E2GO : Free Your Hands for Smartphone Interaction

Dong Liang

liangdong@nuaa.edu.cn

Nanjing University of Aeronautics and Astronautics https://orcid.org/0000-0003-2784-3449

Shaoming Yan

Yuanliang Ju

Shanghai Qi Zhi Institute

Rong Quan

Nanjing University of Aeronautics and Astronautics

Huawei Tu

Department of Computer Science and Information Technology, La Trobe University

Article

Keywords:

Posted Date: April 29th, 2024

DOI: https://doi.org/10.21203/rs.3.rs-3909704/v2

Additional Declarations: The authors declare no competing interests.

E^2GO : Free Your Hands for Smartphone Interaction

Shaoming Yan[1] Yuanliang Ju[2] Rong Quan[1] Huawei Tu[3] Dong Liang[1]

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China

[2] Edward S. Rogers Sr. Department of Electrical & Computer Engineering, University of Toronto, Toronto, Canada

[3] Department of Computer Science and Information Technology, La Trobe University, Melbourne, Australia

Abstract

Current eye-gaze interaction technologies for smartphones are considered inflexible, inaccurate, and power-hungry. These methods typically rely on hand involvement and accomplish partial interactions. In this paper, we propose a novel eye-gaze smartphone interaction method named Event-driven Eye-Gaze Operation (E^2GO), which can realize comprehensive interaction using only eyes and gazes to cover various interaction types. Before the interaction, an anti-jitter gaze estimation method was exploited to stabilize human eye fixation and predict accurate and stable human gaze positions on smartphone screens to further explore refined time-dependent eye-gaze interactions. We also integrated an event-triggering mechanism in E^2GO to significantly decrease its power consumption to deploy on smartphones. We have implemented the prototype of E^2GO on different brands of smartphones and conducted a comprehensive user study to validate its efficacy, demonstrating E^2GO's superior smartphone control capabilities across various scenarios. Demo videos

1. Introduction

Smartphones have become indispensable to human life. Besides the traditional finger touch interaction [1–4], a series of novel interaction approaches have also been explored, such as voice [5–7], gesture [8–11] and eye-gaze interaction [12–16]. To control smartphones, eye-gaze interaction involves utilizing eye and gaze information, such as gaze positions, gaze gestures, dwell time, and eye blinks. It allows users to navigate and operate their smartphones seamlessly with only their eyes, frees up hands, and has the potential to improve user experience. Hu *et al.* [17] conducted a study on the effectiveness of gaze input, specifically dwell time and gaze combined with tapping, in navigating touchscreen menus. Their findings showcased notable reductions in user effort. Moreover, eye-gaze smartphone interaction enables people with motor disabilities to operate smartphones as the **eye-machine interface**, which holds significant potential for promoting social civilization.

Extensive research has been conducted on eye-gaze smartphone interactions [13, 18–27], most of which require hand assistance to realize comprehensive gaze control operations. For example, Nagamatsu *et al.* [19] proposed a gaze and touch interface that enables one-hand control of mobile devices. Users can select an object on the screen by focusing on it while simultaneously touching anywhere on the screen. Rivu *et al.* [23] introduced GazeButton, which supports three different input modes based on gaze and finger click. Users can choose an object by looking at it and tapping on the GazeButton or select text by gazing over it while holding the GazeButton. These methods can only achieve partial control operations, which is far from practical eye-gaze smartphone control. Kong *et al.* [13] proposed a smartphone interface driven by gaze and IMU (Inertial Measurement Unit) named EyeMU. They designed seven hand motion gestures based on IMU's output, paired with the users' gazes to realize various smartphone operations and eliminate the need for touch input. However, it still needs hand involvement, making it impractical when both hands of the users are occupied or for people with motor disabilities.

Unlike existing methods, we propose an eye-gaze smartphone interaction method named Event-driven Eye-Gaze-Controlled Operation (E^2GO) to realize comprehensive smartphone control operations just with the users' eyes and gazes. E^2GO first leverages the smartphone's front camera to capture images. Then it exploits face recognition [28] and gazes estimation [29, 30] to capture users' faces and calculate the users' gaze positions and eye blink situations, respectively. We design four pairs of interactive actions based on the users' gaze positions, gaze gestures, dwell time, and eye blinks. Based on the captured eye and gaze-related information and designed actions, E^2GO determines the action the user is performing and adaptively adjusts the instruction of the interaction.

The premise of eye-gaze smartphone interaction is gaze estimation, *i.e.,* estimating the users' gaze positions on the smartphone screen. Accurate gaze estimation is the basis for capturing gaze-related information and determining the

interactive action. However, our tests revealed a persistent issue with existing smartphone gaze estimation methods [13, 31–34]: they consistently encounter the problem of jitter. This issue arises from the relative movements of the user and the devices, as well as the inherent irregular micro-movements of the human eye when fixating on an object, which always leads to inconstant shifts in the detected gaze positions [35] and hampers the effectiveness of interaction. This problem is not prominent during simple eye control interactions. However, it limits the design of more complex time-dependent interactive actions; for example, staring for a while represents a click action. We introduce an Anti-Jitter Strategy (AJS) to stabilize gaze positions in E^2GO. AJS helps distinguish jitter from typical gaze shift by evaluating the fine-grained distance between the gaze positions of successive frames.

Presumably, once activated, if E^2GO continuously executes gaze estimation on every image frame, it would bring colossal energy consumption to smartphones. Therefore, we tend to use the event-triggering mechanism to reduce energy consumption, which only senses object movements. We introduce a Motion Event Detector (MED) to monitor the events between consecutive frames. Instead of performing gaze estimation for every captured image, we only perform gaze estimation when events between two successive frames exceed a threshold during the interaction. Performing gaze estimation exclusively on frames with a substantial number of events, the runtime of the E^2GO can be significantly extended, rendering it more energy-efficient.

We have prototyped E^2GO and deployed it on three different brands of smartphones. Then, we conducted an in-depth user study to evaluate its effectiveness in various aspects. The user study shows that E^2GO achieves accurate and efficient eye-gaze smartphone interactions in multiple conditions. In summary, this study has the following contributions:

- We propose an eye-gaze smartphone interaction method E^2GO that supports users to perform complex interaction tasks with the help of four pairs of interactive actions.
- We introduced an Anti-Jitter Strategy (AJS) for E^2GO to ensure accurate and stable gaze positions on a screen.
- We introduced a Motion Event Detector (MED) for E^2GO to decrease energy consumption significantly.
- The user study results demonstrated that our proposed E^2GO can realize accurate eye-gaze smartphone control in various situations.

2. Related Works

2.1. Gaze Interaction Techniques

Researchers have developed various methods to enable users to interact with computer systems using their gaze. These techniques include gaze-based pointing, text entry, and object manipulation. For example, dwell-time-based selection and gaze gestures have been proposed for accurate and efficient gaze-based pointing [36, 37]. In the realm of gaze-based text entry, dwell-time-based virtual keyboards and eye-typing techniques have been explored. Dwell-time-based virtual keyboards present a grid of characters, and users select a character by fixating on it for a certain period. This method enables text input solely through gaze and has been studied extensively to enhance typing speed and accuracy [38–40]. Eye typing techniques leverage predictive algorithms and eye tracking to facilitate faster text entry by inferring intended words or phrases from the user's gaze patterns. Gaze-based object manipulation techniques have also been investigated for tasks such as selecting, manipulating, and resizing virtual objects. By tracking the user's gaze, these techniques enable users to interact with virtual objects in three-dimensional space. For instance, users can select objects by directly fixating on them or manipulate objects by gazing at specific control points or handles [41–43]. These techniques have applications in various domains, including virtual reality and 3D modeling.

2.2. Gaze Estimation on Mobile Devices

Gaze estimation on mobile devices is much more challenging due to limited computational resources and constrained hardware [44–46]. These methods often leverage machine learning algorithms and utilize built-in sensors, such as front-facing cameras and inertial sensors, to estimate the user's gaze direction [47]. Moreover, gaze estimation models have been proposed to provide superior accuracy in predicting gaze direction [29, 30]. These models use a convolutional neural network to learn complex patterns in eye images to improve gaze estimation accuracy. Model compression and efficient network architectures have been explored to reduce the computational requirements of deep learning models, making them more suitable for deployment on mobile devices with limited resources [14]. Integrating machine learning with mobile device sensors has paved the way for new forms of interaction on smartphones and other portable devices. Looking forward, we anticipate that these advancements will continue to evolve, enabling more robust and accurate gaze-based interaction in a wide range of mobile contexts [48, 49].

2.3. Eye-controlled Approach for Mobile Devices

Eye gaze interactions on mobile devices predate the advent of smartphones and tablets equipped with front-facing cameras. Drewes *et al.* [50] were pioneers in exploring eye-gaze interaction for controlling mobile phone applications. They experimented with two control schemes based on dwell-time and gaze gestures, demonstrating the feasibility and potential of eye-gaze interaction on mobile phones. EyePhone [51] introduced an approach to control mobile appli-

cations by tracking users' eye fixations on the screen and their eye blinks. Nagamatsu *et al.* [19] proposed a gaze-and-touch interaction interface, allowing users to select objects on a smartphone screen by fixating their eyes on the target and touching them anywhere on the screen. Pfeuffer *et al.* [52] study the integration of gaze and touch interaction on tablets, finding that although users perform slightly slower, they can achieve one-handed use with less physical effort. While these methods offered innovative ways to interact with smartphones using eye gaze, they couldn't cover all smartphone operations effectively. Therefore, they may not be practical for widespread deployment. In contrast, EyeMU [13] combined IMU-tracked motion gestures with accurate gaze position estimation. This approach covered many smartphone operations, reducing the need for constant thumb-tapping and swiping. However, it still required users to move their hands continuously, which could be physically demanding and less accessible to individuals with motor disabilities.

