Enhancing Smartphone Eye Tracking with Cursor-Based Interactive Implicit Calibration

Chang Liu
Department of Computer Science and Technology
Tsinghua University
BeiJing, China
c-liu21@tsinghua.org.cn

Xiangyang Wang
Department of Computer Science and Technology
Tsinghua University
Beijing, China
xiangyang24@mails.tsinghua.edu.cn

Chun Yu*
Department of Computer Science and Technology
Tsinghua University
Beijing, China
chunyu@tsinghua.edu.cn

Yingtian Shi
School of Interactive Computing
Georgia Institute of Technology
Atlanta, Georgia, USA
yshi457@gatech.edu

Chongyang Wang
West China Hospital
Sichuan University
Chengdu, China
mvrjustid@gmail.com

Ziqi Liu
Tsinghua University
Beijing, Beijing, China
liuziqi21@mails.tsinghua.edu.cn

Chen Liang
Computational Media and Arts Thrust
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, Guangdong, China
lliangchenc@163.com

Yuanchun Shi
Department of Computer science and Technology
Tsinghua University
Beijing, China
Qinghai University
Xining, China
shiyc@tsinghua.edu.cn

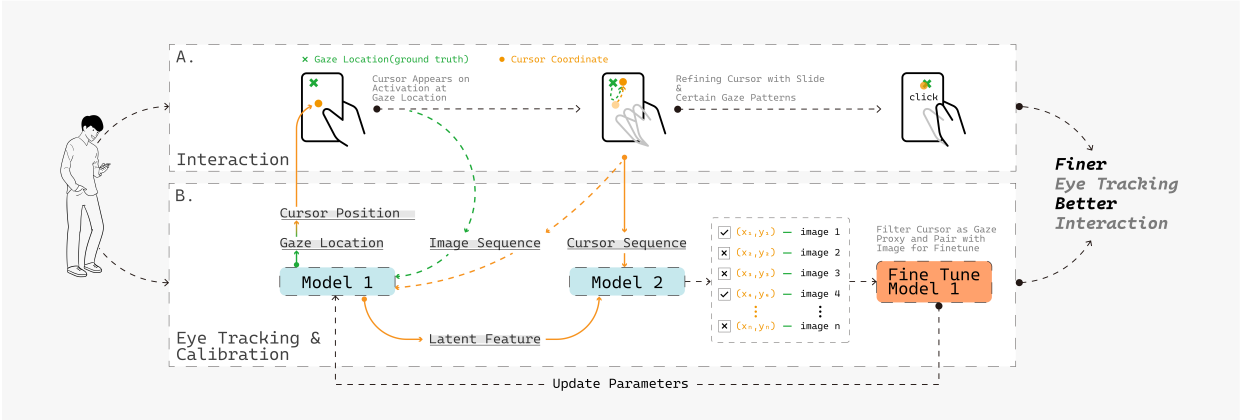


Figure 1: The Workflow of COMETIC (Cursor Operation Mediated Eye-Tracking Implicit Calibration). The system can be divided into the interaction phase and the eye-tracking & calibration phase. (A) The cursor appears at the estimated gaze location when the user activates it. Due to the estimation error, the user refines the cursor position by sliding their thumb. Upon releasing the thumb, a click is executed at the cursor's current position. (B) Model 1 is used for eye-tracking, taking the image sequence as input and outputting the gaze location (cursor position at activation). Model 2 assists in fine-tuning Model 1 by selecting cursor coordinates from the refinement process that can serve as proxies for gaze location. These selected data points are then paired with images and used to fine-tune Model 1. In conclusion, our system leverages data implicitly collected during user interactions to improve eye-tracking accuracy and enhance the interaction experience.

*Corresponding Author

CHI '25, Yokohama, Japan

Abstract

The limited accuracy of eye-tracking on smartphones restricts its use. Existing RGB-camera-based eye-tracking relies on extensive

ACM ISBN 979-8-4007-1394-1/25/04
https://doi.org/10.1145/3706598.3713936

datasets, which could be enhanced by continuous fine-tuning using calibration data implicitly collected from the interaction. In this context, we propose COMETIC (Cursor Operation Mediated Eye-Tracking Implicit Calibration), which introduces a cursor-based interaction and utilizes the inherent correlation between cursor and eye movement. By filtering valid cursor coordinates as proxies for the ground truth of gaze and fine-tuning the eye-tracking model with corresponding images, COMETIC enhances accuracy during the interaction. Both filtering and fine-tuning use pre-trained models and could be facilitated using personalized, dynamically updated data. Results show COMETIC achieves an average eye-tracking error of 278.3 px (1.60 cm, 2.29°), representing a 27.2% improvement compared to that without fine-tuning. We found that filtering cursor points whose actual distance to gaze is 150.0 px (0.86 cm) yields the best eye-tracking results.

CCS Concepts

• **Human-centered computing → Interaction techniques**; **Ubiquitous and mobile computing systems and tools**; **User models**.

Keywords

Eye Tracking, Implicit Calibration, Mobile Devices, Personalization

ACM Reference Format:

Chang Liu, Xiangyang Wang, Chun Yu, Yingtian Shi, Chongyang Wang, Ziqi Liu, Chen Liang, and Yuanchun Shi. 2025. Enhancing Smartphone Eye Tracking with Cursor-Based Interactive Implicit Calibration. In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26–May 01, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 22 pages. https://doi.org/10.1145/3706598.3713936

1 Introduction

Eye tracking has been employed in various fields and demonstrates significant potential [8, 29, 36, 40, 41]. Current eye-tracking on smartphones performs with poor accuracy due to two primary reasons: (1) computer vision-based algorithms lack personalization for individual users[15]; (2) users frequently change their postures while using smartphones, but eye-tracking calibration typically occurs only once during the initial setup[1, 13, 49]. Most solutions address these issues by optimizing computer vision methods, such as increasing dataset size[22, 30] or introducing more complex network architectures[65, 78], which partially mitigates personalization issues but fails to resolve posture changes. Other approaches utilize attention on the screen to optimize eye-tracking results in real-time[68] but struggle with modeling complex eye movements in natural, unconstrained environments[32].

We propose a more effective approach: introducing an intuitive, low-effort interaction method that simplifies eye behavior through coordinated hand-eye actions, from which gaze position proxies can be derived during the interaction. Compared to natural eye behavior in an unconstrained context, studies have demonstrated a strong correlation between eye movement and cursor trace during cursor interactions[42, 43]. Additionally, research shows that integrating a cursor into smartphone interactions can enhance user experience[16], especially for single-handed usage. Therefore, we adopt the cursor as an interactive method to provide information for eye-tracking calibration.

We introduce **COMETIC** (Cursor Operation for Mobile Eye-Tracking Implicit Calibration), a continuous, implicit, and interactive eye-tracking calibration method for smartphones. Our approach integrates a low-effort cursor interaction technique, enabling the system to continuously collect and filter valid cursor coordinates as gaze proxies. Calibration is then achieved by gathering the corresponding image data and fine-tuning the eye-tracking model.

Our cursor interaction is similar to MAGIC Pointing[70]. When the interaction is activated, a cursor is triggered at the current estimated gaze position, and the user refines the cursor's position by sliding. Upon releasing the finger, the system registers a click at the cursor's location. Since the cursor's activation position depends on eye-tracking, the interaction experience improves as eye-tracking accuracy is enhanced.

We conducted a data collection experiment using our cursor interaction method. Analyzing the data reveals a strong correlation between eye movement and cursor motion, manifested in two specific patterns: (1) the distance between the gaze and cursor decreases progressively over time, and (2) the gaze tends to "follow" the movement of the cursor. This provides support for extracting effective coordinates from the cursor trajectory and using them as gaze position proxies for the online fine-tuning of the eye-tracking model.

Our approach involves two deep neural network models. Model 1 takes image sequences as input and outputs gaze position sequences. Model 2 processes the cursor coordinate sequence and generates labels indicating whether each cursor coordinate can be a proxy for gaze position. Since both the cursor and the image contain valuable gaze information, leveraging transfer learning to share knowledge between the two tasks could provide additional advantages [11, 66]. Specifically, Model 2 not only takes the cursor coordinate sequence as input but also incorporates the latent features extracted from the video in Model 1.

In practical use, pre-trained Model 1 and Model 2 are loaded at the start. As the user interacts, the system generates an initial gaze estimation using Model 1 to determine the cursor's starting position. During interactions, cursor coordinates and corresponding image sequences are continuously collected, with Model 2 producing labels for filtering valid data pairs used to fine-tune Model 1. As Model 1's parameters are updated, the latent features of video from Model 1 are also refined, improving the labeling accuracy of Model 2. This iterative adjustment of both models leads to progressively more accurate eye tracking.

Offline evaluation demonstrates that our method achieved an optimal average eye tracking error of 278.3 px (1.60 cm, 2.29°), representing a 27.2% improvement compared to the results without fine-tuning. Filtering cursor points whose actual distance from the gaze at 150.0 px (0.86 cm) yields the best calibration results. Real-time evaluation demonstrates that our method achieved an optimal average eye-tracking error of 446.7 px (2.57 cm, 3.68°), representing a 50.0% improvement compared to the results without fine-tuning. Our primary contributions are: (1) proposing an interaction-integrated eye-tracking calibration method; (2) developing a two-model system that achieves calibration by filtering valid data and fine-tuning; (3) analyzing hand-eye coordination behaviors

during the interaction process to ensure the system's effectiveness; and (4) evaluating the eye-tracking error after calibration.

In the rest of our paper, Section 2 lists related research. Section 3 details the design space of our interaction method. Section 4 describes the data collection experiments. Section 5 presents the statistical analysis of hand-eye behavior in interaction data. Section 6 outlines our algorithmic framework. Section 7 provides an offline evaluation of our method. Section 8 provides a real-time evaluation of our method. Section 9 discusses the limitations of the paper and suggests possible future work. Section 10 provides a conclusion of the entire paper.

2 Related Work

2.1 Eye Tracking on Smartphone

In recent years, eye-tracking technology has attracted researchers' interest especially the technology on the smartphone. With the enhancement of the computational power and sensing ability of the smartphone [32], researchers have designed various kinds of interaction technology with eye-tracking, such as checking the notifications in the top bar of the smartphone using gaze combined with gesture [29], enhancing voice command with gaze on the target object [41], substituting tapping and swiping with gaze-based interactions[36] and so on. All of these downstream applications are based on accurate eye-tracking, for which the researchers have developed plenty of methods to solve this problem which can be classified into two categories. One of them is model-based methods, in which the eye is described as an optical axis and other parameters [4, 23, 58, 59, 63]. The other is learn-based methods, in which the gaze point will be estimated by machine learning methods [9, 54], which is more commonly used in smartphones. Many works utilized RGB [47, 71] or RGB-D [14, 35, 44, 67] images to generate the gaze points. Researchers have extensively explored the feature extraction ability of various models based on CNN like VGG [72, 74], ResNet [26, 71] and so on, which is shown to be effective for the image. To capture information between adjacent frames, models based on RNN were introduced by researchers to extract temporal information [26, 77]. As for the features, face [10, 17, 73] and eye [3, 48] images cropped from face tracking are the most commonly used. To enhance the information input, some works take the facial landmarks as extra features [46, 69]. Moreover, the transformer was utilized by researchers to capture the cross-attention between left and right eyes, which is effective in EM-Gaze [76]. In our study, we also use the ResNet model to extract information from images and use them to generate the gaze point.

2.2 Implicit Calibration

Although learning-based methods can achieve relatively good gaze tracking performance on datasets, calibration remains necessary in practical applications to improve accuracy by "adjusting and customizing the gaze output to reflect the spatial geometry of the camera, the screen, and personal differences" [12, 32]. Calibration is often categorized into two types, one of which is referred to as explicit calibration, where users are required to actively focus on provided gaze points. In this regard, the most common methods are 5-point [23, 47] and 9-point [7, 34]. Some studies also require users to actively track a moving object with their gaze to obtain a sufficient number of calibration points [12, 33, 55–57]. The other type is implicit calibration, where the gaze points are estimated by users' natural interactions. In this context, some researchers take the events of touch [24, 61], mouse and keyboard [21, 25, 37] for calibration, while some other researchers analyze the saliency map [68] or the attention of users' view [2, 28] to find out where the potential gaze points are. In our works, a pilot study was conducted to explore the relationship between the cursor and the gaze points during user interactions. What we found is that the distance between them is close enough to replace the gaze position with the cursor position in some time, which suggests the potential of implicit calibration. In another way, it means that the algorithm can obtain the latest gaze points(substituted by selected cursor positions) to fine-tune the model to achieve continuous calibration and keep accuracy.

2.3 Hand-eye Coordination in Interaction

The application of hand-eye coordination in interaction has been widely studied. In earlier studies, researchers have already discussed the consistency between gaze and cursor behavior in web searching [20] and programming debugging [6]. It is precisely due to this consistency that many gaze-related works use eye movement positions to substitute certain hand operations, such as text selection [50, 51], cursor movement control, swiping, clicking, and returning on mobile screens [36], and so on. These studies utilize eye movement information to replace hand control, thereby achieving better interaction effects. The article by Liebling et al. [38] provides a detailed analysis of the consistency between eye gaze and mouse movements, which points out that in actual PC operations, gaze and mouse exhibit strong consistency for two-thirds of the time, while for the remaining one-third, there is some degree of difference between their behaviors. Furthermore, the study [62] analyzes data on touch and gaze on tablets, suggesting that "on average, 2 fixations occur before and after the tap moment, within a 2-second window centered on the tap moment." In addition, researchers have explored the patterns of hand-eye coordination in different forms of interaction, such as handwriting letters [75], hovering [60], target searching [5], and browsing search results [52]. Recent studies also indicated that in certain interaction modes, mouse and eye movements exhibit consistency [42], or that eye positions can be modeled based on mouse positions using information from the x-axis [43]. In our work, we also observed this hand-eye coordination through a series of statistical analyses. By using the cursor position to replace the specific gaze location, we applied this in our continuous calibration process, yielding promising results.

