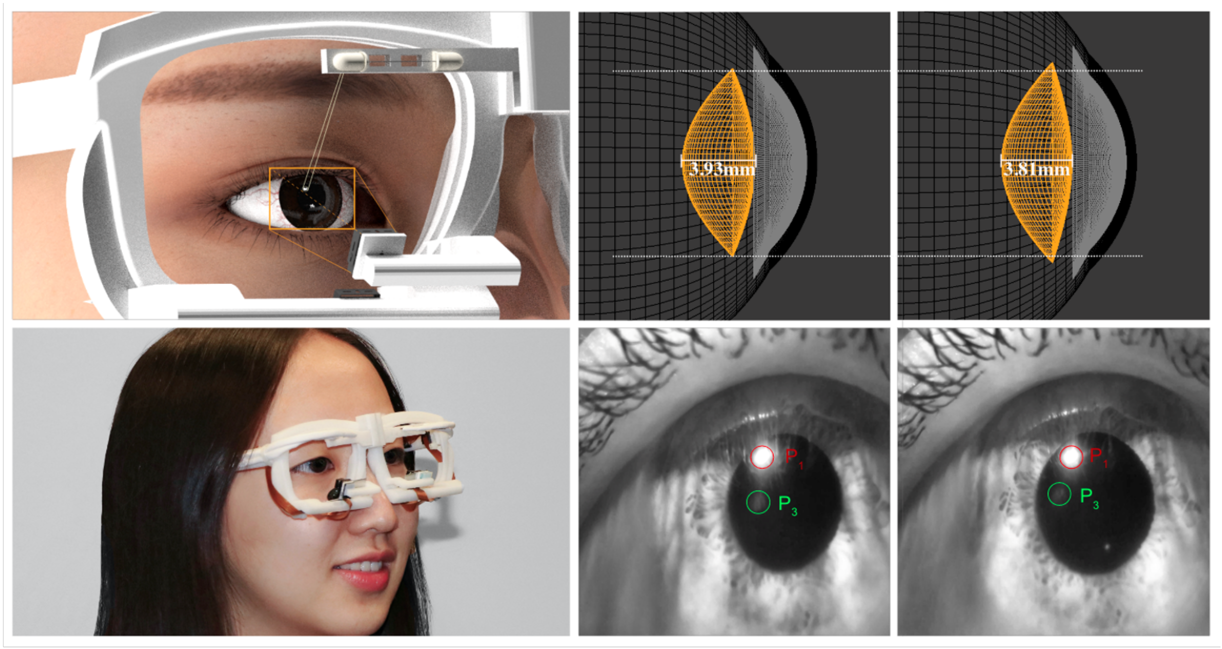
Improved vergence and accommodation via Purkinje Image tracking with multiple cameras for AR glasses

Conny Lu[*] Praneeth Chakravarthula[†] Yujie Tao[‡] Steven Chen[§] Henry Fuchs[¶]

University of North Carolina at Chapel Hill



We design a multi-camera multi-LED based compact eye tracker for improved accommodation and gaze estimation in AR eyeglasses, by effectively tracking the sensitive higher-order Purkinje images. We achieve significant improvements in estimating monocular vergence and accommodation compared to the state-of-the-art.

The ciliary muscle constricts making lens thicker to accommodate near-distances. Purkinje reflections from the cornea (P1) and the anterior lens surface (P3) are captured by eye tracking camera.

The ciliary muscle relaxes causing the lens to flatten to accommodate far distances. This causes the Purkinje image from the dynamic anterior lens surface (P3) to move, while Purkinje image from cornea (P1) is stationary.

Abstract

We present a personalized, comprehensive eye-tracking solution based on tracking higher-order Purkinje images, suited specifically for eyeglasses-style AR and VR displays. Existing eye-tracking systems for near-eye applications are typically designed to work for an on-axis configuration and rely on pupil center and corneal reflections (PCCR) to estimate gaze with an accuracy of only about 0.5°to 1°. These are often expensive, bulky in form factor, and fail to estimate monocular accommodation, which is crucial for focus adjustment within the AR glasses.

Our system independently measures the binocular vergence and monocular accommodation using higher-order Purkinje reflections from the eye, extending the PCCR based methods. We demonstrate that these reflections are sensitive to both gaze rotation and lens accommodation and model the Purkinje images' behavior in simulation. We also design and fabricate a user-customized eye tracker using cheap off-the-shelf cameras and LEDs. We use an end-to-end convolutional neural network (CNN) for calibrating the eye tracker for the individual user, allowing for robust and simultaneous estimation of vergence and accommodation. Experimental results show that our solution, specifically catering to individual users, outperforms state-of-the-art methods for vergence and depth estimation, achieving an accuracy of 0.3782°and 1.108 cm respectively.

Index Terms: Augmented Reality—Eye tracking—Eye tracking Techniques—Purkinje images; Human-Computer Interaction

[*]e-mail:connylu@cs.unc.edu
[†]e-mail:cpk@cs.unc.edu
[‡]e-mail:yujiet@cs.unc.edu
[§]e-mail:stevenc@unc.edu
[¶]e-mail:fuchs@cs.unc.edu

1 Introduction

We have recently seen dramatic developments in AR and VR with commercially available devices that offer a resolution, tracking, and latency sufficient to provide compelling experiences to consumers. In the next few years, the field will be ready to integrate more sophisticated technologies and solve long-standing problems in AR displays, such as wide field-of-view (FOV), high resolution, large eye box, and accommodation support, all in an eyeglasses form factor. Such glasses will provide a comfortable view, simultaneously, of both real and virtual imagery.

Vergence and accommodation The distance to which the eyes verge to maintain a single binocular vision of an object is called the vergence distance, and the distance to which the eyes must accommodate to bring the image of that point in space to sharp focus is the accommodation depth. Vergence and accommodation are neurally coupled, i.e., one drives the other. To maintain a well-focused view of an object, the vergence distance and the accommodation distance should match the focal distance of the object in space. While this naturally occurs in people with normal vision, it is mismatched for people with refractive errors in eyes, e.g., near-sighted or far-sighted. The situation becomes complex when it comes to VR and AR displays, where the focal, vergence and accommodation depth are often mismatched for all people [5].

Need for vergence and accommodation tracking Both real and virtual imagery can be simultaneously presented at a correct focal depth if rapid, accurate, and robust eyetracking were included, resulting in a system that can measure the user's object of attention in both the real and virtual world. Accurate and rapid vergence tracking can potentially also enable gaze-contingent rendering and displays [2, 41]. However, note that the vergence depth can only approximate the accommodation depth of the eye [48] and typically occurs only in users with normal vision. Therefore, we must not only measure vergence accurately but also separately measure monocular accommodation. Suppose we can determine the accommodation supported by the eye's lens. In that case, the virtual imagery in an AR display can be set to the focal distance that matches the user's current accommodation distance [6,7]. Furthermore, any prescription power for compensating the unmet accommodation needs due to refractive errors in the eye can also be applied via an external focus adjustment for viewing the real-world [5]. Such a display combined with robust monocular vergence and accommodation tracking of eyes can also potentially act as auto-focus "everyday" prescription eyeglasses, with the virtual display content turned off.

Achieving enhanced eyetracking Estimating both monocular vergence and accommodation independently and simultaneously requires tracking of not just the pupil but also the continuous change in lens shape. Sophisticated equipment like an autorefractor or a Shack-Hartmann wavefront can measure the change in lens accommodation by analyzing the reflected wavefront from the retina through the eye but are extremely bulky to go with eyeglasses-style displays. Instead, we look at imaging the Purkinje reflections, the reflections from various surfaces of the eye, using miniature cameras that fitted into the eyeglasses frame. Since these reflections occur from curved surfaces of the eye, they are sensitive to eye gaze changes and accommodation and are well-suited for vergence and accommodation tracking.

We believe that the future AR eyeglasses will be individually owned and personalized akin to the current day personal smartphones. Such personalized AR eyeglasses will have eyetrackers and prescription correction lenses, also catering to the individual user. In this work, we demonstrate a compact eyeglasses-style eyetracker based on tracking higher-order Purkinje reflections from the eye. Our eye tracker is customized for a specific user, and our approach to achieving Purkinje images based vergence and accommodation tracking are threefold:

Designing the eyeglasses: The higher-order Purkinje reflections occurring from the deeper layers of the eye are typically faint and often difficult to capture robustly and consistently. To this end, we introduce a sophisticated, physically-accurate, and anatomically-informed synthetic eye model to simulate the Purkinje reflections from the eye. We use this synthetic eye model to optimize for the positions of cameras and infrared LEDs to robustly and consistently capture the Purkinje reflections. We experimentally find that our eye tracker outperforms the existing state-of-the-art eyetracking methods.

Estimating gaze and accommodation: The relationship between the position of the Purkinje reflections in space and the corresponding gaze and accommodation change is non-linear. To measure the eye's underlying gaze and accommodation from the Purkinje images as seen by the eyetracking cameras, we use a convolutional neural network (CNN) as the non-linear function approximator. It parameterizes the relationship between the gaze vergence and accommodation depth, and the Purkinje images. This allows us not only to estimate the vergence and accommodation robustly but also at faster frame rates.

