ORIGINAL ARTICLE

Enhancing gaze estimation accuracy in wearable eye-tracking devices using neural networks

Jerome Charton[1] · Tianpeng Zhang[2] · Na Li[2] · Quanzheng Li[1]

Received: 27 November 2024 / Accepted: 4 May 2025

Abstract
Eye-tracking devices are convenient for interpreting human behaviors and intentions, enabling contactless human–computer interaction, such as in medical image interpretation using eye-gaze tracking. Recent advances in wearable eye-tracking devices have further allowed wearers to move freely and use them in regular activities. However, gaze estimation from wearable devices tends to be less precise than that from standard stationary eye-tracking devices. This is due to device design constraints and a lack of interpretation of the relationship between the scene and the wearer. In this work, we propose to enhance the accuracy of gaze estimation in wearable eye-tracking devices through a framework that incorporates two neural networks, *CorNN* and *CalNN*. The *CorNN* corrects the bias induced by the distance between the observer and the gaze locations, primarily resulting from the parallax and lens distortion effects. Meanwhile, the *CalNN* focuses on improving calibration specific to each wearer. To collect precise training data for these networks, we have implemented an automated robotic data collection pipeline. The proposed framework was demonstrated on the Pupil Labs Invisible eye-tracking device and tested on 11 wearers, showing improved average gaze estimation accuracy for all wearers.

Keywords Head-mounted eye-tracking device · Neural network-based parallax correction · Accuracy improvement · Pupil Labs Invisible

1 Introduction

Eye trackers are devices capable of estimating the gaze location of users by tracking their eye movements and have become a vital tool for contactless human–computer interaction. The device's output is a gaze location given in pixel coordinates, either on a monitor screen or in a scene camera image. The recent applications of eye-tracking technology have extended into various areas, including robot–human interaction in industrial settings [1–5], driving monitoring systems [6–8], and enhancing the quality of automated medical image segmentation and analysis by assessing visual attention [9–12]. Wearable eye-tracking devices have evolved to meet various task requirements, minimizing usage constraints (e.g., eliminating the need for a chin rest) and streamlining the calibration process (e.g., the Pupil Labs Invisible requires only a wearer-specific offset), enhancing user convenience and allowing for more seamless integration into daily activities [13, 14].

Wearable eye-tracking devices, such as the Pupil Labs Invisible [15] as illustrated in Fig. 1, have especially received much attention due to their portability and versatility. However, challenges remain in developing accurate wearable eye-tracking devices. In particular, adapting the device to the gaze characteristics inherent to different wearers is imperative. Pupil Labs Invisible corrects gaze estimation bias by using a camera scene offset that the wearer manually sets. This offset is added to the estimated gaze location to approximate the intended

https://doi.org/10.1007/s00521-025-11334-y
Published online: 04 June 2025

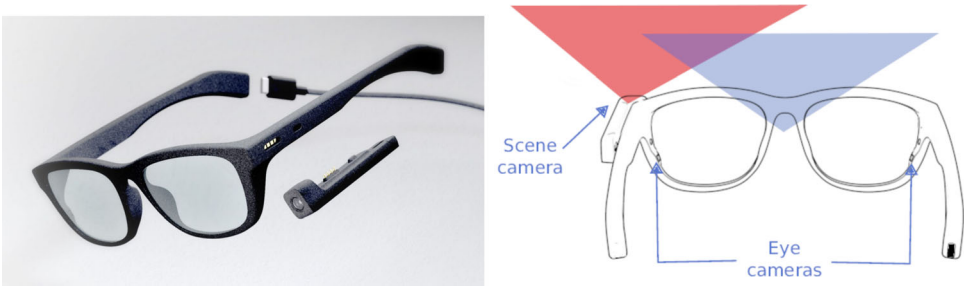


Fig. 1 Configuration of the wearable eye-tracking device used in the framework (Pupil Labs Invisible). Red and blue triangles represent the camera and wearer viewpoints, respectively. (https://pupil-labs.com/products/invisible/)

gaze. However, this simplified correction method does not account for errors due to the position of the eyes, lens distortion of the scene camera, and parallax effects, which influence gaze estimation based on the wearer's distance from the target location [15–19].

In this study, we propose a correction framework for wearable eye-tracking devices using neural networks. Figure 3 outlines the proposed framework. We used a neural network, *CorNN*, to correct the parallax and lens distortion effect. Additionally, we use another network, *CalNN*, to perform wearer-specific calibration. To collect precise training data for these networks, we implemented a robotic data collection system using the UR5e manipulator (see Fig. 2). To validate the correction framework, we collected gaze data from 11 participants and evaluated the gaze estimation accuracy. The proposed framework was evaluated by comparing the average gaze estimation accuracy for all wearers against two baselines. The first baseline was the raw estimation made by the eye-tracking device. The second baseline was a geometrical approach that compensates for the parallax effect, which considers a calibration plane at a 2-meter distance. The proposed framework improved the average gaze estimation accuracy for all wearers compared to both baselines. Specifically, it outperformed the raw estimation