3 Interaction Design Space

In this section, we introduce the interaction method of COMETIC, the interaction design spaces of this approach, the evaluation of design space, and the specific interaction design choices we made in this study.

Our method focuses on the eye-tracking-assisted cursor interaction for single-handed interaction on smartphone. Therefore, We will begin with a brief justification of our choice. Hakka et al. [16] demonstrated that introducing cursor-based interaction to single hand smartphone usage can reduce the physical effort required for

target selection, making it a more ergonomic option. Eye-tracking enhances this further by reducing the need for manual cursor movement [29]. Technologies adopted in Apple Vision Pro, where gaze can effectively guide and control the cursor, leverage eye movements to minimize the reliance on hand motions. Similar technologies can be applied to foldable smartphones, where users must hold the device with both hands and frequently release one hand for interaction. Eye-tracking assisted cursor-based interaction offers a solution to reduce this repetitive process. Users can keep both hand holding the device and achieve target selection by looking at the target, activate the cursor, and refine the cursor with minimal thumb movement.

3.1 Design Space

To effectively implement this method, we identified three key stages in the design of the interaction: Activation, Gaze-Assisted Positioning, and Refinement. In our approach, the user first looks at the target, activates the cursor, and then refines the cursor's position to precisely reach the target. The stages are structured as follows:

Activation (Mode Switch): The first challenge is ensuring that the introduction of cursor-based target selection does not interfere with the original smartphone interactions. This involves designing a seamless activation process, where users can easily switch between standard touch-based input and the cursor-based system, without disrupting their overall experience.

Gaze Assisted Positioning: Once the cursor is activated, the system either continuously repositions the cursor using eye-tracking data, or keeps the cursor fixed at the initial gaze position where activation occurs. Real-time control offers flexibility, allowing users to refine cursor placement as needed, but this can add complexity and create visual distractions. Alternatively, keeping the cursor fixed at the user's gaze position upon activation would avoid visual interference and simplify the process. However, it forces the user to follow a more rigid workflow, where they must first look at the target, activate the cursor, and then refine its position.

Refinement: Unlike Apple Vision Pro, which utilizes multiple cameras to achieve high eye-tracking precision, most standard devices cannot achieve such accuracy. Therefore, an additional refinement stage is required, where users can refine the cursor's position, to ensure precise target selection. Eye-tracking data will no longer be used to reposition the cursor during this stage.

3.1.1 Activation (Mode Switch). Several studies have explored the interaction design for introducing cursors on smartphones [16]. We categorize activation methods into three types: gesture-based activation, multi-modal activation, and activation via external devices or sensors.

Gesture-based activation means defining a hand movement distinct from common smartphone gestures (e.g., tap, double-tap, long press, swipe). Options include bezel swiping [27, 53], tapping or swiping on the back of the phone [64], or shaking the phone, etc.

Multi-modal activation refers to using non-hand-based modalities such as voice or video. One example is to activate the cursor when the user double blinks, with the action detected by an always-on front camera.

External devices or sensors can also be introduced, adding new sensors to different parts of the phone, or using devices like rings that detect specific interactions (e.g., touch or long press) to trigger activation.

3.1.2 Gaze-Assisted Positioning. Once the cursor is activated, the next step is determining how gaze data is used to assist cursor positioning. Eye-tracking can be integrated into cursor positioning in two primary ways: fixed gaze point positioning and dynamic gaze adjustment.

Fixed Gaze Point Positioning: In this method, the cursor is placed at the estimated gaze point immediately upon activation. In other words, the user's gaze location is treated as the starting position for the cursor. While this method is plain to understand, it assumes that the user is already looking at the target first before activation, which may not always be the case. The refinement of the cursor will start at the fixed position.

Dynamic Gaze Adjustment: This method involves continuously adjusting the cursor's position based on the user's gaze after activation. This approach allows for more flexible interaction, as the user does not have to always look at the target before activation. However, a constantly moving cursor on the screen could be distracting. Furthermore, the system might misinterpret eye movements, leading to potential instability. Dynamic gaze adjustment will be stopped once the user starts to refine the cursor. In practice, for touch input, refinement starts when the thumb touches the screen, and for smartphone movement or head movement, it begins when the movement exceeds a specific threshold.

3.1.3 Refinement. Based on previous work [31], we propose three possible methods for cursor refinement: thumb sliding, smartphone movement, head movement.

Thumb Sliding: The most common refinement method involves the user sliding their thumb across the screen to move the cursor. Users can achieve fine-grained control over the cursor's movement, which is especially useful for tasks that require precision, such as selecting small on-screen elements.

Smartphone Movement: Another refinement technique leverages the phone's internal sensors, such as the IMU or the rear camera's optical flow, to detect motion or orientation changes. Users can refine the cursor position by tilting, rotating, or moving the phone in space. This finger-free method can be particularly useful when touch-based interactions are not feasible. However, excessive phone movement can lead to user fatigue over prolonged periods, and Both IMU and optical flow methods may struggle to achieve the level of precision required for fine-grained cursor adjustments.

Head Movement: When the phone is held in a relatively fixed posture, head movements can be employed to refine the cursor. Small shifts in head orientation translate into cursor movement, offering a hands-free solution similar to phone motion-based refinement. However, as with phone movement, this method also suffers from issues of user fatigue and limited precision.

3.2 Evaluation of Design Space

We conducted a user study to evaluate different combinations of the three key stages mentioned above. In the pilot study, participants reported that the "Dynamic Gaze-Assisted Positioning" is very distracting due to the always-on cursor and the inaccurate eye-tracking. Therefore, we will only evaluate 3 *Activations* ×

3 $Refinements$ = 9 different combinations under the "Static Gaze-Assisted Positioning".

3.2.1 Apparatus. As shown in Figure 2a, participants are instructed to complete the experiment on a Huawei P40 smartphone with their right hands. The resolution of the smartphone screen is 1200 × 2486, and the size is 6.9 cm × 14.3 cm. We used a touch sensor connected to an Arduino Uno to achieve activation.

A Tobii Eye Tracker 5 is employed to capture the gaze coordinates on a 29-inch screen (3840 px × 2560 px, 65 cm × 36.5 cm), which are then converted to the coordinates on smartphone screen. Specifically, the eye tracker is first mounted on a custom 3D-printed stand to ensure its stability in relation to the smartphone. Next, the stand and the eye tracker are fixed at the bottom of the computer screen during calibration (Figure 3a). The orange box indicates the smartphone body, while the blue box indicates the smartphone screen. The position and size of the smartphone screen relative to the computer screen are recorded (top: 1036 px, left: 1684 px, height: 936 px, width: 450 px, Figure 3b). During experiments, participants need to hold the stand. The coordinates on the smartphone use the top-left corner as the origin, with the positive x-axis extending rightward and the positive y-axis extending downward. To translate x coordinate from the computer screen to the smartphone screen, we subtract it with the left margin (1684 px), divide it by the width on the computer screen (450 px), and multiply it with the width of smartphone screen (1200 px). To translate y coordinate from the computer screen to the smartphone screen, we subtract it with the top margin (1036 px), divide it by the height on the computer screen (936 px), and multiply it with the height of smartphone screen (2486 px). Refer to Equation 1 and 2 for details.

$$x_{smartphone} = \frac{x\_screen - left\_margin_{on\_screen}}{width_{on\_screen}} \times width_{smartphone} \tag{1}$$

$$y_{smartphone} = \frac{y\_screen - top\_margin\_screen}{height_{on\_screen}} \times height_{smartphone} \tag{2}$$

3.2.2 Participants and Procedure. We recruited 9 participants (4 Males, 5 Females, aged 19-28, M=22) from the university. The entire experiment lasts about an hour. The participants were compensated at a rate of $15 USD per hour.

The experiment evaluates three types of activation: gesture (double tap, abbreviated as "tap"), multi-modal (double blink, abbreviated as "blink"), and external device (touch sensor, abbreviated as "device"); as well as three types of refinement: thumb sliding (abbreviated as "thumb"), smartphone movement (achieved using IMU, abbreviated as "imu"), and head movement (abbreviated as "head"). This results in nine different combinations. The experimental order was counterbalanced using a Latin square to ensure that each combination appeared once in every sequence.

At the start of the experiment, participants were instructed to finish the inherent calibration of Tobii Eye Tracker. Next, they iterated through nine combinations, selecting ten 200 px × 200 px targets that were randomly displayed on the smartphone screen for each combination (Figure 2b). A touch sensor connected to an Arduino Uno is used for activation. When the activation type is "device," the touch sensor is placed on the back of the smartphone, within easy reach of the participant's index finger. For "tap" and "blink" activations, the process is implemented via a Wizard of Oz approach, where the experimenter observes the participant's actions and manually controls the touch sensor to trigger the activation. After activation, the cursor was positioned at the gaze coordinate on the smartphone screen, estimated and converted from the eye tracker. Participants need to move the cursor into the target using the corresponding refinement method. If the activation method is "tap" or "blink", participants need to re-conduct the activation to finish selection. If the activation method is "touch", participants can either lift their thumb from the screen, or lift their index finger from the touch sensor, to finish selection. At the end of each iteration, participants need to rate the six dimensions from NASA-LTX (mental demand, physical demand, efficiency, performance, effort, frustration), as well as the learning cost and the covertness of the combination.

3.2.3 Design Space Evaluation Results. Figure 4 and Table 1 shows the subjective ratings of different combinations across various dimensions. We reordered the dimensions so that indicators with lower values being better performance (mental demand, physical demand, effort, frustration, learning cost) are on the left, while those with higher values being better performance (efficiency, performance, covertness) are on the right. The "rank" in Table 1 is first calculated based on each dimension's value-performance relationship, then averaged across the overall rank.

Among different combinations, "blink, touch", "tap, touch", "device touch" are ranked at top three. The Kruskal-Wallis H test shows no significant differences between these three combinations across all eight dimensions. Table 2 shows the number of combinations significantly outperformed by each of them. Notably, "device, touch" outperforms others across most dimensions, followed by "tap, touch," while 'blink, touch" significantly outperforms only a few combinations

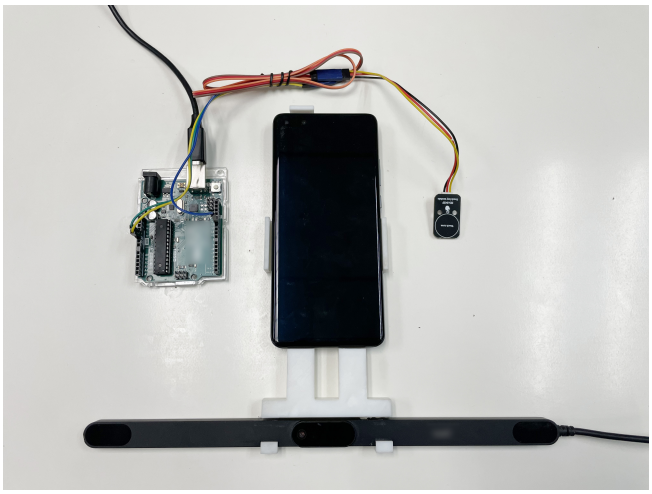
3.3 Implementation in This Study

Based on the evaluation result in Section 3.2.3, we use touch sensor (device) for activation, fixed gaze point positioning for gaze-assisted positioning (static), and thumb sliding (touch) for cursor refinement.

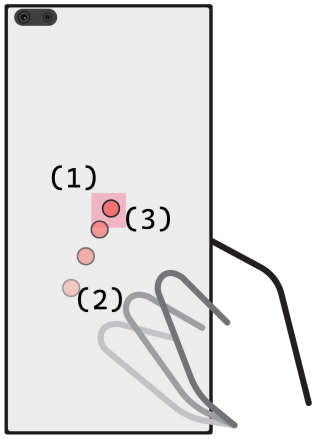
The interaction process is as follows: Users touch the sensor on the back to activate the cursor at the gaze estimation location. Then they slide to move the cursor. A click is registered when they lift their finger.

4 Data Collection Experiment

In this section, we will collect the data necessary for eye-tracking and calibration. Specifically, we will collect cursor traces on the smartphone, video from the front camera, and gaze positions on the smartphone using a Tobii eye tracker. There are three main reasons for collecting this data. First, we aim to understand the correlation between eye movement and cursor traces, thereby demonstrating the potential of cursor-based interactive calibration for eye-tracking. Second, we aim to design and implement algorithms for eye-tracking and calibration using this data. Finally, we need to evaluate the effectiveness of our algorithms for this dataset.

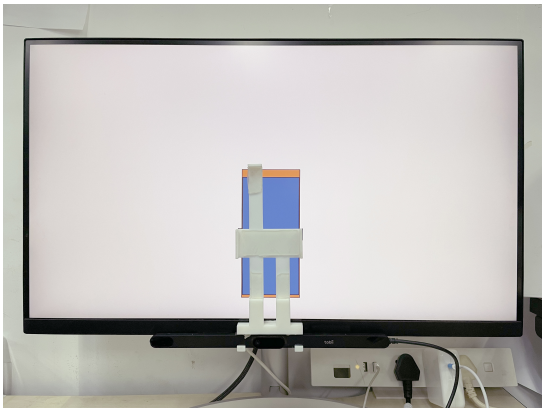


(a) Smartphone, Eye Tracker in 3D-Printed Stand, and Touch Sensor with Arduino

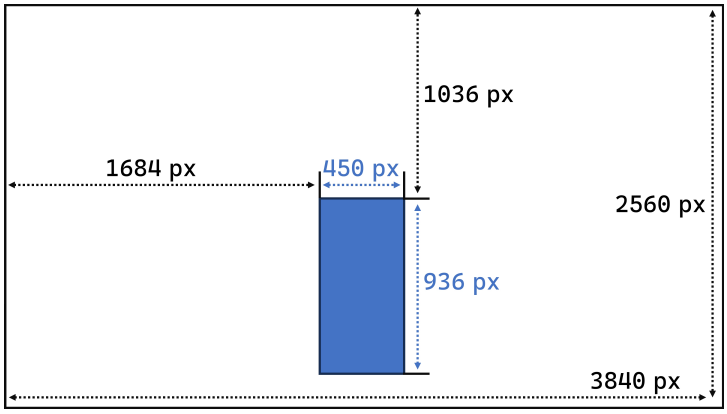


(b) Target Selection Task (Use "touch" for Refinement as An Example)

Figure 2: The Apparatus and Experimental Task. The Target Selection Procedure in Figure 2b: (1) the target is randomly displayed on the screen; (2) the participant activated and refined the cursor (we use touch as an example); (3) the target is selected at deactivation (lift of thumb) as the cursor is positioned inside the target.