Collecting ground truth training data: We design a special multi-plane display setup with adjustable focal planes to collect the ground truth calibration data required to train the neural network. The network is then used to estimate the gaze and accommodation from the Purkinje images. We anticipate that the calibration data can be collected directly via an eyeglasses-style focus-supporting near-eye display in the future.

This serves as a best first approximate for the configuration of cameras and LEDs on the eyetracking eyeglasses. We further refine to precisely fit the eye physiology of each user, which we note is a one-time effort. Specifically, we make the following contributions in this work:

- We introduce higher-order Purkinje image tracking as a robust method for fast and continuous estimation of vergence and accommodation, independently and simultaneously, achieving improved accuracy.
- We design and fabricate customized eyeglasses, hosting multiple cameras and LEDs for robust and consistent tracking of higher-order Purkinje images in AR eyeglasses. To this end, we also create an improved anatomically-informed digital 3D model of the eye to simulate the behavior of Purkinje images.
- We propose a convolutional neural network to learn the non-linear mapping between the characteristics of the Purkinje images captured by multiple cameras, and the vergence and accommodation of the eye. We also create an annotated dataset. We will release all code, datasets, and design models.
- We assess our proposed user-specific eyetracking solution experimentally with a hardware prototype and show that our eyetracker achieves improved accuracy for both monocular and binocular vergence and accommodation estimation.

2 Related Work

In this section, we review eyetracking for head-mounted and near-eye displays for virtual and augmented reality, emphasizing camera-based eyetracking techniques.

2.1 Eyetracking systems

Eyetracking systems for near-eye displays can be broadly classified into three categories: 1) electro-oculography(EOG), 2) magnetic eyetracking, and 3) video-oculography (VOG). Our work belongs to video-oculography.

Video-oculography (VOG) uses camera-based eyetrackers to track the features in eyes such as the limbus (border of the cornea and sclera), pupil, and reflections from the cornea and other surfaces of the eye (Purkinje reflections). A common technique used to perform VOG based eyetracking is pupil center corneal reflection (PCCR). In PCCR, one or more infrared (IR) light sources are used in combination with IR cameras to track the corneal glint relative to the pupil center during the eye motion [21]. Extending this methodology, the center of the iris is tracked instead of the center of the pupil [50]. Dual-Purkinje image-based eyetrackers make use of not only corneal reflections but also reflections from the posterior lens surface to track the gaze angle [8, 11] and sometimes gaze depth [31]. Dual-Purkinje gaze trackers are generally more robust than PCCR tracking, but are typically bulky, relatively difficult, and restricted to controlled laboratory environments, making it difficult to use in AR eyeglasses [8]. Recent work by Lee et al. [31] attempts to implement dual-Purkinje eye tracking in a compact form factor that achieves a compromised gaze accuracy of only 0.96° horizontal and 1.6° vertical, and a 4.59 cm error in accommodation depth. Our system instead tracks reflections from the interior surfaces of the eye lens, similar to early attempts by Itoh et al. [23].

2.2 Camera-based gaze estimation

Camera-based gaze estimation methods utilize computer-vision techniques to predict the line of sight for each user's eye at any given instant, thus continuously tracking the gaze. Such gaze estimation and tracking approaches can be primarily categorized into three methods: feature-based, model-based, and appearance-based [66].

Feature-based gaze estimation localizes key features of the eye, such as pupil (or iris) center and eye corners, and map these features to the corresponding gaze coordinates on the screen space [47,58]. On the other hand, model-based methods use a geometric model of the human eye that is fitted to the pupil image frames captured by the camera for gaze estimation [53,59,61]. Appearance-based gaze estimation methods utilize eye images, sometimes along with a set of features extracted from the images, to learn the mapping between the images and the corresponding gaze via machine learning techniques [45,52,67]. Recent research has also seen success with a combination of the three methods [25,60,62].

2.3 Vergence and accommodation depth estimation

As discussed above, tracking both vergence and accommodation is necessary for comfortable viewing experience in AR eyeglasses for all users [5]. Common techniques for 3D gaze tracking can be categorized into gaze ray-casting, vergence-based, accommodation-based, and vestibulo-ocular reflex-based methods. Gaze ray-casting methods measure the depth to the first object in the scene where the imaginary gaze vectors from both eyes intersect [9,55]. However, this technique works well only when the gaze vectors intersect at an object and fail to address occlusion ambiguity. Vergence-based methods, on the other hand, do not rely on objects in the scene but on the fact that the two eyes roll in opposite directions to foveate on objects at different depths and simultaneously focus on them. Using the gaze disparity between the left and right eyes [12, 14], and/or inter-pupillary distance [3,35], the vergence between the two eyes and hence the fixation depth can be calculated via triangulation. Accommodation-based methods leverage the anatomical changes in the eyes, such as changing lens curvature during accommodation, to estimate the focal depth. Popular monocular depth estimation devices are the autorefractor [34] and the Shack-Hartmann wavefront sensor [37]. These, which are often used for medical and research purposes, provide a good estimate of accommodation but are very bulky and impractical to be integrated into AR eyeglasses.

2.4 Rendering anatomically correct eye models

An eyeglasses-style eye tracker requires an optimized placement of cameras and IR LEDs to efficiently and robustly track higher-order Purkinje images. A near-accurate anatomically correct 3D graphics eye model is necessary for designing such eyetracking eyeglasses, and hence we review several relevant synthetic eye models here.

Świrski et al. [14] were perhaps the early researchers to use a simulated eye model and synthetic gaze for eyetracking. Anatomically-aware graphics model of the eye from then increasingly provides a fineness to model eye movement and eye texture. Generative adversarial network (GAN) has been leveraged to improve the quality of synthetic eye image [49]. Wood et al. [63] created a comprehensive and detailed eye model, including the sclera, pupil, iris, and cornea, using a collection of controllable eye-region scans. This model exhibits both realistic shape changes such as pupillary dilation and textures such as iris and scleral veins. Kim et al. [25], extended Wood's model with additional anatomical details and offered a more sophisticated shading and higher resolution of rendering. However, the above eye models do not simulate the eye's accommodation behavior, which involves the crystalline lens shape change that is essential for tracking monocular accommodation in our eyetracking system.

As a more precise optical model of the human eye, the Arizona model [46] defines a series of surfaces including cornea, lens, and retina by radius, conic constant, IOR, etc. The accommodative mechanism in a young eye lens was quantitatively described by the geometric model from Reilly [43]. More recently, Zoulinakis et.al. [69] studied in detail the accommodation states as described by the three widely used optical eye models: Navarro [36], Arizona [46] and Liou-Brennan [33] found that all three models were able to simulate accommodation with the expected results.

2.5 Eye-gaze datasets

Existing gaze estimation methods that rely on data-driven approaches such as machine learning or deep learning, leverage both real and synthetic eye-gaze datasets. While real data can lead to potentially robust learning, synthetic eye datasets can relieve researchers from manual data collection and labeling. We review a few popular datasets used for learning-based gaze estimation, including both remote and near-eye gaze tracking.

Remote gaze tracking The *MPIIGaze* [68] includes 214k full-face images of 15 users, collected from remote laptop cameras. *UT Multi-view* [52] has 64K data from 50 testers from multiple views. *Columbia Gaze* [51] created a gaze dataset of 56 people and 5880 images that focused on gaze locking. *EYEDIAP* [19] collected eye image data from RGB and RGB-D cameras from 16 people. *GazeCapture* [26] contains data from over 1450 people of almost 2.5M frames. *BIOID* [24] database consists of 1521 images, each 384×288 pixel, of 23 different people. *WebGazer* [38] is a 51-participant benchmark dataset from webcam videos.

Near-eye gaze tracking *InvisibleEye* [56] is a dataset of 280k recorded images with millimeter-size RGB cameras. *NVGaze* [25] dataset has both real data and synthetic data of 2M infrared images of eyes rendered at 1280×960 resolution. *SynthesEyes* [63] has 11.4k eye images sampled at different illumination conditions. Świrski and Dodgson [54] created an IR image dataset with 159 frames. *LPW* [57] has 66 eye region videos for the pupil detection evaluation. *ElSe* [18] contributed 55,000 IR eye images and *ExCuSe* [16] provided 38,000 new, hand-labeled eye images.

Our work draws inspiration from a variety of prior art but offers new features and implementation choices. We especially leverage the sensitivity of third Purkinje reflections from the anterior eye-lens surface to vergence and accommodation changes, and robustly track them with our CNN calibrated multi-camera and multi-LED eye tracker, all in an eyeglasses form factor.