2.4. Event Cameras

Event cameras are a relatively new technology. Since the seminal work [53], they have gained increasing interest due to their appealing properties, which allow them to perform well in challenging scenarios for standard cameras, such as high speed, high dynamic range, and low power consumption. See [54] for a recent survey. Typical scenarios where event cameras offer advantages over other sensing modalities include real-time interaction systems, such as robotics or wearable electronics [55], where operation under uncontrolled lighting conditions, latency, and power are important [56]. Event cameras are used for object tracking [57], surveillance and monitoring [58], and object/gesture recognition [59, 60]. They are also profitable for depth estimation [61], optical flow estimation [62], HDR image reconstruction [63] and Simultaneous Localization and Mapping (SLAM) [64]. Anastasios *et al.*[65] developed a hybrid frame-event-based near-eye gaze tracking system that boasts a remarkable data acquisition speed surpassing 10,000 Hz and achieves impressive accuracies ranging from 0.45° to 1.75° for fields of view spanning from 45° to 98°. However, the system requires an event camera device as an auxiliary tool, posing challenges for its application to mobile devices for gaze interaction.

3. Method

Once triggered, E^2GO exploits an Anti-Jitter Strategy (AJS) to estimate both accurate and stable human gaze positions from the images captured by the front cameras of the smartphones. Based on the gaze positions and the current smartphone page, E^2GO captures other necessary eye and gaze-related information. According to the obtained information, E^2GO determines the interaction action the user is performing and adjusts the smartphone page based on the interaction action. We design four pairs of eye-gaze intereaction actions for E^2GO, enabling users to navigate through various pages of their smartphones solely using their eyes and gazes. It's worth noting that during the execution of E^2GO, we exploit a Motion Event Detector (MED) to detect the number of events between two adjacent frames continuously. Compared to performing gaze estimation for each frame, the MED-based execution method of gaze estimation is more power-efficient.

In the following, we will first introduce our AJS-based gaze estimation. Next, we will take a closer look at the four pairs of interaction actions and how they are performed. Then, we will describe our MED-based power-efficient means for E^2GO.

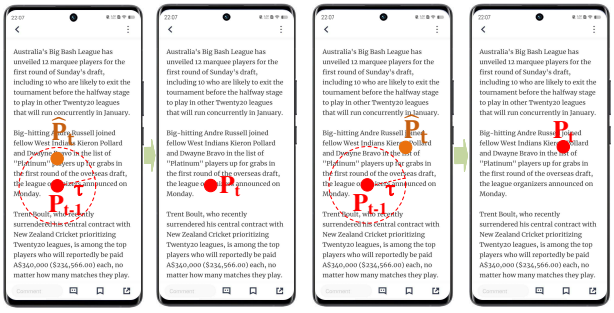


Figure 1. **Detailed Illustration of AJS.** Left: The gaze fixation is a jitter and ignored. Right: the gaze fixation is real and kept.

3.1. AJS-based Gaze Estimation

When users are staring at a spot, their gaze position is slightly shaken around that spot, mainly due to the micro-saccades, micro-tremors, and imperceptible small head movements [35]. The slight shake phenomenon is called the jitter problem. The jitter problem also occurs in eye-gaze smartphone interaction. It worsens when users interact with their smartphones in a moving environment, disrupting the eye-gaze interaction. Our work aims to address the jitter problem in gaze estimation by proposing a method based on anti-jitter strategy. Specifically, we introduce AJS into EyeMU's [66] gaze estimation method to construct an AJS-based gaze estimation method. Next, we will introduce our AJS-based gaze estimation process in detail.

Once triggered, E^2GO employs the smartphone's front-facing camera to capture images at 30fps. For the captured image at time t, we first use MediaPipe Face Mesh [28] for face detection, creating a mesh with 468 3D face landmarks, as illustrated in the Input Face Landmarks Image in Supplementary Fig. S1. Based on the face landmarks, we first calculate the user's eye corner coordinates, face area size, and head pose (pitch, yaw, and roll) and then segment out both of the user's eyes. As shown in Supplementary Fig. S1,

the user's eye corner coordinates, face area size, head pose, and eye images are input into a deep neural network [66] to primarily predict the user's gaze position $\hat{P}_t$ on the screen.

After obtaining the primary gaze position $\hat{P}_t$, we calculate the distance $D(\hat{P}_t, P_{t-1})$ between $\hat{P}_t$ and the previous gaze position P_{t-1}. As shown in Fig. 1, if $D(\hat{P}_t, P_{t-1})$ is lower than the certain threshold τ, then we treat $\hat{P}_t$ as a jitter and ignore it, making P_{t-1} the finally predicted gaze position of time t, *i.e.,* $P_t = P_{t-1}$. On the contrary, if $D(\hat{P}_t, P_{t-1})$ is larger than τ, then we consider $\hat{P}_t$ as the finally predicted gaze position of time t, *i.e.,* $P_t = \hat{P}_t$.

By detecting and removing the jitters, the AJS-based gaze estimation method only retains the natural gaze fixations and thus predicts both accurate and stable gaze positions on smartphone screens. Exploiting the AJS-based gaze estimation method can block up the side effects of the jitter problem on eye-gaze smartphone interaction well in the gaze estimation stage. Based on this, E^2GO can also be used generally in moving environments, such as when users walk, take buses, etc. We have evaluated the effects of E^2GO in static and moving situations in user study. The user study results have demonstrated the effectiveness of E^2GO in both situations.

3.2. Eye-related Information Collection

We design four pairs of interaction actions based on the user's eye and gaze-related information. After obtaining the user's gaze positions on the screen, we continue to detect the state of the user's eyes.

In Supplementary Fig. S1, the upper and lower eyelids' coordinates from the face landmarks allow us to determine whether the user's eyes are open or closed. Specifically, we define the highest and lowest coordinates on the vertical axis of the left eyelid as P_1 and P_2, and those of the right eyelid as P_3 and P_4, respectively. We define the left eye's height as $Eye_{left} = P_1 - P_2$, and the right eye's height as $Eye_{right} = P_3 - P_4$.

Considering the different sizes of the users' eyes and the variations in the distance between users and screens, E^2GO employs a dynamic adaptive method to detect the opening and closing state of users' eyes. Taking the left eye as an example, after continuously collecting 50 values of Eye_{left}, we take their average as the eye height when the user's left eye is open, denoted as Eye_{left}^{open}. During the interaction, if Eye_{left} is smaller than 50% of Eye_{left}^{open}, the left eye is considered closed. On the other hand, if Eye_{left} is larger than 80% of Eye_{left}^{open}, the left eye is considered open.

3.3. The Eye-gaze Interaction Actions

As mentioned, we have studied the most prevalent information display means of current mobile applications and developed four pairs of eye-gaze interaction actions correspondingly. These actions can substitute the users' frequent tapping and swiping operations when using smartphones. In eye-tracking metrics, the number of fixation counts reflects the different areas on the screen that the user focuses on, which helps to understand the user's level of interest in specific information [67]. Then, we classify the triggering areas of interactive actions based on these counts, avoiding high-frequency areas with fixed counts to protect the user's viewing experience. Fig. 2 illustrates the designed actions. The first two actions are based on the users' gaze positions on the screens, the third and fourth actions are based on the gaze positions and the dwell time, the fifth and sixth actions are based on the users' gaze positions and gaze gestures. The last two actions are based on the users' gaze positions and eye blinks. Next, we will describe each action and how it is detected in detail.

3.3.1 Gazing at the Screen to Swipe the Page

The most common way to display information on smartphones is to arrange content from top to bottom, and users need to swipe up or down to see more content as they browse these pages. In response to this situation, we divided the screen into three parts from top to bottom: "Top", "Middle", and "Bottom". As illustrated in Fig. 2 (a)-(b), when the users' gaze falls on the "Bottom" (or "Top") area, E^2GO automatically swipes the smartphone page up (or down) to bring the content in the "Bottom" (or "Top") area to the "Middle" area. Then, the users browse the "Middle" area for content they are interested in.

3.3.2 Staring at the Screen to Switch the Page

Sometimes, mobile applications display information from left to right, as seen in various apps' photo browsing functions, and users need to swipe quickly to the left or right to switch pages. For such situations, we divide the smartphone screen into three areas from left to right with a ratio of 1:3:1, and define the three areas as "Left", "Middle", and "Right", respectively. When the user's gaze falls on the "Left" (or "Right") area for more than one second, E^2GO automatically switches to the "Left" (or "Right") page.

As shown in Fig. 2 (c)-(d), when the user's gaze falls on the "Left" (or "Right") area, E^2GO starts counting. If the gaze position remains on the area for more than one second, E^2GO determines that the user is staring at the "Left" (or "Right") area and switches to the "Left" (or "Right") page.