(a) Eye tracker and 3D-printed stand.



(b) Layout on screen.

Figure 3: The eye tracker and 3D-printed stand is attached to the screen during the calibration of eye tracker. The boxes in the screen indicate the position of the smartphone relative to the screen.

4.1 Apparatus

The apparatus used in this section is almost identical to that in Section 3.2.1's design space experiment. There are two differences: (1) the 3D-printed stand is placed on a metal stand to minimize any potential phone movement caused by swiping (Figure 5a); (2) we do not use touch sensor for activation in this experiment.

4.2 Participants and Procedure

A total of 24 participants (7 females, aged 20-30, M = 24.0) were recruited from the university campus for this experiment. None of the participants had previously participated in any related pilot studies. During the experiment, participants sat comfortably in front of the smartphone and its stand. The entire experiment lasted approximately 30 minutes. Participants were compensated at a rate of $15 USD per hour. The experiment was divided into two phases:

Manual Calibration Phase: In this phase, a 3×3 grid of points was sequentially displayed on the smartphone screen from top to bottom and left to right (see Figure 5b(1)). Each point remained on screen for 2 seconds, during which participants were instructed to focus on the point without blinking unless switching between points. The purpose of this phase was to collect the position of points and their corresponding eye-tracking data for calibration. In the pilot study, we attempted to fix the smartphone and its 3D-printed stand in front of the screen for calibration using the Tobii eye tracker's built-in calibration system, but the results were

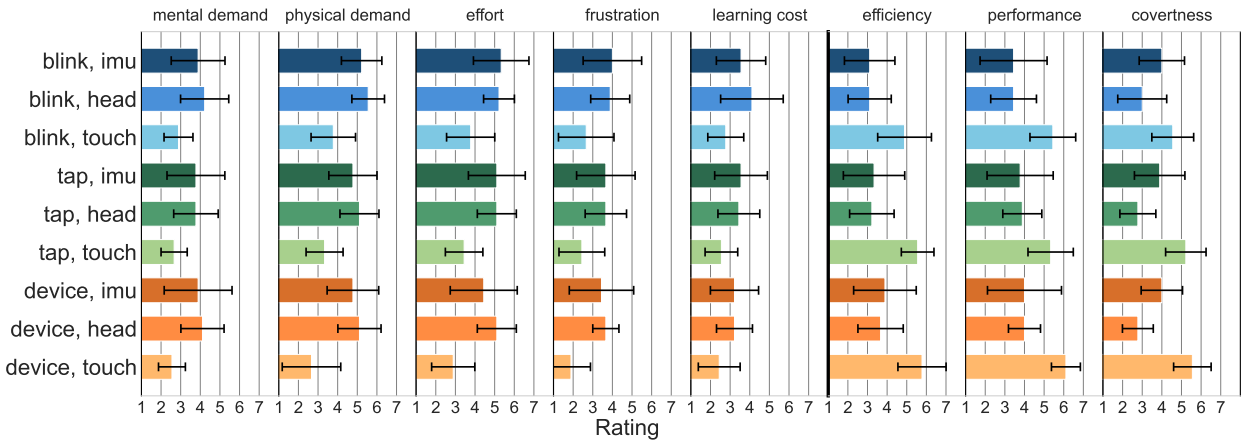


Figure 4: The Subjective Ratings for Different Interaction Combinations

combination	mental demand	physical demand	effort	frustration	learning cost	efficiency	performance	covertness	rank
blink, imu	3.9 ± 1.4	5.2 ± 1.0	5.3 ± 1.4	4.0 ± 1.5	3.6 ± 1.3	3.1 ± 1.3	3.4 ± 1.7	4.0 ± 1.2	7.8
blink, head	4.2 ± 1.2	5.6 ± 0.8	5.2 ± 0.8	3.9 ± 1.0	4.1 ± 1.6	3.1 ± 1.1	3.4 ± 1.2	3.0 ± 1.2	8.2
blink, touch	2.9 ± 0.7	3.8 ± 1.1	3.8 ± 1.2	2.7 ± 1.4	2.8 ± 0.9	4.9 ± 1.4	5.4 ± 1.2	4.6 ± 1.1	2.9
tap, imu	3.8 ± 1.5	4.8 ± 1.2	5.1 ± 1.4	3.7 ± 1.5	3.6 ± 1.3	3.3 ± 1.6	3.8 ± 1.7	3.9 ± 1.3	5.9
tap, head	3.8 ± 1.1	5.1 ± 1.0	5.1 ± 1.0	3.7 ± 1.1	3.4 ± 1.1	3.2 ± 1.1	3.9 ± 1.0	2.8 ± 0.9	5.9
tap, touch	2.7 ± 0.7	3.3 ± 0.9	3.4 ± 1.0	2.4 ± 1.2	2.6 ± 0.8	5.6 ± 0.8	5.3 ± 1.2	5.2 ± 1.0	2.1
device, imu	3.9 ± 1.7	4.8 ± 1.3	4.4 ± 1.7	3.4 ± 1.6	3.2 ± 1.2	3.9 ± 1.6	4.0 ± 1.9	4.0 ± 1.1	4.9
device, head	4.1 ± 1.1	5.1 ± 1.1	5.1 ± 1.0	3.7 ± 0.7	3.2 ± 0.9	3.7 ± 1.2	4.0 ± 0.8	2.8 ± 0.8	6.4
device, touch	2.6 ± 0.7	2.7 ± 1.5	2.9 ± 1.1	1.9 ± 1.0	2.4 ± 1.1	5.8 ± 1.2	6.1 ± 0.7	5.6 ± 1.0	1.0

Table 1: Mean, Standard Deviation, and Rank of Subjective Ratings for Different Combinations

combination	mental demand	physical demand	effort	frustration	learning cost	efficiency	performance	covertness
device, touch	5	6	6	6	6	6	0	6
tap, touch	2	6	6	4	5	2	0	3
blink, touch	1	3	4	5	2	1	0	3

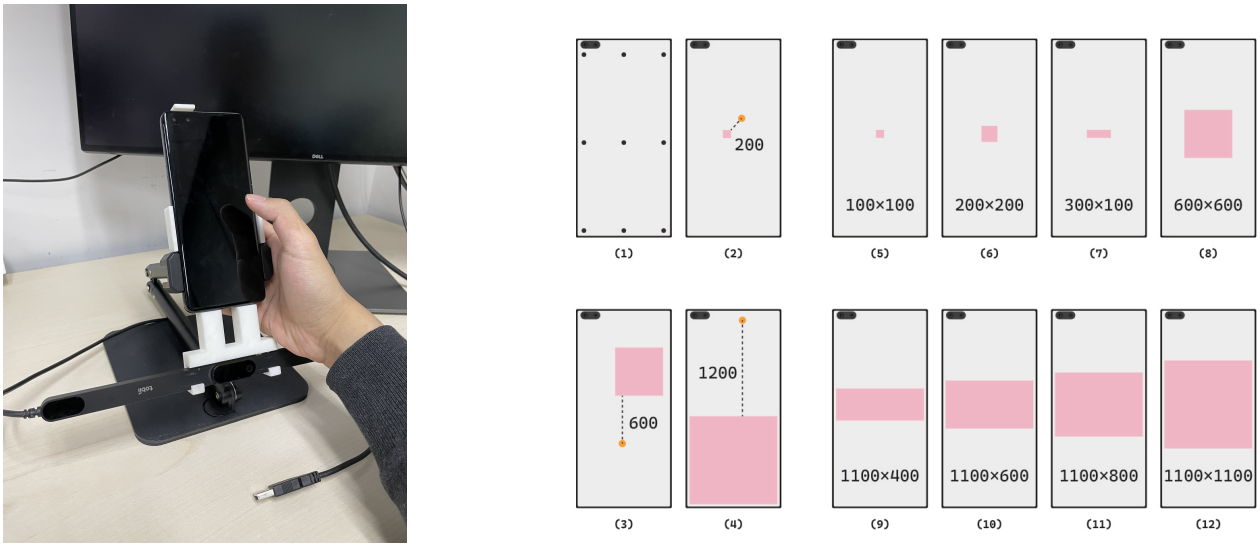
Table 2: Number of Combinations Significantly Outperformed by the Given Combination in Each Dimension

target size (px)	100 × 100	200 × 200	300 × 100	600 × 600
random location number	16	16	16	8
corners and edges	top left, top middle, top right, bottom left, bottom middle, bottom right			

target size (px)	1100 × 400	1100 × 600	1100 × 800	1100 × 1100
random location number	8	8	8	8
corners and edges	top, bottom			

Table 3: Target Configuration during Experiment

unsatisfactory. Therefore, we manually collected calibration data during the experiment and later performed a nine-point calibration on the collected data.



(a) Apparatus for Experiment

(b) Layouts in Experiment.

Figure 5: Apparatus and Layouts in Experiment. For subfigures in Figure5b: (1) shows the layout during calibration. (2)-(4) show the distance between the cursor and target in three different distances. (5)-(12) show eight targets of different sizes.

Cursor-Based Target Selection Phase: In this phase, both a target and a cursor appeared on the smartphone screen. Participants moved the cursor by sliding their thumb on the screen. The task was considered successful if the cursor was moved inside the target when the participant lifted their thumb. If the cursor was not placed inside the target, it would reset to its starting position. The cursor movement followed a variable control-display (CD) ratio based on the sliding speed[45], where faster movements translated to greater cursor displacement.

As shown in Table 3, the targets have 8 different sizes (Figure5b(5)-(12) show their sizes relative to the screen) and appeared at random locations on the screen for multiple times. In addition, the target appeared at specific positions along the corners and edges of the screen at least once, yielding a total of 120 unique combinations of target positions and sizes. Positions along the corners and edges of the screen are further divided into two categories, since for targets with the width of 1100 px, there will not be much difference between left and right, given the screen width is 1200 px.

The cursor was circular, with a diameter of 90 pixels. The distance from the cursor to the target was categorized into three levels: 200, 600, and 1200 pixels, simulating varying levels of eye-tracking estimation accuracy (Figure5b(2)-(4) show three types of distances). The distance is measured from the center of the cursor to the nearest edge or corner of the target. The relative position between the cursor and the target was randomly generated. Since the goal of the experiment is to understand and model the relationship between eye movement and cursor behavior, we did not use the activation method mentioned in section 3.3 but randomly generated the cursor positions to increase data diversity. This results in a total of $120 target-size combinations \times 3 distance levels = 360 distinct trials$.

We will collect the following data during the experiment: the size and position of the target, the sequence of cursor positions, the sequence of eye tracking data (gaze positions), and the video from the front camera.

4.3 Data Pre-processing

4.3.1 Manual Calibration of Eye Tracking Data. As mentioned before, during the Manual Calibration Phase, we collected sequences of gaze coordinates provided by the Tobii eye tracker 5 when participants fixated on nine specific calibration points on the screen. We mapped each gaze coordinate to the corresponding calibration point and used the RANSAC (Random Sample Consensus) method to compute a homography matrix that minimizes the distance between gaze-calibration pairs. This matrix was then applied to move the gaze coordinates during the Cursor-Based Target Selection Phase. In the rest of the paper, we will use the transformed coordinates as the ground truth for gaze on the smartphone screen.

4.3.2 Data Segmentation and Alignment. During the Cursor-Based Target Selection Phase, participants complete 360 trials. Both front camera and cursor data are recorded on the smartphone and segmented by trial, while the eye tracker continuously captures data. Signals sent from the smartphone to the computer at the start and end of each trial are used to segment the eye-tracking data. Since video, cursor, and eye-tracking data are collected on different devices with varying sampling rates, we resample the eye-tracking and cursor data to align them with the video timestamps for synchronization.

4.3.3 Image Cropping. As previously mentioned, we used the front camera of the smartphone to record videos during the experiment. These videos typically capture the participant's head and part of the upper body. To facilitate subsequent neural network training, we crop the original images into three parts. Using Mediapipe [39], we detect the participant's face, left eye, and right eye in the video

frames, and crop the images accordingly to generate three new video segments. Additionally, we record the coordinates and size of these three cropped sections relative to the original frame as a video info sequence.

5 Statistical Analysis of Gaze and Cursor Behavior

In this section, we will analyze the distance between the gaze positions and the cursor coordinates. Figure 6 presents a waterfall chart of the gaze-to-cursor distance across all participants. We can see that approximately 50% of the data points have a distance smaller than 200 px, and 84% have a distance smaller than 600 px. As mentioned in section 4.2, only in one-third of the trials, the initial cursor-to-target distance is 200 px. Therefore, we believe there is a clear correlation between the movement of the user's gaze and the cursor.

We observed two primary patterns in the relationship between the user's gaze and the cursor: (1) The distance between the gaze and the cursor gradually decreases over time. We attribute this to the fact that in certain cases, the user consistently focuses on the target without looking at the cursor, so the distance decreases as the cursor moves closer to the target. (2) The gaze-to-cursor distance remains stable for a period. This is because, in some cases, the user's gaze follows the cursor's movement.

In the following sections, we will discuss these two patterns in detail.

5.1 Gaze-Cursor Distance Convergence

Figure 7 shows the mean distance between the gaze and cursor over time, with time normalized. It is evident that the mean distance between them gradually decreases as time progresses. At about 25% of the time, the distance between gaze and cursor is about 400 px. At about 70% of the time, the mean distance is about 200 px. This suggests that as the user moves the cursor closer to the target, the gaze position also converges toward the cursor. We believe that in the final stages of cursor control, gaze position can, to some extent, be substituted by cursor position. In real use cases, this provides a certain amount of gaze ground truth, enabling us to gather the data needed for subsequent fine-tuning of the model. In other words, this cursor behavior at the end of control sequences offers a possible reference point for gaze position.