3 Purkinje Images

Purkinje-Sanson images, or simply Purkinje images, are reflections from the anatomical structure of the eye named after Czech anatomist Jan Purkyně and French physician Louis Sanson. Due to the imperfections, usually at least four Purkinje images are formed from the anterior (outer) and posterior (inner) surfaces of the eye's cornea and crystalline lens. Specifically, Purkinje image I (P1) is the reflection of light from the outer surface of the cornea, Purkinje image II (P2) from the inner surface of the cornea, Purkinje image III (P3) from the outer surface of the lens and the Purkinje image IV (P4) from the inner surface of the crystalline lens. Due to the imperfections in the eye's optical system, the light reflected from the lens surfaces can sometimes be reflected back into the eye at the cornea giving rise to fainter fifth and the sixth *entoptic* Purkinje reflections. The typical four Purkinje images play an important role in modern applications such as measuring tilt and decentration of intraocular lens [13], assessing the polarization optics of the cornea and lens [42] and detecting fake iris [30].

3.1 Intensity and motion of the Purkinje images

Purkinje images in the eyes are caused by incoming light from the world reflecting at several interfaces of the eye where a change in refractive index is seen, namely the anterior and posterior cornea and lens surfaces, before hitting the retina. A ray entering the eye undergoes both reflection and refraction at these interfaces according to Fresnel laws, as shown in Fig. 1, and results in the four common Purkinje images as discussed above. Further, the intensity of these images are also guided by the Fresnel equation

$$\mathcal{I} = \left|\frac{(n_2 - n_1)}{(n_2 + n_1)}\right|^2 \tag{1}$$

where n_1 and n_2 are the refractive indices before and after the reflecting surface. Parameterizing the eye using the well known Arizona eye model [46], it can be easily seen that Purkinje image I is the brightest and Purkinje image II is the faintest, with Purkinje images III and IV having about the same intensity. Given that Purkinje images I and II from the *static and thin* cornea are formed at the same location, this corneal glint is typically used as a feature for gaze tracking.

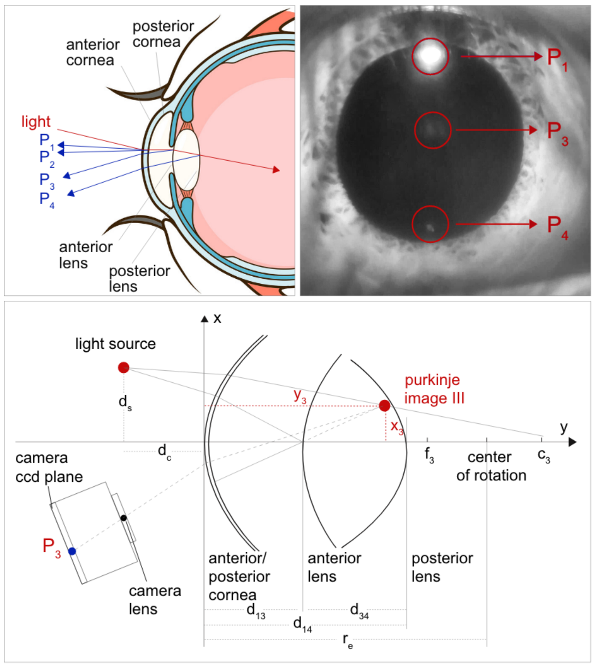


Figure 1: Top left: Purkinje images formation. Top right: Purkinje image in actual capturing. Bottom: Simplified eye structures from a mirror system standpoint, detailing the formation of P3 image captured by camera. P_1, P_2, P_3, and P_4 stand for Purkinje image I, II, III, and IV.

The *static* cornea supplies most of the focal power in the eye, while the *dynamic* crystalline lens provides the additional adjustments for accommodation. When the eye verges and accommodates to different points in 3D space, it causes the eyeball to rotate within the socket and the crystalline lens to change its shape to provide for the required accommodation power. This change in lens shape results in a change in the path of reflected light from the anterior and posterior surfaces of the crystalline lens, making Purkinje image III and Purkinje image IV appear in slightly different locations. Of course, the location of Purkinje reflections also change with gaze, when the eyeball moves inside the socket. Therefore, robustly tracking these reflections can provide improved gaze estimates and give information about the change in lens shape, hence the accommodation of the eye [1, 4, 10, 28].

3.2 Sensitivity to gaze and accommodation

The effect of gaze and accommodation is different on the different Purkinje images, owing to the respective profiles of surface that causes these reflections within the eye. We analyze the sensitivity of Purkinje images with gaze and accommodation by modeling the reflection and refraction at various surfaces of the eye, as shown in Fig. 1 [30]. To this end, we simulate an eye model with the characteristics of various anatomical surfaces parameterized by the Arizona eye model [46]. Raytracing through various surfaces of the eye and plotting the position of Purkinje images on the imaging sensor yield the relative sensitivities of Purkinje image I, III and IV with change in gaze relative to the pupil center, shown in Fig. 2. In simulation and experiments, we find that the third reflection is more sensitive to gaze (and accommodation) than the others, relative to the pupil center. However, as discussed in Sect. 3.1, not only is the intensity of the third Purkinje image significantly less compared to the first (Fig. 1), it is also optically formed on a different focal plane compared to the rest of the Purkinje images, making it out of focus and difficult to detect robustly together with the others.

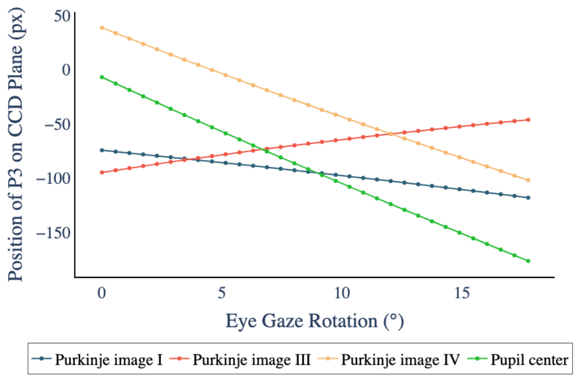


Figure 2: Purkinje images and pupil center position change during eye rotation.

3.3 Capturing the Purkinje images

As discussed above, the higher-order Purkinje images are sensitive to gaze and accommodation changes, and if tracked robustly, they can reveal information about the eye accommodation states. However, they are often difficult to capture due to their significantly reduced brightness. Moreover, the third Purkinje image, which we find to be most sensitive for improved eye tracking, is optically formed on a distant focal plane as compared to the others resulting in a defocus, making it even harder to capture. Due to the increased sensitivity, these reflections are also prone to quickly fall out of the field of view of the eye tracking camera with gaze changes.

To robustly track these reflections, the camera needs to be consistently in focus while also spanning a wide field of view to potentially track all of the movements of Purkinje images. Although the field of view of small eye tracking cameras that can be integrated into an eyeglass frame is typically narrow, a multitude of such cameras combined with multiple illuminating infrared LED sources cover a wider tracking field of view. Each camera and LED combination can then cover a smaller tracking field for the Purkinje images. Moreover, having these cameras slightly defocused can image all of the Purkinje images. Note that our concentration in this paper is only the third Purkinje image and how such a multi-camera multi-LED eye tracker design is robust enough to track Purkinje image III reflections. We optimize our eye tracker glasses to cover the working field of view with two cameras and two IR LEDs, which we discuss in the following sections.

4 System Setup

We devise an eyetracking system that achieves improved vergence and accommodation estimation in a compact eyeglasses-style form factor. Designing compact near-eye eye trackers for AR displays is a relatively an under-investigated problem. We note that much of the prior art in gaze tracking utilized on-axis configuration of eye tracking cameras suitable for VR use cases but unsuitable for AR as the cameras block the available field of view (e.g. NVGaze [25]). While Lee et al. [31] and Wu et al. [64] explored an unoptimized off-axis camera configuration, the closest work to ours is the InvisibleEye [56] tracker, in which the camera positions are optimized based on gaze tracking accuracy on synthetic datasets. In our work, we consider not only the typical pupil center and corneal glint but also the third Purkinje reflections. Given that higher-order Purkinje images are difficult to capture, as discussed in Sect. 3, we employ multiple cameras and multiple LEDs. However, this poses a challenging problem of designing the eyeglasses and the illuminating and imaging sources for efficient and consistent tracking of Purkinje images.

We want our eye tracker to be accurate, compact, and customizable to various users. To this end, we first model user and environment variables in the simulation and then optimize the positions of multiple cameras and LEDs by analyzing characteristics of Purkinje images. We also parameterize the movement of eyeglasses to simulate slippage and create a synthetic dataset to evaluate its impact on eyetracking. With the computed positions of cameras and LEDs, we design an eyeglasses frame that fits the user and 3D print it. Cameras and LEDs are then attached to the frame at corresponding slots. In the following sections, we lay out our mechanisms in simulation and hardware setup.

4.1 Simulation

One major benefit of simulation is its capacity to sample a large set of parameters to model both user and environment variables, which would be hard to implement in a real setup. We specifically focus on modeling detailed eye structures, such as the anterior and posterior cornea and eye lens, iris, pupil, and sclera. Current state-of-the-art works in eye rendering [25, 62, 63] have not paid attention to the eye lens, which is essential to the formation of high-level Purkinje images.

