Wearable Eye-Tracking System for Synchronized Multimodal Data Acquisition

Minqiang Yang, *Member, IEEE*, Yujie Gao, Longzhe Tang, Jian Hou, and Bin Hu, *Fellow, IEEE*

***Abstract*— Eye-tracking technology is extensively utilized in affective computing research, enabling the investigation of emotional responses through the analysis of eye movements. Integration of eye-tracking with other modalities, allows for the collection of multimodal data, leading to a more comprehensive understanding of emotions and their relationship with physiological responses. This paper presents a novel head-mounted eye-tracking system for multimodal data acquisition with a completely redesigned structure and improved performance. We propose a novel method for pupil-fitting with high efficiency and robustness based on deep learning and RANSAC, which gets better performance of pupil segmentation when it is partially occluded, and build a 3D model to obtain gaze points. Existing eye trackers for multi-modal synchronous data collection either have limited device support or suffer from significant synchronization delays. Our proposed hard real-time synchronization mechanism implements microsecond level latency with low cost, which facilitates multimodal analysis for affective computing research. The uniquely designed exterior effectively reduces facial occlusion, making it more comfortable for the wearer while facilitating the capture of facial expressions.**

***Index Terms*— Wearable eye tracker, eye movements, hard real-time synchronization, affective computing.**

I. Introduction

AS ONE of the most prominent features on the face, eye movements are often subconsciously generated by people and can reflect their attentional, cognitive, and emotional states [1], [2], [3]. Nowadays, eye tracking has been widely applied in psychology [4], [5], medicine [6], [7], neuroscience [8], [9], computer science [10], [11], [12], affective computing and other fields.

Manuscript received 4 July 2023; revised 19 September 2023 and 1 November 2023; accepted 6 November 2023. Date of publication 14 November 2023; date of current version 6 June 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFA0706200; in part by the National Natural Science Foundation of China under Grant 62227807; in part by the Natural Science Foundation of Gansu Province, China, under Grant 22JR5RA488; in part by the Fundamental Research Funds for the Central Universities under Grant lzujbky-2023-16; and in part by the Supercomputing Center, Lanzhou University. This article was recommended by Associate Editor A. Liu. *(Corresponding author: Bin Hu.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by the present study is approved by the Tianshui Third People's Hospital Medical Ethics Committee.

The authors are with the School of information Science and Engineering, Lanzhou University, Lanzhou 730000, China (e-mail: yangmq@lzu.edu.cn; bh@lzu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2023.3332814.

Digital Object Identifier 10.1109/TCSVT.2023.3332814

Eye-tracking technology is extensively employed in affective computing to precisely gauge and scrutinize human emotions [13], [14]. It offers a non-invasive and unbiased method of measuring emotional states, as it does not necessitate verbal or self-reported information from participants. Advancements in hardware technology led to a shift in the focus of eye-tracking technology towards developing high-precision and high-stability systems, while researchers propose various optimization algorithms related to eye-tracking [15]. Hansen and Ji [16] conduct comprehensive research on various eye models, eye detection techniques, and gaze estimation models for video-based eye-tracking algorithms. They also review the applications of these algorithms and discussed factors such as eye models, corneal refraction, and user glasses that could potentially result in errors in eye-tracking. Swirski proposes a robust real-time pupil recognition algorithm and 3D eye model fitting algorithm [17], which can accurately identify images with large offset angles of the pupil. The 3D eye model fitting algorithm does not require user calibration, and the calculated gaze vector accuracy is about two degrees [18]. Dan [19], [20] and Kar [21] propose a polynomial fitting model suitable for two-dimensional scenes, which uses the characteristics of the human eye based on multivariate regression to derive the mapping relationship between the center of the pupil and the true visual direction. They use different correction algorithms to improve the accuracy and robustness of the algorithm. However, these technologies are susceptible to head movement during testing, requiring subjects to use headrests, chinrests, or bite bars to keep their heads still. Moreover, these technologies only apply to two-dimensional environments, and when the scenario extends to three-dimensional space, the model's performance will decrease significantly. To address this issue, Mansourya [22] propose a 3D gaze prediction algorithm based on 2D pupil position, which directly maps the 3D gaze point prediction task by mapping the 2D pupil location in the pupil camera image to the 3D gaze direction in the scene camera. Huang et al. [23] design a single-camera single-light source eye-tracking system by constructing a binocular eye model to estimate gaze. By calculating the 3D eye model's gaze from the 2D pupil position, the method can predict the 3D fixation point to some extent. With the widespread applications of machine learning and deep learning techniques, gaze-tracking algorithms undergo developments in terms of accuracy and efficiency, effectively driving the innovation of gaze-tracking technology. Wang [24] proposes an improved DLSR-ANN method based on direct

least squares regression for 2D gaze estimation. Cornia [25] and Yuan [26] use the Attention network for Gaze Prediction. Wu [27] proposes a Modulation-based Adaptive Network (MANet), which is the first method utilizing high-level cues of the eye-specific regions to modulate face features in the gaze estimation task.

Head-mounted eye trackers, unlike desktop eye trackers, provide greater freedom of head and body movement of subjects, which makes them more suitable for affective computing applications. Since the development of a portable head-mounted eye tracker by Land and Lee in 1994 [28], many companies commercialize eye-tracking devices, such as Tobii Pro Glasses 3, Pupil Core, and Pupil Invisible [29], [30]. These devices achieve remarkable accuracy, with gaze estimation accuracy of less than one degree. There are a number of papers that test these devices in a variety of ways [31], [32]. However, despite the availability of many commercial eye trackers and accompanying software on the market, there are still several issues that need to be addressed: External factors such as lighting, occlusion by the eyelid or pupil, and blinking can easily affect the eye-tracking algorithm and cause errors, thereby impacting its robustness. The lack of synchronization between eye movements and facial expressions during data collection can cause misalignment of data, which could potentially affect the accuracy of the results and make it unsuitable for multimodal fusion research. Additionally, the current eye-tracking devices may occlude facial landmarks and result in the loss of facial data.

Based on the issues mentioned above, this article proposes an eye-tracking system that includes wearable eye-tracking hardware equipment, algorithms, software, and other components. We redesign the structure of the eye tracker with the goal of minimizing facial occlusion and optimizing the multimodal synchronous acquisition process, which facilitates the collection of high-quality multimodal data. The eye camera can capture images at a high frame rate of 480 frames per second (fps), providing more detailed information about ocular movements, such as microsaccades. In addition, the system features infrared light and filters to effectively reduce environmental interference from ambient light. Moreover, the use of a gyroscope enables the adjustment of gaze point, reducing the influence of head movement on gaze point. The system also ensures flexibility, portability, foldability, stability, and user comfort for use in various scenarios. In summary, our main contributions are listed as follows:

(1) We propose a pupil-fitting method with high efficiency and robustness based on deep learning and Random Sample Consensus (RANSAC) [33]. It utilizes deep learning and clustering to filter out non-pupil edge points in the images and performs pupil ellipse complementation on closed-eye images. It also contains a pupil-fitting algorithm based on the RANSAC algorithm with higher accuracy, speed, and robustness. We design a long-short queue updating algorithm to determine the center and radius of the eyeball, serving as the foundation for constructing a three-dimensional model for computing the optical axis vector.

(2) We propose a low-cost, hard real-time synchronization solution based on PREEMPT_RT [34], [35], which significantly reduces latency between processes and greatly benefits the synchronization of multimodal data collection. Our proposed solution supports the Linux operating system, reducing the overhead of software porting.

(3) We design three styles of wearable eye-tracking devices tailored to different usage scenarios, substantially reducing facial obstructions and facilitating the synchronized collection of facial expressions and eye movement data.

TABLE I
DESCRIPTION OF SOME NOTATIONS

Symbols	Description	Reference
$\bar{d}$	The average distance within class.	Equation 3
$\mathbf{v}_{3d}$	The optical axis.	Equation 14
$\mathbf{K}$	The internal reference matrix of camera.	Equation 15 and 17
$\mathbf{R}$	The rotation matrix.	Equation 16
$\mathbf{T}$	The translation matrix.	Equation 16
$\mathbf{v}_{gl}$	Left eye's visual axis vector.	Equation 18 and 19
$\mathbf{v}_{gr}$	Right eye's visual axis vector.	Equation 18 and 19
$g(o, \kappa)$	The gaze point.	Equation 20
loss_a	Error in the long axis of the ellipse.	Table II
loss_b	Error in the short axis of the ellipse.	Table II
loss_d	Error in the center point of the ellipse.	Table II
SSE	Within-Cluster Sum of Square Error.	Tabl III
RME	Root Mean Square.	Table III
$\Delta\theta$	The error angle.	Equation 25

II. UEYE EYE TRACKING SYSTEM

We summarize the key symbols used in this paper with a notation table for better comprehension. We use bold lowercase letters to denote vectors, and bold uppercase letters to denote matrices.

Eye-tracking devices used today for multimodal data collection generally suffer from two problems: limited device support and significant synchronization delays. Head-mounted eye trackers, unlike desktop eye trackers, provide greater freedom of head and body movement of subjects, which makes them more suitable for affective computing applications. However, they often result in greater obstruction of critical facial areas, which hinders the synchronization of facial expressions and eye movements. We present a novel eye-tracking system with a completely redesigned structure and improved performance parameters.

We design a unique eye tracker appearance that effectively reduces facial obstruction, which makes it possible to collect facial data and eye movement data at the same time. By using a hard Real-Time Operating System (RTOS), we achieve synchronized collection of multimodal data, improving the quality and reliability of data collection. In addition, with the addition of infrared lights and filters, we decrease the impact of environmental lighting on data while ensuring the safety and health of subjects. A high frame rate camera enables the collection of more detailed data, including microsaccades and saccades, while the inclusion of a gyroscope module allows for gaze point adjustment, minimizing the impact of head movements on the gaze point.

We use deep learning and clustering to screen the pupil edge points in the image and develop special processing methods to handle half-closed eye scenarios, improving the accuracy and speed of pupil-fitting by refining the RANSAC algorithm. Furthermore, our gaze tracking algorithm demonstrates superior performance in terms of faster execution speed, enhanced robustness, and uncompromised accuracy.