5.2 Gaze Following Cursor

We defined gaze-following cursor behavior using three features: (1) the distance between the gaze and cursor is below a threshold $t_{distance}$, (2) the number of consecutive points where the distance remains below this threshold exceeds a threshold t_{num}, and (3) the angle between the movement directions of the gaze and cursor is below a threshold t_{angle}. With the last threshold t_{angle} fixed to 90°, the first two thresholds $t_{distance}$ and t_{num} were sampled within a certain range at defined intervals. This allowed us to create a heatmap (Figure 8) showing the percentage of data points that meet the behavior definition. X axis shows the value of t_{num} and y axis shows the value of $t_{distance}$. Since our tobii eye tracker mentioned in section 4 is sampled at 30 Hz, multiply t_{num} by 0.03 gives the duration in seconds.

From Figure 8, we observe that when the conditions for defining gaze-following cursor behavior are more relaxed (i.e., $t_{num} \leq 8$ and $400 \leq t_{distance} \leq 500$), over 30% of the data meets these criteria. And when the conditions become stricter(i.e., $t_{num} \leq 12$ and $200 \leq t_{distance} \leq 400$), approximately 10% to 30% of the data still satisfies the requirements.

These results indicate that during the user's control of the cursor, there are chains of consecutive cursor points that not only maintain close proximity to the gaze but also align with the gaze's movement direction. These points provide another possible reference for gaze position.

6 COMETIC Algorithm

In this section, we will discuss the COMETIC algorithm. The core problem COMETIC aims to solve is how to fine-tune a video-to-gaze model during its use, specifically, how to obtain the true gaze positions required for fine-tuning without the eye tracker. Our approach proposes using certain cursor coordinates from the cursor-based interaction process as proxies for gaze positions (i.e., those with a high probability of having a distance from the gaze position below a preset threshold, τ).

The algorithm involves two models with three objectives:

(1) Given a video (image sequence) input, Model 1 generates the corresponding gaze position sequence for the user.
(2) Given a cursor sequence input, Model 2 labels whether each cursor position can be used as a proxy for the gaze position.
(3) With pre-trained Model 1 and Model 2, we aim to leverage new user data to optimize both models simultaneously.

6.1 Model 1: Video to Gaze

Model 1 takes as input both a video (image sequence) and the corresponding positional and size information of the regions of interest (referred to as video information), and outputs the gaze position on the screen. Figure 9 gives the structure of Model 1. As mentioned in Section 4.3.3, the front camera video is cropped into three images: left eye, right eye, and head. These images are first processed through a ResNet-18 [18] followed by two fully connected layers. The features extracted from these images are then concatenated with the video information, resulting in latent features. These latent features are passed through five fully connected layers to produce the final gaze coordinates, which are normalized between 0 and 1 based on the screen's width and height.

6.2 Model 2: Cursor Labeling

Model 2 receives the cursor coordinate sequence and the latent features from Model 1 as inputs, and outputs labels indicating whether each cursor coordinate can serve as a proxy for the gaze position. The label is determined by computing the distance between each cursor coordinate and the corresponding gaze position mentioned in Section 4.3.2. If the distance exceeds the preset threshold τ, the label is False, indicating the cursor coordinate can not serve as a proxy; otherwise, the label is True, indicating the cursor coordinate can serve as a proxy. Figure 9 gives the structure of Model 2. The cursor coordinate sequence first passes through one fully connected layer, followed by an LSTM [19] and another fully connected layer, and is then concatenated with the corresponding latent features

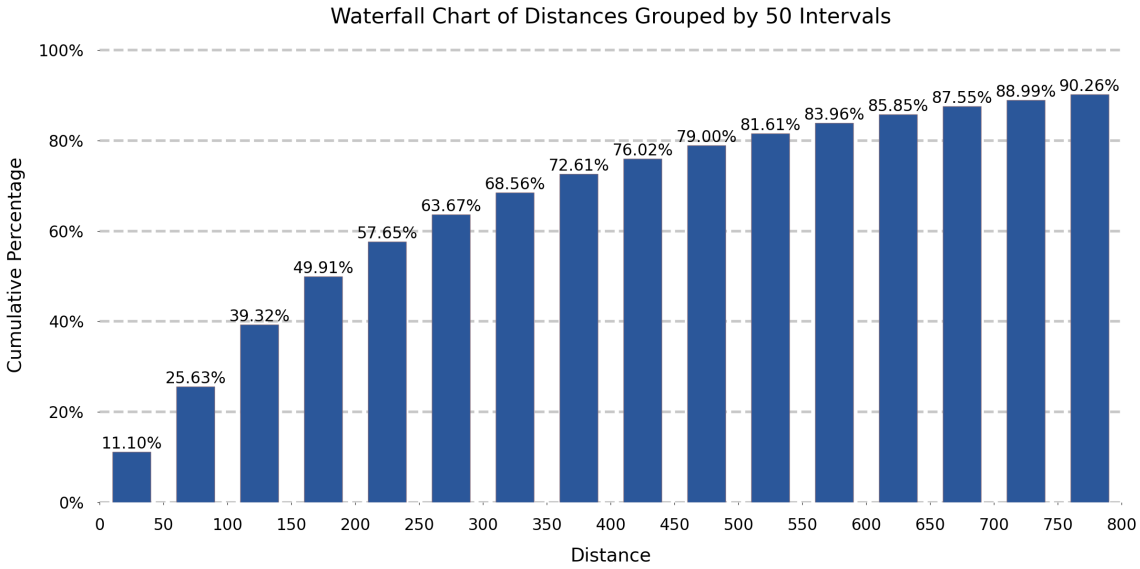


Figure 6: Waterfall chart of distances grouped by 50 intervals

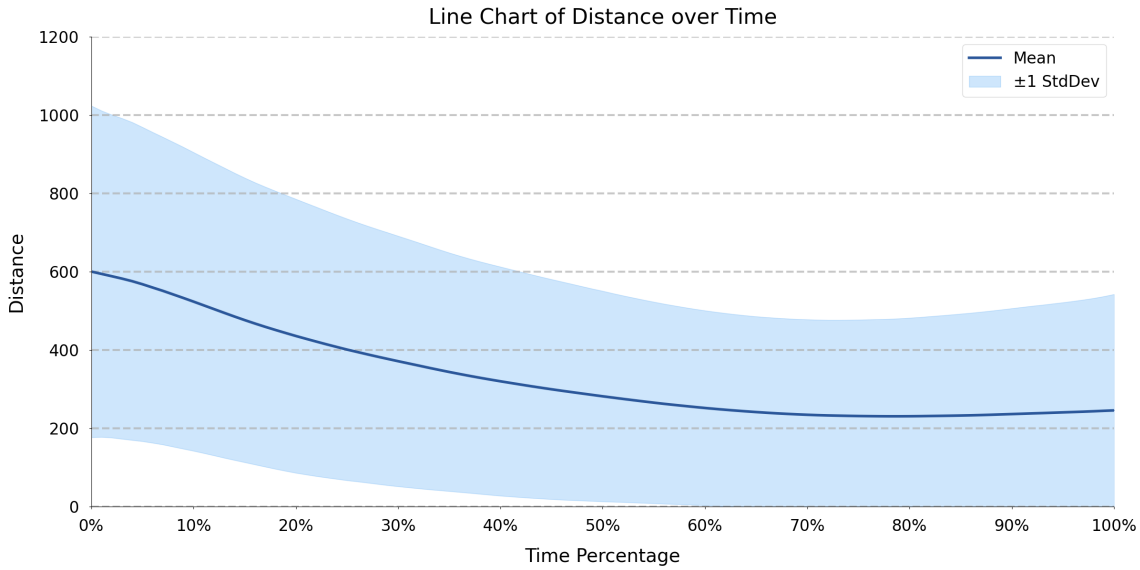


Figure 7: Line chart of distance over time

from the video sequence. Afterward, the data is passed through five more fully connected layers to produce the final labels.

6.3 Online Fine-Tuning with New User Data

As previously mentioned, the primary goal of this work is to allow the system to quickly fine-tune both Model 1 and Model 2 when encountering new users, thereby improving the eye tracking. Specifically, after collecting video and cursor coordinate sequences from a new user during cursor interaction, Model 2 is used to filter the cursor coordinates, selecting those likely to serve as accurate proxies for the gaze position.

Once the filtered cursor coordinates are obtained, they are paired with corresponding images to fine-tune Model 1. Similar to standard transfer learning, a smaller learning rate and fewer epochs are adopted. However, unlike the typical approach, we fine-tune the entire model instead of only the final layers.

After fine-tuning Model 1, its internal parameters are adjusted to better extract features for the new user. At this point, we repeat the filtering process with Model 2, using new latent features from Model 1, selecting new cursor coordinates, and continuing the fine-tuning of Model 1. This iterative fine-tune process further refines the model's ability to achieve better eye tracking for new users.

Figure 8: Heatmap of number threshold and different distance threshold

7 Offline Evaluation with Dataset

In this section, we will evaluate our approach using the previously collected dataset, focusing on three main scenarios:

(1) We will first assess the gaze error of Model 1. It takes video as input and outputs gaze positions. We assess the gaze error when using cursor coordinates as a proxy for gaze positions in training, under the condition that the distance between the cursor and gaze is below different thresholds τ.
(2) Next we will assess Model 2, which takes the latent features of video and cursor sequence as input and outputs labels determining whether cursor coordinates can serve as a proxy for gaze positions. We will evaluate its true positive rate under different τ values. Additionally, we will analyze the difference between the gaze-cursor distance and the τ value of false positive cases (abbreviated as "FP difference").
(3) We will monitor the performance changes in both Model 1 and Model 2 during the iterative fine-tune process, including the gaze error of Model 1, as well as the true positive rate and FP difference of Model 2.

In our evaluation, we applied a leave-one-out training approach, repeating the training process 24 times. For the first two scenarios, the left-out participant served as the validation set, while the remaining data was used for training. For the iterative fine-tune scenario, the data from participants who were not left out was used to pre-train Model 1 and Model 2. The first 20% of the left-out participant's data (equivalent to 72 target selection trials, which takes approximately 3-4 minutes) was used as the training set for iterative fine-tuning, while the remaining 80% was served for validation.

The distance mentioned in this section is initially measured in pixels. We will convert it to centimeters based on the smartphone size and resolution (Section 3.2.1), and then convert it to angular units assuming a 40cm distance between the participant and the smartphone.

7.1 Gaze Error of Model 1

Figure 10 shows the box plot of gaze error of Model 1 with threshold τ values ranging from 0 to 1000.0 px (5.75 cm). The gaze error is defined as the Euclidean distance between the actual gaze position and the estimated gaze position provided by Model 1. Ideally, the gaze error should be minimized for optimal performance. A τ value of 0 indicates that no cursor coordinates were used to replace the gaze positions, while positive τ values represent the use of cursor

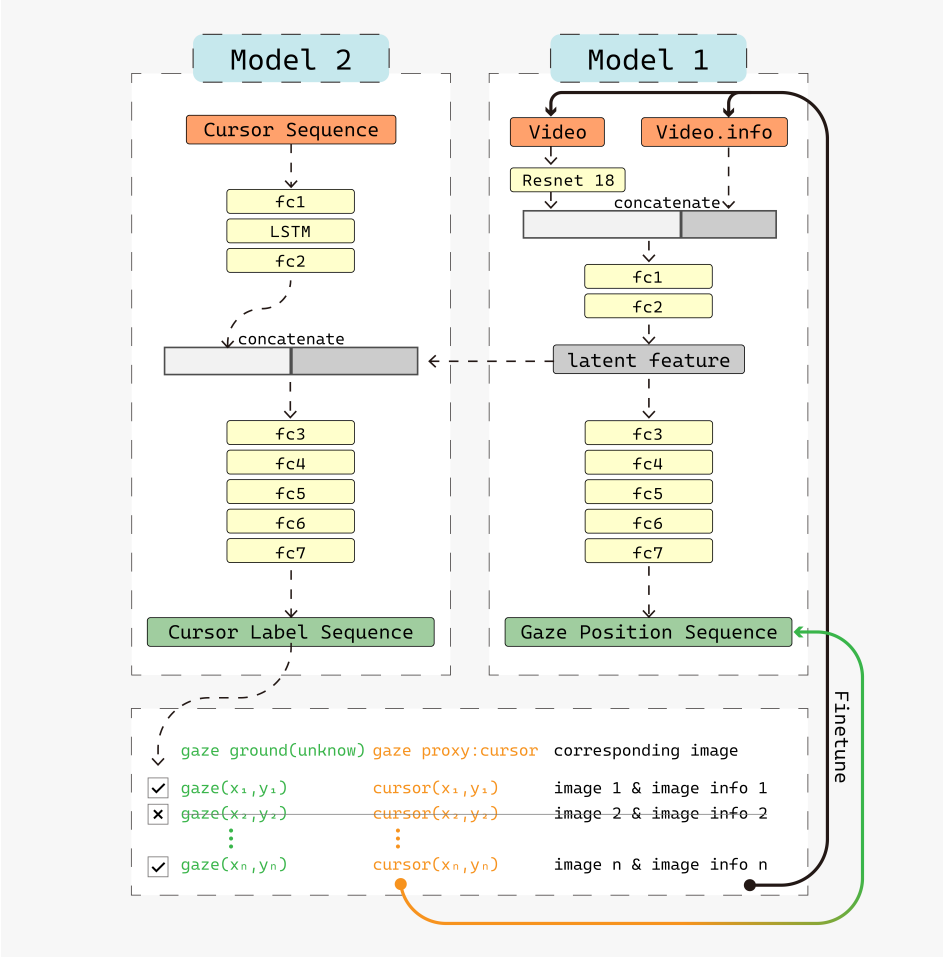


Figure 9: Structure of COMETIC Models. Model 1 takes video and video info as input. Video first goes into resnet and then be concatenated with video info. After passing through two fully connect layers, we get latent features. Latent features will then pass through five fully connected layers to get the gaze position sequences. Model 2 takes the cursor sequence as input. After passing throught fully connected layer 1, LSTM, and fully connected layer 2, it will be concatenated with latent features from Model 1. The result will then pass through five fully connected layers and output cursor label sequence. In the bottom of this figure, it shows that we use the output of Model 2 to filter valid cursor coordinates, pair it with image and image info, and finetune the Model 1.

coordinates in place of gaze positions when the distance between them is less than τ.