Considering other variables that may impact the capturing of Purkinje image III, we also introduce a head model with eyelashes, virtual camera, and LED models with the same parameters as those of the real ones.

4.1.1 Modeling the Eye Parameters

Based on the Arizona eye model [46], we utilize quadric functions to model a series of surfaces, including the anterior and posterior cornea and lens. A common quadric function is represented as [43]:

$$(1+q)\cdot z^2 - 2r\cdot z + x^2 + y^2 = 0 \qquad (2)$$

where q is the conic constant of a specific surface, and r is the radius. For anterior and posterior cornea surfaces, q is -0.25 for both, and r is 7.8 and 6.5 mm, separately. The sclera is modeled as a sphere with an average radius of 13.4 mm. The lens model is created using two quadric functions that vary with accommodation. The anterior surface of the lens has a conic constant of $-7.52+1.29A$ and a radius of $12.0-0.4A$, where A is the accommodation in diopters. The conic constant and the radius of the posterior surface are $-1.35-0.43A$ and $-5.22+0.2A$ [46]. Eye lens parameterization enables flexible lens deformation in response to eye accommodation.

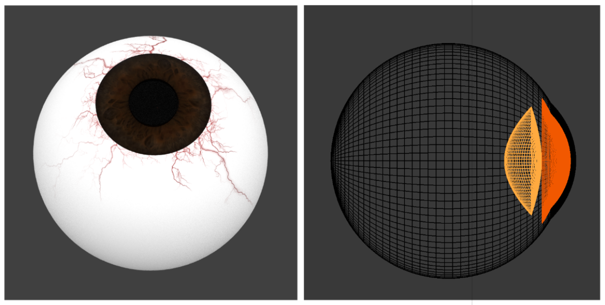


Figure 3: Left: Eye model texture. Right: Eye structure cornea and lens highlighted in orange and yellow respectively.

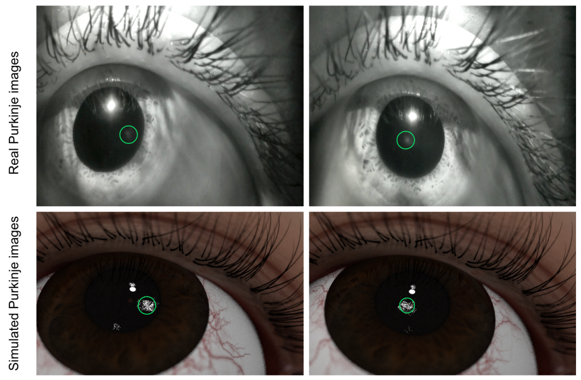


Figure 4: Comparison of Purkinje images capturing from the actual setup (top row) and the simulation (bottom row). It demonstrates the movement of Purkinje image III during eye rotation as well.

The outermost surface of the eye can then be formed by intersecting the anterior cornea and the sclera. Similarly, the eye's inner surface is modeled by intersecting the posterior cornea and the iris's plane, with a hole in the middle to represent the pupil. Fig. 3 provides a visualization of the simulated eye model, including eye texture, cornea and lens surface. Fig. 4 compares the captured Purkinje images and simulated ones.

In addition to the eye model, we also import a head model [1] with eyelashes, to fit with the eyeball. Interocular distance is assumed to be 63 mm [15]. Inspired by [63], we modeled the eyelids movement in consideration of its potential occlusion with Purkinje images in capturing stage. Currently, we utilize the head model closest to our tester's face shape. To be more precise, a reconstructed user face model could be used in the future.

Beyond the user-specific parameters, we also model environmental parameters, specifically cameras and LEDs. We model the virtual camera with a resolution at 1920×1080 and focus length of 35 mm according to the Raspberry Pi Camera we use, and set the focal depth to the anterior lens surface, where Purkinje image III forms. The LED we model provides a directional spotlight with a 3 mm size of soft shadow and a 45 °emitting angle. The specification is from the smallest-size of LED we can find in the consumer market.

4.1.2 Optimizing the Camera and LED Positions

With the model described above, we start to explore the optimal positions for cameras and LEDs given a specific face and eye model. For more precise Purkinje images visualization, we use ray tracing-based rendering. Our goal of optimization is to ensure that, within a

[1] https://www.3dscanstore.com

Algorithm 1: Camera and LED position optimization

```
Initialize position of $Cam_1$ in the bottom center in alignment
  with the eye center;
/* Search $LED_1 = (x,y,z)$, $c_1, c_2$ are thresholds */
for $(x,y,z) \leftarrow (-1.5,-2,0)\ cm$ to $(0,0,3)\ cm$ do
    set $num_{PIII}(x,y,z) = 0$;
    for $(\theta_h, \theta_v) \leftarrow (-12°, -12°)$ to $(12°, 12°)$ do
        Render $I_1$ without lens;
        Render $I_2$ without posterior lens;
        if $\sum_{i,j} 1(I_1(i,j) - I_2(i,j) > c_1) > c_2$ then
            $num_{PIII}(x,y,z) += 1$;
        end
    end
end
Set $LED_1 = argmax_{(x,y,z)} num_{PIII}(x,y,z)$;
Set $Cam_2$ with lowest $y$ and closest $z$ while Purkinje image
  III is able to be seen in the whole FOV;
Update $LED_1$ to ensure that $Cam_1$ and $Cam_2$ can cover two
  connected but not repeated areas;
Set $LED_2$ at the symmetrical point of $LED_1$ relative to the
  eye center;
```

certain range of field of view, both the Purkinje image I and III are visible everywhere the eye rotates. Other constraints we consider include preventing occlusion of the vision, using fewer LEDs to avoid mutual interference between LEDs, using fewer cameras to decrease both the bandwidth and price, and guarantee a large enough field of view.

We describe the algorithm for optimizing the camera and LED positions on our eye tracking eyeglasses in Algorithm 1. We start with one camera and one LED by initializing the camera position cam_1 in the bottom center in alignment with the eye center. It is worth mentioning that we only consider the bottom positions of cameras due to occlusion of eyelashes on the top and obvious distortion of eye images on the side.

To optimize the LED position LED_1, we first discretize the space in a unit of 0.1cm and set the field of view to $\pm 12° \times \pm 12°$. Since the eyeball is modeled symmetrically, we only need to search half of the original space. Considering the compactness requirement, the final search space is set to a 1.5 cm × 2 cm × 3 cm cube in front of the eye. When searching for LED_1, we rotate the eyeball in the whole FOV with a step of 1 degree and expect to see Purkinje image III in as many angles as possible. To identify Purkinje image III, we compute the disparity of two rendered eye images, one without lens, another one with only the anterior lens surface.

We discovered that a single LED can sufficiently illuminate to cover half of the $\pm 12°$field of view, both horizontally and vertically. We also find out that any vertical combination of two LEDs will cause occlusion of the current field of view, so we choose to add another camera above the previous one. We fix the LED_1 and search Cam_2 that satisfies both non-occlusion and compactness requirements. The results show that two cameras are enough for capturing Purkinje images in the specified field of view. We then update the LED_1 to ensure that two cameras can cover two connected but not repeated areas to maximize the field of view. In the end, we add another LED_2 symmetric to the eye center, considering the eye symmetry.

While we initiate the optimization to cover a 12-degree field of view, the result shows that the final camera and LED positions derived from the optimization can cover a 15-degree field of view. In total, we use four LEDs and four cameras for both eyes for the solid capturing of Purkinje image III.

4.2 Hardware

With the optimized camera and infrared LED positions generated from the simulation, we design several slots to put the objects and 3D print the eyeglasses model. The eyetracker is then assembled with four tiny Raspberry Pi cameras with a length of 8 mm, resolution at 1920×1080, and frame rate at 30 fps. To reduce noise, we attach a lens filter to the camera to filter out visible light. We deploy four clear round infrared LEDs as light sources, each with a lens diameter of 3 mm, an IR emitter of 940 nm, and an emitting angle of 30°. The total cost of the setup is around \$100, much cheaper than current commercial ones. The assembled eyeglasses are shown in Fig. 5.

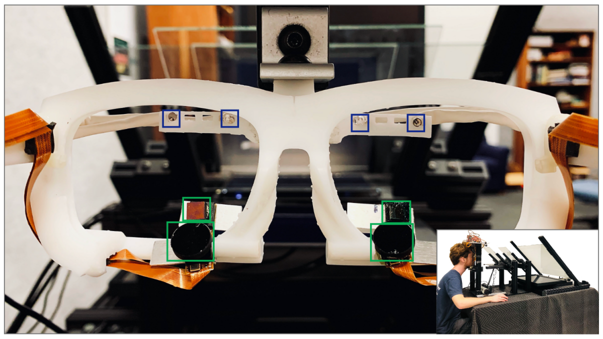


Figure 5: Eyeglasses with multi-plane setup. The LEDs are marked with blue boxes and the cameras are marked in green boxes. The full-view of multi-plane data capturing system is inserted in the bottom-right corner.