3.3.3 Swiftly Shifting Gaze within a Designated Area to Switch Short Videos

Besides text and images, short videos have become another prevalent smartphone content format, accessible through apps like TikTok, YouTube, and Instagram. Users typically swipe quickly up or down to switch between video pages

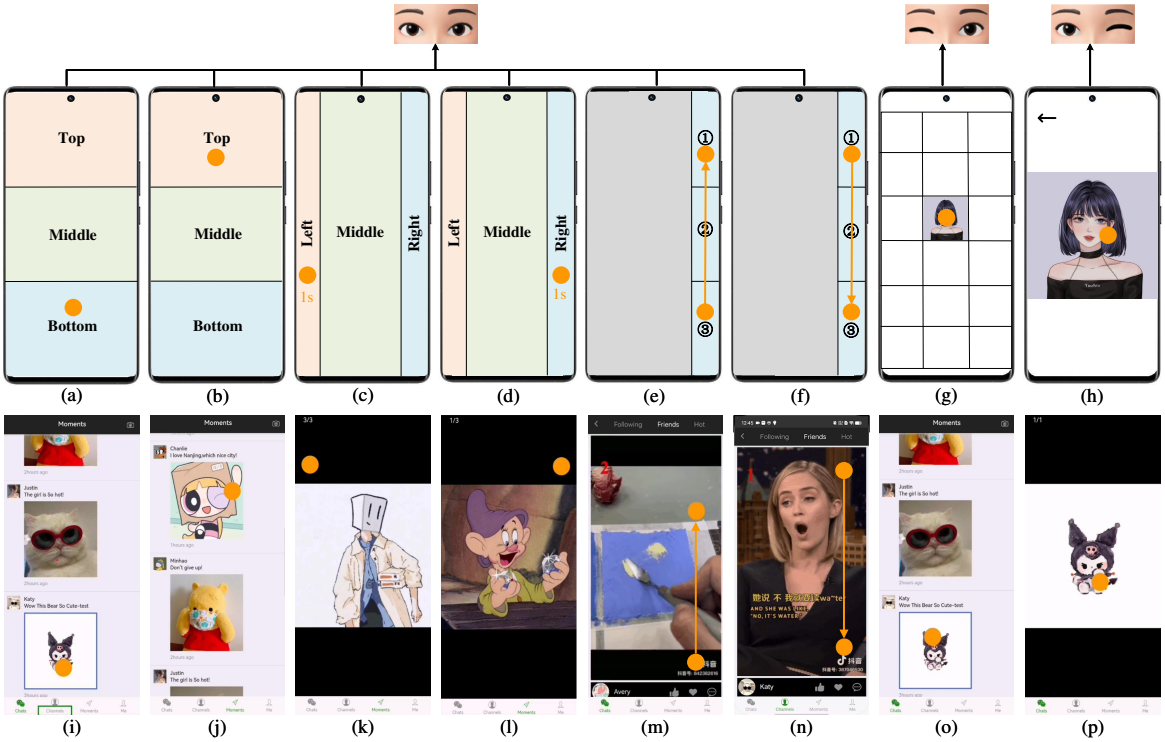


Figure 2. **The first and second rows show the actions and user tests corresponding to the E^2GO. (a)** Gazing at the "Bottom" area to swipe the page up, **(b)** Gazing at the "Top" area to swipe the page down, **(c)** Staring at the "Left" area to switch to the left page, **(d)** Staring at the "Right" area to switch to the right page, **(e)** Swiftly shifting gaze from ③ to ① to switch to the next short video, **(f)** Swiftly shifting gaze from ① to ③ to switch to the previous short video, **(g)** Closing the right eye while keeping the left eye open to click the target, **(h)** Closing the left eye while keeping the right eye open to return. The eye blink images of (g) and (h) mean that executing these two actions needs eye blinks. **(i)-(p)** correspond to the test results of the (a)-(h) action, respectively.

when watching them. In this case, we designed "swiftly shifting gaze from 'Bottom' to 'Up'" and "swiftly shifting gaze from 'Up' to 'Bottom'" actions for the users' eyes to replace their thumbs' quick swiping up and down operations. As shown in Fig. 2 (e)-(f), we designated a 1/5 screen width area on the right as the operation zone and divided it equally into three areas: ①, ②, and ③, from top to bottom.

For short video playback, E^2GO detects the "swiftly shifting gaze from 'Bottom' to 'Up'" action as follows: Once a user's gaze falls on ③, E^2GO starts counting. If the gaze transitions from ③ to① within the subsequent 60 frames, E^2GO interprets this as the "swiftly shifting gaze from 'Bottom' to 'Up'" action. If the gaze position exits the operation area within those 60 frames, E^2GO thinks the action detecting has failed and stops counting. The process for detecting the "swiftly shifting gaze from 'Up' to 'Bottom'" action is analogous.

3.3.4 Blinking one eye to click or enter

We design "closing the right eye while keeping the left eye open" and "closing the left eye while keeping the right eye open" actions to replace the conventional thumb-tapping operation. As depicted in Fig. 2 (g)-(h), when a user focuses on a target on the screen and closes his right eye while keeping his left eye open, E^2GO enters the next page associated with that target. On the contrary, if the user closes his left eye and opens his right eye, E^2GO will navigate back to the previous page.

3.4. Motion Event Detector (MED)

Gaze estimation for the captured image requires face recognition followed by the AJS-based method, consuming significant power on smartphones. Besides, E^2GO employs the smartphone's front-facing camera to capture images at 30fps. In this case, executing gaze estimation for each captured image will make E^2GO a power-inefficient method. Therefore, we utilize the principle of event camera to control the number of times the gaze estimation is executed during the interaction to reduce the power consumption of E^2GO and make it more power–efficient.

Event cameras, also known as dynamic vision sensors [53], differ from traditional frame-based cameras. In a traditional camera, changes in all pixels are captured at

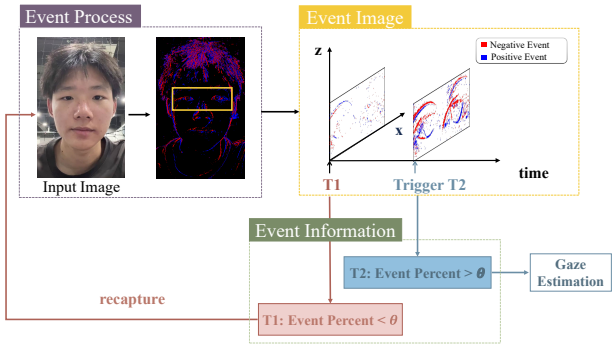


Figure 3. **Detailed Illustration of MED.** T2: When Event Percent exceeds the threshold θ, Gaze Estimation is invoked; T1: When Event Percent is less than θ, the input image is recaptured.

regular intervals. In contrast, each pixel in the event camera has an independent photoelectric sensing module and generates data only when the brightness changes. To be specific, when the brightness at a pixel (x, y) changes, the camera outputs one value $E = (x, y, t, p)$, where t is the timestamp of the event, and p can be either -1 or 1 [68], indicating a decrease or increase in brightness.

Algorithm 1 Motion Event Detector

Input: User's Image I_t, Previous grayscale EC_{t-1}, Threshold θ;
Default Setting: Calculate the number of pixels in an image function $Count$;
$EC_t \leftarrow$ grayscale(I_t)
$EI_t \leftarrow$ Event_Process(EC_t, EC_{t-1})
EP $\leftarrow Count(EI_t, event)/Count(EC_t, all)$
if EP $> \theta$ **then**
    Invoke Gaze Estimation
else
    E^2GO.update_frame(I_{t+1})
end if

Inspired by the event camera, we propose a Motion Event Detector (MED) to detect the number of "events" of each captured image, *i.e.,* each captured image's change compared with the previous frame. We perform gaze estimation only when the number of "events" exceeds a certain threshold. Specifically, we first convert the image captured at time t into a grayscale image, denoted as EC_t. Then, we generate a pseudo Event Image (EI_t) of time t by subtracting EC_t with the grayscale image EC_{t-1} at time $t-1$, as shown in Fig. 3.

After generating the pseudo-event image, we continue to calculate the ratio of event pixels to the total pixels to get the Event Percent (EP) and compare it to our predefined threshold θ (which will be explained in the section 5.6). Gaze estimation is performed if EP exceeds the threshold. Otherwise, the image is recaptured. Algorithm 1 illustrates the details of the MED.

4. Evaluation

4.1. Participants

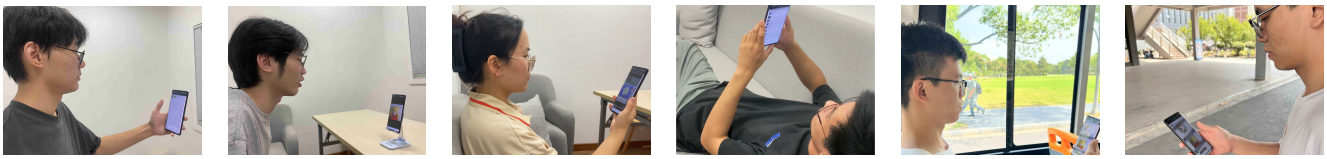
We invited 20 participants to the user study comprising 5 females and 15 males. The participants ranged from 18 to 24 years old, averaging 21. All individuals were young adults and predominantly university students. The participants' height ranged from 1.60 to 1.85 meters, averaging 1.73 meters. Four participants had some experience using eye-gaze-based technology, while the remaining participants had no experience, with an average experience score of 2.1 on a scale from 0 (no experience) to 5 (very experienced). This ensures that our results are not biased by prior familiarity with such technology. Participants reported that they sometimes use their smartphones while walking, with an average score of 3.1 on a scale from 0 (never) to 5 (always). This gave us insights into gaze interaction's real-world applicability and challenges in moving environments. The experiment took approximately one hour per participant, and all participants were compensated for their time. All participants provided explicit informed consent, allowing their data to be used for research and publicly released online. We implemented stringent data anonymization and confidentiality measures to ensure the privacy and data security of the participants. This experiment received ethical approval from our institution.