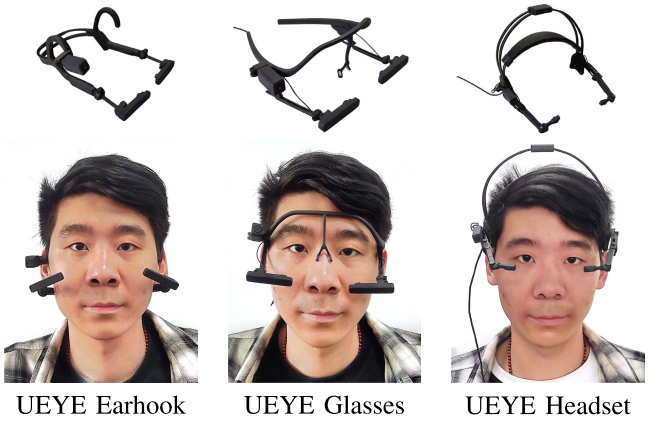


Fig. 1. Appearance of proposed three eye trackers.

We develop three types of eye trackers, i.e., UEYE Earhook, UEYE Glasses, and UEYE Headset, Fig. 1 illustrates their wearing styles respectively. In this article, we will focus on the UEYE Headset to present our work.

A. Appearance Design

While high-performance head-mounted eye trackers are available, they often obstruct the important facial landmarks of subjects, which hinders the multimodal study involving facial behavior. In this article, we design three novelty wearable eye trackers with different mounting structures to minimize the obstruction of the subject's face, which are shown in Fig. 2. This approach also improves the comfortability and makes it easier to conduct multimodal studies involving facial expressions and eye movements.

The UEYE Headset resembles a headset and is equipped with three cameras: one scene camera which captures the world, and two eye cameras. Apart from the eye cameras, the eye tracker does not have any other structure that obstructs the face. Moreover, the size of the eye cameras is small, with a width of 0.5cm and a length of 1.2cm, which preserves facial information to the greatest extent possible. This design is suitable for collecting multimodal data of eye movements and facial expressions, as well as ensuring the subject's comfort.

For eye-tracking experiments, it is important that the eye-tracking device has a stable structure and does not slide during the experiment. The outer ring of the device serves as the overall framework, ensuring the stability of the structure and fixing the cameras and gyroscopes. The elastic strap is retractable to accommodate subjects with different head sizes and automatically tightens after being worn. Additionally, there are semi-circular brackets on both sides that match the shape of the ears, which fixes the device to the ears to prevent slipping during experiments.

Considering the potential requirement for prolonged wearing during experimental sessions, it is crucial for the eye tracker to be lightweight and comfortable for the subjects. Therefore, we choose nylon material with high precision and elasticity in the design to reduce the weight of the device and prevent excessive pressure on the subject.

The eye tracker also features multiple flexible joints, allowing for a wide range of camera rotation to precisely record eye movement data for different subjects and experimental scenarios. The eye cameras have a horizontal adjustment range of $\pm 30°$ and a vertical adjustment range of $\pm 45°$. The extension arm of the eye camera is extendable and rotatable to accommodate various head shapes and foldable for easy storage, avoiding damage to the eye camera and its structure. The scene camera's horizontal adjustment range is $\pm 27°$, and the vertical adjustment range is $\pm 53.5°$. It has been verified that the above adjustment ranges are competent in most scenarios through multiple experiments.

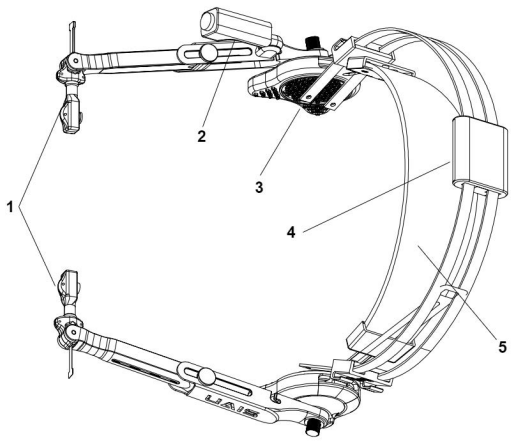


Fig. 2. Structure diagram of UEYE Headset. 1) Eye camera, 2) Scene camera, 3) Semi-circular bracket, 4) Gyroscope, 5) Elastic strap.

B. Hardware Design

1) Hardware Layout: The proposed eye tracker adopts a USB slave scheme, it equips one scene camera and two eye cameras which are responsible for the synchronized capturing of ocular images and encoding. Our proposed eye tracker features infrared light and a gyroscope sensor. The infrared light helps to reduce environmental interference from ambient light, while the gyroscope enables the adjustment of gaze point, reducing the influence of head movement on gaze point. All devices are connected to the host via USB protocol and powered and transmit information through USB. The hardware structure diagram is shown in Fig. 3.

Our eye-tracking device is designed with the following hardware specifications: an eye camera with 480 fps, a resolution of 320 $\times$ 240 pixels, and a field of view of 55° vertically and 70° horizontally; a scene camera with a frame rate of 30 fps, a resolution of 1920 $\times$ 1080 pixels, and a field of view of 52° vertically and 68° horizontally. Additionally, the device features an 850nm near-infrared light source and a corresponding 850nm filter.

2) Elimination of Ambient Light Interference: Infrared light sources play a very important role in the eye tracker. In addition to serving as supplementary lighting for the eye camera and mitigating the impact of ambient light sources, infrared light sources play a crucial role in producing a dark pupil, facilitating the identification of corneal reflections. This can be used for algorithmic research through the pupillary corneal reflection method. We have strict restrictions on the wavelength and power of infrared light sources to ensure the health and safety of subjects. The eye tracker developed in this

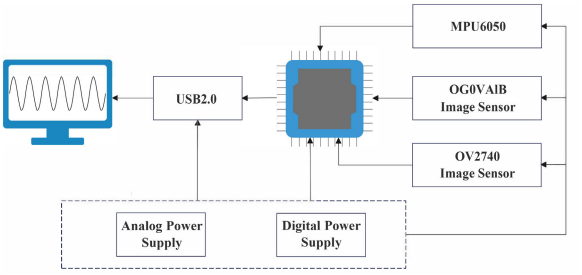


Fig. 3. Hardware structure diagram.

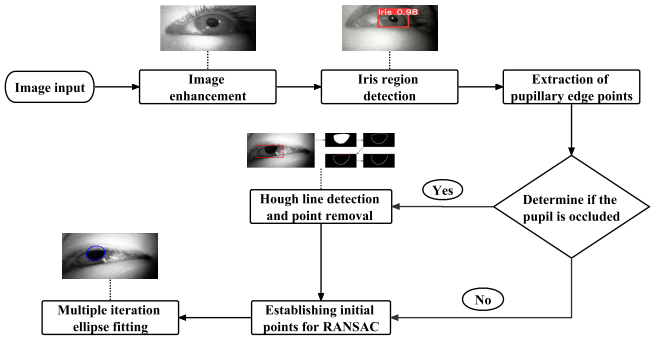


Fig. 4. Algorithm flowchart.

paper has two infrared light sources with a wavelength of 850nm installed on both sides of the eye camera. We also add a corresponding 850nm wavelength optical filter to achieve better results.

3) The Gyroscope Module: As we all know, head movement may affect the accuracy of gaze point estimation. We integrate an MPU6050 gyroscope chip that integrates a 3-axis gyroscope and a 3-axis accelerometer in the eye tracker. This allows us to measure the head movements of subjects during experiments and adjust the gaze point to reduce the influence of head movement.

C. Pupil Recognition Algorithm

During the pupil recognition process, we exclusively focus on the image of the pupil region, so we use the YOLO network to segment the region of pupils. Additionally, we improve the accuracy and fitting speed by introducing a filtering mechanism for edge point sets and implementing completion of pupil edge points in the presence of occlusion, based on the RANSAC [33]. The flowchart of the algorithm is shown in Fig. 4.

1) Deep Learning-Based Iris Region Segmentation: The pupil extraction algorithm proposed in this paper mainly relies on the edge point set and the pupil-fitting algorithm, which require the extraction of edge information from the eye image. However, there are many edge points in the eye image that do not belong to the pupil edge, such as eyelashes and corners of the eyes. Therefore, it is necessary to segment the pupil area to avoid their impact on the fitting speed and robustness of the algorithm.

The frame rate of the eye camera can reach up to 480 fps, which means that the image exposure time is short. Besides, due to the harmful effects of infrared light on the human eye, the illumination intensity is restricted in a safety range. These factors lead to darker images captured by the eye camera. The low contrast of the image can greatly affect the results of subsequent edge recognition algorithms. Before iris segmentation, we enhance the image with histogram equalization [36] to improve the accuracy and robustness of the subsequent algorithm.

We employ a deep neural network based on Yolov5s [37] for iris segmentation. It has the smallest depth and feature map width among the Yolov5 series networks, making it the fastest to train and detect, but with the lowest average precision (AP) for a single-class model. Considering that the pupil segmentation scene in this article is single and the segmented area is relatively large, it is acceptable to sacrifice some recognition accuracy to improve model efficiency. Then we use Labelimg to manually annotate 3000 images for the model's training set.

2) Pupil Edge Point Extraction and Filtering: The edge information of an object is mainly concentrated in the high-frequency band, i.e., pixels with significant gradient changes in the surrounding pixel values. Based on this property, the basic idea of edge extraction is to calculate the gradient changes of the gray values in the image, typically using Sobel operators to calculate the horizontal and vertical gradient changes in the image.

After performing the aforementioned calculations, gradient information for each pixel point can be obtained. Subsequently, the entire image is scanned to determine whether the gradient intensity of a pixel point is the maximum among the pixels with the same direction in its vicinity. If the gradient intensity of the point is the highest, it is considered a candidate edge point. To determine whether the point is an edge point, two thresholds, i.e., $LowerBound$ and $UpperBound$, need to be set for screening. If the gray gradient G of the image is greater than $UpperBound$, the pixel point is determined to be an edge point. When G is less than $LowerBound$, the pixel point is not an edge point. For pixels between the two thresholds, being connected to pixels that have already been identified as true edge points in the neighborhood is used as a criterion for further determination. If they are connected, then the pixel point is also regarded as an edge point.

The edge points obtained in this way contain both pupillary and non-pupillary edge points, so it is necessary to distinguish them. It is known that the edge of the pupil in the image is a continuous closed ellipse. Therefore, considering the edge points of the image as a sample set, clustering is performed based on the closeness of the sample distribution, and tightly connected samples are partitioned into one class.

$$Distance(p,q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2} \tag{1}$$

$$q \in Cluster_i \Leftarrow Distance(p,q) \leq threshold \cap p \in Cluster_i \tag{2}$$

Here, $Distance(p,q)$ represents the Euclidean distance between two points p and q. When it is less than the threshold, they are considered to belong to the same $Cluster_i$ category.