When collecting the dataset, our eyetracker is stuck above an adjustable chin rest to guarantee minimal head movement during the experiments. To capture eye images of the tester looking at different depths, we set up a multi-plane system, as shown in the lower right corner of Fig. 5. The displays are laid down and each with a 45°upward-pointing mirror to form an image reflected in the vertical direction. Such setup allows the user to stare at dots at different depths and guarantee that the images from each display centered at the same horizontal and vertical direction.

Upon the controlling systems, a desktop with a Windows 8 system connects to all four screens. Both cameras and infrared LEDs are connected to a Raspberry Pi Zero. With knowledge of each LED's corresponding field of view coverage in advance, we sync up the Raspberry Pi control program with the display program. Whenever a target point is shown on display, the program turns on the corresponding LED that proved to have visible Purkinje image III.

5 Purkinje Image Based Gaze Estimation

To leverage the detected eye features, we design an end-to-end convolutional neural network (CNN), shown in Fig. 6. Using a CNN has several benefits such as high accuracy, robustness, and efficiency on large datasets. Our model takes images captured by four cameras and feeds them into a Faster R-CNN [44] to generate heatmaps, which are then stacked with the original images to pass to a regressor for the final 3D gaze output. In the following section, we first describe the ground truth generation process and then discuss the gaze estimation model in detail.

5.1 Ground truth generation

5.1.1 Data collection

We first ask the subjects to put their heads on a chin rest and calibrate their head positions by aligning the vertical and horizontal lines at all four displays during the data collection stage. We turn on each LED to make sure there is no hardware malfunction and then start the collection by recording on Raspberry Pi cameras. Random target points will appear on each display, and the subjects are asked to

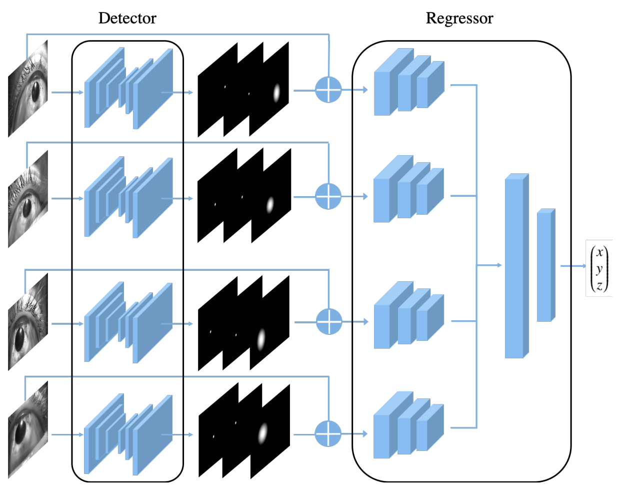


Figure 6: Our network architecture. Images captured by different cameras are first fed into the detector to infer the heatmaps of semantic features. Then the original image and its corresponding heatmaps are concat together as the input of the regressor. The final output of the regressor is the gaze position in 3D space.

stare at the target point. To ensure the tester's attention, we require the tester to click on the target point first, which shrinks by 5 pixels to provide higher acuity and sufficient time to accommodate and maintain on display for one second. Eye images captured during this period for each target point have known (x,y,z) positions. To enforce the sample to cover the entire data space with a feasible sampled data size, 300 target points light up one-by-one randomly on the whole $\pm 15 \times \pm 15^{\circ}$field of view for each display. We also ask the user to stare at uniformly distributed points with a step of two degrees as test data for more discernible visualization of gaze tracking errors.

Our data collection system is controlled by one master Raspberry Pi and four slave Raspberry Pi. Master Raspberry Pi sends a signal to all slave Raspberry Pi, by which each of them controls one camera on the eyetracker. When different target points appear on display, the corresponding LEDs light up as designed by the simulation.

5.1.2 Data processing

Our raw data are video streams from four Raspberry Pi cameras. We synchronize four videos with the recorded timestamps and extract five frames from one video at each target point. In addition to the known target position, we also generate ground truth of pupil coordinates and center coordinates for Purkinje images I and III. In 6, we show that using these features together produces a better result than using one or two of them. For pupil segmentation, we leverage model from DeepVOG [65], which returns us with the center coordinates, the axis of pupil ellipse, and angle of rotation. For Purkinje image I and III, we manually label the center coordinates using VGG Image Annotator [2]. In total, we have 32,500 labeled eye images from five depths, that look into 1625 different target points from five depths, shot by four cameras. Among those target points, 1486 contain Purkinje image III from at least one of the camera's capturing.

5.2 Gaze estimation with neural network

With the data collected from four cameras, we estimate the 3D gaze position with an end-to-end four-branch convolutional neural network. We first use Faster-RCNN to detect semantic eye features, including pupil, Purkinje images I, and III. We then transfer those features into heatmaps to better signify their positions and sizes. Concatenating the heatmaps with their original input images, we feed them into a modified LeNet [29] to generate feature maps. Those feature maps from different branches are then stacked together to feed into a fully connected layer, which derives the final 3D gaze position g. Fig. 6 visualizes the whole pipeline of our algorithm, and we will go into details of each component in the next few paragraphs.

Feature detection For eye features detection, we are inspired by the idea of using Feature Pyramid Network (FPN) [32] to enhance Faster R-CNN [44]. It is proved to be useful for small objects detection, which is suitable for small-sized features such as Purkinje images. We choose ResNet18 [22] as the backbone, which is smaller and faster, as a more complex structure does not improve the results. In the implementation, we adjust the anchor size to $[60, 500]$, based on the average sizes of Purkinje image I, Purkinje image III, and pupil in eye images. Based on the ground truth data, both Purkinje images can be regarded as circles with a radius of 30 pixels, and the pupil has an averaged radius of 250 pixels. We choose the finest-resolution layer $P2$ as the return layer on account of the small size of anchors. In consideration of the generalization of feature detection, all the Faster-RCNN modules share the weights. The outputs of the Faster-RCNN are several bounding boxes of detected features. We remove all the bounding boxes with confidence less than 0.5, and then select the ones with the highest confidence as the final estimated position of our three main eye features. To transfer from bounding boxes to heatmaps, we approximate each bounding box with a straight ellipse and simulate overlaying a piecewise two-dimensional Standard Gaussian distribution on the ellipse, centered on the ellipse center. Regions farther away from the center are blurred, indicating lower credibility in the boundary. Representing features with heatmaps is also beneficial in the absence of eye features. Without changing the network architecture, a fully black image is used to indicate that no feature is detected. Stacking heatmaps with original images allow us to load both Purkinje image information and other eye features that may affect the gaze estimation.

3D Gaze regression For each camera branch, we feed the generated heatmaps, stacked with the corresponding original image, into a regressor. Our regressor is based on LeNet [29] with some modification. It is composed of two convolutional layers, each followed by a max-pooling layer and a ReLU activation. To avoid overfitting, we set a small 3×3 kernel size. All the feature maps from different branches are then stacked with each other and input to two fully connected layers with dimensions 500 and 200 and finally output the 3D gaze point (x, y, z).

The resolution of raw images is 1920x1080, and we scale the size to 720x540 to remove redundant details and preserve eye feature quality. We find that data augmentation improves the tracking accuracy for both gaze angle and accommodation. The input images are contrasted enhanced and gray-scaled before going into the neural network. We minimize the least absolute deviations from the estimated to the true gaze point in 3D.

Training The neural network is implemented by PyTorch [40]. The total datasets with 32,500 images are captured from four cameras with targets at five different depths: 30cm, 35cm, 40cm, 45cm, 50cm. 80% images of the dataset are used as training data and the remaining for the test. Multiple images corresponding to one target will not be placed in both the train and test dataset. In the binocular version, images captured by four cameras are fed into the network simultaneously, while in the monocular version, only left or right two camera images are input. In the training process, we first train Faster-RCNN with the labeled location of the pupil, Purkinje image I, and Purkinje image III. We then freeze its weights and train the regression network separately. The initial learning rate for training Faster-RCNN is set to 0.0003, and 0.0001 for training regressors. They are both decreased by a factor of 0.3 once the number of epoch reaches one of the milestones $[20, 50, 80]$. The model uses Adam

[2]http://www.robots.ox.ac.uk/ vgg/software/via/via-1.0.6.html

Table 1: Quantitative results of semantic feature detection of real cameras frames

	Pupil	**P1**	**P3**
True Positive	100%	100%	100%
False Positive	0%	0%	2.92%
Euclidean Distance (px)	1.8739	2.2699	2.5493

optimizer, and we train for a total of 100 epochs. For both Faster-RCNN and regressors, the batch size is set to 1 to encourage stronger regularization. All training is performed on an NVIDIA GTX 1080 Ti graphics card.

6 Results and Analysis

In this section, we evaluate our proposed user-specific gaze tracking algorithm on our multi-view dataset and provide qualitative and quantitative analysis on major components. We also validate the effectiveness of the third Purkinje images in improving both vergence and accommodation estimation. Experiments demonstrate that our results outperform the state-of-the-art in both vergence and monocular accommodation estimation.