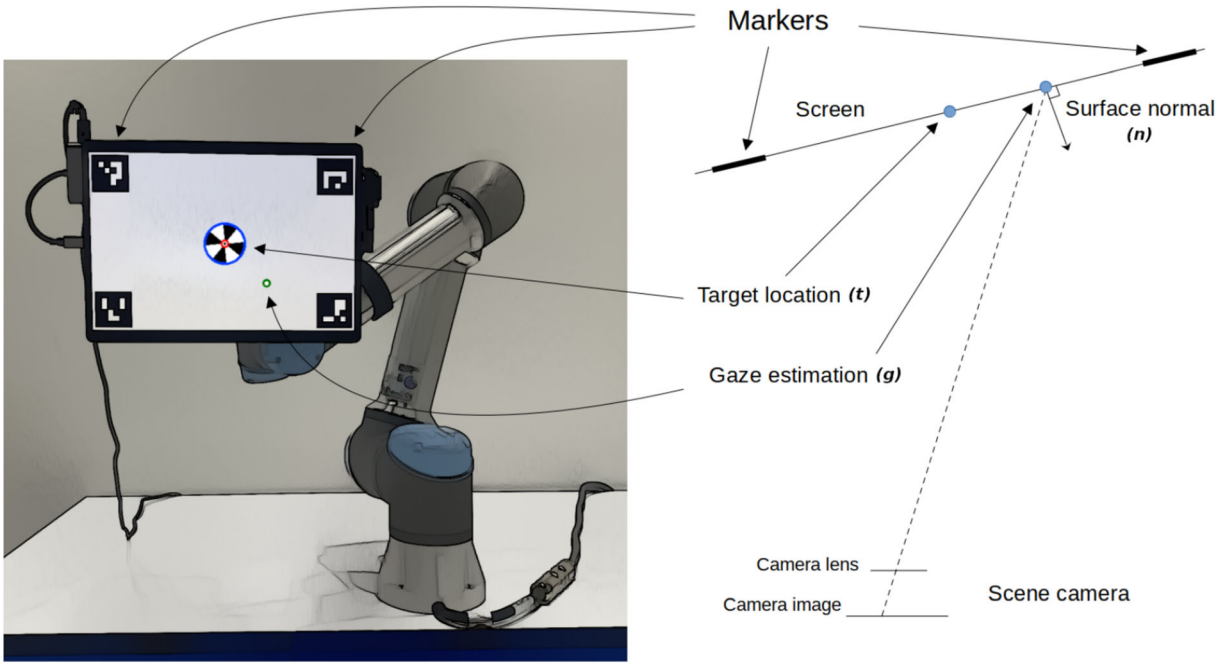


Fig. 2 Actual gaze location vs. gaze estimation. This figure illustrates the screen presented to the eye tracker's wearers and the schematic of the 3D gaze estimation and scene pose reconstruction during a record. Left, the application screen with four ArUco markers, the visual target, and the gaze estimated by the eye-tracking device. Right, a top view of the scene with the 3D locations of the screen and the projection for the gaze estimation onto the screen plane

made by the eye-tracking device and the geometrical approach that accounts for the parallax effect. This demonstrates the effectiveness of the proposed correction framework in enhancing gaze estimation accuracy.

2 Related works

Head-stabilized and remote eye-tracking devices, such as the EyeLink 3000+ and the Tobii Pro X2-30, are widely adopted in research and are known for their remarkable gaze estimation accuracy. However, these products are known to require meticulous calibrations and impose constraints on the user's position to achieve high accuracy. Additionally, they lack portability and must be mounted at a fixed location.

In contrast, wearable *head-mounted eye trackers* enable users to move freely during regular indoor and outdoor activities [14, 20], such as the *Pupil Labs Invisible eye tracker* [15]. These devices resemble a pair of regular eyeglasses (see Fig.1), but are equipped with two *eye cameras* installed in the frame pointed toward the wearer's eyes, capturing the pupil center location and corneal reflections. This allows for estimating the wearer's gaze location relative to the device. Additionally, a *scene camera* is mounted on the left side of the frame, capturing the scene seen by the wearer and enabling the device's self-localization in the wearer's environment. The wearer's gaze location in the environment can thus be determined using the eye and scene cameras.

Despite their portability, existing wearable eye trackers suffer from mediocre gaze estimation accuracy due to the *parallax effect*. The parallax effect is caused by a discrepancy between the scene camera's field of view and the wearer's field of view, as illustrated in Fig. 1. Due to this discrepancy between the two observation points, the estimated gaze locations observed from the scene camera differ from the actual gaze location of the wearer, and that shift increases when the gaze location gets closer.

The parallax effect is widely prevalent in wearable eye-tracking devices, particularly in those equipped with a single scene camera. Additionally, when the distance between the eyes and the scene is less than two meters, the parallax effect often combines with camera lens distortion, leading to a significant degradation in gaze estimation accuracy, as highlighted in [16, 17].

To address the parallax effects, [13] proposed a binocular head-mounted eye-tracking device and used the estimated gaze direction of each eye to calculate the observation plane distance and compensate for the parallax effect as a translation in the scene camera image. [21] suggested gaze estimation for compensating the parallax effect induced by the gap distance between the display and a touch screen. [22] compensated the parallax effect accounted with an SR Research EyeLink II by adding a head-mounted stereo camera synchronized with the eye tracker for reconstructing the observer's 3D scene. [23] presented a fully custom binocular head-mounted eye-tracking device for 3D gaze estimation using an Intel RealSense D435 RGB-D camera. Recently, the Tobii Pro Glasses 3 has integrated a built-in parallax effect compensation tool using an embedded stereo camera [24]. Although proving effective in improving gaze estimation, all these techniques are either device-specific or require customized/additional hardware. Thus, these solutions could be inflexible and costly. We also note that limited effort has been dedicated to improving gaze estimation software, which could offer a more versatile and cost-effective solution to mitigate the impacts of parallax-related effects.

In this paper, we propose a purely software add-on framework for compensating gaze distortions relative to the distance between a monitor screen and any head-mounted eye-tracking device that can provide a gaze estimation as x- and y-coordinates within a scene camera image. This framework stands on a neural network for the device's distortion correction and a second neural network for supporting a user-specific calibration.