Now we have obtained a set of pupil edge points and multiple sets of non-pupil edge points. The non-pupil edge point set is more discrete. We implement an adaptive filtering

Algorithm 1 Pupil Edge Point Filtering Algorithm

Require: Set of edge points $edge\_points$, Threshold distance $threshold$, Threshold ratio $rate$, Maximum number of iterations $max\_iterations$

Initialize $iterations$ as 0, Initialize set $line\_points$ as an empty set
while $iterations < max\_iterations$ **do**
  Calculate the intersect point (ρ_i, θ_i) for points in $edge\_points$
  Fit a line $y = ax + b$ using the point (ρ_i, θ_i)
  for each point (x, y) in $edge\_points$ **do**
    Calculate distance d to the line $y = ax + b$
    if $|d| \leq threshold$ **then**
      Add (x, y) to set $line\_points$
    end if
  end for
  if size($line\_points$) is not increasing or size($line\_points$) $>$ $rate*$ size($edge\_points$) **then**
    if size($line\_points$) > 0 **then**
      Remove points in $line\_points$ from $edge\_points$
      Reset $iterations$ to 0
    else
      Break
    end if
  else
    Increment $iterations$ by 1
  end if
end while

approach for excluding non-pupil edge points by utilizing calculations involving the dispersion degree of the samples and the ratio of samples within a set to the total samples. The dispersion degree is represented by the coefficient of dispersion and the $Pearson$ coefficient.

$$\bar{d} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2} \tag{3}$$

$$CV = \frac{\sum_{i=1}^{N}(\sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2} - \bar{d})^2}{\sqrt{N} * \bar{d}} \tag{4}$$

$$\rho = \frac{cov(X, Y)}{\sigma_x \sigma_y} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}} \tag{5}$$

The pupil may be obstructed by the eyelids in situations where a person blinks or has a semi-closed eye state, resulting in eyelid edges being included in the extracted pupil edge. We propose a non-pupil edge point filtering algorithm based on line detection to optimize the algorithm. Our Algorithm 1 primarily utilizes the Hough transform to detect points that lie on a straight line.

3) Elliptical Pupil-Fitting: The edge point filtering algorithm mentioned above states that when the pupil is covered by the eyelid, non-pupil edge points will be adaptively removed. As a result, the pupil edge points form a non-closed ellipse. So we provide a method to complete the missing pupil edge points.

It is known that an ellipse has rotational invariance, meaning that there exist two points, i.e., $p_1(x_1, y_1)$ and $p_2(x_2, y_2)$, on the edge of the ellipse that is symmetric with respect to the center point (x_0, y_0) of the ellipse.

$$(x_0, y_0) = \left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2}\right) \tag{6}$$

We can estimate the possible center of the ellipse using the gray value of the image. Since the pupil appears black in the image, a negation operation is performed resulting in black having a grayscale value of 255 and white having a grayscale value of 0. The grayscale values of each row and column are then accumulated resulting in a higher accumulation in the pupil region compared to other regions. By calculating the local maximum value, the centroid of the pupil can be found, which corresponds to the possible center of the ellipse. The formulas are as follows:

$$\arg\max_{w\in[0,W]} f(w) = \{w \in [0, W] : f(w) = \sum_{i=1}^{H} I(w, i)\} \tag{7}$$

$$\arg\max_{h\in[0,H]} f(h) = \{h \in [0, H] : f(h) = \sum_{j=1}^{W} I(j, h)\} \tag{8}$$

RANSAC is a general parameter estimation method proposed by Fischler and Bolles [33]. It is used to solve the problem of a large proportion of outliers in the input data. Unlike traditional sampling techniques, RANSAC uses a minimal number of data points and expands it by using consistent data points to continue fitting the model [38]. RANSAC is able to effectively identify and handle outliers, thereby improving the robustness and accuracy of the model.

An improved RANSAC ellipse fitting algorithm with adaptive edge point selection is proposed in this paper. The model fitting speed is related to the initial point selection of the algorithm in the original RANSAC algorithm, which leads to some problems in model efficiency and stability. So we adapt the selection of initial points to achieve adaptive selection and improve the accuracy, efficiency, and robustness of the algorithm. The algorithm process is as Algorithm 2.

$$error(Q, x, y) = \alpha\frac{Q(x, y)}{|\nabla Q(x, y)|} \tag{9}$$

$$inliers = \{(x, y)|error(Q, x, y) < \varepsilon\} \tag{10}$$

Here the error weight α has two specificities. First, compared with the real edge points, the completed edge points have a lower value of α. Second, there is a lower tolerance for edge points with large grayscale gradient changes means a higher value of α, making the ellipse fitting more biased towards the edge parts with large gradient changes.

By incorporating adaptive edge point selection, the improved RANSAC ellipse fitting algorithm is able to better handle varying pupil shapes and sizes in a more efficient and accurate manner.

Algorithm 2 Pupil Ellipse Fitting Algorithm

Require: The center of the ellipse $O(x, y)$, Set of edge points $edge\_points$, Minimum number of interior points $threshold$, Error threshold ε, Number of interior points n, Maximum number of iterations $max\_iterations$

Initialize $iterations$ as 0, Initialize $Q(x, y)$ as an elliptic equation
With O as the origin divide $edge\_points$ into four parts and randomly select one point from each part
Randomly select one point from real pupil edge points
Add the five selected points to $input\_data$
Fit the ellipse $Q(x, y)$ using $input\_data$
while $iterations < max\_iterations$ **do**
  Initialize set $input\_data$ as an empty set
  for each edge point (x, y) in $edge\_points$ **do**
    Calculate error e with Equation 9
    if $e < \varepsilon$ **then**
      Add (x, y) to $input\_data$
    end if
  end for
  Re-fit ellipse $Q(x.y)$ using updated $input\_data$
  Increment $iterations$ by 1
  Calculate the number of interior points n with Equation 10
  if $n > threshold$ **then**
    Break
  end if
end while
return $Q(x, y)$

D. Three-Dimensional Eyeball and Gaze Tracking Model Based on Perspective Projection

We use a gaze prediction model based on a three-dimensional gaze vector that aims to improve the accuracy of gaze prediction by constructing a three-dimensional model of the two-dimensional pupil image and calculating the 3D eyeball model based on perspective projection. This algorithm includes three steps, i.e. estimation of the optical axis vector, calculation of the 3D eyeball model, and calibration.

1) Three-Dimensional Eyeball Model: The 3D eyeball model used in this paper is based on the two-sphere model initially proposed by Le Grand [39]. The two-sphere model is a model that describes the shape of the eyeball, which approximates the geometric shape of the eye and consists of two spherical parts: the eyeball and the pupil sphere. Various parameters of the eye, such as the center of the pupil, the center of corneal curvature, the optical axis, and the visual axis, are usually required in gaze estimation [40]. The pupil, located at the center of the iris, appears black and allows external light to enter [41]. The line connecting the fovea and the center of corneal curvature is called the visual axis, and the vector passing through the centers of the two spheres is defined as the optical axis. The visual axis determines the direction of gaze, and there is a certain angle between the visual axis and the optical axis, which is defined as the kappa angle ($kappa$) and has an angle value of about 5° [42]. In gaze estimation, the horizontal and vertical components of $kappa$ are fixed for each individual, and the visual axis cannot be directly estimated. Therefore, the visual axis and $kappa$ angle of each individual must be obtained through a calibration process to complete the modeling of the 3D eyeball model.

In the two-sphere model, the optical axis can be explained as the connecting line between the two centers of the spheres. Given the fitted ellipse of the pupil in the 2D image, to obtain the optical axis $\mathbf{v}_{3d} = (v_x, v_y, v_z)$, the projection $\mathbf{v}_{2d} = (v'_x, v'_y)$ of the optical axis on the 2D plane needs to be obtained first. The pupil image used as input data has undergone lens distortion correction during acquisition; therefore, the two-sphere model discussed in this paper only considers orthogonal projection in the process of 2D to 3D transformation.

$$R = \frac{d}{\sin\theta} \tag{11}$$

$$\theta = \arccos\frac{b}{a} \tag{12}$$

The 3D optical axis vector is projected onto the 2D image by passing through both the center of the eyeball and the center of the pupil in 2D. Its direction is the same as the short axis of the fitted ellipse of the pupil. According to the rotational invariance of a sphere, these two conditions remain true during pupil movement, while the coordinates and radius of the eyeball remain unchanged. Therefore, an iterative updating algorithm based on a long and short queue is designed to calculate the 2D center coordinates and radius of the eyeball, as is shown in Algorithm 3.

The algorithm described above obtains the coordinates (x_e, y_e) and radius R of the two-dimensional projection of the eyeball on the sphere. Assuming that the center of the eyeball is located on the xoy plane of the three-dimensional coordinate system, we can determine the three-dimensional coordinate equation of the eyeball: $(x - x_e)^2 + (y - y_e)^2 + z^2 = R^2$.

Therefore, we can define the three-dimensional coordinate equation of the pupil sphere as $(x-i)^2+(y-j)^2+(z-k)^2 = r^2$. Simultaneous the equations of the eyeball and the pupil sphere:

$$\begin{cases} (x - x_e)^2 + (y - y_e)^2 + z^2 = R^2 \\ (x - i)^2 + (y - j)^2 + (z - k)^2 = r^2 \end{cases} \tag{13}$$

We solve the equations of the two spheres simultaneously and then find the plane of intersection between the two spheres. Once we have the plane of intersection, we can substitute it into either sphere equation to obtain the projected equation of the intersectant curve on the xoy plane.

This projected curve equation is the ellipse fitting equation from previous calculations. And we know that the minor axis of the ellipse passes through the projection of the pupil sphere center on the xoy plane.