6.1 Dataset

Our dataset is composed of 32,500 eye images from a single subject, a 24-year-old Asian female with normal vision [3]. It contains the subject's data looking at five depths of the plane, each with 325 target points. Different from existing near-eye tracking dataset [17,25,54,56,57,63], our eye dataset captures Purkinje image III, which is useful for both eye vergence and monocular accommodation estimation. Unlike [20,25], our dataset is fully captured by off-axis cameras, in line with the scenarios of using AR glasses, and more challenging than on-axis setup. Our dataset also consists of multi-view eye images that enable robust eye tracking even when eye features are occluded in some views. Multi-view dataset also unleashes the potential to mitigate drift issues in near-eye eyetracking, which is a common but underestimated problem.

6.2 Quantitative and qualitative evaluation

6.2.1 Semantic features detection

In the algorithm, we leverage Faster-RCNN for semantic feature detection, which includes pupil, Purkinje image I and III. To evaluate the result of the detection, we measure both the true positive and false positive detection rate, considering the occlusions of eye features in the images and the scenarios that features are not being captured properly. We also compute the Euclidean distance between the estimated feature center and the ground truth to indicate the accuracy of detection. As shown in Table 1, our feature detection method is able to detect all three features (pupil, Purkinje image I, and Purkinje image PIII) correctly when they indeed exist. No pupil or Purkinje image I are misdetected when they do not exist. In all images without Purkinje image III, only 2.92% of them are falsely detected, which is acceptable given the difficulty of distinguishing noise from Purkinje image III in some capturing. On 720x540 resolution, the averaged Euclidean distance of estimated and real feature center are 1.8739 px, 2.2699 px, 2.5493 px respectively. Such high accuracy allows us to detect Purkinje images' tiny movements and is an important stepping stone for our next gaze tracking.

Fig. 7 illustrates the detected features in highlighted blue, green, and red bounding boxes, along with three heatmaps representing pupil, Purkinje image I, and Purkinje image III.

[3]Data selection and evaluation currently limited only to an individual user due to our institution being closed due to COVID-19

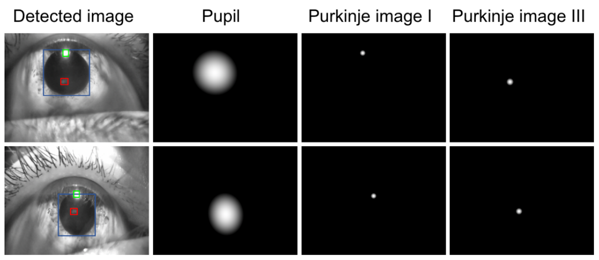


Figure 7: Qualitative results of semantic features detection. The first column shows pupil, Purkinje image I and Purkinje image III highlighted in blue, green, and red bounding box. The three columns on the right illustrate their corresponding heatmaps.

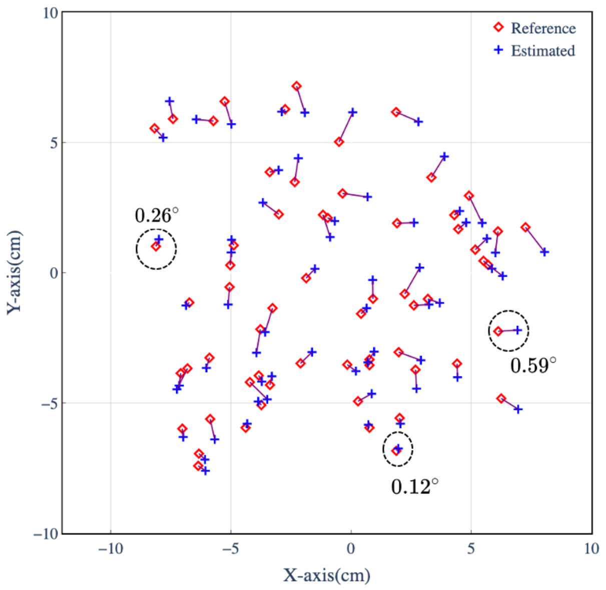


Figure 8: 2D Error visualization at 40cm depth. Reference and estimated points are marked with red square and blue cross, separately. We select three pairs of them to visualize the vergence error, i.e. the average error of the estimated left and right gaze angle and its ground truth.

6.2.2 Vergence and accommodation estimation

We visualize the ground truth target position and the estimated gaze position at a depth of 40cm, which is halfway through the range of depths used in our experiments in Fig. 8. Ground truth target points are marked as red squares, while the calculated gaze points are indicated as blue crosses. A purple line connects each pair of target points and calculated gaze point in the visualization. To better evaluate our results, we transfer the output 3D vector (x,y,z) to yaw and pitch gaze angle (θ,ϕ) to compute the vergence angle error. Our eye tracker estimates gaze with an error of only about 0.37 on average. We further validate the generalizability of our method for estimating the gaze and accommodation at different depths in Fig. 9. The left plot describes the vergence angle error across different depths, while the right one plots the error in estimating the accommodation depth for various distances. The median errors in the vergence angle estimates and the accommodation estimates are consistently low across multiple depths.

6.3 Ablation study

Effectiveness of Purkinje image III We conduct an ablation study to assess the importance of Purkinje image III in estimating vergence and accommodation. As eye features are represented as different layers of heatmaps in our neural network, we conduct ablation studies of different heatmap combinations to assess their efficacy. Table 2 and Table 3 provide the binocular and monocular results of vergence and accommodation estimation, respectively.

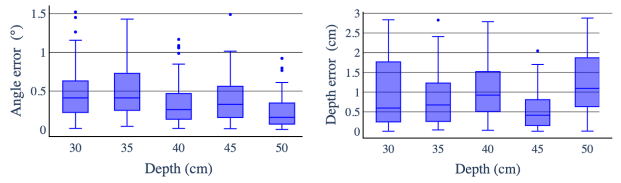


Figure 9: Vergence and accommodation error distribution at five depths. Both figures show that we are able to estimate vergence and accommodation consistently. The estimated vergence errors at five depths are all under 0.5°and the accommodation errors are all close to 1cm.

Table 2: Comparison of using different features in binocular version of our eye tracker.

Input	Angle est. (degree)	Depth est. (cm)
image	0.4846	1.234
image+PIII	0.4362	1.144
image+pupil+PI	0.4143	1.213
image+pupil+PI+PIII	**0.3782**	**1.108**

In terms of vergence, our baseline is to directly input eye images to the regressor without detecting features, which yields 0.4846°accuracy in angle estimation. By adding extra information of detected Purkinje image III, our system achieves a better 0.4362°accuracy with 10% improvement, which indicates the effectiveness of Purkinje image III on this task. The best estimation result comes from binding all features, which yields 0.3782°in angle estimation accuracy with a 22% improvement. As a comparison, we also experiment with only using input images and detected PCCR features, i.e., pupil and Purkinje image I, which achieves a better result than the baseline but is worse than the one combining Purkinje image III. The result corresponds with our theoretical analysis on the relationship between feature movements and rotation angles: Pupil center and Purkinje image III experience the largest distance change while the eye rotates. Experiments also show that using both features at the same time achieves the optimal results.

Similarly, for accommodation estimation, adding Purkinje image III coordinates as an input improve depth estimation from 1.234 cm accuracy to 1.144 cm. The best result 1.108 cm is achieved by inputting images with all the features.

Both binocular and monocular results show the effectiveness of the Purkinje image III. We also found that binocular result is better than monocular one, which is very reasonable considering the additional information another eye may provide compared to the monocular version.

6.4 Comparison with state-of-the-art work

6.4.1 Vergence estimation

We compare our results with current state-of-the-art work in both vergence and accommodation estimation. For the vergence estimation task, we compare with NVGaze [25] and Park's GazeML [39]. NVGaze used a convolutional neural network motivated by Laine et al. [27] and GazeML projects eyeball and iris models onto binary maps for gaze direction regression. On its dataset, NVGaze achieved an absolute estimation error of 0.84°accuracy, with the best case of 0.5°. The deep pictorial gaze model from Park et al. had a gaze angle error of 4.5°on the 15-person MPII [68] dataset. Since none of their datasets contain Purkinje image III, for a fair comparison, we reproduce the network architecture used by NVGaze and borrow the code from GazeML's online repository [4]. We train both methods on our dataset and evaluate with the same metrics. Since the input of GazeML and NVGaze is both one single image, we also only use one branch in the original network architecture. Table 4 presents the performance of those approaches evaluated on our dataset. GazeML achieves a 0.92°angle estimation error, while NVGaze has a better result with a 0.67°error. Our method, however, achieves the best vergence estimation performance, which goes to 0.52°. Note that we compare our network performance only on our single-user dataset. A comparison on a multi-user dataset is left for future work.

Table 3: Comparison of using different features in monocular version of our eye tracker.