4.2. Experiment Design and Procedure

We developed and deployed our system using the MUI front-end framework. Tensorflow.js facilitated the deployment of the model, while the software deployment was carried out with Hbuilder X. Given JavaScript's extensive compatibility with various operating systems, E^2GO can theoretically function as an API, and be deployed on any operating system that supports JavaScript. We tested our system on several Android-based smartphones, including **Vivo X80 Pro, Xiaomi 9, Huawei P30**, and the specific information of these devices is shown in the Supplementary. The diversity of devices bestowed our research project with greater possibilities and flexibility, allowing us to conduct in-depth testing and optimization of E^2GO under various devices and environments.

We experimented to investigate the effects of participants interacting with E^2GO. The study involved various situations, including static postures such as **standing, sitting, slouching**, and **lying**, as well as moving gestures like **walking** and **taking buses**. To assess user performance with E^2GO, we employed the following metrics:

- **Success Rate:** Successful rate of actions in a given time.



(a) Standing (b) Sitting (c) Slouching (d) Lying (e) Taking buses (f) Walking

Figure 4. **Demonstration of participant postures. (a)** Standing, **(b)** Sitting, **(c)** Slouching, **(d)** Lying, **(e)** Taking buses, and **(f)** Walking.

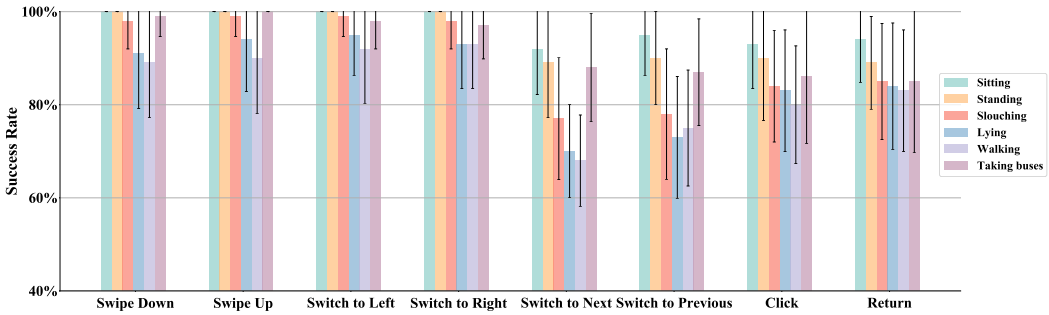


Figure 5. **The test results of E^2GO in different postures.** The bar chart represents the average success rate of the users, and the black error lines represent the standard deviation of the user tests, respectively.

- **Task Time:** Time taken to complete the actions.

During the experimental setup stage, we prepared the necessary devices for the participants and adjusted the posture to control the external variables of the experiment. First, we explained the experiment's tasks and operation methods to the participants, *i.e.,* what kind of interaction actions the users should execute to control each kind of smartphone page. We also provided demonstrations to ensure the participants understood how to perform these interaction actions correctly. Next, we would ask them to execute these actions in different postures, *i.e.,* sitting, standing, slouching, and lying, as shown in Fig. 4. We would also adjust the environmental conditions, such as the distance of the device and the angle of the light, to simulate different usage scenarios. During the operation process, we would record the operation results in detail, including the time they needed to complete each action, whether they completed the action, and whether there were any operation timeouts.

After the experiment, we conducted a survey using the System Usability Scale (SUS) with the participants. SUS was initially developed by Brooke in 1986 and consists of 10 questions, including positive statements in odd-numbered items and negative statements in even-numbered items. Participants are required to rate each question after using the system. SUS is freely available and features straightforward statements, requiring only participant ratings, making implementation quick. Researchers have analyzed SUS scores for a wide range of human-computer interaction products, indicating that SUS is a robust and versatile tool for usability assessment. Moreover, it can achieve evaluation results quickly, even with limited sample sizes [69].

Table 1. **The detailed results of the Least Significant Difference (LSD) test for the success rate of different postures.** The F calibration was performed first to prove that there was a difference between the different groups. The significance level for the mean difference is 0.05.* represents a significant difference between the two postures.

Posture (I)	Posture (J)	Mean Difference (I-J)	P-Value	95% Confidence Interval of Diff.	
				Lower	Upper
Sitting	Standing	.10	.075	-.01	.21
	Slouching	.35	0*	.24	.46
	Lying	.57	0*	.46	.68
	Walking	.65	0*	.54	.76
	taking buses	.21	0*	.10	.32
Standing	Slouching	.25	0*	.14	.36
	Lying	.47	0*	.36	.58
	Walking	.55	0*	.44	.66
	taking buses	.11	.043*	0	.22
Slouching	Lying	.22	0*	.11	.33
	Walking	.30	0*	.19	.41
	taking buses	-.14	.014*	-.25	-.03
Lying	Walking	.08	.148	-.03	.19
	taking buses	-.36	0*	-.47	-.25
Walking	taking buses	-.44	0*	-.55	-.33

5. Results

5.1. The Interaction Actions Success Rates

In the study, participants performed four pairs of actions under two environments: static and moving. Static postures

included standing, sitting, slouching, and lying, while moving postures involved walking and taking buses. The actions performed included swipe down, swipe up, switch to left, switch to right, switch to next, switch to previous, click, and return. Each participant repeated each action five times under these postures.

We plotted the average success rates of all actions performed in six different postures and the standard deviation of test results with black error lines, as shown in Fig. 5. Additionally, we presented the detailed numerical results of the experiments in the Supplementary Table S1.

- **Different postures:** Actions performed in standing and sitting postures had higher success rates ($\geq$ 85%) than other postures, with smaller standard deviations. In particular, "swipe down/up" and "switch to left/right" achieved a success rate of 100% in standing and sitting, demonstrating the effectiveness of our action design. From the Least Significant Difference (LSD) results in Table 1, there was no significant difference ($p > 0.05$) between standing and sitting postures. This is because both standing and sitting postures are commonly used for viewing mobile phones, and users are not affected by body or environmental movements in these postures, thus enhancing the user experience. Upon observation, it was noted that the success rates of actions in slouching, lying, taking buses, and walking experienced a decline, especially in lying and walking, where most actions had success rates below 90%. We believed this was due to the non-parallel alignment of the user's facial and phone planes in these postures. In these states, facial deformations occurred in the images captured by the camera, resulting in changes in the facial information input to AJS-based gaze estimation, further affecting the accuracy of gaze position and the execution of interaction actions.
- **Different actions:** From the perspective of the average success rates of different actions, "swipe down/up" and "switch to left/right" maintained success rates of over 90% in various postures, significantly surpassing other actions. Besides, the standard deviation of these two pairs of actions was also significantly smaller than the remaining. Table 2 also proved that there were no significant differences among these four actions ($p > 0.05$). This might be because these simple actions only require detecting the user's gaze position or dwell time. It was evident that "switch to next/previous" action differed significantly ($p < 0.05$) from other actions, as they entailed complex tracking of the user's gaze transition path within a specific range, with any deviation during execution potentially resulting in action failure. Therefore, the average success rates of these actions in all postures were much lower than that of simple actions, especially in slouching, lying, and walking, where the success rates were even below 80%. In these postures, instability between the user's facial plane and the phone plane prevented the algorithm from accurately capturing facial and eye information. The standard deviation of the "blinking to click/return" action was much higher than other actions. Upon inquiry, we discovered that some users could not perform a single blink, making it nearly impossible to trigger this action during testing. For users who could perform a single blink, the average success rate of this action was close to that of "swipe down/up" action.

Table 2. **The detailed results of the Least Significant Difference (LSD) test for the success rate of different actions.** The F calibration was performed first to prove that there was a difference between the different groups. The significance level for the mean difference is 0.05.* represents a significant difference between the two actions.

Action (I)	Action (J)	Mean Diff. (I-J)	P-Value	95% Confidence Interval of Diff.	
				Lower	Upper
Swipe Down	Swipe Up	-.05	.441	-.18	.08
	Switch to Left	-.06	.368	-.19	.07
	Switch to Right	-.03	.607	-.16	.09
	Switch to Next	.77	0*	.65	.90
	Switch to Previous	.66	0*	.53	.79
	Click	.51	0*	.38	.64
	Return	.48	0*	.35	.60
Swipe Up	Switch to Left	-.01	.898	-.14	.12
	Switch to Right	.02	.797	-.11	.14
	Switch to Next	.82	0*	.70	.95
	Switch to Previous	.71	0*	.58	.84
	Click	.56	0*	.43	.69
	Return	.53	0*	.40	.65
Switch to Left	Switch to Right	.03	.700	-.10	.15
	Switch to Next	.83	0*	.71	.96
	Switch to Previous	.72	0*	.59	.84
	Click	.57	0*	.44	.69
	Return	.53	0*	.41	.66
Switch to Right	Switch to Next	.81	0*	.68	.94
	Switch to Previous	.69	0*	.56	.82
	Click	.54	0*	.41	.67
	Return	.51	0*	.38	.64
Switch to Next	Switch to Previous	-.12	.072	-.24	.01
	Click	-.27	0*	-.39	-.14
	Return	-.30	0*	-.43	-.17
Switch to Previous	Click	-.15	.021*	-.28	-.02
	Return	-.18	.005*	-.31	-.06
Click	Return	-.03	.607	-.16	.09

5.2. The Interaction Actions Task Times

Our experimental results indicated significant differences in the task times of interaction actions across different postures, as shown in Fig. 6 and Supplementary Table S2.