3 Method

The eye-tracking device used in our study is the Pupil Labs Invisible [15] (Fig. 1), a lightweight wearable device that requires only minimal calibration. The device allows a wearer-specific offset to be added to the x- and y-coordinates of the 2D raw gaze location estimates, correcting biases inherent to individual wearers. This offset is manually configurable via the Pupil Labs Invisible Companion application installed on the smartphone connected to the eye-tracking device.

Consider the scenario where a person wearing the eye tracker gazes at a specific location on a monitor screen. In this 3D scene, as illustrated in Figure 2, we define g as the gaze location estimated by the eye-tracking device projected on the screen, n as the direction of the normal of the monitor screen plane, and t as the wearer's actual gaze location on the screen (ground truth for g). We used the two-dimensional raw gaze estimation from the camera scene image in the wearer's three-dimensional real-world scene. These quantities are measured in a three-dimensional Cartesian coordinate system, with the origin being the center of the scene camera and the z-axis aligned with the normal of the scene camera.

Our approach involves two neural networks: a correction neural network (*CorNN*) and a calibration neural network (*CalNN*), as outlined in Fig. 3. The *CorNN* uses gaze estimation (g) and screen orientation (n) as inputs to predict the actual gaze location (t). The *CorNN* is directly responsible for correcting the gaze estimation error induced by the parallax effect and the scene camera lens distortion. The *CalNN* is the reciprocal of *CorNN*, which predicts g from t and n. The *CalNN* is used to calculate *wearer-specific offsets*, an input parameter required by *CorNN* that captures the estimation bias inherent to individual wearers. The details of this framework are described in the following subsections.

3.1 Data collection

We acquire gaze estimations g_i from a set of `fixation points` characterized by gaze target locations t_i (ground truth for g_i) and screen orientations n_i. We developed a graphical interface for data acquisition, as shown in Fig. 2.

The graphical interface displayed four ArUco markers and a visual target at each corner of the screen. The 3D location of the screen was estimated using the four ArUco markers, while the wearers were instructed to focus on the center of a visual target during the data collection. The visual target was a ring-shaped animated object consisting of two rings, a cross, and a black dot in the center. The cross rotated continuously to facilitate visual attention to the central point. The dot and the inner ring were resizable, and we adjusted their sizes to be distinctly visible at the observing distance. We used the realtime-network-api [25] provided by Pupil Labs to access the data from the eye tracker, including the scene camera view and a 2D raw gaze estimation in the scene camera coordinates (in pixels). While collecting data, the wearer-specific offset on the Invisible companion device remained at (0, 0), its default setting.

The training dataset was established on a single-wearer recording, referred to as `Wearer 0`. For this recording, the wearer's head was stabilized on a chin rest, and the monitor screen was mounted on a UR5e robot arm facing the wearer (see Fig. 2). The robot arm was programmed to move the screen through a regular grid of 3D waypoints sequentially, generating various `fixation points` within the field of view of the scene camera. Upon arriving at each location within the grid, the robot arm remains stationary to record a new `fixation point`. The eye-tracking device's estimated 2D and 3D gaze locations and the eventual wearer's head movements were monitored during recording. When the wearer's head and gaze were observed to have stabilized according to the latest device output, we recorded an estimated 3D gaze location g,[1] the real target location t, and the camera's relative orientation to the screen n. Upon completing this recording, the robot arm moved the screen

[1] g was obtained by offsetting the latest gaze estimation using the average difference between the device's outputs and the actual target locations over a window of several samples, excluding outliers.

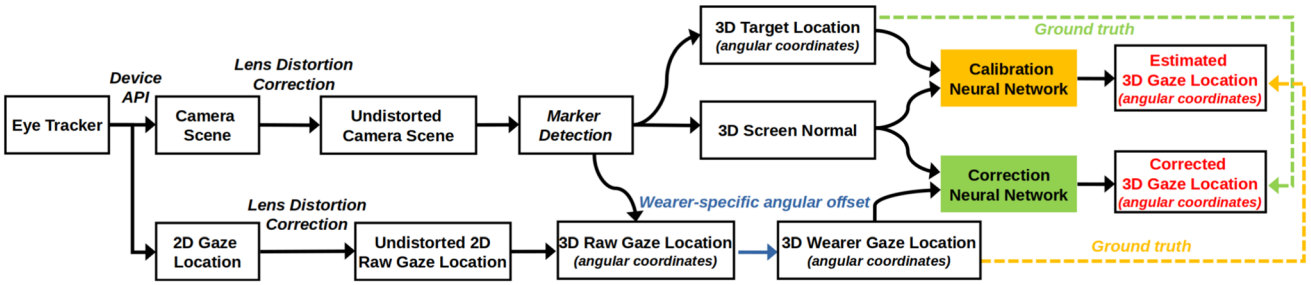


Fig. 3 The proposed framework for wearable eye tracker correction. The eye tracker's camera scene and 2D gaze location are collected using the device's API. Additionally, 2D positions of ArUco markers on the screen are recorded to establish the 3D pose of markers and screen. The camera lens distortion is compensated on both the 2D camera scene images and corresponding 2D gaze locations. The estimated 3D pose of markers and the screen was used for network inputs to correct the 3D gaze location. Green elements are part of the gaze correction neural network. Orange elements are part of the Calibration Neural Network. Dashed arrows show the data used as the ground truth output for the corresponding networks during training. The 3D target location is present for training and calibration purposes only

to the next grid point. Using the robot arm, 10405 samples were recorded within a bounding box about 1.5m wide, 1.2m high, and 1.5m deep, centered along the scene camera axis at about 500 mm from the wearer.