Input	Angle est. (degree)	Depth est. (cm)
image	0.5988	1.467
image+PIII	0.5542	1.286
image+pupil+PI	0.5193	1.349
image+pupil+PI+PIII	**0.4834**	**1.245**

Table 4: Comparison with state-of-the-art methods for gaze estimation.

	GazeML [39]	NVGaze [25]	Ours
Gaze Error (degree)	0.92	0.67	**0.52**

6.4.2 Monocular accommodation estimation

We compare the performance of our monocular accommodation estimation with Lee et al. [31] and Itoh et al. [23]. Lee et al. [31] used the relative positions of the Purkinje image I and IV to the pupil center, inter-distance between these two Purkinje images, and pupil size to calculate the 3D gaze position. Although Lee et al. [31] evaluated their accuracy based on a larger range of depths than ours, i.e. 10 cm to 50 cm, the normal focus range of human eyes starts with 25 cm. Moreover, their accommodation estimation is mainly based on pupil diameter, which is heavily affected by illumination change. They leveraged a multi-layered perceptron (MLP) and achieved an average 4.59 cm error along with the depth. We also compare with results from Itoh et al. [23] as they used Purkinje image III in the experiment as well. However, they did not offer their methods or implementations in detail. The final mean absolute error of their accommodation estimation is 3.15 cm for a 10-degree viewing angle. Nor did they explicitly describe the depth values they used for network training, and only provide a depth range of 15-40 cm, similar to ours 30-50 cm. Their reported average depth estimation error is 3.15 cm. We achieve a 1.108 cm better than both methods. And we also illustrate the improvement of angle estimation by leveraging Purkinje Image III that is not mentioned before.

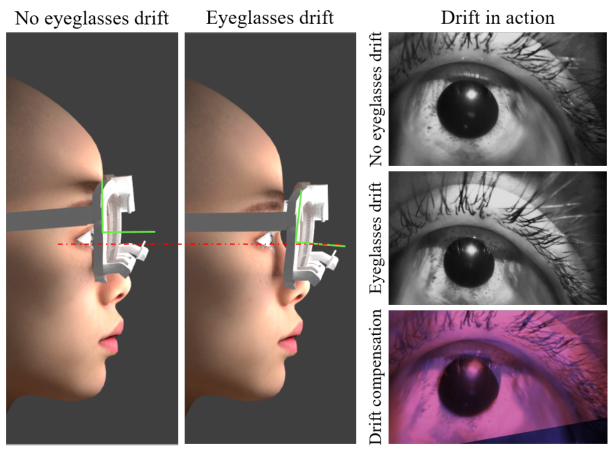
6.5 Errors from eyeglasses slippage

Eyeglasses sliding down the bridge of the nose is a common problem encountered by all people wearing eyeglasses. We anticipate that, in the future, when the tracker is attached to AR eyeglasses, this kind of slippage will be a significant problem. The slippage of eyeglasses would cause the attached eyetracking cameras to drift in space, causing errors in the gaze estimation. We evaluated the simulated drift using our proposed algorithm to assess the impact of eyeglasses slippage on gaze estimation accuracy. To this end, we trained a dataset of simulated eye images captured from cameras attached to the eyeglasses model without any drift. We tested this trained model to evaluate the gaze estimation accuracy on test data consisting of images with varying amounts of eyeglass drift between 1mm to 5mm along the nose bridge, as shown in Fig. 5. The error in estimated gaze angle (degrees) is plotted in Fig. 10. It can be seen in

[4]https://github.com/swook/GazeML

Fig. 10 that eyeglass slippage can significantly impact the accuracy of gaze estimation.

Next, we conduct a controlled laboratory experiment where a user is asked to look at a set of target points displayed at a distance of 40 cm in two different sessions. Despite having a chin rest to hold the head position constant relative to the eye tracker, we notice discrepancies in the image frames captured by the eyetracking cameras, which we account for drifting. We show one such example in Fig. 5. Such slippage could be severe in an eye tracker in-the-wild. However, we note that such errors caused by drift can be compensated for by taking advantage of the multiple cameras and IR LEDs present in our eye tracker. The multiple cameras allow us to estimate the eye's 3D position relative to the eyeglasses frame, making it possible to account for the eyeglasses slippage in the gaze estimation pipeline. To validate this, we show the drift compensation for one frame of a real experiment by overlaying the drift-compensated camera image with that of a non-drifted camera image in Fig. 10. While we currently propose using geometric techniques to estimate the camera drift via detecting landmark features on the captured eye frames, we expect this to be included as part of an end-to-end learning framework in the future.



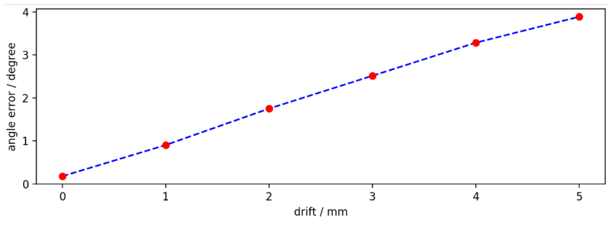


Figure 10: Top left: Drifting of the eyeglasses over the nose bridge. Top right: Errors caused by drifting of the eyeglasses and hence the eye tracking cameras. Notice how the captured frames differ when the eyeglasses drift. We compensate for the eyeglasses drift using geometric techniques and validate by overlaying non-drifted image and the warped drifted image to show alignment. Bottom: The error in gaze estimation caused due to the eyeglasses slippage over the nose bridge.

7 Discussion and Future Work

In this work, we present a comprehensive eyetracking system that captures higher-order Purkinje images for more accurate vergence and monocular accommodation estimation, without requiring additional information from the ambient environment. The first Purkinje image (i.e., corneal glint) and the fourth has been widely used in state-of-the-art near-eye eyetrackers. Contrary to the existing work, we utilize the third Purkinje image from the anterior surface of the eye's crystalline lens for both vergence and accommodation estimation. The third Purkinje image is typically underutilized for its reduced brightness and sensitivity to the camera, LED positions, and eye motion. In this work, we model an anatomical eye, in simulation, to include not only the cornea, sclera, and iris, but also the eye's crystalline lens that causes the higher-order Purkinje images. We use this more advanced eye model to optimize for the placement of cameras and LEDs for robust tracking of Purkinje images. We also demonstrate a prototype eye tracker made from off-the-shelf Raspberry Pi cameras and LEDs, thus providing a cheap alternative to commercial eyetrackers. We use a convolutional neural network that uses the detected eye features for fast and reliable gaze and accommodation estimation, achieving an accuracy of 0.3782 degree and 1.108cm in vergence and accommodation, respectively, outperforming the existing state-of-the-art work eye trackers.

Limitations Our current eye tracker can be improved in several ways and currently has the following limitations. First of all, our anatomical eye model is only an approximate model that might not generalize well for a wide variety of users, potentially leading to minor discrepancies in the simulated and real positions of the Purkinje images. This can result in a one-time effort to manually optimize the positions of the cameras and LEDs, to refine the optimized positions in simulation further. We wish to model the physiological variability of eyes that generalize well to a large set of people and design optimization strategies to eliminate any manual effort in the future.

Our eyetracking solution can be customized and calibrated for each individual user owing to the personal ownership and use of the future AR eyeglasses, similar to today's smartphones. While the proposed eyetracking method can extend to a wide range of users, including those with both normal vision as well as refractive errors in the eyes, we currently evaluate our eye tracker only on a single user with normal vision. Our university remained closed due to the COVID-19 pandemic limiting an extensive analysis of our eye tracker to validate its generalizability to a wide variety of users. Moreover, we also could not demonstrate the use of eye tracker *in-the-wild*. However, we expect that our eyetracking solution will generalize well for a wide range of users and anticipate integrating well into the future AR eyeglasses.

8 Conclusion

We anticipate that the future AR near-eye displays will be in the form factor of everyday prescription eyeglasses, integrating the digital content seamlessly into our ambient real world. For such displays, presenting well-focused imagery of both real and virtual worlds via tracking the user's eyes is of significant importance for comfortable viewing experiences. Moreover, eye tracking can aid such techniques as gaze-contingent rendering that can dramatically reduce the power and compute demands of AR eyeglasses, resulting in extended battery life for prolonged usage.

We demonstrate in this paper, an eye tracker for AR eyeglasses that tracks both the vergence and monocular accommodation robustly via tracking the Purkinje reflections from the various surfaces of the eye using multiple cameras and multiple IR LED sources for illumination. Our eye tracker improves the gaze and accommodation estimation significantly compared to the existing eye trackers. We are excited by the possibility of future AR displays integrating our eyetracker alongside with focus adjusting capabilities for both real and virtual imagery to provide seamless and comfortable augmented reality experiences.

Acknowledgments

The authors gratefully acknowledge the support of NSF grants 1840131 and 1405847, generous gifts from Google and Facebook, assistance with eyeglasses design and simulation by Nolan Scobie and Shalini De Mello, assistance with physical construction by Jim Mahaney, and assistance with early experiments by Alexander Plopski.