Among all actions, "blinking to click/return" was the quickest to complete, while "switch to next/previous" took the longest time. Across different postures, actions were fastest to complete in the sitting posture, while they were relatively slower in walking. These differences in task times might be attributed to various factors, including the complexity of the action, cognitive load, attention allocation, accuracy requirements of the action, and user experience and proficiency. For instance, a relaxed posture like lying may require more cognitive resources, leading to longer task times. Similarly, the walking posture affected the smooth-

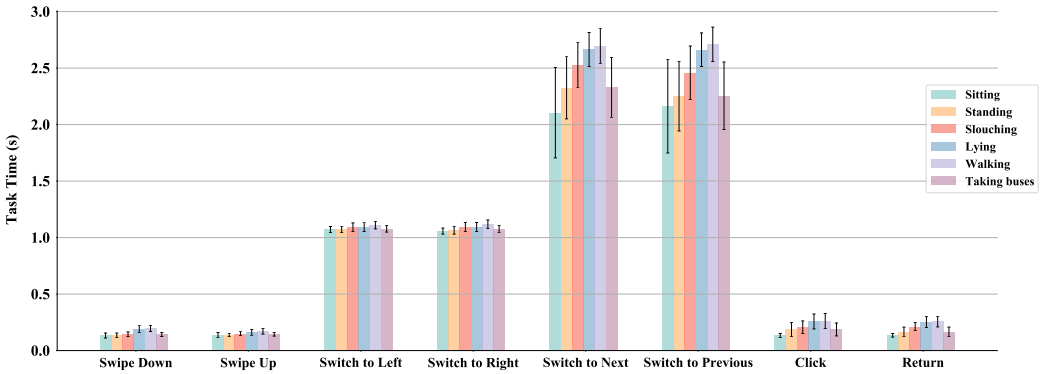


Figure 6. **Task times of E^2GO across different postures.** The bar chart represents the average task times, and the black error lines represent the standard deviation of the tests.

Table 3. **The mean success rate of the participant's eye-gaze smartphone interaction tests in different scenarios.**

	Swipe Down	Swipe Up	Switch to Left	Switch to Right	Switch to Next	Switch to Previous	Click	Return
No glasses	100%	100%	100%	100%	92%	93%	95%	95%
No glare	100%	100%	100%	100%	93%	94%	96%	94%
Mild glare	97%	98%	99%	98%	87%	86%	90%	87%
Severe glare	95%	96%	95%	94%	84%	83%	80%	81%

ness of interaction actions, particularly in the vertical direction, leading to longer times for "switch to next/previous" action.

In conclusion, different postures significantly affects the task time performance of interaction actions. Understanding these differences is crucial for optimizing the design of E^2GO and improving user experience. Future research can further explore the impact of these factors on the time performance of eye-gaze smartphone interaction actions across different postures.

5.3. The Impact of Eyeglasses Glare on E^2GO

For participants wearing glasses, reflection on the glasses may affect their visual perception and operation. We simulated the following four scenarios (no glasses, no glare, mild glare, and severe glare) by adjusting the light intensity and angle around the participant seated to represent the conditions one might encounter when using E^2GO. Supplementary Fig. S2 illustrates the scenarios for these four cases, while Table 3 displays the success rates for each action under these scenarios.

From the test results, as the glare intensity increased, there was a noticeable decrease in the success rates for each action, particularly for the "switch to next", "switch to previous", "click", and "return" actions. The success rates for these actions decreased from 93%, 94%, 96% and 94% under no glare to 84%, 83%, 80% and 81% under severe glare, while the other four actions remained relatively stable.

Eyeglass glare introduces reflected light from the light source into the eyes, disrupting the normal transmission of light. This can result in visual blurring, flickering, or disturbance, reducing the ability to discern details accurately. Glare from eyeglasses reduces the contrast in a scene, making object edges and details less clear, which makes it difficult for participants to accurately locate and manipulate targets during visual operations such as "swipe up", "swipe down", "switch to left" and "switch to right". The reflected light from eyeglass glare interferes with visual tasks by reflecting surrounding environmental light sources, including indoor or outdoor sunlight. These reflections disrupt participants' focus on their intended targets, possibly causing glare or discomfort to the eyes. Eyeglass glare requires extra eye effort to filter and adapt to this interference, leading to eye fatigue and discomfort, reducing the ability to sustain long-term focused attention. In summary, eyeglass glare negatively affects visual perception and task performance by causing visual interference, reducing contrast, creating reflections, and contributing to eye fatigue.

5.4. The Optimal Distance for E^2GO

We conducted a pilot study to analyze the E^2GO at different distances while participants were seated. Fig. 7 illustrates the success rates of each action at various distances (30 cm, 50 cm, 70 cm, 90 cm). Based on the average success rates and participant feedback, it was found that a distance of approximately 50 cm was optimal. Therefore, we suggested that participants maintain this distance during the formal experiment.

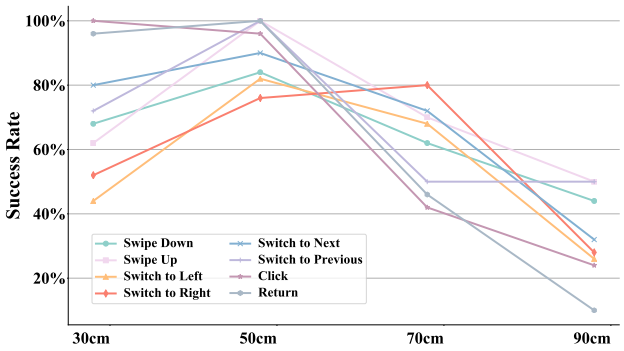


Figure 7. **Performance of E^2GO across different distances.**

Regarding E^2GO at different distances, the greater distance requires users to exert more effort to focus accurately on screen elements, potentially leading to eye fatigue and reducing interaction accuracy. Furthermore, environmental factors at different distances impact eye-gaze smartphone interaction. At greater distances, there may be more interference factors, such as variations in ambient lighting or other visual distractions. These factors disrupt users' gaze, diminishing the quality and effectiveness of the interaction. On the other hand, closer distances provide higher visual accuracy, as users find it easier to focus on and track targets precisely, enhancing their interactions' accuracy and stability. Additionally, being closer allows for a broader visual perception, enabling users to better comprehend the elements and layout on the screen, leading to more informed decision-making and actions. However, it is essential to note that when the distance is too close, there is lens distortion in the smartphone's camera, leading to some degree of facial deformation. This affects the input facial information in gaze estimation, further influencing the accuracy of the output gaze positions. Therefore, considering all these factors, we determined that the optimal distance for our experiment is 50 cm.

5.5. The Comparison of Page Settings

We conducted a pilot study to determine the division ratio of the interface. For "swipe down/up" and "switch to left/right", we evaluated four division ratios: 1:1:1, 1:2:1, 1:3:1, and 1:4:1, as shown in Table 4. Supplementary Fig. S3 shows the interface at different scales.

Table 4. **The success rate when the page is divided differently.** The best results are in bold.

	Swipe Down/Up	Switch to Left/Right
1:1:1	**99%**	90%
1:2:1	95%	92%
1:3:1	92%	**95%**
1:4:1	90%	93%

Results showed that the page division ratio had a varying degree of impact on different actions. A relatively balanced page division ratio (1:1:1) performed the best in "swipe down/up" actions. This is because a balanced ratio ensures that users trigger the action more easily, making reading the information in the middle section easier. On the other hand, when the middle division area has a larger ratio (1:3:1), the probability of users accidentally triggering "switch to left/right" action while viewing page information is lower.

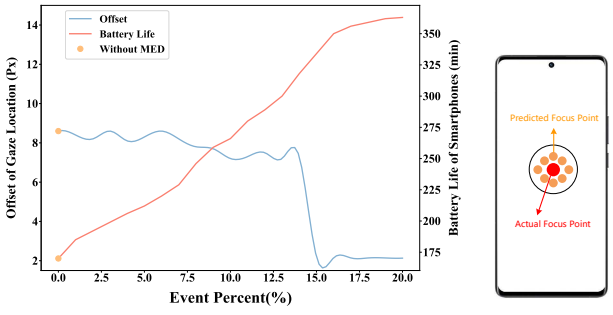
5.6. The Power-Efficient Ability of MED

As shown in Fig. 8 (b), we placed a red dot in the center of the screen and asked the participants to stare at the dot for three minutes. We calculated the offset based on the predicted range of fluctuations in the gaze positions and recorded the change in the battery life of the mobile devices on a fully charged during testing.

According to the principles of MED, when the Event Percent exceeded the threshold θ, E^2GO would re-invoke the AJS-based gaze estimation to predict. Therefore, we evaluated the impact of different thresholds on gaze position stability and the battery life of mobile devices. The results are shown in Fig. 8 (a) and Table 5.

Table 5. **Partial detailed results of the relationship between Event Percent and Battery Life.**

Event Percent(%)	Battery Life (min)
0%	170
1%	185
5%	220
10%	280
15%	350



(a) Result (b) Design

Figure 8. **Experiment of MED**. **(a)** illustrating the change in battery life and gaze position offset for different Event Percent. **(b)** depicting the testing method.

In our test results, we observed that when the threshold was small, small differences in eye images between neighboring frames also exceeded the threshold, leading to frequent calls to the AJS-based gaze estimation. On the other

Table 6. **The System Usability Scale Content and Average Score.**

Question No.	Statement	Average Score
1	I think that I would like to use this product frequently.	4.55
2	I found the product unnecessarily complex.	2
3	I thought the product was easy to use.	4.25
4	I think that I would need the support of a technical person to be able to use this product.	2
5	I found the various functions in the product were well integrated.	4.2
6	I thought there was too much inconsistency in this product.	1.65
7	I imagine that most people would learn to use this product very quickly.	4.45
8	I found the product very awkward to use.	1.6
9	I felt very confident using the product.	4.85
10	I needed to learn a lot of things before I could get going with this product.	1.55

hand, when the threshold was large, the Event Percent remained below the threshold even if the eye image changed significantly. In this case, the gaze estimation was rarely called to re-predict the gaze position, adversely affecting the user interaction experience.