The testing dataset includes data from ten independent wearers. The recordings use regular monitor screens mounted on a desk. Similar to the setting for training data collection, the wearers' heads were stabilized on a chin rest facing the screen. Instead of moving the screen to create different fixation points, the screen is now fixed. At the same time, the visual target appears at various locations on the screen sequentially, drawing a grid of ten columns by eight rows for each recorded distance. The recording is repeated for each wearer at distances 50, 65, 80, 100, and 130 cms from the screen, resulting in a record of 400 samples per test wearer. We also include the data from **Wearer 0** as a control group.

3.2 Preprocessing training data

The collected data are preprocessed before training the neural networks through the following steps.

3.2.1 Converting Cartesian coordinates to angular coordinates

Let the p be either g (gaze estimate) or t (target location). The first step of preprocessing is to convert p into angular coordinates $\dot{p} = (\theta, \phi, d)$, where θ and ϕ as the angles between p and the normal of the camera (z-axis) along the axes x and y, respectively, and d as the distance between p and the camera's nodal point. This conversion is necessary since the error tolerance of eye-tracking devices is associated with angles.

3.2.2 Augmenting the data set

We introduce an augmentation technique to our training dataset that artificially generates gaze estimation values with various potential screen orientations. Suppose a `fixation point` (g, t, n) is recorded from the wearer, where g becomes $\dot{g}$ after conversion to angular coordinates. When a screen orientation n^+ differs from n while the target t remains at the same location, the eye-tracking device would return a gaze estimation g^+, whose angular representation $\dot{g}^+$ has the same ϕ and θ values as $\dot{g}$ but a different distance d^+.

For each t, we generate 201 simulated n^+ values such that their angles with line (O, t) remain below 60°, where O is the center of the camera $O = (0, 0, 0)$. For each n^+, we compute its corresponding g^+, such that the ϕ and θ components of its angular representation are the same as $\dot{g}$. Only its component d is modified so that g^+ lies on the plane (t, n^+).

3.2.3 Adding the wearer-specific offset

The gaze estimation in angular coordinates $\dot{g}_i$ needs adjustment through the *wearer-specific offset* to account for the biases inherent to individual wearers. We define the wearer-specific offset ρ as an angular shift along the x-axis and y-axis of the camera, (θ, ϕ). This offset is a weighted average of the difference between $\dot{t}_i$ and $\dot{g}_i$, calculated according to Eq. (1),

$$\rho = (\rho_\theta, \rho_\phi) = \frac{\sum_i \omega_i (\dot{t}_{i_\theta} - \dot{g}_{i_\theta}, \dot{t}_{i_\phi} - \dot{g}_{i_\phi})}{\sum_i \omega_i}, \tag{1}$$

where ω_i are weights that prioritize records closer to a defined point $C = (0, 0, 2000)$ situated 2 ms from the device, as suggested by the manufacturer.

$$\omega_i = \left(1 - \frac{\|t_i - C\|}{\max_j \|t_j - C\|}\right)^2 \tag{2}$$

We then compute the shifted gaze estimation set $\{\dot{g}'_i\} = \{(\theta'_i, \phi'_i, d'_i)\}$ by adding (ρ_θ, ρ_ϕ) to the gaze estimation in angular coordinates $\dot{g}_i = (\theta_i, \phi_i, d_i)$ and re-projecting onto the screen plane,

$$\dot{g}'_i = (\theta'_i, \phi'_i, d'_i) = (\theta_i + \rho_\theta, \phi_i + \rho_\phi, d'_i), \tag{3}$$

where d'_i is calculated so that $\dot{g}'_i$ lies on the screen plane (t, n).

Remark 1 The primary issue of calculating the offset according to Eqs. (1) and (2) is that the offset puts much more weight on samples close to the manufacturer-specified location $C = (0, 0, 2000)$ than the remaining samples. However, later on, we want our calibration process to consider samples from locations of different distances equally, rather than overly focusing on a predetermined location. Therefore, when adjusting our framework to the testing wearers, we use *CalNN* to calculate the wearer-specific offset that does not bias toward any specific location (see Sect. 3.4).

3.3 Networks structure and training

The *CorNN* and *CalNN* have identical network architectures but are trained independently. The neural networks comprise 20 fully connected layers with 500 hidden units. Each fully connected layer is followed by a leaky rectified linear unit (ReLU) and dropout layer [26]. We applied a dropout rate of 0.3 to obtain robustness against noisy measurements. A residual connection between the gaze coordinates and the neural network output was used [27]. The residual connection of angular coordinate data provides robust correction by preventing angular warping. *CorNN* takes $(n, \dot{g}')$ as input and is trained to predict $\dot{t}$, while *CalNN* takes $(n, \dot{t})$ as input and predicts $\dot{g}'$. We use the mean squared error MSE (Eq. (4) and (5)) as the loss function for training, as follows:

$$L_{CorNN}(n, \dot{g}') = \sum_i (\dot{t}_i - CorNN(n_i, \dot{g}'_i))^2, \tag{4}$$

$$L_{CalNN}(n, \dot{t}) = \sum_i (\dot{g}'_i - CalNN(n_i, \dot{t}_i))^2, \tag{5}$$

where L_{CorNN} and L_{CalNN} are losses of *CorNN* and *CalNN*, respectively. The neural networks were implemented in Python 3.7 using Keras 2.11.0. Training, validation, and testing were performed on a 12GB memory GTX

Titan Xp workstation (NVIDIA Santa Clara, California, USA) over 3000 epochs with a batch size of 100,000 samples.

