arm fatigue, which is where the user might start to feel the difference in the perceived physical demand. At the same time, temporal demand and effort scores are on the lower side for the *speedup* methods which may indicate that lower hand movement did have a small impact on the perceived effort and time taken. However, Friedman tests did not show any significant effects.

5 DISCUSSION

5.1 Hand Movement and Speed

Our proposed implicit gaze-assisted methods significantly reduced hand movement without loss of input speed, accuracy, and mental demand. The study results support H1, H2, and H3. Although the input speeds of our methods are slightly higher than the wrist-only baseline, the differences are not statistically significant. H4 and H5 are not supported by the study results. This suggests that users adjust to the cursor speed up and reduce the amount of effort they are putting in.

5.2 Final Trace

Speedup's final trace length is significantly longer than that of the wrist-only method (Figure 5). This is because *Speedup* can cause overshooting as previously mentioned. However, the overshooting does not affect the error rates because the gesture decoder can tolerate the overshooting and still decode the correct word.

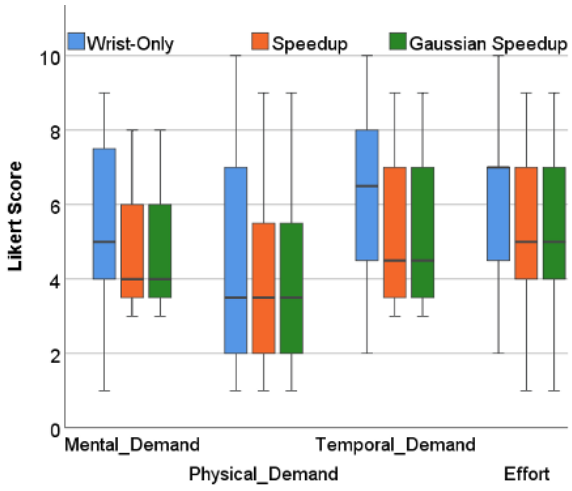


Figure 10: Subjective scores.

The final trace length of *Gaussian speedup* is similar to the wrist-only method. This is because the Gaussian speedup rate α' prevented the overshooting problem and allows users to have precise control around gaze fixation. A test set with higher perplexity might be more prone to overshooting errors. Thus, despite its lower reduction in hand movement, we believe the *Gaussian speedup* method can be highly advantageous in both significantly reducing the amount of hand motion and allowing precise control around the gaze fixation.

5.3 Learning overhead

As seen in Figure 9b, the gaze-assisted methods' initial speeds are similar to Wrist-Only and are reasonably high at 16 WPM in block 1. Thus, the learning overhead of the gaze-assisted methods is low. Since we used implicit eye gaze information to speed up the cursor, our user interface for gesture typing is exactly the same as typing in mid-air using hand motion.

5.4 The value of α

In this paper, we just picked an appropriate α to prove the effectiveness of the proposed methods. We acknowledge that this α value might not be the optimal one. Searching for the optimal α value could be future work if it exists. And further research is needed to investigate the factors affecting the optimal α values, such as the keyboard size, eye tracking accuracy, the base speed of the wrist cursor, and so on. Moreover, a single alpha value may not work well across all users. The optimal α value may be participant dependent.

5.5 Implicit Gaze

We used the alignment ($\cos(\theta)$, $0^\circ \leq \theta \leq 90^\circ$) between the wrist cursor's moving direction and the direction toward the gaze fixation as the indication of how likely the gaze fixation is the target. Even if an experienced user is able to gesture type without ever looking at the next key, our method ensures the baseline, no speed-up performance at the minimum. It will be interesting to investigate the performance of the speedup methods over long-term usage. Although the experiment results show that our heuristic methods work well, there is space to explore implicit gaze information for text input using machine learning approaches or as an input to the decoder.

6 CONCLUSION

In this work, we used the user's implicit eye gaze to reduce the hand movement for gesture typing in the air in virtual reality. We assumed that the alignment between the wrist cursor's moving direction and the direction toward the gaze fixation can be used as the likelihood of the gaze fixation location being the target. With this assumption, we proposed two methods, *Speedup* and *Gaussian Speedup* to speed up the cursor towards the gaze fixation direction. The methods allow the users to move their gaze freely anywhere in the scene, while opportunistically using gaze fixations on the keyboard for speeding up the trace. In *Gaussian Speedup*, we further propose a dynamically changing speedup rate that reduces the hand movement and at the same time allows precise control near the target. Through a user study, we demonstrated that the *Speedup* and *Gaussian Speedup* methods reduced the hand movement by 30% and 22% without any loss of input speed and error rate. We believe that the implicit use of gaze in real-time to enable adaptive interactions is a compelling, but nascent space and hope that our work spurs future work in this area.