RM-ANOVA revealed that different τ values significantly affected the gaze error ($F_{8.37,192.45} = 38.20, p < 0.01$). Table 4 shows the post-hoc Bonferroni analysis under different τ values. The gaze errors for the τ combinations below are significantly higher than those for the τ above. It reveals τ values of 700.0 px (4.03 cm) is a threshold, where gaze error below this tau value is typically significantly lower than gaze error above it.

7.2 True Positive Rate and FP Difference of Model 2

Model 2 predicts whether the Euclidean distance between the cursor coordinate and the gaze position is less than the threshold τ. A higher true positive rate means a larger proportion of actual positives are correctly identified by the model. Figure 11 shows the true positive rate of Model 2 for five participants across τ values ranging from 100.0 px (0.58 cm) to 1000.0 px (5.75 cm). True positive rate starts around 0.3 and increases to approximately 0.9 at τ = 750.0 px(4.31 cm), eventually stabilizing around 0.95 at τ = 1000.0 px(5.75 cm).

Figure 12 illustrates the FP difference, with τ ranging from 100.0 px (0.58 cm) to 1000.0 px (5.75 cm). A smaller difference in false positives means that when Model 2 misclassifies an object, the gaze-cursor distance of the misclassified object is closer to the corresponding τ value. A smaller difference is preferable. However, ANOVA results indicate no significant variation across different τ

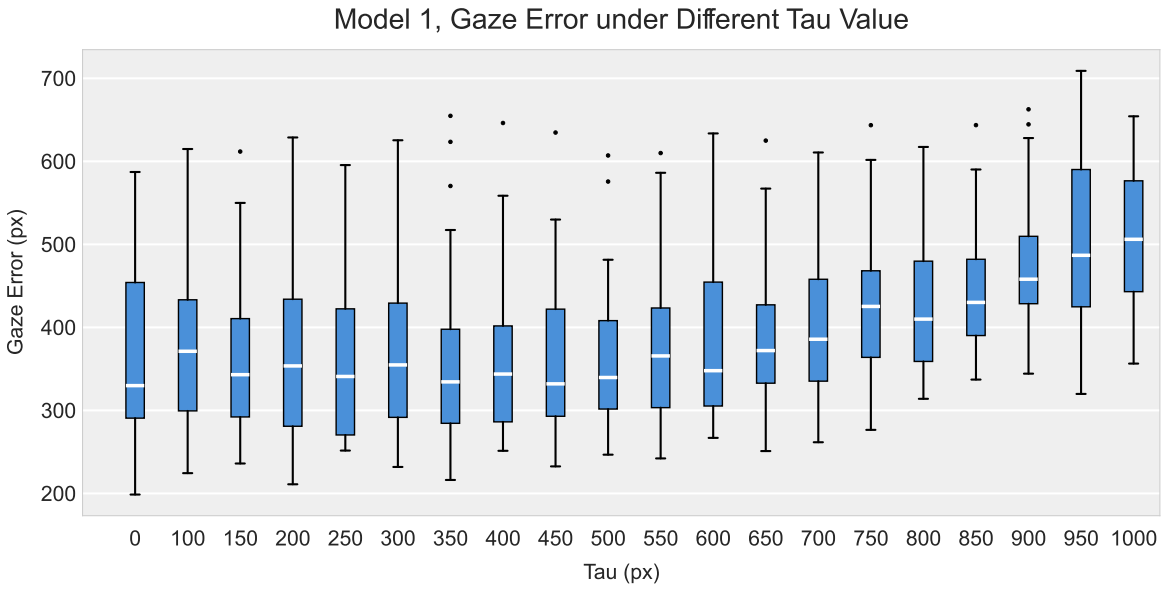


Figure 10: Gaze Error of Model 1 Given Different τ values

τ	0	100	150	200	250	300	350	400	450	500
τ with significantly greater gaze error	550 to 1000	700 to 1000	600, 700 to 1000	700 to 1000	600 to 1000	700 to 1000	700 to 1000	700 to 1000	600 to 1000	600 to 1000

τ	550	600	650	700	750	800	850	900	950	1000
τ with significantly greater gaze error	700 to 1000	750 to 1000	700 to 1000	750 to 1000	900 to 1000	900 to 1000	900 to 1000	950 to 1000	/	/

Table 4: τ Values with and Corresponding τ Ranges with Significantly Greater Gaze Error.

values, with the average distance remaining consistently around 183.0 px (1.05 cm).

7.3 Performance During Iterative Fine-tune Process

Table 5 presents the mean, standard deviation (std) of the minimal gaze error results across iterations during iterative fine-tuning for different τ values, as well as the improvement compared to the model without fine-tuning. The best calibration performance and largest improvement were observed at $\tau = 150.0$ px(0.86 cm), where the gaze error reached 278.3 px (1.60 cm, 2.29°), representing a 27.2% improvement over the result without fine-tune.

Figure 13, 14, and 15 shows the gaze error of Model 1, true positive rate of Model 2, and FP difference of Model 2, given different τ values and iterations. Blue and green markers indicate the values of participants for different iterations (iter 1 to 5), while orange markers represent the results from the pre-trained model without fine-tuning (iter 0). As noted earlier, there are five rounds of iterative fine-tuning, the markers with higher brightness indicating a smaller iteration index and lower brightness indicating a larger iteration index.

7.3.1 Performance of Model 1 in Iterative Fine-tuning. As shown in Figure 13, the gaze error without fine-tuning is significantly larger than those with fine-tuning. Additionally, after iteration 1, the gaze error increases with the τ value and the iteration. RM-ANOVA result indicates a significant effect from $\tau * iteration$ ($F_{10.36,238.37} = 23.92, p < 0.001$). Post-hoc LSD analysis supports the conclusion above.

Table 6 shows the significance of gaze error between current and previous iteration under different τ values. A "-" indicates the current iteration has significantly lower gaze error, "+" indicates significantly higher, and "/" indicates no significant difference. It is evident that the gaze error of iteration 0 (without fine-tuning) is significantly higher than those of iteration from 2 to 5. Furthermore, fine-tuning no longer yields significant improvements after the fourth iteration, and when τ exceeds 300.0 px (1.73 cm), fine-tuning leads to worse performance. This may be because higher τ

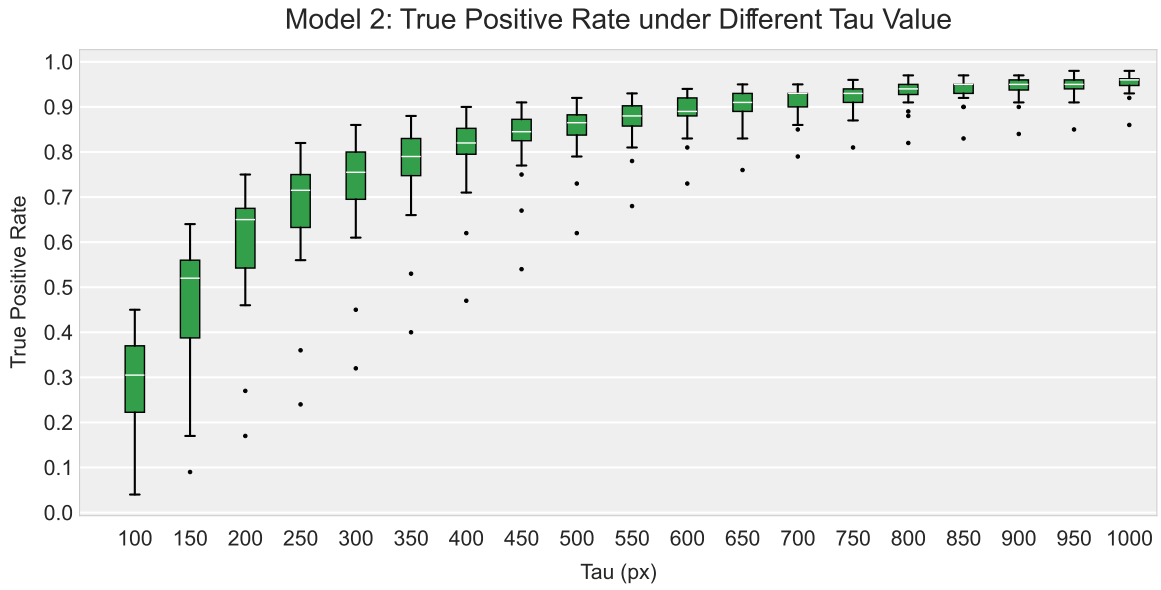


Figure 11: True Positive Rate of Model 2 Given Different τ Values

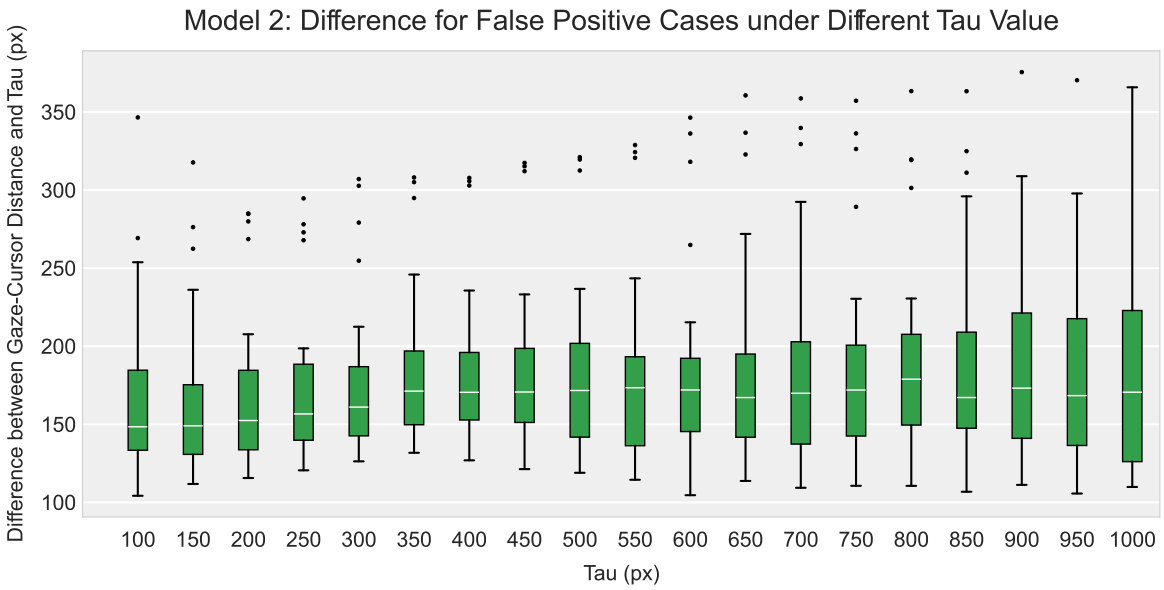


Figure 12: FP Difference of Model 2 Given Different τ Values

values inherently mean the information used for fine-tuning is less accurate.

For iteration 0 (without fine-tuning) and iteration 1, there are no significant differences in gaze error across different τ values. Table 7 shows the top three τ values with the smallest gaze error, and their corresponding τ ranges with significantly greater gaze error for iterations from 2 to 5. For each iteration, there is a threshold-like τ value, above which the gaze error is significantly greater than for the top three τ values with the smallest gaze error. Furthermore, as the iterations increase, this threshold value continues to decrease.

In summary, we suggest to use a τ value smaller than 300.0 px (1.73 cm) in practical applications.