3.4 Framework usages

With the networks trained, the first step of adapting the framework to a new wearer λ is to estimate their wearer-specific offsets $\rho^{\lambda} = (\rho^{\lambda}_{\theta}, \rho^{\lambda}_{\phi})$ as introduced in Sect. 3.2.3. *CalNN* plays an important role in calculating the offsets for new wearers. We collect a set of calibration `fixation points` on the new wearer $\{(g_k, n_k, t_k) | k \in (\text{Samples from } \lambda)\}$, without manually setting any offsets on the companion application. $\{n_k, t_k\}_k$ are used as inputs for the *CalNN*. The wearer-specific ρ^{λ} is estimated as the average difference between $CalNN(n, \dot{t})$ and the angular representation of the captured $\{g_k\}$:

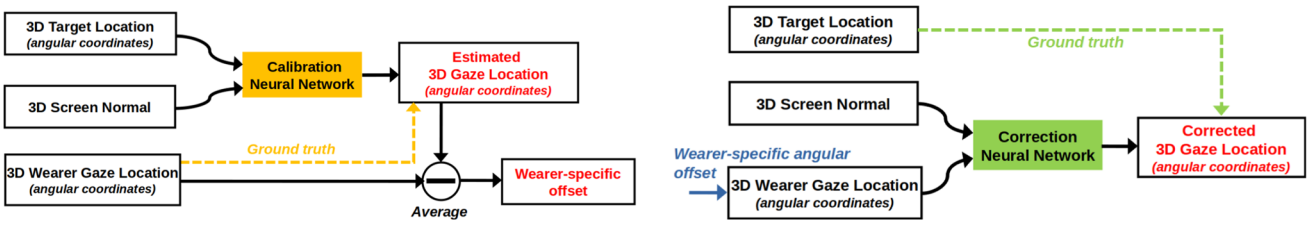
$$(\rho^{\lambda}_{\theta}, \rho^{\lambda}_{\phi}, *) = \frac{1}{m} \sum_{k} CalNN(n_k, \dot{t}_k) - \dot{g}_k, \tag{6}$$

See Fig. 4a for an illustration. Note that *CalNN* enables us to treat all samples with equal weights, not biasing toward samples near any specific locations like Eq. (1) and (2).

We then offset the angular representation of $\dot{g}_k$ with the ρ^{λ} value to become $\dot{g}'_k$. *CorNN* then takes $(\dot{g}'_k, n)$ as inputs and outputs a corrected gaze location estimate (see Fig. 4b).

4 Evaluation

We evaluated the proposed framework for improving the accuracy of gaze estimation. A total of 20 wearers participated in the testing experiment, named below `Wearers 1 to 20`. In addition, the results show the performance of each studied method upon `Wearer 0`, who contributed to the training data. `Wearer 0` is presented here as a control test. His results are not included in cross-wearer measurements. The wearers' heads were stabilized on a chin rest facing the center of the screen. Unlike the training data collection procedure, the testing experiment used regular monitor screens mounted on a desk: The screen was fixed while the visual target appeared at different locations on the screen, stepping through a grid of ten columns by eight rows sequentially in one recording. The recording is repeated for each wearer at distances 50, 65, 80, 100, and 130 cms from the screen, resulting in 400 samples per test wearer. `Wearers 0 to 10` had regular visions without correction and no makeup. The `Wearer 10` had regular vision and wore makeup. The `Wearers 11 to 20` had vision corrections of factor -2 or less. The `Wearer 20` wore contact lenses during the experiment. We calculated the testing wearers' wearer-specific offsets with the help of the calibration network *CalNN*(t, n) as explained in Sect. 3.4. These wearer-specific offsets are then used with the *CorNN* to correct the gaze estimations of the testing wearers. The gaze estimation accuracy for our framework is compared to the *Baseline* estimation and the



(a) Calibration Neural Network (b) Gaze Correction Neural Network

Fig. 4 Gaze calibration and correction networks (dashed arrows indicate the ground truth in training)

CorPara methods. In the *Baseline* method, the wearer-specific offsets are calculated using Eq. (1) and the *CorNN* correction is not applied. The *CorPara* uses the *Baseline*'s data and applies a correction as presented in [13] with a calibration plane at a distance of 2 m (value determined to optimize the average correction upon every test wearer). Both results are formalized in angular coordinates (θ, ϕ, d). The angular error of an estimated gaze location γ with a given ground truth location τ is:

$$Angular\ Error(\gamma, \tau) = \|(\theta_\tau, \phi_\tau) - (\theta_\gamma, \phi_\gamma)\| \tag{7}$$

In addition to comparing the proposed framework to the *Baseline* and *CorPara* methods, we have also conducted an ablation study over the proposed network to observe the impact of each component on the whole framework's performance.

5 Results

Figure 6 shows the results of absolute angular error from the 21 participants. Figure 5 shows the average accuracy obtained with the *Baseline* method, the parallax correction method *CorPara*, and the proposed correction framework *CorNN* over every testing `wearer`. Figure 5 shows the proposed method has improved the average accuracy of every wearer, with an edge for wearers without vision correction and makeup. It is also noticeable that contact lenses significantly degrade the device's accuracy. Most `wearers` benefit from similar improvements to the control test wearer, i.e., `Wearer 0`. *CorNN* does not always perform a better correction compared with the *CorPara*, as with `Wearers 6, 15, 18,` and `20`. However, it did not deteriorate the average accuracy like *CorPara* did on `Wearers 4, 5, 7, 8, 10, 11, 13, 14,` and `16`. In addition, even when *CorPara* improves the average accuracy, it also introduces variability in some cases. Across wearers, *CorNN* consistently offers a strong balance of low error and low variance, which is ideal in practical systems.