Based on the elliptic equation and the short axis equation, we can obtain the center of the pupil sphere (x_p, y_p, z_p). Then we combine it with the previously calculated coordinate of the eyeball center to obtain the optical axis of the two-sphere model. At this point, the modeling of the eyeball two-sphere

Algorithm 3 Stable Eye Radius Calculation Algorithm

Require: Video frames of the pupil, Initial eyeball center $O(x_0, y_0)$, Lengths of the long and short queues n_l and n_s, Efficiency of iterative updates α

Initialize queues Q_l and Q_s
while Video frames are not processed completely **do**
  if Length of Q_s is n_s **then**
    Calculate the mean value of the elements in Q_s as the short-term eyeball radius $\bar{R}'$ and add it to the long queue Q_l
    Remove half of the elements in Q_s that significantly differ from $\bar{R}'$
  else
    Calculate the equation of the short axis l: $y = kx + b$ with the pupil ellipse and find point $O'(x, y)$ on this line
    Let $OO' \perp l$ and take a point A on the line OO' that $OA = \alpha OO'$
    Take point A as the new center point O of the eyeball.
    Calculate the Euclidean distance d between OO'
    Calculate eyeball radius R' using Equations 11 and 12:
    Add R' to the short queue Q_s
  end if
  if Length of Q_l is n_l **then**
    Calculate the mean value of the elements in Q_l as the long-term eyeball radius
    When a new element is added to Q_l, remove the element with the largest difference from the average value
  end if
end while
return $O(x_0, y_0)$, R

model is completed.

$$\mathbf{v}_{3d} = (x_p - x_e, y_p - x_e, z_p) \tag{14}$$

2) Gaze Point Prediction Model Based on Three-Dimensional Gaze Vector: The model based on three-dimensional gaze vectors involves modeling the human eye and gaze target in three-dimensional space coordinates, fixing the positional relationship between the two eyes in space, and calculating the visual axis vector and its intersection point in three-dimensional space.

However, achieving this requires a coordinate system transformation. In this paper, three coordinate systems are defined: the image coordinate system, the camera coordinate system, and the world coordinate system. The image coordinate system is a two-dimensional Cartesian coordinate system about the image plane, represented in pixels as (u, v), while the camera coordinate system describes the three-dimensional spatial position of the image with respect to the camera, represented by (x_c, y_c, z_c). The world coordinate system, also known as the measurement coordinate system, is a three-dimensional Cartesian coordinate system that can be used as a reference to describe the spatial position of a camera and an object being measured. The coordinates are represented as (x_w, y_w, z_w), and the position of the world coordinate system can be freely determined based on the actual situation. It is important to note that the origin and rotation direction of the world coordinate system differ from those of the camera coordinate system. Since the above three-dimensional models are obtained in the image coordinate system, the eye model needs to be transformed twice to obtain a model in the world coordinate system and estimate the gaze point in this coordinate system. Fig. 5 illustrates the relationship between the camera coordinate system and the image coordinate system.

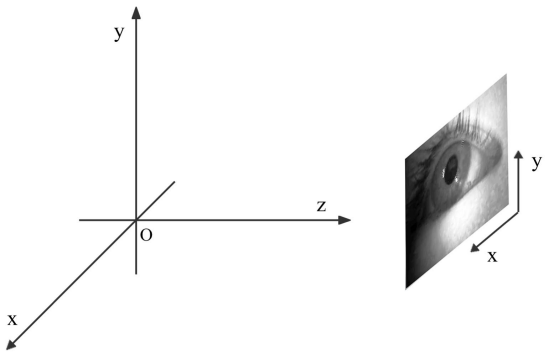


Fig. 5. Camera coordinate system and image coordinate system.

According to the principle of the pinhole camera model, light from an object passes through the aperture and forms an inverted image on the camera's image plane. As a result, the origin of the camera coordinate system is located at the optical center of the camera, with the x-axis and y-axis parallel to the x-axis and y-axis of the image coordinate system, respectively. The z-axis is perpendicular to the xoy plane and points outward. Given these differences, the transformation between the image coordinate system and the camera coordinate system can be defined using perspective projection. This is shown in Equation 15:

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{d_x} & 0 & u_0 \\ 0 & \frac{f}{d_x} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{15}$$

Here, $\mathbf{K}$ represents the camera's intrinsic matrix, where f is the focal length and d_x and d_y represent the length in actual that each pixel on the x and y axes corresponds to. u_0 and v_0 denote the offsets of the image coordinate system relative to the center of the camera coordinate system, ideally corresponding to half of the image's width and height under ideal conditions.

In addition, the transformation between the camera coordinate system and the world coordinate system mainly involves translation and rotation, which can be expressed by the following formula in Equation 16:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \mathbf{R} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \mathbf{T} \tag{16}$$

Here, $\mathbf{T}$ is the rotation matrix and $\mathbf{T}$ is the translation matrix from the camera coordinate system to the world coordinate system.

Based on the previous discussions on the transformation between the different coordinate systems, the transformation between the world coordinate system and the image pixel coordinate system can be expressed as shown in Equation 17:

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{R}_{3*3} & \mathbf{T}_{3*1} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \mathbf{P}_{3*4} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{17}$$

Then we can obtain the unit vector of the optical axis in the world coordinate system. However, it is important to note that the optical axis may not accurately reflect the subject's gaze direction. Therefore, it is necessary to model the position of the binocular visual axis and determine the kappa angle through calibration.

First, a simple visual axis and optical axis angle model is defined where the unit vector of the optical axis is $\mathbf{v}_o = [0\ 0\ 1]^T$, and the angle between the visual axis and the optical axis is represented as $\kappa = (\alpha, \beta)$. Then, the left and right eye's visual axis vectors can be expressed as shown in Equation 18:

$$\mathbf{v}_{gl} = \begin{pmatrix} \cos(\beta)\sin(\alpha) \\ \sin(\beta) \\ -\cos(\beta)\cos(\alpha) \end{pmatrix}, \mathbf{v}_{gr} = \begin{pmatrix} \cos(\beta)\sin(\alpha) \\ \sin(\beta) \\ \cos(\beta)\cos(\alpha) \end{pmatrix} \tag{18}$$

Expressing the optical axis in terms of horizontal and vertical angles, i.e.θ and φ, the above Equation 18 can be extended to Equation 19:

$$\mathbf{v}_{gl} = \begin{pmatrix} \cos(\varphi+\beta)\sin(\theta+\alpha) \\ \sin(\varphi+\beta) \\ -\cos(\varphi+\beta)\cos(\theta+\alpha) \end{pmatrix}, \quad \mathbf{v}_{gr} = \begin{pmatrix} \cos(\varphi+\beta)\sin(\theta+\alpha) \\ \sin(\varphi+\beta) \\ \cos(\varphi+\beta)\cos(\theta+\alpha) \end{pmatrix} \tag{19}$$

Finally, we can compute the intersection point of the left and right eye's visual axis vectors in space to estimate the subject's current gaze point. The three-dimensional coordinate of the gaze point is defined as $g(o, \kappa) = (g_x, g_y, g_z)$, where $o = (o_x, o_y, o_z)$ represents the three-dimensional coordinate of the eyeball. The gaze point $g(o, \kappa)$ can be determined using Equation 20:

$$\begin{aligned} g(o,\kappa) &= o_l + k_l \cdot \begin{pmatrix} \cos(\varphi_l+\beta_l)\sin(\theta_l+\alpha_l) \\ \sin(\varphi_l+\beta_l) \\ -\cos(\varphi_l+\beta_l)\cos(\theta_l+\alpha_l) \end{pmatrix} \\ &= \emptyset_r + k_r \cdot \begin{pmatrix} \cos(\varphi_r+\beta_r)\sin(\theta_r+\alpha_r) \\ \sin(\varphi_r+\beta_r) \\ \cos(\varphi_r+\beta_r)\cos(\theta_r+\alpha_r) \end{pmatrix} \end{aligned} \tag{20}$$

In the above-mentioned model, we use corneal reflection to constrain the three-dimensional gaze point model. The hardware device used in this paper is a dual-camera dual-light source, which produces a corneal reflection spot on the left and right pupil images, respectively. The following constraint equations, shown in Equations 21 and 22, are obtained through the law of reflection. Here, q_l,q_r represent the reflection spots on the left and right corneal surfaces, respectively. Similarly, o_l,o_r represent the coordinates of the centers of the left and right eyes, respectively. l_l,l_r represent the positions of the infrared light sources in space, and c is the optical center of the camera.

$$(l_l - c) \times (q_l - c) \bullet (o_l - c) = 0 \tag{21}$$

$$(l_r - c) \times (q_r - c) \bullet (o_r - c) = 0 \tag{22}$$

However, because the kappa angles of different subjects are different, calibration needs to be estimated in advance. In the individual calibration process, subjects are required to focus on N calibration points on the screen: $g_i^*, i = 1, 2, \ldots, N$. Then, the model parameters can be optimized by minimizing the distance between the predicted gaze point and the actual fixation point, as is shown in Equation 23:

$$\kappa^* = \arg\min_{\kappa} \sum_i \left\| g_i - g(o_i^*, \kappa) \right\| \tag{23}$$

Here, g_i represents the actual coordinates of the calibration point, and $g(o_i^*, \kappa)$ represents the predicted gaze point coordinates. A five-point calibration was used in the experiment to optimize the model parameters, determine the kappa angle for each subject, and build a 3D gaze-tracking model.

E. Hard Real-Time Synchronization Mechanism

The synchronous collection of multimodal data is the foundation of multimodal fusion analysis. Therefore, we need to ensure low latency between different modalities of data as well as between paradigm and data to achieve synchronization as much as possible. The precise control of timing and synchronization for input and output signals is typically achieved through computer software such as E-Prime [43]. However, these software have limited support for devices [44] and always require specialized hardware devices and physical connections, which is costly. General Purpose Operating System (GPOS) based process synchronization may be affected by multiple factors including operating system architecture, preemptive scheduling, multitasking processing, etc. These factors may lead to huge latencies, resulting in reduced efficiency and performance of synchronization operations.