REFERENCES

[1] Sunggeun Ahn, Seongkook Heo, and Geehyuk Lee. 2017. Typing on a smartwatch for smart glasses. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces*. 201–209.

[2] Doug A. Bowman, Christopher J. Rhoton, and Marcio S. Pinho. 2002. Text Input Techniques for Immersive Virtual Environments: An Empirical Comparison. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 46, 26 (sep 2002), 2154–2158. https://doi.org/10.1177/154193120204602611

[3] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1€ filter: a simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2527–2530.

[4] Alexander De Luca, Roman Weiss, and Heiko Drewes. 2007. Evaluation of eye-gaze interaction methods for security enhanced PIN-entry. In *Proceedings of the 19th australasian conference on computer-human interaction: Entertaining user interfaces*. 199–202.

[5] Heiko Drewes and Albrecht Schmidt. 2009. The MAGIC touch: Combining MAGIC-pointing with a touch-sensitive mouse. In *IFIP Conference on Human-Computer Interaction*. Springer, 415–428.

[6] Tafadzwa Joseph Dube, Kevin Johnson, and Ahmed Sabbir Arif. 2022. Shapeshifter: Gesture Typing in Virtual Reality with a Force-Based Digital Thimble. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI EA '22)*. Association for Computing Machinery, New York, NY, USA, Article 230, 9 pages. https://doi.org/10.1145/3491101.3519679

[7] Jun Gong, Zheer Xu, Qifan Guo, Teddy Seyed, Xiang 'Anthony' Chen, Xiaojun Bi, and Xing-Dong Yang. 2018. WrisText. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*. ACM Press, New York, New York, USA, 1–14. https://doi.org/10.1145/3173574.3173755

[8] Gabriel González, José P. Molina, Arturo S. García, Diego Martínez, and Pascual González. 2009. Evaluation of Text Input Techniques in Immersive Virtual Environments. In *New Trends on Human-Computer Interaction*. Springer London, London, 1–10. https://doi.org/10.1007/978-1-84882-352-5_11

[9] Mitchell Gordon, Tom Ouyang, and Shumin Zhai. 2016. WatchWriter. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. ACM Press, New York, New York, USA, 3817–3821. https://doi.org/10.1145/2858036.2858242

[10] Tovi Grossman, Xiang Anthony Chen, and George Fitzmaurice. 2015. Typing on glasses: Adapting text entry to smart eyewear. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. 144–152.

[11] Aakar Gupta, Cheng Ji, Hui-Shyong Yeo, Aaron Quigley, and Daniel Vogel. 2019. RotoSwype: Word-Gesture Typing using a Ring. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*. ACM Press, New York, New York, USA, 1–12. https://doi.org/10.1145/3290605.3300244

[12] Aakar Gupta, Thomas Pietrzak, Nicolas Roussel, and Ravin Balakrishnan. 2016. Direct Manipulation in Tactile Displays. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 3683–3693. https://doi.org/10.1145/2858036.2858161

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

[14] HTC. 2019. Vive. https://www.vive.com

[15] Anke Huckauf and Mario Urbina. 2007. Gazing with pEYE: new concepts in eye typing. In *Proceedings of the 4th Symposium on Applied Perception in Graphics and Visualization*. 141–141.

[16] Robert J. K. Jacob. 1991. The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look at is What You Get. *ACM Trans. Inf. Syst.* 9, 2 (apr 1991), 152–169. https://doi.org/10.1145/123078.128728

[17] Haiyan Jiang, Dongdong Weng, Zhenliang Zhang, Feng Chen, Haiyan Jiang, Dongdong Weng, Zhenliang Zhang, and Feng Chen. 2019. HiFinger: One-Handed Text Entry Technique for Virtual Environments Based on Touches between Fingers. *Sensors* 19, 14 (jul 2019), 3063. https://doi.org/10.3390/s19143063

[18] Youngwon R. Kim and Gerard J. Kim. 2016. HoVR-type: smartphone as a typing interface in VR using hovering. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology - VRST '16*. ACM Press, New York, New York, USA, 333–334. https://doi.org/10.1145/2993369.2996330

[19] 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*. 1–9.