Figure 6 shows how the estimation accuracy changes as the distance between the wearer and the screen changes. Figure 6a shows that the *Baseline* method's accuracy improves with the distance within a short range of [450, 1300] millimeters with improvement factors within about $2 \sim 3\times$. This tendency is also observable in *CorNN*'s results but with a lower impact, between $1 \sim 2\times$ (see Fig. 6d). Figure 6c , *CorPara* slightly improves

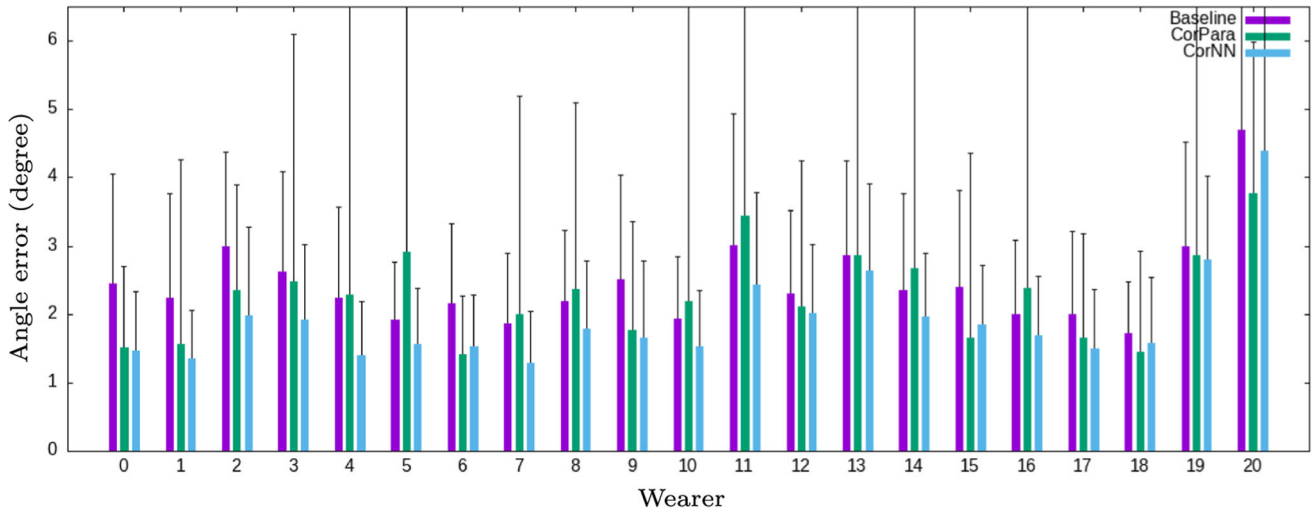


Fig. 5 Per-wearer average angular error and standard deviation. Angular distance between a target and the gaze estimations from the *Baseline*, *CorPara*, and *CorNN* methods. The overall test `wearers` (excluding `Wearer 0`) averages and standard deviations were 2.5(1.6), 2.3(4.0), and 2.0(1.4), respectively