In order to address the issues mentioned before, we propose a hard real-time synchronization mechanism based on PREEMPT_RT RTOS, which effectively guarantees deterministic synchronization latency for real-time tasks. The PREEMPT_RT patch on the Linux kernel, developed by a group of kernel developers [34], [35], optimizes mechanisms such as process scheduling, interrupts, and signals. It can meet hard real-time requirements and has been widely used [45], [46]. The main advantage of PREEMPT_RT over other hard RTOSs is that it is compatible with various Linux distributions and can use the development and execution environments of existing Linux distributions, reducing the amount of work involved in developing and porting applications. By modifying the Linux kernel and adding the PREEMPT_RT patch, we can artificially put process switching in a controllable state. High-priority tasks, by increasing the process priority, can preempt low-priority tasks, allowing our four camera processes to maintain a stable and synchronized state, which increases predictability and reduces maximum latency differences. First, load the main process to initialize related resources and start child processes then increase their priorities respectively and

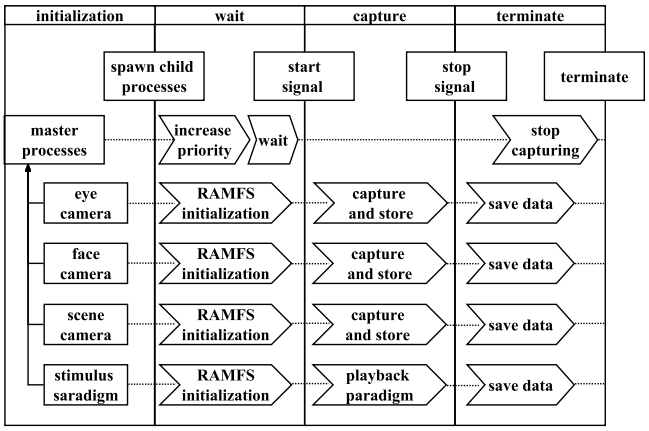


Fig. 6. Synchronization mechanism diagram.

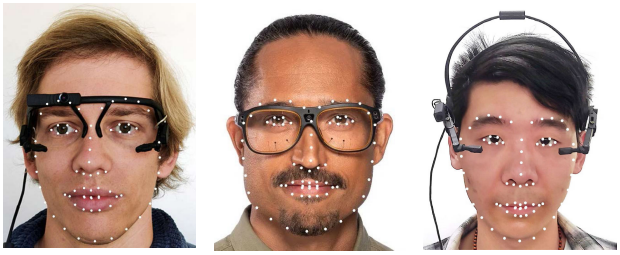


Fig. 7. Facial landmark annotation image.

wait for synchronization signals. When the main process sends synchronization signals to each child process, data collection begins. The collected data is stored in the Random Access Memory File System (RAMFS) first to reduce latency and then transferred to the disk file system after the data collection is completed. Fig. 6 illustrates the mechanism we implemented for synchronized data acquisition.

III. Results and Analysis

We conducted experiments to analyze the performance of the facial occlusion detection, pupil-fitting algorithm, gaze point prediction model, and hard real-time synchronization mechanism of the eye tracker that we designed. The experimental platform uses a Linux operating system with version Ubuntu18.04, and the hardware consists of an i5-9400F CPU, 16G memory, and NVIDIA GeForce GTX 1660Ti. The tracking application on Ubuntu is available at https://gitee.com/defeattroy/ueye-desk-linux.

A. Facial Occlusion Analysis

Reducing facial occlusion is an important purpose of the eye tracker that we designed. Therefore, we conducted a facial key point occlusion analysis on it.

We utilize dlib to annotate the 68 facial landmarks and output the annotated image by OpenCV, as shown in Fig. 7. It can be seen that our eye tracker does not cause any occlusion on the facial landmarks, while both Pupil Core and Tobii Glasses 3 cause significant interference in the eyebrow area.

B. Pupil-Fitting

We validate the performance of the proposed pupil-fitting algorithm on four publicly available datasets: CASIA-Iris-Thousand, CASIA-Iris-Lamp, CASIA-Iris-Syn [47], and IIT-Delhi-Iris [48]. The algorithm is compared with the traditional RANSAC algorithm and the deep learning algorithm Depp-VOG [49], and their performance is tested under extreme scenarios. We randomly select 500 images from the database as the test set and manually label the ellipse that fits the pupil's edge as groundtruth using the VGG Image Annotator.

The CASIA-Iris-V4 iris image dataset is a publicly available dataset released by the Chinese Academy of Sciences, containing a total of 54,607 iris images. In this paper, three subsets of this dataset are utilized:

CASIA-Iris-Syn comprises 10,000 synthetic iris images belonging to 1,000 categories and introduces intra-class variations such as deformation, blur, and rotation. CASIA-Iris-Lamp records texture elastic deformations caused by changes in pupil dilation and contraction under different lighting conditions. Therefore, it is suitable for researching non-linear iris normalization and robust iris feature representation. CASIA-Iris-Thousand is the first publicly available iris dataset with a thousand subjects, making it suitable for studying the uniqueness of iris features and for developing new iris classification and indexing methods.

The IIT-Delhi-Iris dataset originates from iris images of students and faculty in New Delhi, India. A total of 1,120 images are currently available from 224 subjects, comprising 176 males and 48 females, with ages ranging from 14 to 55 years.

Apart from the aforementioned publicly available datasets, this paper also uses an independently collected iris database. It contains 200 videos from 200 subjects and was captured using a wearable eye-tracking device developed in our laboratory, with a resolution of 320×200 pixels. These pupil videos encompass natural eye movements and blinking, resulting in images with pupil occlusions and blurriness due to eye motion. This dataset serves to better validate algorithms' performance in extreme scenarios.

Fig. 8 visually displays the algorithm's fitting results and demonstrates that the fitted ellipse circumference effectively covers the edge of the pupil and has high fitting accuracy across the four public datasets.

Table II displays the fitting results of the three algorithms on the four datasets. During the experiment, we find that all three algorithms performed well in identifying pupils, but significant errors occurred when pupils were occluded, leading to a notable impact on the performance evaluation. As a result, we conduct the error analysis only when correctly identifying pupils. The proposed algorithm had slightly smaller errors compared to the other two algorithms but without statistical significance. From the perspective of real-time performance, our pupil-fitting algorithm has a clear advantage in efficiency, and the frame rate in real-time analysis can be stabilized at above 80Hz. We first perform pupil region extraction, significantly reducing the computational workload for subsequent algorithms. Second, we make improvements to the RANSAC algorithm, improving the advantages of initial point selection and thereby enhancing the algorithm's efficiency.

TABLE II
COMPARING THE PUPIL-FITTING RESULTS OF RELATED ALGORITHMS

Datasets	Method	loss_a	loss_b	loss_d	Recall	Precision	F1	IoU	Average Time (ms)
CASIA-Iris-Lamp	RANSAC	0.648	0.774	1.487	96.32%	98.72%	97.51%	95.14%	38.47
	DeepVOG	0.522	**0.756**	1.143	95.94%	98.40%	97.15%	94.66%	43.5
	Ours	**0.483**	0.891	**0.996**	**97.52%**	**99.31%**	**98.34%**	**96.86%**	**11.49**
CASIA-Iris-Thousand	RANSAC	0.784	0.927	1.684	98.13%	96.27%	97.19%	94.55%	76.93
	DeepVOG	**0.404**	**0.576**	1.028	98.37%	96.74%	97.34%	95.20%	52.8
	Ours	0.613	0.936	**1.003**	**98.34%**	**97.19%**	**97.76%**	**95.61%**	**13.39**
CASIA-Iris-Syn	RANSAC	0.784	1.221	1.134	96.39%	99.03%	97.69%	96.72%	43.82
	DeepVOG	**0.406**	**0.822**	1.006	**98.14%**	99.28%	98.70%	**97.22%**	49.2
	Ours	0.565	0.876	**0.954**	97.89%	**99.96%**	**98.92%**	97.12%	**12.96**
IITD-Iris-Database	RANSAC	0.548	0.758	1.509	96.91%	99.18%	98.03%	96.17%	45.64
	DeepVOG	**0.406**	0.737	1.083	**99.21%**	97.23%	98.21%	96.46%	45.4
	Ours	0.452	**0.729**	**0.973**	98.08%	**99.37%**	**98.72%**	**97.48%**	**11.2**

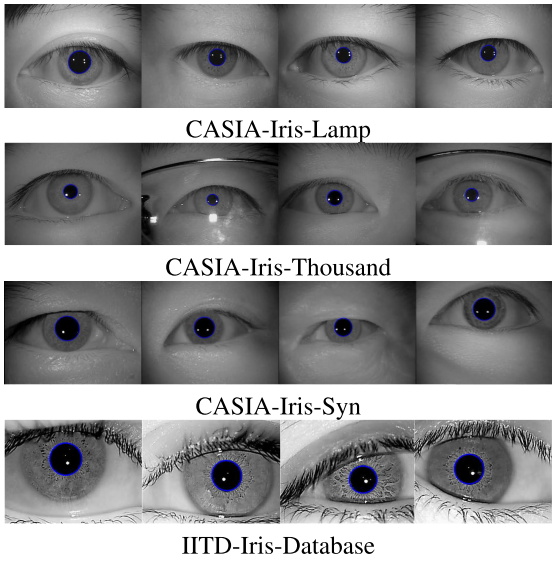


Fig. 8. The display of pupil ellipse fitting results.

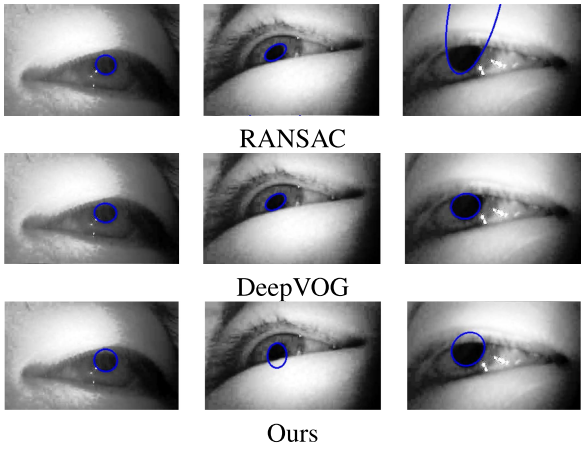


Fig. 9. The fitting results of three algorithms.

While the fitting results presented earlier are based on identifying pupils, we also conduct an experiment to test the algorithm's performance under occlusion conditions. We manually screen images with pupil occlusion from a pupil database collected and established independently in our laboratory and obtained a pupil occluded dataset containing 200 images. The pupil regions are labeled manually, and effective recognition is defined as having an IoU greater than 80% between predicted values and true values. The experiment was repeated ten times, and the recognition rates of RANSAC and DeepVOG are 23.8% and 56.3%, respectively. In contrast, the recognition rate of our proposed algorithm reaches 83.8%. Fig. 9 displays the fitting results of three algorithms. It is evident that our proposed pupil-fitting algorithm in this paper has significantly better robustness under occlusion conditions compared to the first two algorithms.