7.3.2 Performance of Model 2 in Iterative Fine-tuning. As shown in Figure 14, the relationship between true positive rate and τ value is consistent with that in Figure 11. RM-ANOVA result shows that neither the interaction effect $\tau * iter$ or the effect of iter is significant,

τ value	0	100	150	200	250	300	350	400	450	500
Mean (px)	413.1	279.1	278.3	282.2	281.4	286.6	284.1	289.4	292.8	293.3
Mean (cm)	2.37	1.60	1.60	1.62	1.62	1.65	1.63	1.66	1.68	1.69
Mean (°)	3.40	2.30	2.29	2.32	2.32	2.36	2.34	2.38	2.41	2.41
Std (px)	227.7	90.8	83.7	90.2	84.5	100.0	85.8	85.0	90.4	93.2
Std (cm)	1.31	0.52	0.48	0.52	0.49	0.57	0.49	0.49	0.52	0.54
Std (°)	1.87	0.75	0.69	0.74	0.70	0.82	0.71	0.70	0.74	0.77
Improvement	0	26.9%	27.2%	26.1%	26.6%	24.9%	25.7%	24.3%	23.9%	23.0%

τ value	550	600	650	700	750	800	850	900	950	1000
Mean (px)	295.3	293.1	301.2	305.7	303.3	304.5	308.0	302.7	312.9	314.0
Mean (cm)	1.70	1.69	1.73	1.76	1.74	1.75	1.77	1.74	1.80	1.81
Mean (°)	2.43	2.41	2.48	2.52	2.50	2.51	2.53	2.49	2.58	2.58
Std (px)	89.1	93.6	92.3	92.0	92.9	89.7	95.8	94.7	101.0	92.2
Std (cm)	0.51	0.54	0.53	0.53	0.53	0.52	0.55	0.54	0.58	0.53
Std (°)	0.73	0.77	0.76	0.76	0.77	0.74	0.79	0.78	0.83	0.76
Improvement	22.4%	23.3%	21.3%	19.8%	20.8%	20.8%	19.3%	20.9%	18.3%	18.1%

Table 5: Mean, Standard Deviation and Improvement to No Fine-tuning of Minimal Gaze Error under Different τ for Iterative Result

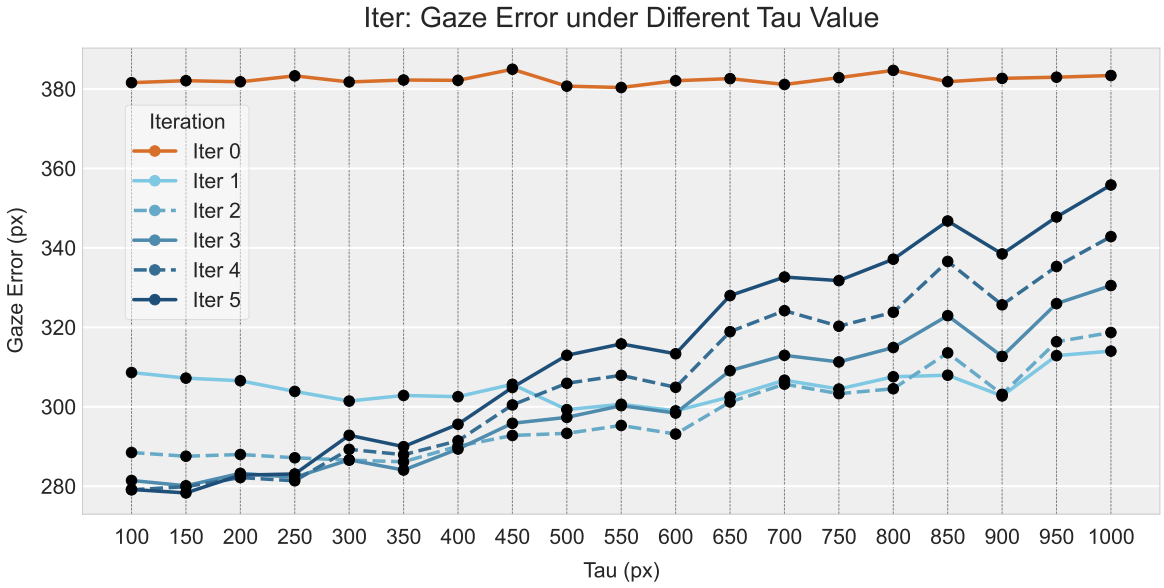


Figure 13: Gaze Error of Model 1 Given Different τ Values and Iterations

τ	100	150	200	250	300	350	400	450	500	550	600	650	700	750	800	850	900	950	1000
iter 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
iter 2	-	-	-	-	-	-	-	-	/	/	/	/	/	/	/	/	/	/	/
iter 3	-	-	/	/	/	/	/	/	/	+	+	+	+	+	+	+	+	+	+
iter 4	/	/	/	/	/	+	/	+	+	+	+	+	+	+	+	+	+	+	+
iter 5	/	/	/	/	+	/	+	+	+	+	+	+	+	+	+	+	+	+	+

Table 6: Significance Gaze Error Difference between Current and Previous Iteration under Different τ Values

indicating the fine-tuning does not improve the true positive rate of Model 2.

As shown in Figure 15, the FP difference without fine-tuning is significantly larger than that with fine-tuning. The FP difference

iteration	iter=2			iter=3			iter=4			iter=5		
top three τ with least gaze error	250	300	350	100	150	250	100	150	250	100	150	200
τ with significantly greater gaze error	650 to 1000	650 to 1000	650 to 1000	450 to 1000	450 to 1000	450 to 1000	450 to 1000	400 to 1000	400 to 1000	300 to 1000	300 to 1000	400 to 1000

Table 7: Top Three τ Values with the Smallest Gaze Error and Corresponding τ Ranges with Significantly Greater Gaze Error for Different Iterations

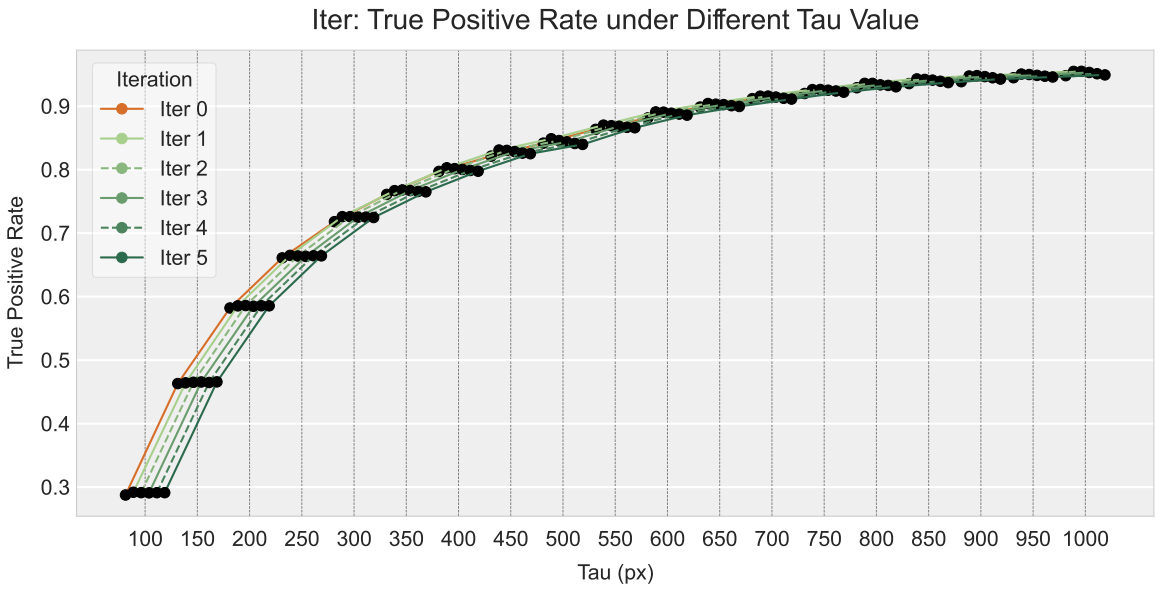


Figure 14: True Positive Rate of Model 2 Given Different τ Values and Iterations

increases with the τ value, as well as iterations when τ value is large. RM-ANOVA result indicates a significant effect from $\tau * iteration$ ($F_{9.38,215.7} = 2.8, p < 0.001$). Post-hoc LSD analysis supports the conclusion above.

Table 8 shows the difference of FP difference between current and previous iteration under different τ values. A "-" indicates the current iteration has significantly lower FP difference, "+" indicates significantly higher, and "/" indicates no significant difference. It is evident that the FP difference of iteration 0 (without fine-tuning) is significantly higher than those of iteration from 2 to 5. Furthermore, fine-tuning no longer yields significant improvements after the third iteration, and when τ exceeds 450.0 px (2.59 cm), fine-tuning constantly leads to worse performance.

In summary, we prove that the model structure in Section 6 which introduces latent features of video from Model 1 into Model 2 could decrease the difference between the gaze-cursor distance and the τ value of false positive cases. Additionally, we suggest to use a τ value smaller than 450.0 px (2.59 cm) in practical applications.

7.4 Summary of Evaluation

In conclusion, our iterative method achieved an optimal gaze error of 278.3 px (1.60 cm, 2.29°) when τ is150.0 px (0.86 cm) and iteration is 5, representing a 27.2% improvement compared to the results without fine-tune. The τ value has a significant impact on Model 1's gaze error and Model 2's true positive rate, but has no significant impact on Model 2's FP difference. In practical application, we recommend using a τ value smaller than 300.0 px (1.73 cm). Iteration has a significant impact on Model 1's gaze error and Model 2's FP difference, but has no significant impact on Model 2's true positive rate. For both Model 1's gaze error and Model 2's FP difference, those of iteration 0 (without fine-tuning) are significantly worse than those of iteration from 1 to 5 (after fine-tuning). More iterations of fine-tuning improve performance when the τ value is small, but reduce performance when the τ value is large. This may be because higher τ values inherently lead to less accurate information for fine-tuning. Furthermore, the improvement observed with increasing iterations highlights the effectiveness of the model structure described in Section 6, where latent video features from Model 1 are incorporated into Model 2 to enhance performance.

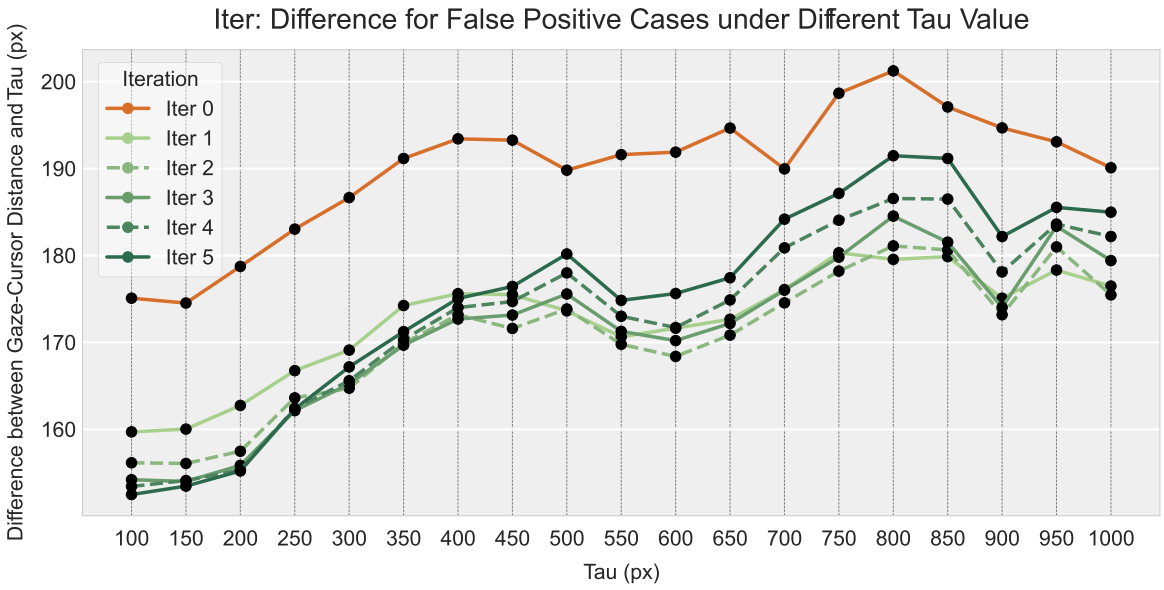


Figure 15: FP difference of Model 2 Given Different τ Values and Iterations

τ	100	150	200	250	300	350	400	450	500	550	600	650	700	750	800	850	900	950	1000
iter 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
iter 2	-	-	-	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/
iter 3	-	/	/	/	/	/	/	/	/	/	/	/	/	/	+	/	/	/	+
iter 4	/	/	/	/	/	/	/	/	+	/	/	/	+	+	/	+	+	/	/
iter 5	-	/	/	/	+	/	/	+	+	+	+	+	+	+	+	+	+	/	/

Table 8: Significance of Difference of FP Difference between Current and Previous Iteration under Different τ Values

8 Real-Time Evaluation

8.1 Apparatus, Participants and Procedure

The apparatus used in this Section is identical to that in Section 3.2.1's design space experiment.

A total of 14 participants (5 females, aged 19-31, M = 24.4) were recruited from the university for this experiment. None of the participants had previously participated in the data collection experiment. During the experiment, participants held the 3D-printed stand with the Tobii eye tracker, used the touch tensor attached to the back of the smartphone to achieve activation, and used thumb slide to achieve refinement. This time participants **did not use** the metal stand to prevent shakes from swipes, and were told to use the smartphone naturally. The entire experiment lasted 45 minutes to an hour. Participants were compensated at a rate of $15 USD per hour.

The procedure is almost identical to that in Section 4.2. There are three differences: (1) Participants will conduct five rounds in one experiment; (2) In each round, participants will first conduct manual calibration, followed by 50 target selection tasks; (3) Participants could rest for five minutes between each round.

We will collect the following data during the experiment: the size and position of the target, the sequence of cursor positions, the sequence of eye tracking data (gaze positions), and the video from the front camera. We used all the data in Section 4 to train a model at $\tau =$ 150.0 px (0.86 cm) and $iteration = 5$, which is then used as the initial model in the experiment. After each round, the collected data is pre-processed as mentioned in Section 4.3, and then used to fine-tune the model in real-time. The updated model is applied in the next round of the experiment.

8.2 Real-time Evaluation Result

Table 9 shows the mean and standard deviation of gaze error for five rounds, and for the overall minimum, as well as the improvement of gaze error mean compared to that without fine-tuning. The overall minimum is the minimal gaze error among five rounds. We focus on the overall minimum because when fine-tuning with the same meta data, its effects vary across participants due to differences in their data distributions. The gaze error of i-th round is computed as follows: (1) Use the model from the $i-1$-th round, which was trained on data from rounds 1 to $i-1$, and the front camera data collected in the i-th round to compute the $gaze_{estimate}$. (2) Eye tracker data from the i-th round is used as the $gaze_{truth}$. (3) The gaze error is computed as the Euclidean distance between $gaze_{estimate}$ and $gaze_{truth}$. Therefore, the data of Round 1 demonstrates the mean and std without fine-tuning, the data of Round 2 demonstrates the

Round	1	2	3	4	5	minimal
Mean (px)	893.6	801.7	590.1	517.2	538.0	446.7
Mean (cm)	5.14	4.61	3.39	2.97	3.09	2.57
Mean (°)	7.32	6.57	4.85	4.25	4.42	3.68
Std (px)	393.0	645.2	139.7	160.8	177.7	112.2
Std (cm)	2.26	3.71	0.8	0.92	1.02	0.64
Std (°)	3.23	5.39	1.15	1.32	1.46	0.92
Improvement	/	10.3%	34.0%	42.1%	39.8%	50.0%

Table 9: Mean, Standard Deviation of Gaze Error for Five Rounds and for the Overall Minimum, and Improvement Compared to the Mean without Fine-tuning

mean and std computed using data from Round 2 and the model from Round 1, etc.