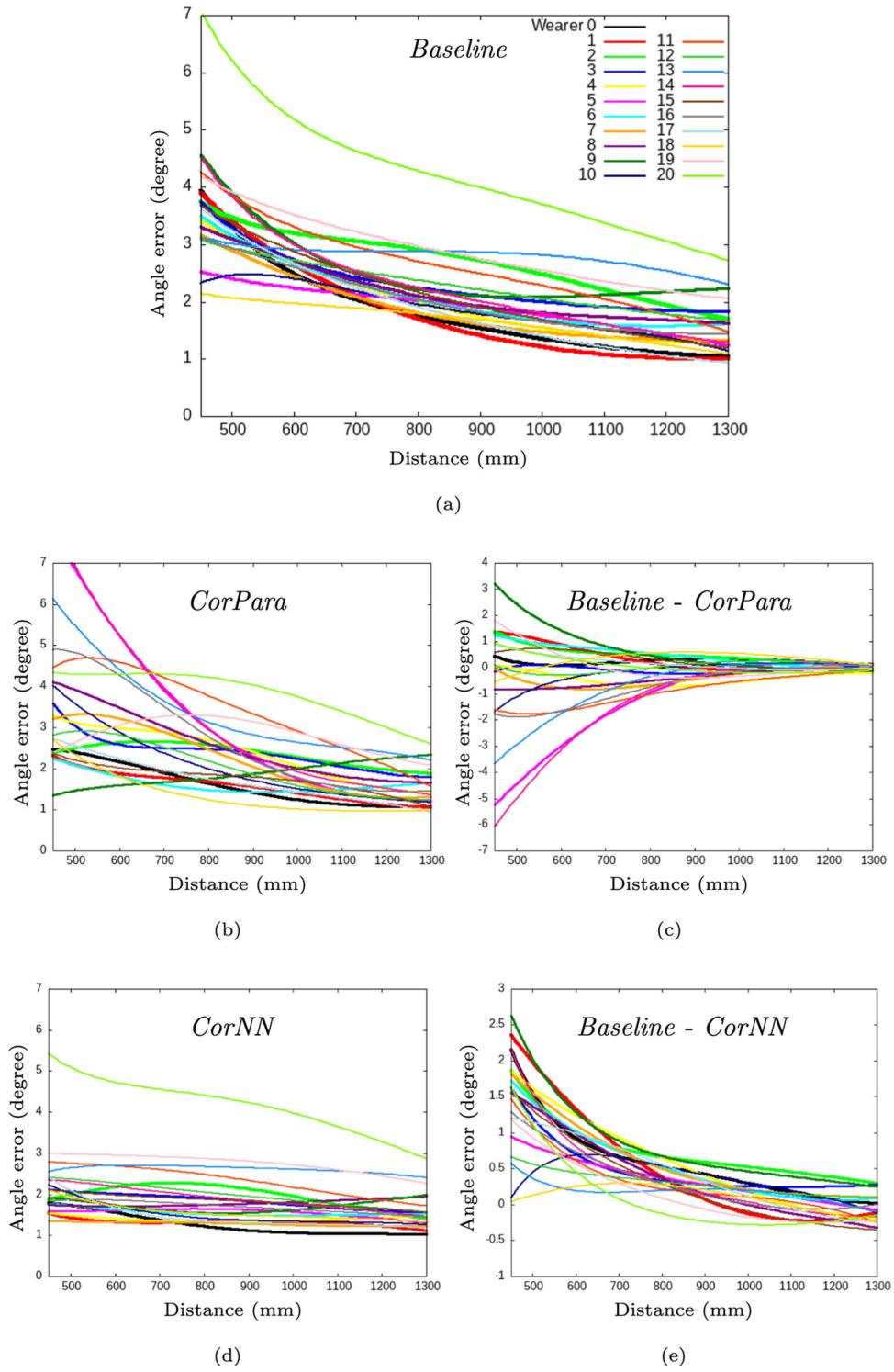


Fig. 6 Angular distance between a target and the gaze estimations from the *Baseline* method and the output of the *CorNN* framework. **a** Angular error tendencies of the *Baseline* method. **b** Angular error tendencies of the *CorPara*. **c** Improvement due to the *CorPara* ((**a**) minus (**b**)). **d** Angular error tendencies of the *CorNN*. **e** Improvement due to the *CorNN* ((**a**) minus (**d**))

the accuracy in mid-range and long-range distances for most of the wearers and at short-range distances for half of them. However, it also deteriorates the accuracy for half of the wearers up to $1\times$ their original values. Based on Fig. 6e , the correction network had overall significantly increased the accuracy at a very short distance $\lesssim 1m$ for most wearers, with an improvement between 0.5° and 2° and had a positive impact at a longer range for some of them. This result matches our expectations since lens distortion and parallax effects are most significant in short ranges. Thus, the benefit of applying the networks should be the most prominent. The only exceptions were the

Table 1 Average angular error variations in percentage, if mean absolute error (MAE) loss was used instead of mean squared error (MSE), Cartesian representation was used instead of the angular representation, the residual was not used, and the augmentation was not used

Variable	MAE loss	Cartesian rep	w/o Residual	w/o Augment
Avg. Ang. Err	+1.7%	+4.8%	+3.1%	+6.9%

Table 2 Average angular error variations in percentage when using 500, 300, or 100 hidden units and 20 or 10 layers (X marks the original configuration)

Layers/Hidden units	500	300	100
20	X	+2.7%	+6.6%
10	+1.7%	+3.0%	+6.6%

accuracy of the `Wearer 10`'s measurements at mid-range (around $1m$) and several wearers at distances over $1m$. The deterioration varies between $0.1°$ and $0.3°$ approximately.

An ablation study has been conducted to evaluate the impact of each characteristic of the proposed framework introduced in Sect. 3. Tables 1 and 2 report the average angular error variations upon all testing wearers of the framework with various configurations as a percentage between the original and modified frameworks.

This ablation study showed that, on the one hand, the angular representation, the training data augmentation, and the number of hidden units had the most significant performance impact; on the other hand, the choice of the loss function, the use of the residual, and the number of layers contributed half as much to the framework's performance.

6 Conclusion

Our proposed method aims to improve head-mounted eye-tracking devices' accuracy by attenuation for the distortions implied essentially by the parallax effect, considering the distance between the observer and the gazed object. This method consists of embedding gaze coordinates into the 3D real-world scene and using two neural networks, one for correction and one for calibration. The method was trained on a single-wearer data set and tested on eleven wearers using the Pupil Labs Invisible eye-tracking device. The results of those tests have shown that the proposed method could significantly improve the device's accuracy for every tested wearer compared to the *Baseline* method. Further study will include multi-wearer training and few-shot learning techniques to optimize the correction network to its current wearer, integrating the wearer-specific parameters into the neural network architecture.

Author Contributions Jerome Charton contributed as the main author—jcharton@mgh.harvard.edu. Tianpeng Zhang contributed as the second author (Robotic assistance). Na Li contributed as co-principal investigator. Quanzheng Li contributed as principal investigator.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Declarations

Conflict of interests The authors have no relevant financial or non-financial interests to disclose.

Ethical approval Does not apply to this study.

Consent to participate Does not apply to this study.

Consent to publication The authors affirm that human research participants provided informed consent for publication of the images in Figs. 5 and 6.