C. Gaze Tracking System

Experiments are designed to evaluate the gaze-tracking system proposed in this paper. The experimenters are divided into examiners and subjects where the examiners operate the eye-tracking device's upper computer software to complete the calibration and gaze point prediction, while the subjects observe a test paradigm displayed on another screen.

1) Experiment: The experiment mainly consists of two parts: model calibration and gaze point prediction. The subject is required to sequentially fixate on the five circular points that appear on the screen at a distance of 80 cm during calibration.

During gaze point prediction, the subject is required to perform three sets of tests at distances of 60 cm, 80 cm, and 100 cm from the screen to compare the accuracy changes of the gaze-tracking model at different depths. Five test points are sequentially shown in the center, upper left, lower left, upper right, and lower right corners of the screen to test the gaze-tracking algorithm's performance. During the experiment, the subjects are not required to strictly control their head movements and are allowed to move their heads naturally.

2) Model Performance Analysis: We mainly present the performance of the gaze-tracking system at different depths in terms of two indicators: precision and accuracy of predicted point distribution. Gaze points are estimated at three different depths of 60 cm, 80 cm, and 100 cm after calibration at a distance of 80 cm from the screen.

To compare with the 3D model, we implement a gaze point prediction model based on multiple regression. Tables III and IV show the precision and accuracy under ten parallel experiments at three different depths. It can be seen that the multiple regression and the 3D model exhibit similar accuracy and perform well in gaze prediction at the calibration plane distance (80 cm). Averaging accuracy across different depths, the average precision for multiple regression is

TABLE III

PRECISION OF GAZE PREDICTION MODEL BASED ON THREE-DIMENSIONAL GAZE VECTOR

Method	Sample	SSE			RMS		
		60 cm	80 cm	100 cm	60 cm	80 cm	100 cm
Multiple Regression	s1	0.307	0.111	0.129	0.171	0.124	0.110
	s2	0.348	0.117	0.137	0.181	0.132	0.116
	s3	0.645	0.250	0.184	0.303	0.244	0.124
	s4	0.166	0.195	0.379	0.094	0.204	0.375
	s5	0.128	0.392	0.397	0.115	0.171	0.209
	s6	0.194	0.101	0.375	0.148	0.104	0.250
	s7	0.143	0.116	0.602	0.101	0.123	0.300
	s8	0.255	0.212	0.296	0.159	0.119	0.142
	s9	0.189	0.181	0.154	0.191	0.148	0.136
	s10	0.223	0.168	0.222	0.206	0.156	0.219
	Mean	0.260	0.184	0.288	0.167	0.153	0.198
3D Model	s1	0.343	0.224	0.999	0.110	0.208	0.759
	s2	0.230	0.382	0.935	0.169	0.200	0.446
	s3	0.543	0.462	1.441	0.233	0.326	0.373
	s4	0.225	0.239	0.924	0.211	0.222	0.261
	s5	0.365	0.367	1.067	0.237	0.306	0.201
	s6	0.331	0.114	0.368	0.188	0.143	0.297
	s7	0.232	0.562	0.885	0.157	0.129	0.372
	s8	0.406	0.212	0.815	0.309	0.119	0.279
	s9	0.309	0.207	0.382	0.253	0.185	0.300
	s10	0.562	0.181	1.067	0.129	0.135	0.301
	Mean	0.355	0.295	0.888	0.199	0.197	0.349

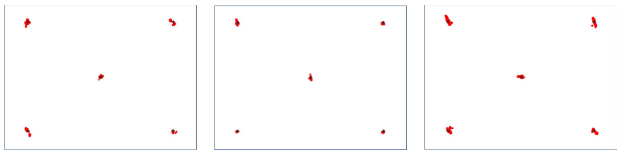
TABLE IV

ACCURACY OF GAZE PREDICTION MODEL BASED ON THREE-DIMENSIONAL GAZE VECTOR

Method	Sample	60 cm	80 cm	100 cm
Multiple Regression	s1	1.478 ± 0.384	0.449 ± 0.105	0.776± 0.063
	s2	1.329 ± 0.169	0.476 ± 0.111	0.824 ± 0.067
	s3	2.983 ± 0.268	0.409 ± 0.199	0.874 ± 0.171
	s4	3.317 ± 0.168	0.542 ± 0.122	0.756 ± 0.266
	s5	1.409 ± 0.125	0.512 ± 0.157	0.671 ± 0.312
	s6	3.379 ± 0.179	0.500 ± 0.086	0.730 ± 0.291
	s7	2.296 ± 0.103	0.671 ± 0.108	1.005 ± 0.447
	s8	1.694 ± 0.261	0.570 ± 0.108	0.784 ± 0.279
	s9	1.101 ± 0.202	0.460 ± 0.169	1.038 ± 0.091
	s10	2.663 ± 0.259	0.339 ± 0.120	0.634 ± 0.125
	Mean	2.165 ± 0.212	0.493 ± 0.129	0.809 ± 0.211
3D Model	s1	0.700 ± 0.322	0.598 ± 0.231	1.445 ± 1.082
	s2	0.784 ± 0.257	0.679 ± 0.316	1.161 ± 0.842
	s3	0.520 ± 0.391	0.514 ± 0.332	1.715 ± 1.211
	s4	0.948 ± 0.204	0.637 ± 0.246	1.080 ± 0.769
	s5	0.504 ± 0.331	0.480 ± 0.229	1.633 ± 1.134
	s6	0.757 ± 0.286	0.608 ± 0.079	1.159 ± 0.332
	s7	0.705 ± 0.117	0.590 ± 0.328	2.128 ± 0.692
	s8	0.723 ± 0.211	0.570 ± 0.108	0.878 ± 0.409
	s9	1.231 ± 0.314	0.281 ± 0.182	0.679 ± 0.316
	s10	0.590 ± 0.328	0.314 ± 0.108	1.633 ± 1.134
	Mean	0.746 ± 0.276	0.527 ± 0.216	1.351 ± 0.792

approximately 1.156°, while the 3D model's average precision is approximately 0.875°. The method of multiple regression performs better in terms of precision. However, the 3D gaze prediction model shows decreasing precision as the distance between the subject and the gaze point increases. The predicted gaze point set is also more scattered, leading to poorer accuracy.

3) Error Analysis: As the distance between the human eyes and the fixation point increases, the accuracy of the fixation point prediction model based on the three-dimensional gaze vector decreases. The gaze point prediction can be expressed by Equation 24, which is determined by the eyeball coordinates o and the κ angle. The visual axis vector is obtained from the three-dimensional eyeball model generated by the pupil region in the image and the *kappa* angle is obtained from calibration. However, the process has errors, which cause angle errors (θ, γ) propagated to the visual vector, leading to a dispersed distribution of predicted gaze points. When the distance between the eye and the gaze target increases, the error is magnified accordingly. Solving this problem will be the focus of our future research.



(60cm) (80cm) (100cm)

Fig. 10. A gaze prediction model based on eye-tracking data is used to predict the distribution of fixation points in different depth scenes.

TABLE V

COMPARISON OF GAZE TRACKING ALGORITHMS

Algorithms	Mean Average Precision	Algorithm Types
Brolly [50]	0.8°	Multiple Regression
Blignaut [51]	0.87°	Multiple Regression
Beymer [52]	0.6°	3D Modeling
Shih [53]	$< 1 \sim 2°$	3D Modeling
Shih [54]	$< 1 \sim 2°$	3D Modeling
Newman [55]	1°	3D Modeling
Zhang [56]	$0.3 \sim 0.4°$	Cross-ratio-based
Arar [57]	1°	Cross-ratio-based
Tan [58]	0.5°	Appearance-based
P-CFB [59]	1.43°	Contour-feature-based
Hybrid [59]	1.18°	Hybrid method
Reinders [60]	4.5Pix	Eye Shape-based
Ours	$0.527 \sim 0.875°$	3D Modeling

$$g_p(o,\kappa) = o + k \cdot \begin{pmatrix} \cos(\varphi+\beta+\theta)\sin(\theta+\alpha+\gamma) \\ \sin(\varphi+\beta+\theta) \\ -\cos(\varphi+\beta+\theta)\cos(\theta+\alpha+\gamma) \end{pmatrix} \tag{24}$$

The formula for the error angle is given as Equation 25. Taking the derivative of $\Delta\theta$ with respect to Δd yields Equation 26 and Equation 27. Here, m is much smaller than a and b, and aince a and b are determined, the value of m approaches 1, meaning that $f'(\Delta\theta)$ approaches infinity. Therefore, if there is a small increase in Δd, the value of $\Delta\theta$ will increase greatly, causing the prediction distribution of the gaze point to become dispersed as the gaze depth increases, resulting in instability of the model.

$$\Delta\theta = \arccos\frac{a^2+b^2-\Delta d^2}{2ab} \tag{25}$$

$$f'(\Delta\theta) = \frac{\Delta d}{ab\sqrt{1-m^2}} \tag{26}$$

$$m = \frac{a^2+b^2-\Delta d^2}{2ab} \tag{27}$$

In summary, the error in accuracy is mainly caused by imprecision in estimating the visual axis vector, which leads to dispersed predicted gaze point distributions due to error propagation in earlier algorithms, thus affecting the accuracy of gaze estimation. If we can solve the errors caused by algorithm jitter and improve the accuracy of predicting gaze points at long distances, the gaze point prediction model will have even better results. Table V shows the comparison between our system and others.

TABLE VI
ABLATION EXPERIMENTS

	Priority	Sync	Idle System (ms)			Cpu Overload System (ms)			Memory Overload System (ms)		
			Latency	Jitter	Maximum	Latency	Jitter	Maximum	Latency	Jitter	Maximum
GPOS	×	×	0.143	0.159	0.726	4.470	2.998	112.2	10.30	7.398	111.6
	✓	×	0.142	0.228	0.536	2.800	1.604	96.12	7.930	4.182	86.33
	×	✓	0.085	0.113	0.403	3.700	2.715	80.12	9.870	6.377	87.63
	✓	✓	0.088	0.125	0.224	4.260	2.411	125.3	9.240	5.871	48.27
RTOS	×	×	0.228	0.281	0.664	3.870	2.973	96.02	9.150	10.27	119.1
	✓	×	0.113	0.032	0.463	0.118	0.015	0.853	0.761	0.224	1.912
	×	✓	0.081	0.102	0.891	3.030	2.265	92.95	8.080	5.335	91.82
	✓	✓	0.086	0.059	0.226	0.068	0.005	0.139	0.451	0.120	1.175

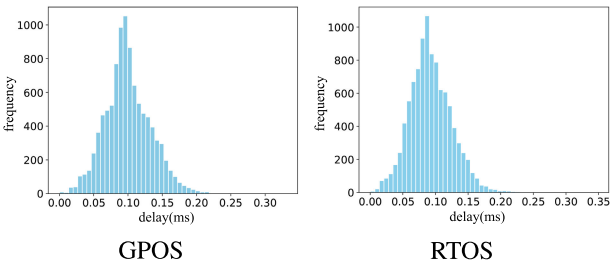


Fig. 11. Synchronization latency benchmark on idle system.