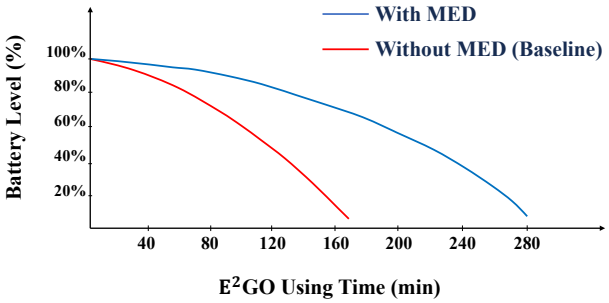


Figure 9. **A comparison of the usage time for the battery life with and without MED for E^2GO.**

It should be noted that the predicted gaze positions only fluctuated within a small range since the user was constantly looking at the red dot on the screen. However, when the threshold was set too large, the offset of gaze position decreased dramatically. Under the circumstances, gaze estimation only re-invoked when the user blinked causing a large change in the eye image, resulting in the predicted gaze positions remaining constant. While the battery life is longer at larger thresholds, this will leave the model mostly at rest and will not be invoked even when the user's eyes are changing normally, which is contrary to our original intent. Therefore, we set the threshold θ to 10%. After that, we tested the difference between with MED and without it, as shown in Fig. 9. It is obvious that the runtime of E^2GO increases by about 50% with the help of MED.

5.7. Evaluation results of System Usability Scale

System Usability Scale (SUS) provides an overall usability assessment metric consisting of 10 questions, with subscales 4, 5, and 10 consisting of "Validity", subscales 2, 3, 7, and 8 consisting of "Usability" and subscales 1, 6, and 9 consisting of "Satisfaction". The scores of 1, 2, 3, 4, and 5 indicate that they strongly disagree, disagree, are neutral, agree, and strongly agree, respectively. At the end of the experiment, we surveyed the subjects on the system usability scale, as shown in Table 6.

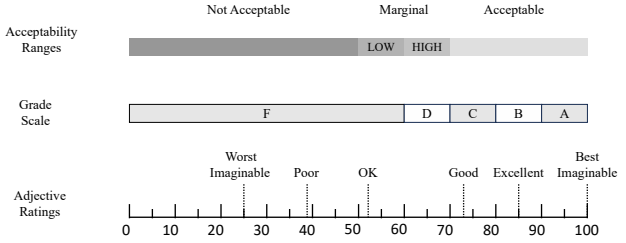


Figure 10. **A comparison of mean System Usability Scale (SUS) scores by quartile, adjective ratings, and the acceptability.**

According to the SUS scoring rule, the score of the odd-numbered items is subtracted by 1, and the score of the even-numbered items is subtracted by 5. The sum of the two scores is 2.5 times of the score of the SUS. The total score of the SUS was calculated to be **83.8**. According to the SUS score evaluation system (shown in Figure. 10), if the score exceeds 70, the system can be evaluated as high acceptance range. It can be found that most of the participants are satisfied with E^2GO.

6. Limitations and Future Work

Despite our best efforts to design and execute a comprehensive experiment, some limitations should be acknowledged:

- **Participant Variation:** Even though we aimed to recruit a diverse group of participants, there were inherent limitations due to individual differences in factors such as physical abilities (e.g. the ability to perform distinct single eye blinks), familiarity with eye-gaze interaction actions, and personal preferences.
- **Controlled Environment:** While we adjusted the environmental conditions to simulate different usage scenarios, these conditions inevitably differed from the myriad of real-life situations and environments in which users interacted with their devices.

- **Limited Gestures and Postures:** The gestures and postures included in our experiment represent a subset of all possible gestures and postures. Other gestures and postures not explored in this study may yield different results.
- **Experiment Duration:** The experiments may not fully capture the effects of prolonged usage and fatigue over time.

While these limitations may impact the generalizability of the findings, they provide valuable directions for future research and offer insights that can inform the design and evaluation of eye-gaze interfaces.

7. Conclusion

This paper introduces and validates the E^2GO system, an Event-Driven Eye-Gaze Operation framework designed to facilitate hands-free smartphone interactions. E^2GO addresses critical challenges associated with existing eye-gaze interaction methods, such as the limited perceptual capabilities of smartphone front cameras, the complexity of relative motion between the device and the user's eyes, and the constraints of smartphone computing and power resources. By integrating a lightweight gaze estimation module with a robust anti-shake, E^2GO achieves reliable gaze estimation even in less-than-ideal conditions. The system's utilization of event cameras not only optimizes operational efficiency but also significantly reduces power consumption, which is paramount for mobile devices. The tailored design of E^2GO, with its robust and distinguishable eye-gaze interaction actions, supports a variety of smartphone interfaces and usage contexts. The extensive user study conducted has demonstrated E^2GO's precise control capabilities, marking it as a promising step forward in the domain of eye-gaze smartphone interactions. Future work may explore additional gaze gestures, further optimizations for power efficiency, and broader applications across different devices and user demographics.

References

[1] Chao Shen, Yong Zhang, Xiaohong Guan, and Roy A Maxion. Performance analysis of touch-interaction behavior for active smartphone authentication. *IEEE Transactions on Information Forensics and Security*, 11(3):498–513, 2015. 1

[2] Huy Viet Le, Sven Mayer, and Niels Henze. Infinitouch: Finger-aware interaction on fully touch sensitive smartphones. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, pages 779–792, 2018.

[3] Surjya Ghosh, Kaustubh Hiware, Niloy Ganguly, Bivas Mitra, and Pradipta De. Emotion detection from touch interactions during text entry on smartphones. *International Journal of Human-Computer Studies*, 130:47–57, 2019.

[4] Xiang Chen, Kent W Nixon, Hucheng Zhou, Yunxin Liu, and Yiran Chen. {FingerShadow}: An {OLED} power optimization based on smartphone touch interactions. In *6th Workshop on Power-Aware Computing and Systems (HotPower 14)*, 2014. 1

[5] S Sriwati, E Eruinsyah, S Karim, and F Rahman. Control of electronic devices using smartphone-based voice identification. *IOP Conference Series: Materials Science and Engineering*, 662(2):022004, 2019. 1

[6] Zaid Sh Alattar, Tarek Abbes, and Faouzi Zerai. Smartphone-key: Hands-free two-factor authentication for voice-controlled devices using wi-fi location. *IEEE Transactions on Network and Service Management*, 2023.

[7] Matthew B Hoy. Alexa, siri, cortana, and more: an introduction to voice assistants. *Medical reference services quarterly*, 37(1):81–88, 2018. 1

[8] Myungin Lee. Entangled: A multi-modal, multi-user interactive instrument in virtual 3d space using the smartphone for gesture control. *The International Conference on New Interfaces for Musical Expression*, 2021. 1

[9] David Rozado, T Moreno, Javier San Agustin, FB Rodriguez, and Pablo Varona. Controlling a smartphone using gaze gestures as the input mechanism. *Human–Computer Interaction*, 30(1):34–63, 2015.

[10] Yang Qifan, Tang Hao, Zhao Xuebing, Li Yin, and Zhang Sanfeng. Dolphin: Ultrasonic-based gesture recognition on smartphone platform. In *2014 IEEE 17th International Conference on Computational Science and Engineering*, pages 1461–1468. IEEE, 2014.

[11] Chi-Huang Hung, Ying-Wen Bai, and Hsu-Yao Wu. Home outlet and led array lamp controlled by a smartphone with a hand gesture recognition. In *2016 IEEE international conference on consumer electronics (ICCE)*, pages 5–6. IEEE, 2016. 1

[12] Lucas Paletta, Helmut Neuschmied, Michael Schwarz, Gerald Lodron, Martin Pszeida, Stefan Ladstätter, and Patrick Luley. Smartphone eye tracking toolbox: accurate gaze recovery on mobile displays. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 367–68, 2014. 1

[13] Andy Kong, Karan Ahuja, Mayank Goel, and Chris Harrison. Eyemu interactions: Gaze+ imu gestures on mobile devices. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 577–585, 2021. 1, 2, 3

[14] Michael Xuelin Huang, Jiajia Li, Grace Ngai, and Hong Va Leong. Screenglint: Practical, in-situ gaze estimation on smartphones. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 2546–2557, 2017. 2

[15] Joonbeom Park, Seonghoon Park, and Hojung Cha. Gazel: Runtime gaze tracking for smartphones. In *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10. IEEE, 2021.

[16] Mengyu Qiu, Quan Rong, Dong Liang, and Huawei Tu. Visual scanpath transformer: Guiding computers to see the world. In *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 223–232. IEEE, 2023. 1

[17] Yanfei Hu Fleischhauer, Hemant Bhaskar Surale, Florian Alt, and Ken Pfeuffer. Gaze-based mode-switching to enhance interaction with menus on tablets. In *Proceedings of*

the 2023 Symposium on Eye Tracking Research and Applications, ETRA '23, New York, NY, USA, 2023. Association for Computing Machinery. 1

[18] H. Drewes, A. D. Luca, and A. Schmidt. Eye-gaze interaction for mobile phones. In *International Symposium on International Conference on Mobile Technology, Applications, DBLP*, 2007. 1

[19] Takashi Nagamatsu, Michiya Yamamoto, and Hiroshi Sato. Mobigaze: Development of a gaze interface for handheld mobile devices. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*, pages 3349–3354. Association for Computing Machinery, 2010. 1, 3

[20] Emiliano Miluzzo, Tianyu Wang, and Andrew T Campbell. Eyephone: activating mobile phones with your eyes. In *Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds*, pages 15–20, 2010.