References

1. Shen Y, Mo X, Krisciunas V, Hanson D, Shi BE (2023) Intention estimation via gaze for robot guidance in hierarchical tasks. In: Annual Conference on Neural Information Processing Systems pp 140–164. PMLR
2. Berg J, Lottermoser A, Richter C, Reinhart G (2019) Human-robot-interaction for mobile industrial robot teams. Proc CIRP 79:614–619
3. Di Maio M, Dondi P, Lombardi L, Porta M (2021) Hybrid manual and gaze-based interaction with a robotic arm. In: 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pp 1–4 . IEEE
4. Chadalavada RT, Andreasson H, Schindler M, Palm R, Lilienthal AJ (2020) Bi-directional navigation intent communication using spatial augmented reality and eye-tracking glasses for improved safety in human-robot interaction. Robot Comput-Integr Manuf 61:101830
5. Kratzer P, Bihlmaier S, Midlagajni NB, Prakash R, Toussaint M, Mainprice J (2020) Mogaze: a dataset of full-body motions that includes workspace geometry and eye-gaze. IEEE Robot Autom Lett 6(2):367–373
6. Čegovnik T, Stojmenova K, Jakus G, Sodnik J (2018) An analysis of the suitability of a low-cost eye tracker for assessing the cognitive load of drivers. Appl Ergon 68:1–11
7. Xu J, Min J, Hu J (2018) Real-time eye tracking for the assessment of driver fatigue. Healthc Technol Lett 5(2):54–58
8. Akshay S, Abhishek M, Sudhanshu D, Anuvaishnav C (2021). Drowsy driver detection using eye-tracking through machine learning. In: 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC), pp 1916–1923 . IEEE
9. Khosravan N, Celik H, Turkbey B, Cheng R, McCreedy E, McAuliffe M, Bednarova S, Jones E, Chen X, Choyke P, et al (2017) Gaze2segment: a pilot study for integrating eye-tracking technology into medical image segmentation. In: Medical Computer Vision and Bayesian and Graphical Models for Biomedical Imaging: MICCAI 2016 International Workshops, MCV and BAMBI, Athens, Greece, October 21, 2016, Revised Selected Papers 8, pp 94–104 . Springer
10. Wang B, Aboah A, Zhang Z, Bagci U (2023) Gazesam: What you see is what you segment. arXiv preprint arXiv:2304.13844
11. Wang S, Ouyang X, Liu T, Wang Q, Shen D (2022) Follow my eye: using gaze to supervise computer-aided diagnosis. IEEE Trans Med Imag 41(7):1688–1698
12. Ma C, Zhao L, Chen Y, Wang S, Guo L, Zhang T, Shen D, Jiang X, Liu T (2023) Eye-gaze-guided vision transformer for rectifying shortcut learning. IEEE Trans Med Imag. https://doi.org/10.1109/TMI.2023.3287572
13. Narcizo F.B, Hansen D.W (2015) Depth compensation model for gaze estimation in sport analysis. In: 2015 IEEE International Conference on Computer Vision Workshop (ICCVW). IEEE, USA . https://doi.org/10.1109/iccvw.2015.107
14. Cognolato M, Atzori M, Müller H (2018) Head-mounted eye gaze tracking devices: an overview of modern devices and recent advances. J Rehabil Assist Technol Eng. https://doi.org/10.1177/2055668318773991
15. Tonsen M, Baumann C, Dierkes K (2020) A high-level description and performance evaluation of pupil invisible. ArXiv **abs/2009.00508**
16. ACM, Pittsburgh, Pennsylvania (2012) https://doi.org/10.1145/2370216.2370366
17. Ehinger BV, Groß K, Ibs I, König P (2019) A new comprehensive eye-tracking test battery concurrently evaluating the pupil labs glasses and the EyeLink 1000. PeerJ 7:7086. https://doi.org/10.7717/peerj.7086
18. Narcizo FB, Santos FED, Hansen DW (2021) High-accuracy gaze estimation for interpolation-based eye-tracking methods. Vision 5(3):41. https://doi.org/10.3390/vision5030041
19. Hooge ITC, Niehorster DC, Hessels RS, Benjamins JS, Nyström M (2022) How robust are wearable eye trackers to slow and fast head and body movements? Behav Res Methods. https://doi.org/10.3758/s13428-022-02010-3
20. Franchak J.M, Yu C (2022) Beyond Screen Time: Using Head-mounted Eye Tracking to Study Natural Behavior 62: 61–91. Elsevier, ??? . https://doi.org/10.1016/bs.acdb.2021.11.001 . https://linkinghub.elsevier.com/retrieve/pii/S0065240721000409 Accessed 12 Sep 2023
21. Khamis M, Buschek D, Thieron T, Alt F, Bulling A (2018) EyePACT. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 1(4):1–18. https://doi.org/10.1145/3161168
22. Gibaldi A, DuTell V, Banks M.S (2021) Solving parallax error for 3d eye tracking. In: ACM Symposium on Eye Tracking Research and Applications. ACM, Stuttgart, Germany . https://doi.org/10.1145/3450341.3458494

23. Liu M, Li Y, Liu H (2020) 3d gaze estimation for head-mounted eye tracking system with auto-calibration method. IEEE Access 8:104207–104215. https://doi.org/10.1109/ACCESS.2020.2999633
24. Tobii: Tobii Pro Glasses 3 | Latest in wearable eye tracking — tobii.com. https://www.tobii.com/products/eye-trackers/wearables/tobii-pro-glasses-3. Accessed 11 Feb 2023
25. Prietz P, Kassner M, García M (2023). Pupil Labs Realtime Network API . https://github.com/pupil-labs/realtime-network-api
26. Gal Y, Ghahramani Z (2016) Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: International Conference on Machine Learning 1050–1059.PMLR
27. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition pp 770–778

Authors and Affiliations

Jerome Charton[1] · Tianpeng Zhang[2] · Na Li[2] · Quanzheng Li[1]

✉ Jerome Charton
jcharton@mgh.harvard.edu

[1] Harvard Medical School, Massachusetts General Hospital, Boston, USA

[2] School of Engineering and Applied Sciences (SEAS) at Harvard University, Boston, USA