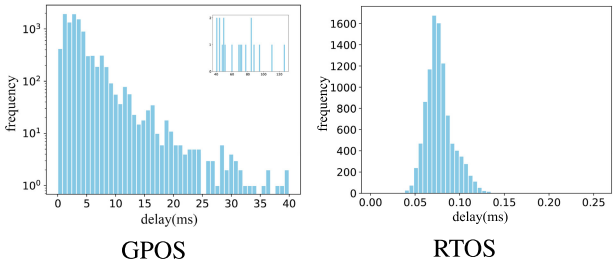


Fig. 12. Synchronization latency benchmark on CPU overload system.

D. Synchronous Performance Benchmark

In order to verify the feasibility of process synchronization under this system, we conduct three sets of experiments using an eye-tracking system on a general purpose Linux operating system and PREEMPT_RT. We benchmark the maximum process latency between the real-time tasks under idle, high CPU load, and high memory load systems. A deterministic real-time system always exhibits a normal distribution of latency in linear scale, while there are occasional instances of huge latencies in GPOS, so we use linear scale and logarithmic scale to show the latency distribution for RTOS and GPOS respectively.

The Linux stress testing tool "Stress" is used to increase the system load, which applies a load on the computer's CPU, memory, IO, and disk to test system performance. The first set of experiments is conducted under no system load as a comparison experiment, with results shown in Fig. 11. In the second set of experiments, we increase CPU load by running 60 processes in the background to constantly preempt the processor, resulting in the data shown in Fig. 12. In the third set of experiments, we increase the memory load and test the system under heavy memory usage, with results shown in Fig. 13. Each experiment is conducted 10,000 times while calculating the maximum process latency difference between the four camera processes.

Under an idle system, the maximum delay between a GPOS process and a hard RTOS process is within 0.2ms. The results

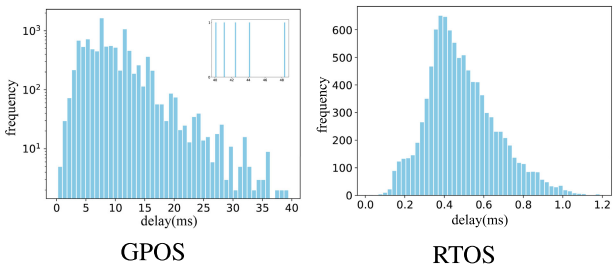


Fig. 13. Synchronization latency benchmark on memory overload system.

of latencies show a normal distribution, which is consistent with actual scenarios. However, under high CPU load conditions, the delay of GPOS processes is significantly larger, with a maximum of up to 120ms which leads to serious frame loss and misalignment. Under a hard RTOS, the maximum delay of processes remains within 0.2ms, which is the same as under idle conditions and follows a normal distribution. Under high memory load conditions, the delay of GPOS is still significant, with a maximum of up to 50 milliseconds. However, with a hard RTOS, the maximum delay of processes remains within 1.2ms and follows a normal distribution, which is also within an acceptable range.

After three sets of experiments and 60,000 tests of the maximum delay of processes, we find that modifying the Linux kernel to a hard RTOS and increasing its priority can ensure that the system remains stable under high load conditions. That guarantees accurate synchronization of eye movement, facial expression data, and scene images, and collection of high-quality multimodal data. With a hard RTOS, the maximum delay difference of four cameras is guaranteed to be within 1 millisecond, which meets the acceptable error range of the eye-tracking system designed in this paper.

In order to demonstrate the roles of the various modules in our proposed hard real-time synchronization mechanism, we conduct ablation experiments, and the results are shown in Table VI. When RAMFS is not used and data is stored directly on the hard drive, the frame rate is significantly reduced, falling far below the expected design frame rate. At the same time, the average latency exceeds 200ms. Conducting ablation experiments without storing data in RAMFS would be meaningless. Therefore, the ablation experiments store data in RAMFS.

From the table, it can be observed that under GPOS, due to differences in system scheduling, increasing process priorities and using synchronization signals can sometimes lead to an increase in latency discrepancies. However, under RTOS, adjusting process priorities and introducing

synchronization signals have a noticeable effect in reducing latency discrepancies.

IV. Conclusion

This paper presents the design of a wearable eye-tracking system capable of synchronously collecting multimodal data. We carry out several key tasks to achieve this goal. First, we design a wearable eye tracker that is flexible, portable, stable, and comfortable, and minimizes facial occlusion as much as possible, making it suitable for multimodal data collection in a variety of scenarios. Second, we add infrared light sources and filters to eliminate the effects of ambient light on data collection and provide an optimal data collection environment. Third, we propose a pupil extraction algorithm based on RANSAC, which first segmented the iris region and then fitted an elliptical pupil region using edge points. Then, we use a gaze prediction model based on a three-dimensional annotation vector. The intersection point between the left and right eye visual axis vectors in space is calculated as the current gaze point. Finally, we design a hard real-time synchronization scheme to facilitate the synchronous collection of multimodal data, which ensures data synchronization even under a high load.

The experimental results demonstrate the effectiveness of our proposed eye-tracking system. In addition, the eye tracker we designed is compatible with Pupil Labs software [61], providing excellent expandability for the system. Overall, our work provides a robust wearable eye-tracking system that can synchronously collect multimodal data.

References

[1] J. Henderson and F. Ferreira, *The Interface of Language, Vision, and Action: Eye Movements and the Visual World*. New York, NY, USA: Psychology Press, 2013.

[2] S. Caldani, C.-L. Gerard, H. Peyre, and M. P. Bucci, "Visual attentional training improves reading capabilities in children with dyslexia: An eye tracker study during a reading task," *Brain Sci.*, vol. 10, no. 8, p. 558, Aug. 2020.

[3] M. Yang, X. Feng, R. Ma, X. Li, and C. Mao, "Orthogonal-moment-based attraction measurement with ocular hints in video-watching task," *IEEE Trans. Computat. Social Syst.*, vol. 10, no. 3, pp. 900–909, Jun. 2023.

[4] Y. Wang, Z. Lv, and Y. Zheng, "Automatic emotion perception using eye movement information for E-healthcare systems," *Sensors*, vol. 18, no. 9, p. 2826, Aug. 2018.

[5] F. Mayrand, F. Capozzi, and J. Ristic, "A dual mobile eye tracking study on natural eye contact during live interactions," *Sci. Rep.*, vol. 13, no. 1, p. 11385, Jul. 2023.

[6] M. Yanoff and J. S. Duker, *Ophthalmology*. Amsterdam, The Netherlands: Elsevier, 2008.

[7] V. Delvigne, H. Wannous, T. Dutoit, L. Ris, and J.-P. Vandeborre, "PhyDAA: Physiological dataset assessing attention," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 5, pp. 2612–2623, May 2022.

[8] H. K. Raza, H. Chen, T. Chansysouphanthong, and G. Cui, "The aetiologies of the unilateral oculomotor nerve palsy: A review of the literature," *Somatosensory Motor Res.*, vol. 35, nos. nos. 3–4, pp. 229–239, Oct. 2018.

[9] S. Li, M. C. Duffy, S. P. Lajoie, J. Zheng, and K. Lachapelle, "Using eye tracking to examine expert-novice differences during simulated surgical training: A case study," *Comput. Hum. Behav.*, vol. 144, Jul. 2023, Art. no. 107720.

[10] A. Kovari, J. Katona, and C. Costescu, "Evaluation of eye-movement metrics in a software debugging task using gp3 eye tracker," *Acta Polytechnica Hungarica*, vol. 17, no. 2, p. 5776, 2020.

[11] H. Liu and I. Heynderickx, "Visual attention in objective image quality assessment: Based on eye-tracking data," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 7, pp. 971–982, Jul. 2011.

[12] M. Paul and M. M. Salehin, "Spatial and motion saliency prediction method using eye tracker data for video summarization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 6, pp. 1856–1867, Jun. 2019.

[13] J. Z. Lim, J. Mountstephens, and J. Teo, "Emotion recognition using eye-tracking: Taxonomy, review and current challenges," *Sensors*, vol. 20, no. 8, p. 2384, Apr. 2020.

[14] M. Oliva and A. Anikin, "Pupil dilation reflects the time course of emotion recognition in human vocalizations," *Sci. Reports*, vol. 8, no. 1, p. 4871, 2018.

[15] J.-X. Mi, Y. Gao, S. Yuan, and W. Li, "Accurate and robust eye center localization by deep voting," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 8, pp. 4070–4082, Aug. 2023.

[16] D. W. Hansen and Q. Ji, "In the eye of the beholder: A survey of models for eyes and gaze," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 3, pp. 478–500, Mar. 2010.

[17] L. Świrski, A. Bulling, and N. Dodgson, "Robust real-time pupil tracking in highly off-axis images," in *Proc. Symp. Eye Tracking Res. Appl.*, Mar. 2012, pp. 173–176.

[18] L. Swirski and N. Dodgson, "A fully-automatic, temporal approach to single camera, glint-free 3D eye model fitting," in *Proc. PETMEI*, 2013, p. 111.

[19] D. Su, Y. F. Li, and H. Chen, "Region-wise polynomial regression for 3D mobile gaze estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 907–913.

[20] D. Su, Y.-F. Li, and H. Chen, "Cross-validated locally polynomial modeling for 2-D/3-D gaze tracking with head-worn devices," *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 510–521, Jan. 2020.

[21] A. Kar and P. Corcoran, "A review and analysis of eye-gaze estimation systems, algorithms and performance evaluation methods in consumer platforms," *IEEE Access*, vol. 5, pp. 16495–16519, 2017.

[22] M. Mansouryar, J. Steil, Y. Sugano, and A. Bulling, "3D gaze estimation from 2D pupil positions on monocular head-mounted eye trackers," in *Proc. 9th Biennial ACM Symp. Eye Tracking Res. Appl.*, Mar. 2016, pp. 197–200.