[21] Xinbo Zhao, Huan Xie, and Xiaochun Zou. Gaze-gesture interaction for mobile phones. In *2012 7th International Conference on Computing and Convergence Technology (ICCCT)*, pages 1030–1033. IEEE, 2012.

[22] Morten Lund Dybdal, Javier San Agustin, and John Paulin Hansen. Gaze input for mobile devices by dwell and gestures. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 225–228, 2012.

[23] Sheikh Rivu, Yasmeen Abdrabou, Thomas Mayer, Ken Pfeuffer, and Florian Alt. Gazebutton: enhancing buttons with eye gaze interactions. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*, pages 1–7, 2019. 1

[24] Luqian Ren, Haoxian Huang, Hao Wang, and Zhuo Yang. Gazegrid: A novel interaction method based on gaze estimation. In *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)*, pages 1–5. IEEE, 2021.

[25] Radiah Rivu, Yasmeen Abdrabou, Ken Pfeuffer, Mariam Hassib, and Florian Alt. 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*, page 1–8, 2020.

[26] Radiah Rivu, Yasmeen Abdrabou, Ken Pfeuffer, Mariam Hassib, and Florian Alt. 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*, pages 1–8, 2020.

[27] Omar Namnakani, Yasmeen Abdrabou, Jonathan Grizou, Augusto Esteves, and Mohamed Khamis. Comparing dwell time, pursuits and gaze gestures for gaze interaction on handheld mobile devices. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–17, 2023. 1

[28] Google. Mediapipe face mesh. In *https://developers.google.com/mediapipe*, 2021. 1, 3

[29] Joe Lemley, Shabab Bazrafkan, and Peter Corcoran. Deep learning for consumer devices and services: Pushing the limits for machine learning, artificial intelligence, and computer vision. *IEEE Consumer Electronics Magazine*, 6(2):48–56, 2017. 1, 2

[30] Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. Learning-by-synthesis for appearance-based 3d gaze estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1821–1828, 2014. 1, 2

[31] Darius Miniotas and Oleg Špakov. An algorithm to counteract eye jitter in gaze-controlled interfaces. *Information Technology and Control*, 30(1), 2004. 2

[32] Moaaz Hudhud Mughrabi, Aunnoy K Mutasim, Wolfgang Stuerzlinger, and Anil Ufuk Batmaz. My eyes hurt: Effects of jitter in 3d gaze tracking. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pages 310–315. IEEE, 2022.

[33] Pingmei Xu, Krista A Ehinger, Yinda Zhang, Adam Finkelstein, Sanjeev R Kulkarni, and Jianxiong Xiao. Turkergaze: Crowdsourcing saliency with webcam based eye tracking. *arXiv preprint arXiv:1504.06755*, 2015.

[34] Yifei Huang, Minjie Cai, Zhenqiang Li, and Yoichi Sato. Predicting gaze in egocentric video by learning task-dependent attention transition. In *Proceedings of the European conference on computer vision (ECCV)*, pages 754–769, 2018. 2

[35] Juno Kim and Stephen Palmisano. Effects of active and passive viewpoint jitter on vection in depth. *Brain research bulletin*, 77(6):335–342, 2008. 2, 3

[36] Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. Fast gaze typing with an adjustable dwell time. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 357–360, 2009. 2

[37] Misahael Fernandez, Florian Mathis, and Mohamed Khamis. Gazewheels: Comparing dwell-time feedback and methods for gaze input. In *Proceedings of the 11th Nordic Conference on Human-Computer Interaction: Shaping Experiences, Shaping Society*, pages 1–6, 2020. 2

[38] Xueshi Lu, Difeng Yu, Hai-Ning Liang, Wenge Xu, Yuzheng Chen, Xiang Li, and Khalad Hasan. Exploration of hands-free text entry techniques for virtual reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 344–349. IEEE, 2020. 2

[39] Wenxin Feng, Jiangnan Zou, Andrew Kurauchi, Carlos H Morimoto, and Margrit Betke. Hgaze typing: Head-gesture assisted gaze typing. In *ACM Symposium on eye tracking research and applications*, pages 1–11, 2021.

[40] Yogesh Kumar Meena, Hubert Cecotti, KongFatt Wong-Lin, and Girijesh Prasad. Design and evaluation of a time adaptive multimodal virtual keyboard. *Journal on Multimodal User Interfaces*, 13:343–361, 2019. 2

[41] Chang Liu, Alexander Plopski, and Jason Orlosky. Orthogaze: Gaze-based three-dimensional object manipulation using orthogonal planes. *Computers & Graphics*, 89:1–10, 2020. 2

[42] Difeng Yu, Xueshi Lu, Rongkai Shi, Hai-Ning Liang, Tilman Dingler, Eduardo Velloso, and Jorge Goncalves. Gaze-supported 3d object manipulation in virtual reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2021.

[43] Songpo Li, Xiaoli Zhang, and Jeremy D Webb. 3-d-gaze-based robotic grasping through mimicking human visuomotor function for people with motion impairments. *IEEE*

Transactions on Biomedical Engineering, 64(12):2824–2835, 2017. 2

[44] Andreas Bulling and Hans Gellersen. Toward mobile eye-based human-computer interaction. *IEEE Pervasive Computing*, 9(4):8–12, 2010. 2

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

[46] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. Eye tracking for everyone. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2176–2184, 2016. 2

[47] Keiko Sakurai, Mingmin Yan, Koichi Tanno, Hiroki Tamura, et al. Gaze estimation method using analysis of electrooculogram signals and kinect sensor. *Computational intelligence and neuroscience*, 2017, 2017. 2

[48] Chenglei Wu, Zhihao Tan, Zhi Wang, and Shiqiang Yang. A dataset for exploring user behaviors in vr spherical video streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, pages 193–198, 2017. 2

[49] Andreas Bulling. Pervasive attentive user interfaces. *Computer*, 49(01):94–98, 2016. 2

[50] Heiko Drewes, Alexander De Luca, and Albrecht Schmidt. Eye-gaze interaction for mobile phones. In *Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology*, pages 364–371, 2007. 2

[51] Emiliano Miluzzo, Tianyu Wang, and Andrew T Campbell. Eyephone: activating mobile phones with your eyes. In *Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds*, pages 15–20, 2010. 2

[52] Ken Pfeuffer and Hans Gellersen. Gaze and touch interaction on tablets. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, page 301–311, New York, NY, USA, 2016. Association for Computing Machinery. 3

[53] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. A 128× 128 120 db 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, 43(2):566–576, 2008. 3, 5

[54] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020. 3

[55] Fuyou Liao, Feichi Zhou, and Yang Chai. Neuromorphic vision sensors: Principle, progress and perspectives. *Journal of Semiconductors*, 42(1):013105, 2021. 3

[56] Shih-Chii Liu, Bodo Rueckauer, Enea Ceolini, Adrian Huber, and Tobi Delbruck. Event-driven sensing for efficient perception: Vision and audition algorithms. *IEEE Signal Processing Magazine*, 36(6):29–37, 2019. 3

[57] Tobi Delbruck and Manuel Lang. Robotic goalie with 3 ms reaction time at 4% cpu load using event-based dynamic vision sensor. *Frontiers in neuroscience*, 7:223, 2013. 3

[58] Martin Litzenberger, Bernhard Kohn, Ahmed Nabil Belbachir, Nikolaus Donath, Gerhard Gritsch, Heinrich Garn, Christoph Posch, and Stephan Schraml. Estimation of vehicle speed based on asynchronous data from a silicon retina optical sensor. In *2006 IEEE intelligent transportation systems conference*, pages 653–658. IEEE, 2006. 3

[59] Jun Haeng Lee, Tobi Delbruck, Michael Pfeiffer, Paul KJ Park, Chang-Woo Shin, Hyunsurk Ryu, and Byung Chang Kang. Real-time gesture interface based on event-driven processing from stereo silicon retinas. *IEEE transactions on neural networks and learning systems*, 25(12):2250–2263, 2014. 3

[60] Arnon Amir, Brian Taba, David J. Berg, Timothy Melano, Jeffrey L. McKinstry, Carmelo di Nolfo, Tapan Kumar Nayak, Alexander Andreopoulos, Guillaume J. Garreau, Marcela Mendoza, Jeffrey A. Kusnitz, Michael V. DeBole, Steven K. Esser, Tobi Delbrück, Myron Flickner, and Dharmendra S. Modha. A low power, fully event-based gesture recognition system. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7388–7397, 2017. 3

[61] Paul Rogister, Ryad Benosman, Sio-Hoi Ieng, Patrick Lichtsteiner, and Tobi Delbruck. Asynchronous event-based binocular stereo matching. *IEEE Transactions on Neural Networks and Learning Systems*, 23(2):347–353, 2011. 3

[62] Ryad Benosman, Charles Clercq, Xavier Lagorce, Sio-Hoi Ieng, and Chiara Bartolozzi. Event-based visual flow. *IEEE transactions on neural networks and learning systems*, 25(2):407–417, 2013. 3

[63] Matthew Cook, Luca Gugelmann, Florian Jug, Christoph Krautz, and Angelika Steger. Interacting maps for fast visual interpretation. In *The 2011 International Joint Conference on Neural Networks*, pages 770–776, 2011. 3

[64] Hanme Kim, Stefan Leutenegger, and Andrew J Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI 14*, pages 349–364. Springer, 2016. 3

[65] Anastasios N. Angelopoulos, Julien N.P. Martel, Amit P. Kohli, Jörg Conradt, and Gordon Wetzstein. Event-based near-eye gaze tracking beyond 10,000 hz. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2577–2586, 2021. 3

[66] Andy Kong, Karan Ahuja, Mayank Goel, and Chris Harrison. Eyemu interactions: Gaze+ imu gestures on mobile devices. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 577–585, 2021. 3, 4

[67] Juni Nurma Sari, Ridi Ferdiana, Paulus Insap Santosa, and Lukito Edi Nugroho. An eye tracking study: exploration customer behavior on web design. In *Proceedings of the International HCI and UX Conference in Indonesia*, CHIuXiD '15, page 69–72, New York, NY, USA, 2015. Association for Computing Machinery. 4

[68] Yuji Nozaki and Tobi Delbruck. Temperature and parasitic photocurrent effects in dynamic vision sensors. *IEEE Transactions on Electron Devices*, 64(8):3239–3245, 2017. 6

[69] Aaron Bangor, Philip T. Kortum, and James T. Miller. An empirical evaluation of the system usability scale. *International Journal of Human–Computer Interaction*, 24:574 – 594, 2008. 7

Supplementary Material

Supplementary Figures

- **Figure S1:** E^2GO Gaze Estimation Pipeline.
- **Figure S2:** The different eye scenarios.
- **Figure S3:** Schematic diagram of page division ratios.