As a result, our method shows a gaze error mean of 893.6 px (5.14 cm, 7.32°) without fine-tuning. Among the rounds, the fourth round has the lowest gaze error mean at 517.2 px (2.98 cm, 4.25°), representing a 42.1% improvement compared to the no fine-tuning condition. The mean of overall minimum gaze error across all rounds is 446.7 px (2.57 cm, 3.68°), which is a 50.0% improvement over the no fine-tuning condition. Compared to offline performance, it outperforms real-world performance by 37.8%. We propose several possible explanations for this discrepancy:

(1) Significant posture changes occurred between rounds during the real-time experiment, as participants had to put down and pick up the smartphone for each round. In contrast, our data used in the offline evaluation was collected while participants consistently held the smartphone, maintaining stable postures.
(2) Less stable camera footage during interaction due to thumb taps and swipes, as well as head movements, since the real-time experiment did not use the metal stand.
(3) Less data for fine-tuning. The average time consumption of the fine-tuning in the real-time experiment from Round 1 to Round 5 is 22.09 s, 40.98 s, 67.24 s, 84.79 s, 109.25 s. The average time consumption of offline training on our dataset is around 350 s. This difference reflects the variation in the amount of data, which directly impacts the fine-tuning performance.

8.3 Empirical Comparison with Other Calibration Method

Table 10 shows the empirical comparison of gaze error between COMETIC and other existing methods. We listed both the offline gaze error from Section 7, as well as the real-time gaze error from Section 8. The results indicate that the gaze error of our method is comparable to that of existing methods.

9 Discussion

9.1 Eye-Tracking Performance and Interaction Experience

On the smartphone mentioned in Section 4 with a resolution of 1200×2486, the typical icon layout is 4-column by 7-row. The offline minimal gaze error computed from our dataset in Section 7.3 is 278.3 px (1.60 cm, 2.29°), enabling icon-level gaze selection and analysis for a 4-column by 8-row layout. In contrast, the real-time minimal gaze error computed in real-time conditions is 446.7 px (2.57 cm, 3.68°). While this only supports a 2-column by 5-row layout, it is still sufficient for effective target selection using our method's cursor interaction. Although both methods show significantly improved performance compared to the absence of calibration, the offline performance is 37.8% better than the real-time performance. As mentioned in Section 8.2, this discrepancy may be attributed to: (1) significant posture changes caused by putting down and picking up the smartphone, (2) the less stable camera footage caused by swipe shakes and head movements, (3) the less amount of data for fine-tuning.

9.2 Potential Exploration on Gaze Behavior

Although this study experimented with three different interaction distances (200, 600, and 1200 px), no further investigation was conducted into how these distances affect user behavior and, subsequently, the impact of data validity on model training performance. We believe that future work could build upon this study to explore these aspects. Furthermore, if interaction distances do influence data validity (e.g., longer distances leading to more gaze-following behaviors and providing more valuable data), we could adjust the eye-tracking outcomes to guide users toward specific actions or behaviors, enriching the data and ultimately improving model outcomes.

9.3 Impact of Content on Gaze Behavior in Real-Time Interaction Scenarios

One of the key limitations of this study is that it does not thoroughly explore how varying content backgrounds in real-life settings affect gaze behavior during interaction and, subsequently, the accuracy of our implicit calibration. Different types of content—whether they are text-heavy interfaces, video playback, or interactive elements—can lead to distinct attractions in eye movement during cursor interaction.

However, as discussed in section 2, the use of a cursor in interactive tasks provides constraint on the user's gaze behavior, helping to mitigate some of the unpredictability found in more natural, unconstrained interactions. The presence of the cursor helps direct the user's attention toward specific areas of the interface, making the eye movements more predictable and easier to model.

Project	Year	Devices	Sensor	Calibration Method	Error(cm)	Error(°)
COMETIC (offline)	2024	Mobile	RGB	Cursor interaction	1.60 cm	2.29°
COMETIC (real-time)	2024	Mobile	RGB	Cursor interaction	2.57 cm	3.68°
PACE (real-time) [21]	2016	PC	RGB	Interaction event	3.09 cm	2.56°
GazeRefineNet (offline) [48]	2020	PC	RGB	Visual saliency	2.75 cm	2.49°
GazeL (offline) [47]	2021	Mobile	RGB	Touch event	1.58 cm	-
vGaze (real-time) [68]	2021	Mobile	RGB-D	Image attention	1.51 cm	-
iTracker (offline) [30]	2016	Mobile	RGB	Explicit calibrate with 13 points	1.34 cm	-

Table 10: Empirical Comparison between COMETIC and Existing Methods

Moreover, from an interaction design perspective, there are several strategies that could be employed to further reduce the influence of content on gaze behavior. For example, activating the cursor could dim the areas outside its immediate surroundings, directing the user's attention more effectively toward the active region. This approach minimizes distractions from other on-screen elements, allowing for more consistent gaze behavior during cursor-based interactions.

9.4 Limitation on Efficiency

The calibration method used in this study fundamentally involves collecting additional training data and fine-tuning the eye-tracking model. Compared to traditional calibration techniques, this inevitably presents efficiency challenges. Standard multi-point calibration methods typically take around 30 seconds to 1 minute to complete. In contrast, our approach requires significantly more time. Utilizing an NVIDIA GeForce RTX 3090, each iteration of model fine-tuning takes approximately 350 seconds, therefore five rounds of interaction took roughly 30 minutes. This time could further increase as more user data accumulates, making the process increasingly time-intensive.

We believe that one of the primary reasons for the inefficiency lies in the fact that the current model is still learning how to effectively extract user-specific features during the fine-tuning phase, rather than simply establishing a robust mapping between features and gaze positions.

With advancements in computer vision, using existing models to extract effective features tends to be a better solution. For example, by using segmentation techniques, the eye region could be divided into key areas such as the pupil, cornea, eyelids, and eyebrows, which could then be parameterized and fed into the eye-tracking model. In this approach, the dimensionality eye-tracking model's input is significantly reduced, and the model would only need to focus on the mapping. This approach might even enable the use of simpler machine learning methods to handle the mapping process, thereby dramatically reducing the time required for calibration.

Another issue observed in our study is that, due to the inherent differences in data distributions across users, the optimal meta-data for achieving the best fine-tuning results may also vary significantly between individuals. In this study, we used the same meta-data for all participants, which likely contributed to the considerable variation in training outcomes. We believe that developing algorithms capable of quickly adapting to personalized data, potentially by leveraging the aforementioned feature extraction techniques, is an important avenue for future research.

Despite the issues mentioned above, we still believe that implicit calibration during interaction is more suitable than explicit calibration. There are three main reasons for this:

(1) For eye tracking on mobile devices using cameras, frequent calibration is often needed to maintain accuracy. If users are required to perform explicit calibration frequently, it would inevitably disrupt their normal use.
(2) The current fine-tuning process, which takes 350 seconds, uses data from around 72 instances of cursor usage. We could further reduce calibration time by fine-tuning with fewer instances of cursor usage. As shown in Section 8.2, a smaller amount of data reduces the training time, but at the cost of lower eye-tracking accuracy. Identifying an optimal number of cursor usage events for effective calibration is one of the directions for future work.
(3) Advancements in hardware will further decrease the time consumption for fine-tuning model.

9.5 Limitation on Privacy

The calibration method also introduces privacy issues, as fine-tuning requires GPU processing, necessitating the upload of front-camera data to the cloud. We believe there are two ways to mitigate this issue:

(1) We could deploy a deep neural network model on the smartphone to encode the data from the front camera, and retrain the models using encoded data as input. This method avoids directly uploading videos containing user facial data to the cloud.
(2) As mentioned in Section 9.4, our method suffers from low efficiency since it needs to achieve feature extraction as well as mapping features to gaze positions simultaneously. Utilizing a reliable feature extraction model may decrease the number of parameters, potentially enabling full model deployment on a smartphone.

10 Conclusion

In this paper, we presented COMETIC, an interaction-integrated method for implicit, continuous eye-tracking calibration on smartphones. Our system leverages low-effort cursor interactions to collect calibration data during cursor refinement on the smartphone. By filtering valid cursor coordinates and using them as gaze position proxies, our approach achieves unobtrusive calibration and maintains eye-tracking accuracy during the interaction. Offline evaluation on our dataset demonstrated a 27.2% improvement in

eye tracking accuracy, achieving a mean error of 278.3 px (1.60 cm, 2.29°). The real-time evaluation demonstrates a 50.0% improvement, achieving a mean error of 446.7 px (2.57 cm, 3.68°). Future work could further optimize the system by exploring the impact of content variability and efficiency, with the goal of enhancing real-time performance in more complex situations.

Acknowledgments

This work is supported by the Natural Science Foundation of China under Grant No. 62132010, Beijing Key Lab of Networked Multimedia, Institute for Artificial Intelligence, Tsinghua University (THUAI), Beijing National Research Center for Information Science and Technology (BNRist), 2025 Key Technological Innovation Program of Ningbo City under Grant No. 2022Z080, Beijing Municipal Science and Technology Commission, Administrative Commission of Zhongguancun Science Park No.Z221100006722018, Science and Technology Innovation Key R&D Program of Chongqing, and Natural Science Foundation of China under Grant No. 62402271.

References

[1] Fares Alnajar, Theo Gevers, Roberto Valenti, and Sennay Ghebreab. 2013. Calibration-Free Gaze Estimation Using Human Gaze Patterns. In *2013 IEEE International Conference on Computer Vision*. 137–144. https://doi.org/10.1109/ICCV.2013.24

[2] Dima Amso, Sara Haas, and Julie Markant. 2014. An Eye Tracking Investigation of Developmental Change in Bottom-up Attention Orienting to Faces in Cluttered Natural Scenes. *PloS one* 9 (01 2014), e85701. https://doi.org/10.1371/journal.pone.0085701

[3] Jun Bao, Buyu Liu, and Jun Yu. 2022. An individual-difference-aware model for cross-person gaze estimation. *IEEE Transactions on Image Processing* 31 (2022), 3322–3333.

[4] Pascal Bérard, Derek Bradley, Markus Gross, and Thabo Beeler. 2016. Lightweight eye capture using a parametric model. *ACM Trans. Graph.* 35, 4, Article 117 (jul 2016), 12 pages. https://doi.org/10.1145/2897824.2925962

[5] Hans-Joachim Bieg, Lewis L Chuang, Roland W Fleming, Harald Reiterer, and Heinrich H Bülthoff. 2010. Eye and pointer coordination in search and selection tasks. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*. 89–92.

[6] Monchu Chen and Veraneka Lim. 2013. Eye gaze and mouse cursor relationship in a debugging task. In *HCI International 2013-Posters' Extended Abstracts: International Conference, HCI International 2013, Las Vegas, NV, USA, July 21-26, 2013, Proceedings, Part I 15*. Springer, 468–472.

[7] Shiwei Cheng, Qiufeng Ping, Jialing Wang, and Yijian Chen. 2022. EasyGaze: Hybrid eye tracking approach for handheld mobile devices. *Virtual Reality & Intelligent Hardware* 4, 2 (2022), 173–188.

[8] Shiwei Cheng, Jialing Wang, Xiaoquan Shen, Yijian Chen, and Anind Dey. 2022. Collaborative eye tracking based code review through real-time shared gaze visualization. *Frontiers of Computer Science* 16, 3 (2022), 163704.

[9] Yihua Cheng, Yiwei Bao, and Feng Lu. 2022. Puregaze: Purifying gaze feature for generalizable gaze estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 436–443.

[10] Yihua Cheng, Haofei Wang, Yiwei Bao, and Feng Lu. 2024. Appearance-based gaze estimation with deep learning: A review and benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

[11] Javier de Lope and Manuel Graña. 2022. Deep transfer learning-based gaze tracking for behavioral activity recognition. *Neurocomputing* 500 (2022), 518–527.

[12] Heiko Drewes, Ken Pfeuffer, and Florian Alt. 2019. Time-and space-efficient eye tracker calibration. In *Proceedings of the 11th ACM symposium on eye tracking research & applications*. 1–8.

[13] Johanna Dunaway, Kathleen Searles, Mingxiao Sui, and Newly Paul. 2018. News Attention in a Mobile Era. *Journal of Computer-Mediated Communication* 23, 2 (03 2018), 107–124. https://doi.org/10.1093/jcmc/zmy004 arXiv:https://academic.oup.com/jcmc/article-pdf/23/2/107/24460575/zmy004.pdf

[14] Kenneth Alberto Funes Mora, Florent Monay, and Jean-Marc Odobez. 2014. Eyediap: A database for the development and evaluation of gaze estimation algorithms from rgb and rgb-d cameras. In *Proceedings of the symposium on eye tracking research and applications*. 255–258.

[15] Nishan Gunawardena, Jeewani Anupama Ginige, and Bahman Javadi. 2022. Eye-tracking Technologies in Mobile Devices Using Edge Computing: A Systematic Review. *ACM Comput. Surv.* 55, 8, Article 158 (dec 2022), 33 pages. https://doi.org/10.1145/3546938

[16] Kyohei Hakka, Toshiyuki Ando, Buntarou Shizuki, and Shin Takahashi. 2019. Pressure-Based One-Handed Interaction Technique for Large Smartphones Using Cursor. (2019).

[17] Junfeng He, Khoi Pham, Nachiappan Valliappan, Pingmei Xu, Chase Roberts, Dmitry Lagun, and Vidhya Navalpakkam. 2019. On-device few-shot personalization for real-time gaze estimation. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*. 0–0.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[19] S Hochreiter. 1997. Long Short-term Memory. *Neural Computation MIT-Press* (1997).

[20] Jeff Huang, Ryen White, and Georg Buscher. 2012. User see, user point: gaze and cursor alignment in web search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 1341–1350. https://doi.org/10.1145/2207676.2208591

[21] Michael Xuelin Huang, Tiffany CK Kwok, Grace Ngai, Stephen CF Chan, and Hong Va Leong. 2016. Building a personalized, auto-calibrating eye tracker from user interactions. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. 5169–5179.