[23] C.-W. Huang and W.-C. Tan, "An approach of head movement compensation when using a head mounted eye tracker," in *Proc. IEEE Int. Conf. Consum. Electron.-Taiwan (ICCE-TW)*, May 2016, pp. 1–2.

[24] J. Wang, G. Zhang, and J. Shi, "2D gaze estimation based on pupil-glint vector using an artificial neural network," *Appl. Sci.*, vol. 6, no. 6, p. 174, Jun. 2016.

[25] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "Predicting human eye fixations via an LSTM-based saliency attentive model," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5142–5154, Oct. 2018.

[26] M. Yuan and D. Xu, "Spatio-temporal feature pyramid interactive attention network for egocentric gaze prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 10, pp. 5790–5801, Oct. 2023.

[27] Y. Wu, G. Li, Z. Liu, M. Huang, and Y. Wang, "Gaze estimation via modulation-based adaptive network with auxiliary self-learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 8, pp. 5510–5520, Aug. 2022.

[28] M. F. Land and D. N. Lee, "Where we look when we steer," *Nature*, vol. 369, no. 6483, pp. 742–744, Jun. 1994.

[29] K. Dierkes, M. Kassner, and A. Bulling, "A fast approach to refraction-aware eye-model fitting and gaze prediction," in *Proc. 11th ACM Symp. Eye Tracking Res. Appl.*, Jun. 2019, p. 19.

[30] B. Petersch and K. Dierkes, "Gaze-angle dependency of pupil-size measurements in head-mounted eye tracking," *Behav. Res. Methods*, vol. 54, no. 2, pp. 763–779, Apr. 2022.

[31] I. T. C. Hooge, D. C. Niehorster, R. S. Hessels, J. S. Benjamins, and M. Nyström, "How robust are wearable eye trackers to slow and fast head and body movements?" *Behav. Res. Methods*, p. 115, Nov. 2022, doi: 10.3758/s13428-022-02010-3.

[32] V. Delvigne, H. Wannous, T. Dutoit, L. Ris, and J.-P. Vandeborre, "Evaluating the Tobii Pro Glasses 2 and 3 in static and dynamic conditions," *Behav. Res. Methods*, Aug. 2023. [Online]. Available: https://doi.org/10.3758/s13428-023-02173-7

[33] M. A. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[34] F. Reghenzani, G. Massari, and W. Fornaciari, "The real-time Linux kernel: A survey on preempt_rt," *ACM Comput. Surv.*, vol. 52, no. 1, p. 136, 2019.

[35] Y.-C. Wang and K.-J. Lin, "Enhancing the real-time capability of the Linux kernel," in *Proc. 5th Int. Conf. Real-Time Comput. Syst. Appl.*, 1998, pp. 11–20.

[36] Y. Qi et al., "A comprehensive overview of image enhancement techniques," *Arch. Comput. Methods Eng.*, vol. 29, no. 1, pp. 583–607, Jan. 2022.
[37] (2020). *Ultralytics, YOLOv5, GitHub Repository*. [Online]. Available: https://github.com/ultralytics/yolov5/
[38] K. G. Derpanis, "Overview of the RANSAC algorithm," *Image Rochester NY*, vol. 4, no. 1, p. 23, 2010.
[39] Y. Le Grand, *Light, Colour and Vision*. New York, NY, USA: Dover, 1957.
[40] E. D. Guestrin and M. Eizenman, "General theory of remote gaze estimation using the pupil center and corneal reflections," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 6, pp. 1124–1133, Jun. 2006.
[41] K. R. Senior et al., *The Eye: The Physiology of Human Perception*. New York, NY, USA: The Rosen Publishing Group, 2010.
[42] Z. Zhu and Q. Ji, "Novel eye gaze tracking techniques under natural head movement," *IEEE Trans. Biomed. Eng.*, vol. 54, no. 12, pp. 2246–2260, Dec. 2007.
[43] W. Schneider, A. Eschman, and A. Zuccolotto, *E-Prime Users Guide*. Pittsburgh, PA, USA: Psychology Software Tools, 2012.
[44] D. Bridges, A. Pitiot, M. R. MacAskill, and J. W. Peirce, "The timing mega-study: Comparing a range of experiment generators, both lab-based and online," *PeerJ*, vol. 8, p. e9414, Jul. 2020.
[45] J. Vestin, A. Kassler, and J. Åkerberg, "FastReact: In-network control and caching for industrial control networks using programmable data planes," in *Proc. IEEE 23rd Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, vol. 1, Sep. 2018, pp. 219–226.
[46] G. K. Adam, N. Petrellis, and L. T. Doulos, "Performance assessment of Linux kernels with PREEMPT_RT on ARM-based embedded devices," *Electronics*, vol. 10, no. 11, p. 1331, Jun. 2021.
[47] Chinese Academy of Sciences. *CASIA.v4*. Accessed: Jun. 15, 2023. [Online]. Available: http://biometrics.idealtest.org/
[48] A. Kumar and A. Passi, "Comparison and combination of iris matchers for reliable personal authentication," *Pattern Recognit.*, vol. 43, no. 3, pp. 1016–1026, Mar. 2010.
[49] Y.-H. Yiu et al., "DeepVOG: Open-source pupil segmentation and gaze estimation in neuroscience using deep learning," *J. Neurosci. Methods*, vol. 324, Aug. 2019, Art. no. 108307.
[50] X. L. C. Brolly and J. B. Mulligan, "Implicit calibration of a remote gaze tracker," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshop*, 2004, p. 134.
[51] P. Blignaut, "Mapping the pupil-glint vector to gaze coordinates in a simple video-based eye tracker," *J. Eye Movement Res.*, vol. 7, no. 1, pp. 1–11, Mar. 2013.
[52] D. Beymer and M. Flickner, "Eye gaze tracking using an active stereo head," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, Jun. 2003, p. II451.
[53] S.-W. Shih and J. Liu, "A novel approach to 3-D gaze tracking using stereo cameras," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 34, no. 1, pp. 234–245, Feb. 2004.
[54] S.-W. Shih, Y.-T. Wu, and J. Liu, "A calibration-free gaze tracking technique," in *Proc. 15th Int. Conf. Pattern Recognit. (ICPR)*, 2000, pp. 201–204.
[55] R. Newman, Y. Matsumoto, S. Rougeaux, and A. Zelinsky, "Real-time stereo tracking for head pose and gaze estimation," in *Proc. 4th IEEE Int. Conf. Autom. Face Gesture Recognit.*, Mar. 2000, pp. 122–128.
[56] Z. Zhang and Q. Cai, "Improving cross-ratio-based eye tracking techniques by leveraging the binocular fixation constraint," in *Proc. Symp. Eye Tracking Res. Appl.*, Mar. 2014, pp. 267–270.
[57] N. M. Arar, H. Gao, and J.-P. Thiran, "A regression-based user calibration framework for real-time gaze estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 12, pp. 2623–2638, Dec. 2017.
[58] K.-H. Tan, D. J. Kriegman, and N. Ahuja, "Appearance-based eye gaze estimation," in *Proc. 6th IEEE Workshop Appl. Comput. Vis. (WACV)*, Dec. 2002, pp. 191–195.
[59] C.-C. Lai, S.-W. Shih, and Y.-P. Hung, "Hybrid method for 3-D gaze tracking using glint and contour features," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 1, pp. 24–37, Jan. 2015.
[60] M. Reinders, "Eye tracking by template matching using an automatic codebook generation scheme," in *Proc. 3rd Annu. Conf. ASCI, ASCI, Delft*, 1997, pp. 85–91.
[61] M. Kassner, W. Patera, and A. Bulling, "Pupil: An open source platform for pervasive eye tracking and mobile gaze-based interaction," in *Proc. ACM Int. Joint Conf. Pervas. Ubiquitous Comput., Adjunct Publication*, 2014, pp. 1151–1160.

Minqiang Yang (Member, IEEE) received the Ph.D. degree in computer science from Lanzhou University. He is currently an Associate Professor with the Gansu Provincial Key Laboratory of Wearable Computing, School of Information Science and Engineering, Lanzhou University. His current research interests include affective computing, image processing, machine learning, and automatic depression detection. He has published more than 20 papers on IEEE magazines, IEEE journals, and leading conferences.

Yujie Gao received the B.S. degree from the Beijing Institute of Technology in 2021. He is currently pursuing the M.S. degree with the Gansu Provincial Key Laboratory of Wearable Computing, School of Information Science and Engineering, Lanzhou University. His main research interests include affective computing, image processing, and machine learning.

Longzhe Tang received the B.S. and M.S. degrees from Lanzhou University, Lanzhou, China, in 2020 and 2023, respectively. His main research interests include affective computing, image processing, and machine learning.

Jian Hou received the M.S. degree from Lanzhou University, Lanzhou, China, in 2023. His main research interests include affective computing, image processing, and machine learning.

Bin Hu (Fellow, IEEE) received the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Science, in 1998. He is currently a (Full) Professor and the Dean of the School of Medical Technology, Beijing Institute of Technology, China. He is also an Adjunct Professor and the former Dean of the School of Information Science and Engineering, Lanzhou University, Lanzhou, China. He is a Chinese National Distinguished Expert, the Chief Scientist of 973 projects, and the National Advanced Worker in 2020. He was elected as a fellow of the Institution of Engineering and Technology (IET). He is a member of the Steering Council of the ACM China Council and the Vice-Chair of the China Committee of the International Society for Social Neuroscience. He serves as the Editor-in-Chief for IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS. He is also the TC Co-Chair of Computational Psychophysiology in the IEEE Systems, Man, and Cybernetics Society (SMC) and the TC Co-Chair of Cognitive Computing in IEEE SMC. He is a member of the Computer Science Teaching and Steering Committee and the Science and Technology Committee. He (co)authored more than 400 publications (more than 10 000 citations, H-index 51). His awards include the 2014 China Overseas Innovation Talent Award, the 2016 Chinese Ministry of Education Technology Invention Award, the 2018 Chinese National Technology Invention Award, and the 2019 WIPO-CNIPA Award for Chinese Outstanding Patented Invention. He is a Principal Investigator for large grants, such as the National Transformative Technology Early Recognition and Intervention Technology of Mental Disorders Based on Psychophysiological Multimodal Information, which have extensively promoted the development of objective, quantitative diagnosis, and non-drug interventions for mental disorders.