Supplementary Tables

- **Table S1:** Specific values of E^2GO success rates in different postures.
- **Table S2:** Specific values of E^2GO task times in different postures.

Specific parameters of experimental devices

- **Vivo X80 Pro:**
  a 6.78-inch screen, a Qualcomm Snapdragon 8Gen1 processor, a 32MP front camera, 16GB of RAM, and a 4700mAh battery capacity.
- **Xiaomi 9:**
  a 6.39-inch screen, a Qualcomm Snapdragon 855 processor, a 20MP front camera, 6GB of RAM, and a 3300mAh battery capacity.
- **Huawei P30:**
  a 6.1-inch screen, a HiSilicon Kirin 980 processor, a 32MP front camera, and 8GB RAM and a 3650mAh battery capacity.

More results

E^2GO's test and demo videos can be found at Github.

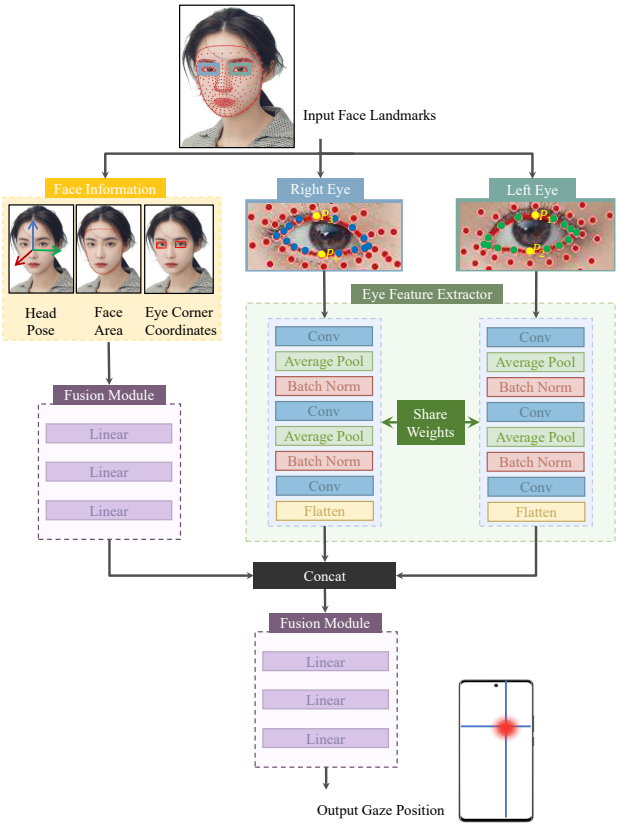
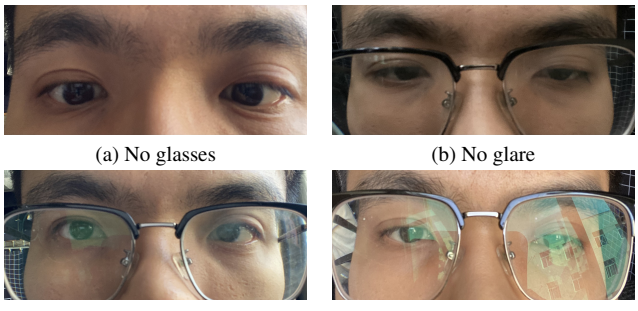


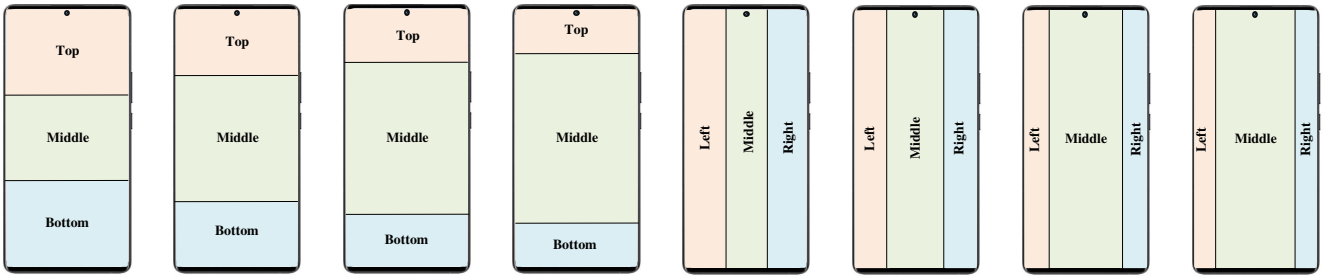
Figure S1. E^2GO **Gaze Estimation Pipeline.** The model intercepts the binocular image from the input image, extracts the facial feature information (head pose, face area, and eye corner coordinates), and gets the user's gaze position using the feature extraction and feature fusion modules.



(a) No glasses (b) No glare

(c) Mild glare (d) Severe glare

Figure S2. **The different eye scenarios. (a)** No glasses: Users test without glasses. **(b)** No glare: Testing without glare by users wearing glasses. **(c)** Testing with mild glare by users wearing glasses. **(d)** Testing with severe glare by users wearing glasses.



(a) T/M/B 1:1:1 (b) T/M/B 1:2:1 (c) T/M/B 1:3:1 (d) T/M/B 1:4:1 (e) L/M/R 1:1:1 (f) L/M/R 1:2:1 (g) L/M/R 1:3:1 (h) L/M/R 1:4:1

Figure S3. **The schematic diagram of page division ratios. (a)-(d) Swipe up/down action page display:** Top/Middle/Bottom (T/M/B) from 1:1:1 to 1:4:1; **(e)-(h) Switch to left/right action page display:** Left/Middle/Right (L/M/R) from 1:1:1 to 1:4:1.

Table S1. **Specific values of E^2GO success rates in different postures.**The data is expressed in the form of mean success rate (standard deviation): it refers to the mean success rates of the actions in different postures and the standard deviations for the 20 testers.

Postures	Sitting	Standing	Slouching	Lying	Walking	taking buses
Swipe Down	100% (0.000)	100% (0.000)	98% (0.300)	91% (0.589)	89% (0.589)	99% (0.218)
Swipe Up	100% (0.000)	100% (0.000)	99% (0.218)	94% (0.557)	90% (0.592)	100% (0.000)
Switch to Left	100% (0.000)	100% (0.000)	99% (0.218)	95% (0.433)	92% (0.583)	98% (0.300)
Switch to Right	100% (0.000)	100% (0.000)	98% (0.300)	93% (0.477)	93% (0.477)	97% (0.357)
Switch to Next	92% (0.490)	89% (0.589)	77% (0.654)	70% (0.500)	68% (0.490)	88% (0.583)
Switch to Previous	95% (0.433)	90% (0.500)	78% (0.700)	73% (0.654)	75% (0.622)	87% (0.572)
Click	93% (0.477)	90% (0.671)	84% (0.600)	83% (0.654)	80% (0.632)	86% (0.714)
Return	94% (0.458)	89% (0.497)	85% (0.622)	84% (0.678)	83% (0.654)	85% (0.766)

Table S2. **Specific values of E^2GO task times in different postures.** The data is expressed in the form of mean task time (standard deviation): it refers to the mean task times(s) of the actions in different postures and the standard deviations for the 20 testers.

Postures	Sitting	Standing	Slouching	Lying	Walking	taking buses
Swipe Down	0.133 (0.222)	0.135 (0.021)	0.145 (0.020)	0.191 (0.029)	0.197 (0.027)	0.143 (0.018)
Swipe Up	0.139 (0.021)	0.139 (0.016)	0.149 (0.017)	0.161 (0.025)	0.170 (0.024)	0.144 (0.015)
Switch to Left	1.071 (0.027)	1.072 (0.029)	1.093 (0.037)	1.092 (0.039)	1.111 (0.034)	1.077 (0.028)
Switch to Right	1.058 (0.027)	1.066 (0.035)	1.094 (0.041)	1.094 (0.039)	1.118 (0.038)	1.076 (0.034)
Switch to Next	2.106 (0.400)	2.325 (0.274)	2.530 (0.199)	2.664 (0.151)	2.698 (0.155)	2.331 (0.265)
Switch to Previous	2.164 (0.415)	2.250 (0.307)	2.458 (0.237)	2.663 (0.148)	2.711 (0.152)	2.255 (0.299)
Click	0.134 (0.017)	0.186 (0.064)	0.207 (0.056)	0.258 (0.066)	0.262 (0.068)	0.187 (0.057)
Return	0.136 (0.016)	0.165 (0.043)	0.212 (0.035)	0.253 (0.047)	0.257 (0.047)	0.165 (0.043)