[22] Qiong Huang, Ashok Veeraraghavan, and Ashutosh Sabharwal. 2016. TabletGaze: Unconstrained Appearance-based Gaze Estimation in Mobile Tablets. arXiv:1508.01244 [cs.CV] https://arxiv.org/abs/1508.01244

[23] Sinh Huynh, Rajesh Krishna Balan, and JeongGil Ko. 2021. imon: Appearance-based gaze tracking system on mobile devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 4 (2021), 1–26.

[24] Xinhui Jiang, Yang Li, Jussi PP Jokinen, Viet Ba Hirvola, Antti Oulasvirta, and Xiangshi Ren. 2020. How we type: Eye and finger movement strategies in mobile typing. In *Proceedings of the 2020 CHI conference on human factors in computing systems*. 1–14.

[25] Pawel Kasprowski and Katarzyna Harezlak. 2016. Implicit calibration using predicted gaze targets. In *Proceedings of the ninth Biennial ACM symposium on eye tracking research & applications*. 245–248.

[26] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik, and Antonio Torralba. 2019. Gaze360: Physically unconstrained gaze estimation in the wild. In *Proceedings of the IEEE/CVF international conference on computer vision*. 6912–6921.

[27] Sunjun Kim, Jihyun Yu, and Geehyuk Lee. 2012. Interaction techniques for unreachable objects on the touchscreen. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*. 295–298.

[28] Christof Koch and Shimon Ullman. 1987. *Shifts in Selective Visual Attention: Towards the Underlying Neural Circuitry*. Springer Netherlands, Dordrecht, 115–141. https://doi.org/10.1007/978-94-009-3833-5_5

[29] Andy Kong, Karan Ahuja, Mayank Goel, and Chris Harrison. 2021. EyeMU Interactions: Gaze + IMU Gestures on Mobile Devices. In *Proceedings of the 2021 International Conference on Multimodal Interaction* (Montréal, QC, Canada) *(ICMI '21)*. Association for Computing Machinery, New York, NY, USA, 577–585. https://doi.org/10.1145/3462244.3479938

[30] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. 2016. Eye Tracking for Everyone. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2176–2184. https://doi.org/10.1109/CVPR.2016.239

[31] Mikko Kytö, Barrett Ens, Thammathip Piumsomboon, Gun A Lee, and Mark Billinghurst. 2018. Pinpointing: Precise head-and eye-based target selection for augmented reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–14.

[32] Yaxiong Lei, Shijing He, Mohamed Khamis, and Juan Ye. 2023. An End-to-End Review of Gaze Estimation and its Interactive Applications on Handheld Mobile Devices. *ACM Comput. Surv.* 56, 2, Article 34 (sep 2023), 38 pages. https://doi.org/10.1145/3606947

[33] Yaxiong Lei, Yuheng Wang, Tyler Caslin, Alexander Wisowaty, Xu Zhu, Mohamed Khamis, and Juan Ye. 2023. DynamicRead: Exploring Robust Gaze Interaction Methods for Reading on Handheld Mobile Devices under Dynamic Conditions. *Proc. ACM Hum.-Comput. Interact.* 7, ETRA, Article 158 (may 2023), 17 pages. https://doi.org/10.1145/3591127

[34] Runtong Li, Huimin Ma, Rongquan Wang, and Jiawei Ding. 2021. Device-adaptive 2D gaze estimation: a multi-point differential framework. In *International Conference on Image and Graphics*. Springer, 485–497.

[35] Dongze Lian, Ziheng Zhang, Weixin Luo, Lina Hu, Minye Wu, Zechao Li, Jingyi Yu, and Shenghua Gao. 2019. RGBD based gaze estimation via multi-task CNN. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 2488–2495.

[36] Dong Liang, Shaoming Yan, Yuanliang Ju, Rong Quan, and Huawei Tu. 2024. E2GO : Free Your Hands for Smartphone Interaction. https://doi.org/10.21203/

rs.3.rs-3909704/v2

[37] Daniel J Liebling and Susan T Dumais. 2014. Gaze and mouse coordination in everyday work. In *Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing: adjunct publication*. 1141–1150.

[38] Daniel J. Liebling and Susan T. Dumais. 2014. Gaze and mouse coordination in everyday work. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication* (Seattle, Washington) *(UbiComp '14 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 1141–1150. https://doi.org/10.1145/2638728.2641692

[39] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. 2019. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172* (2019).

[40] Shijian Luo, Yi Hu, and Yuxiao Zhou. 2017. Factors attracting Chinese Generation Y in the smartphone application marketplace. *Frontiers of Computer Science* 11 (2017), 290–306.

[41] Sven Mayer, Gierad Laput, and Chris Harrison. 2020. Enhancing Mobile Voice Assistants with WorldGaze. 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, 1–10. https://doi.org/10.1145/3313831.3376479

[42] Alexandre Milisavljevic, Fabrice Abate, Thomas Le Bras, Bernard Gosselin, Matei Mancas, and Karine Doré-Mazars. 2021. Similarities and differences between eye and mouse dynamics during web pages exploration. *Frontiers in Psychology* 12 (2021), 554595.

[43] Alexandre Milisavljevic, Kevin Hamard, Coralie Petermann, Bernard Gosselin, Karine Doré-Mazars, and Matei Mancas. 2018. Eye and Mouse Coordination During Task: From Behaviour to Prediction. 86–93. https://doi.org/10.5220/0006618800860093

[44] Kenneth Alberto Funes Mora and Jean-Marc Odobez. 2013. Person independent 3d gaze estimation from remote rgb-d cameras. In *2013 IEEE International Conference on Image Processing*. IEEE, 2787–2791.

[45] Mathieu Nancel, Emmanuel Pietriga, Olivier Chapuis, and Michel Beaudouin-Lafon. 2015. Mid-air pointing on ultra-walls. *ACM Transactions on Computer-Human Interaction (TOCHI)* 22, 5 (2015), 1–62.

[46] Cristina Palmero, Javier Selva, Mohammad Ali Bagheri, and Sergio Escalera. 2018. Recurrent cnn for 3d gaze estimation using appearance and shape cues. *arXiv preprint arXiv:1805.03064* (2018).

[47] Joonbeom Park, Seonghoon Park, and Hojung Cha. 2021. GAZEL: Runtime Gaze Tracking for Smartphones. In *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. 1–10. https://doi.org/10.1109/PERCOM50583.2021.9439113

[48] Seonwook Park, Emre Aksan, Xucong Zhang, and Otmar Hilliges. 2020. Towards end-to-end video-based eye-tracking. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*. Springer, 747–763.

[49] Ken Pfeuffer, Melodie Vidal, Jayson Turner, Andreas Bulling, and Hans Gellersen. 2013. Pursuit calibration: making gaze calibration less tedious and more flexible. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 261–270. https://doi.org/10.1145/2501988.2501998

[50] Radiah Rivu, Yasmeen Abdrabou, Ken Pfeuffer, Mariam Hassib, and Florian Alt. 2020. Gaze'N'Touch: Enhancing Text Selection on Mobile Devices Using Gaze. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3334480.3382802

[51] Sheikh Rivu, Yasmeen Abdrabou, Thomas Mayer, Ken Pfeuffer, and Florian Alt. 2019. GazeButton: enhancing buttons with eye gaze interactions. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications* (Denver, Colorado) *(ETRA '19)*. Association for Computing Machinery, New York, NY, USA, Article 73, 7 pages. https://doi.org/10.1145/3317956.3318154

[52] Kerry Rodden, Xin Fu, Anne Aula, and Ian Spiro. 2008. Eye-mouse coordination patterns on web search results pages. In *CHI'08 extended abstracts on Human factors in computing systems*. 2997–3002.

[53] Volker Roth and Thea Turner. 2009. Bezel swipe: conflict-free scrolling and multiple selection on mobile touch screen devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1523–1526.

[54] Nachiappan Valliappan, Na Dai, Ethan Steinberg, Junfeng He, Kantwon Rogers, Venky Ramachandran, Pingmei Xu, Mina Shojaeizadeh, Li Guo, Kai Kohlhoff, et al. 2020. Accelerating eye movement research via accurate and affordable smartphone eye tracking. *Nature communications* 11, 1 (2020), 4553.

[55] Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: spontaneous interaction with displays based on smooth pursuit eye movement and moving targets. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Zurich, Switzerland) *(UbiComp '13)*. Association for Computing Machinery, New York, NY, USA, 439–448. https://doi.org/10.1145/2493432.2493477

[56] Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2015. Pursuits: Spontaneous Eye-Based Interaction for Dynamic Interfaces. *GetMobile: Mobile Comp. and Comm.* 18, 4 (jan 2015), 8–10. https://doi.org/10.1145/2721914.2721917

[57] Melodie Vidal, Ken Pfeuffer, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: eye-based interaction with moving targets. 3147–3150. https://doi.org/10.1145/2468356.2479632

[58] Congyi Wang, Fuhao Shi, Shihong Xia, and Jinxiang Chai. 2016. Realtime 3D eye gaze animation using a single RGB camera. *ACM Trans. Graph.* 35, 4, Article 118 (jul 2016), 14 pages. https://doi.org/10.1145/2897824.2925947

[59] Kang Wang, Shen Wang, and Qiang Ji. 2016. Deep eye fixation map learning for calibration-free eye gaze tracking. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications* (Charleston, South Carolina) *(ETRA '16)*. Association for Computing Machinery, New York, NY, USA, 47–55. https://doi.org/10.1145/2857491.2857515

[60] Pierre Weill-Tessier and Hans Gellersen. 2018. Correlation between gaze and hovers during decision-making interaction. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*. 1–5.

[61] Pierre Weill-Tessier, Jayson Turner, and Hans Gellersen. 2016. How do you look at what you touch? A study of touch interaction and gaze correlation on tablets. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*. 329–330.

[62] Pierre Weill-Tessier, Jayson Turner, and Hans Gellersen. 2016. How do you look at what you touch? a study of touch interaction and gaze correlation on tablets. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications* (Charleston, South Carolina) *(ETRA '16)*. Association for Computing Machinery, New York, NY, USA, 329–330. https://doi.org/10.1145/2857491.2888592

[63] Quan Wen, Derek Bradley, Thabo Beeler, Seonwook Park, Otmar Hilliges, Junhai Yong, and Feng Xu. 2020. Accurate real-time 3D gaze tracking using a lightweight eyeball calibration. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 475–485.

[64] Pui Chung Wong, Hongbo Fu, and Kening Zhu. 2016. Back-mirror: Back-of-device one-handed interaction on smartphones. In *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications*. 1–5.

[65] Erroll Wood and Andreas Bulling. 2014. EyeTab: model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications* (Safety Harbor, Florida) *(ETRA '14)*. Association for Computing Machinery, New York, NY, USA, 207–210. https://doi.org/10.1145/2578153.2578185

[66] Yang Xing, Jianlin Tang, Hong Liu, Chen Lv, Dongpu Cao, Efstathios Velenis, and Fei-Yue Wang. 2018. End-to-end driving activities and secondary tasks recognition using deep convolutional neural network and transfer learning. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1626–1631.

[67] Yunyang Xiong, Hyunwoo J Kim, and Vikas Singh. 2019. Mixed effects neural networks (menets) with applications to gaze estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 7743–7752.

[68] Songzhou Yang, Yuan He, and Meng Jin. 2021. vGaze: Implicit Saliency-Aware Calibration for Continuous Gaze Tracking on Mobile Devices. In *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*. 1–10. https://doi.org/10.1109/INFOCOM42981.2021.9488668

[69] Jun-Seok Yun, Youngju Na, Hee Hyeon Kim, Hyung-Il Kim, and Seok Bong Yoo. 2022. HAZE-Net: High-frequency attentive super-resolved gaze estimation in low-resolution face images. In *Proceedings of the Asian Conference on Computer Vision*. 3361–3378.

[70] Shumin Zhai, Carlos Morimoto, and Steven Ihde. 1999. Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Pittsburgh, Pennsylvania, USA) *(CHI '99)*. Association for Computing Machinery, New York, NY, USA, 246–253. https://doi.org/10.1145/302979.303053

[71] Xucong Zhang, Seonwook Park, Thabo Beeler, Derek Bradley, Siyu Tang, and Otmar Hilliges. 2020. ETH-XGaze: A Large Scale Dataset for Gaze Estimation Under Extreme Head Pose and Gaze Variation. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 365–381.

[72] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2015. Appearance-based gaze estimation in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4511–4520.

[73] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2017. It's written all over your face: Full-face appearance-based gaze estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 51–60.

[74] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2017. Mpiigaze: Real-world dataset and deep appearance-based gaze estimation. *IEEE transactions on pattern analysis and machine intelligence* 41, 1 (2017), 162–175.

[75] Xiaoyan Zhao, Beibei Chao, Wei Hu, Guihuan Feng, and Bing Luo. 2017. Eyes Never Lie! Hand Eye Coordination Patterns Analysis for Text-graph Separation. In *Proceedings of the Fifth International Symposium of Chinese CHI*. 60–64.

[76] Jinchao Zhou, Guoan Li, Feng Shi, Xiaoyan Guo, Pengfei Wan, and Miao Wang. 2023. EM-Gaze: eye context correlation and metric learning for gaze estimation.

Visual Computing for Industry, Biomedicine, and Art 6, 1 (2023), 8.

[77] Xiaolong Zhou, Jianing Lin, Zhuo Zhang, Zhanpeng Shao, Shenyong Chen, and Honghai Liu. 2020. Improved itracker combined with bidirectional long short-term memory for 3D gaze estimation using appearance cues. *Neurocomputing* 390 (2020), 217–225.

[78] Anjie Zhu, Qianjing Wei, Yilin Hu, Zhangwei Zhang, and Shiwei Cheng. 2018. MobiET: A New Approach to Eye Tracking for Mobile Device. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers* (Singapore, Singapore) *(UbiComp '18)*. Association for Computing Machinery, New York, NY, USA, 862–869. https://doi.org/10.1145/3267305.3274174