[20] Per-Ola Kristensson and Shumin Zhai. 2004. 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*. 43–52.

[21] Per Ola Kristensson and Shumin Zhai. 2007. Command Strokes with and Without Preview: Using Pen Gestures on Keyboard for Command Selection. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. ACM, New York, NY, USA, 1137–1146. https://doi.org/10.1145/1240624.1240797

[22] Chandan Kumar, Ramin Hedeshy, I Scott MacKenzie, and Steffen Staab. 2020. TAGSwipe: Touch Assisted Gaze Swipe for Text Entry. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.

[23] Manu Kumar, Andreas Paepcke, and Terry Winograd. 2007. EyePoint: practical pointing and selection using gaze and keyboard. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 421–430.

[24] Andrew Kurauchi, Wenxin Feng, Ajjen Joshi, Carlos Morimoto, and Margrit Betke. 2016. EyeSwipe: Dwell-free text entry using gaze paths. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 1952–1956.

[25] I. Scott MacKenzie. 2015. A Note on Calculating Text Entry Speed. http://www.yorku.ca/mack/RN-TextEntrySpeed.html. [Online; Accessed: 2022-09-23].

[26] 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* (Ft. Lauderdale, Florida, USA) *(CHI EA '03)*. Association for Computing Machinery, New York, NY, USA, 754–755. https://doi.org/10.1145/765891.765971

[27] Päivi Majaranta. 2012. Communication and text entry by gaze. In *Gaze interaction and applications of eye tracking: Advances in assistive technologies*. IGI Global, 63–77.

[28] Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. 2009. Fast gaze typing with an adjustable dwell time. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 357–360.

[29] Anders Markussen, Mikkel Rønne Jakobsen, Kasper Hornbæk, Anders Markussen, Mikkel Rønne Jakobsen, and Kasper Hornbæk. 2014. Vulture. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*. ACM Press, New York, New York, USA, 1073–1082. https://doi.org/10.1145/2556288.2556964

[30] Yogesh Kumar Meena, Hubert Cecotti, K Wong-Lin, and Girijesh Prasad. 2016. A novel multimodal gaze-controlled hindi virtual keyboard for disabled users. In *2016 ieee international conference on systems, man, and cybernetics (smc)*. IEEE, 003688–003693.

[31] Raphael Menges, Chandan Kumar, and Steffen Staab. 2019. Improving user experience of eye tracking-based interaction: Introspecting and adapting interfaces. *ACM Transactions on Computer-Human Interaction (TOCHI)* 26, 6 (2019), 1–46.

[32] Carlos H Morimoto and Arnon Amir. 2010. Context switching for fast key selection in text entry applications. In *Proceedings of the 2010 symposium on eye-tracking research & applications*. 271–274.

[33] Martez E Mott, Shane Williams, Jacob O Wobbrock, and Meredith Ringel Morris. 2017. Improving dwell-based gaze typing with dynamic, cascading dwell times. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2558–2570.

[34] Taihei Ogitani, Yoshitaka Arahori, Yusuke Shinyama, and Katsuhiko Gondow. 2018. Space Saving Text Input Method for Head Mounted Display with Virtual 12-key Keyboard. In *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)*. IEEE, 342–349. https://doi.org/10.1109/AINA.2018.00059

[35] Diogo Pedrosa, Maria da Graça Pimentel, and Khai N Truong. 2015. Filteryedping: A dwell-free eye typing technique. In *Proceedings of the 33rd annual acm conference extended abstracts on human factors in computing systems*. 303–306.

[36] Manuel Prätorius, Dimitar Valkov, Ulrich Burgbacher, and Klaus Hinrichs. 2014. DigiTap: an eyes-free VR/AR symbolic input device. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology - VRST '14*. ACM Press, New York, New York, USA, 9–18. https://doi.org/10.1145/2671015.2671029

[37] Tobii Pro. 2016. The Tobii Pro fixation filters (eye movement classification). https://tobii.23video.com/the-tobii-pro-fixation-filters-eye-movement. [Online; Accessed: 2022-10-14].

[38] Sayan Sarcar, Prateek Panwar, and Tuhin Chakraborty. 2013. EyeK: an efficient dwell-free eye gaze-based text entry system. In *Proceedings of the 11th asia pacific conference on computer human interaction*. 215–220.

[39] Korok Sengupta, Raphael Menges, Chandan Kumar, and Steffen Staab. 2019. Impact of variable positioning of text prediction in gaze-based text entry. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. 1–9.

[40] Sheng Shen, He Wang, and Romit Roy Choudhury. 2016. I Am a Smartwatch and I Can Track My User's Arm. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services* (Singapore, Singapore) *(MobiSys '16)*. Association for Computing Machinery, New York, NY, USA, 85–96. https://doi.org/10.1145/2906388.2906407

[41] Jeongmin Son, Sunggeun Ahn, Sunbum Kim, and Geehyuk Lee. 2019. Improving Two-Thumb Touchpad Typing in Virtual Reality. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems - CHI EA '19*. ACM Press, New York, New York, USA, 1–6. https://doi.org/10.1145/3290607.3312926

[42] Zhipeng Song, Zhichao Cao, Zhenjiang Li, Jiliang Wang, and Yunhao Liu. 2021. Inertial motion tracking on mobile and wearable devices: Recent advancements and challenges. *Tsinghua Science and Technology* 26, 5 (2021), 692–705. https://doi.org/10.26599/TST.2021.9010017

[43] 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*. ACM Press, New York, New York, USA, 1–13. https://doi.org/10.1145/3173574.3174221

[44] Outi Tuisku, Päivi Majaranta, Poika Isokoski, and Kari-Jouko Räihä. 2008. Now Dasher! Dash away! Longitudinal study of fast text entry by eye gaze. In *Proceedings of the 2008 symposium on Eye tracking research & applications*. 19–26.

[45] UploadVR. 2019. Oculus Claims Breakthrough in Hand-tracking Accuracy. https://www.roadtovr.com/oculus-claims-breakthrough-in-hand-tracking-accuracy/

[46] Cheng-Yao Wang, Wei-Chen Chu, Po-Tsung Chiu, Min-Chieh Hsiu, Yih-Harn Chiang, and Mike Y Chen. 2015. PalmType: Using palms as keyboards for smart glasses. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. 153–160.

[47] Jacob O Wobbrock, James Rubinstein, Michael W Sawyer, and Andrew T Duchowski. 2008. Longitudinal evaluation of discrete consecutive gaze gestures for text entry. In *Proceedings of the 2008 symposium on Eye tracking research & applications*. 11–18.

[48] Zheer Xu, Weihao Chen, Dongyang Zhao, Jiehui Luo, Te-Yen Wu, Jun Gong, Sicheng Yin, Jialun Zhai, and Xing-Dong Yang. 2020. BiTipText: Bimanual Eyes-Free Text Entry on a Fingertip Keyboard. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376306

[49] Zheer Xu, Pui Chung Wong, Jun Gong, Te-Yen Wu, Aditya Shekhar Nittala, Xiaojun Bi, Jürgen Steimle, Hongbo Fu, Kening Zhu, and Xing-Dong Yang. 2019. TipText: Eyes-Free Text Entry on a Fingertip Keyboard. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 883–899. https://doi.org/10.1145/3332165.3347865

[50] Naoki Yanagihara and Buntarou Shizuki. 2018. Cubic Keyboard for Virtual Reality. In *Proceedings of the Symposium on Spatial User Interaction - SUI '18*. ACM Press, New York, New York, USA, 170–170. https://doi.org/10.1145/3267782.3274687

[51] Xin Yi, Chun Yu, Weijie Xu, Xiaojun Bi, and Yuanchun Shi. 2017. COMPASS: Rotational Keyboard on Non-Touch Smartwatches. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 705–715. https://doi.org/10.1145/3025453.3025454

[52] 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*. 4479–4488.

[53] 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

[54] Mingrui Ray Zhang, Shumin Zhai, and Jacob O. Wobbrock. 2022. TypeAnywhere: A QWERTY-Based Text Entry Solution for Ubiquitous Computing. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 339, 16 pages. https://doi.org/10.1145/3491102.3517686