Authors' contributions PC conceived the idea, conducted the preliminary research and initial experiments, built the early eyeglasses prototypes and proposed the overall approach. XL designed and fabricated the eyeglasses prototype, collected the data, proposed and implemented the algorithms, conducted the evaluation. YT designed and fabricated the eyeglasses prototype, collected and analyzed the data. SC assisted in analyzing the data and evaluation. HF supervised the overall research. All authors contributed to the final manuscript.

References

[1] A method for the objective measurement of accommodation speed of the human eye*: The accommodation speed and its modification due to fatigue, eserine, pilocarpine and homatropine §.

[2] K. Akşit, P. Chakravarthula, K. Rathinavel, Y. Jeong, R. Albert, H. Fuchs, and D. Luebke. Manufacturing application-driven foveated near-eye displays. *IEEE transactions on visualization and computer graphics*, 25(5):1928–1939, 2019.

[3] F. Alt, S. Schneegass, J. Auda, R. Rzayev, and N. Broy. Using eye-tracking to support interaction with layered 3d interfaces on stereoscopic displays. In *Proceedings of the 19th international conference on Intelligent User Interfaces*, pp. 267–272, 2014.

[4] N. Brown. The change in shape and internal form of the lens of the eye on accommodation. *Experimental eye research*, 15(4):441–459, 1973.

[5] P. Chakravarthula, D. Dunn, K. Akşit, and H. Fuchs. Focusar: Autofocus augmented reality eyeglasses for both real world and virtual imagery. *IEEE transactions on visualization and computer graphics*, 24(11):2906–2916, 2018.

[6] P. Chakravarthula, Y. Peng, J. Kollin, H. Fuchs, and F. Heide. Wirtinger holography for near-eye displays. *ACM Transactions on Graphics (TOG)*, 38(6):213, 2019.

[7] P. Chakravarthula, Y. Peng, J. Kollin, F. Heide, and H. Fuchs. Computing high quality phase-only holograms for holographic displays. In *Optical Architectures for Displays and Sensing in Augmented, Virtual, and Mixed Reality (AR, VR, MR)*, vol. 11310, p. 1131006. International Society for Optics and Photonics, 2020.

[8] T. N. Cornsweet and H. D. Crane. Accurate two-dimensional eye tracker using first and fourth purkinje images. *JOSA*, 63(8):921–928, 1973.

[9] N. Cournia, J. D. Smith, and A. T. Duchowski. Gaze-vs. hand-based pointing in virtual environments. In *CHI'03 extended abstracts on Human factors in computing systems*, pp. 772–773, 2003.

[10] A. Cramer. Mededeelingen uit het gebied der ophthalmologie. *Tijdschrfit Ned. Maatsch. tot bevordering der Geneeskunst*, pp. 99–119, 1851.

[11] H. D. Crane and C. M. Steele. Generation-v dual-purkinje-image eyetracker. *Applied Optics*, 24(4):527–537, 1985.

[12] B. C. Daugherty, A. T. Duchowski, D. H. House, and C. Ramasamy. Measuring vergence over stereoscopic video with a remote eye tracker. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, pp. 97–100, 2010.

[13] A. de Castro, P. Rosales, and S. Marcos. Tilt and decentration of intraocular lenses in vivo from purkinje and scheimpflug imaging: validation study. *Journal of Cataract & Refractive Surgery*, 33(3):418–429, 2007.

[14] A. T. Duchowski, D. H. House, J. Gestring, R. Congdon, L. Świrski, N. A. Dodgson, K. Krejtz, and I. Krejtz. Comparing estimated gaze depth in virtual and physical environments. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pp. 103–110, 2014.

[15] A. T. Duchowski, D. H. House, J. Gestring, R. I. Wang, K. Krejtz, I. Krejtz, R. Mantiuk, and B. Bazyluk. Reducing visual discomfort of 3d stereoscopic displays with gaze-contingent depth-of-field. In *Proceedings of the acm symposium on applied perception*, pp. 39–46, 2014.

[16] W. Fuhl, T. Kübler, K. Sippel, W. Rosenstiel, and E. Kasneci. Excuse: Robust pupil detection in real-world scenarios. In *International Conference on Computer Analysis of Images and Patterns*, pp. 39–51. Springer, 2015.

[17] W. Fuhl, T. Santini, G. Kasneci, W. Rosenstiel, and E. Kasneci. Pupilnet v2. 0: Convolutional neural networks for cpu based real time robust pupil detection. *arXiv preprint arXiv:1711.00112*, 2017.

[18] W. Fuhl, T. C. Santini, T. Kübler, and E. Kasneci. Else: Ellipse selection for robust pupil detection in real-world environments. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, pp. 123–130, 2016.

[19] K. A. Funes Mora, F. Monay, and J.-M. Odobez. Eyediap: A database for the development and evaluation of gaze estimation algorithms from rgb and rgb-d cameras. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pp. 255–258. ACM, 2014.

[20] S. J. Garbin, Y. Shen, I. Schuetz, R. Cavin, G. Hughes, and S. S. Talathi. Openeds: Open eye dataset. *arXiv preprint arXiv:1905.03702*, 2019.

[21] E. D. Guestrin and M. Eizenman. General theory of remote gaze estimation using the pupil center and corneal reflections. *IEEE Transactions on biomedical engineering*, 53(6):1124–1133, 2006.

[22] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[23] Y. Itoh, J. Orlosky, K. Kiyokawa, T. Amano, and M. Sugimoto. Monocular focus estimation method for a freely-orienting eye using purkinje-sanson images. In *2017 IEEE Virtual Reality (VR)*, pp. 213–214. IEEE, 2017.

[24] O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz. Robust face detection using the hausdorff distance. In *International conference on audio-and video-based biometric person authentication*, pp. 90–95. Springer, 2001.

[25] J. Kim, M. Stengel, A. Majercik, S. De Mello, D. Dunn, S. Laine, M. McGuire, and D. Luebke. Nvgaze: An anatomically-informed dataset for low-latency, near-eye gaze estimation. p. 550. ACM, 2019.

[26] K. Krafka, A. Khosla, P. Kellnhofer, H. Kannan, S. Bhandarkar, W. Matusik, and A. Torralba. Eye tracking for everyone. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2176–2184, 2016.

[27] S. Laine, T. Karras, T. Aila, A. Herva, S. Saito, R. Yu, H. Li, and J. Lehtinen. Production-level facial performance capture using deep convolutional neural networks. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 1–10, 2017.

[28] M. A. Langenbeck. *Klinische beiträge aus dem Gebiete der Chirurgie und Ophthalmologie*, vol. 2. Dieterich, 1850.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[30] E. C. Lee, K. R. Park, and J. Kim. Fake iris detection by using purkinje image. In *International Conference on Biometrics*, pp. 397–403. Springer, 2006.

[31] J. W. Lee, C. W. Cho, K. Y. Shin, E. C. Lee, and K. R. Park. 3d gaze tracking method using purkinje images on eye optical model and pupil. *Optics and Lasers in Engineering*, 50(5):736–751, 2012.

[32] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

[33] H.-L. Liou and N. A. Brennan. Anatomically accurate, finite model eye for optical modeling. *JOSA A*, 14(8):1684–1695, 1997.

[34] E. Mallen, J. Wolffsohn, B. Gilmartin, and S. Tsujimura. Clinical evaluation of the shin-nippon srw-5000 autorefractor in adults. *Ophthalmic and Physiological Optics*, 21(2):101–107, 2001.

[35] E. G. Mlot, H. Bahmani, S. Wahl, and E. Kasneci. 3d gaze estimation using eye vergence. In *HEALTHINF*, pp. 125–131, 2016.

[36] R. Navarro, J. Santamaría, and J. Bescós. Accommodation-dependent model of the human eye with aspherics. *JOSA A*, 2(8):1273–1280, 1985.

[37] D. R. Neal, J. Copland, and D. A. Neal. Shack-hartmann wavefront sensor precision and accuracy. In *Advanced Characterization Techniques for Optical, Semiconductor, and Data Storage Components*, vol. 4779, pp. 148–160. International Society for Optics and Photonics, 2002.

[38] A. Papoutsaki, A. Gokaslan, J. Tompkin, Y. He, and J. Huang. The eye of the typer: a benchmark and analysis of gaze behavior during

typing. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*, pp. 1–9, 2018.

[39] S. Park, A. Spurr, and O. Hilliges. Deep pictorial gaze estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 721–738, 2018.

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds., *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

[41] A. Patney, M. Salvi, J. Kim, A. Kaplanyan, C. Wyman, N. Benty, D. Luebke, and A. Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics (TOG)*, 35(6):179, 2016.

[42] B. K. Pierscionek and R. A. Weale. Investigation of the polarization optics of the living human cornea and lens with purkinje images. *Applied optics*, 37(28):6845–6851, 1998.

[43] M. A. Reilly. A quantitative geometric mechanics lens model: insights into the mechanisms of accommodation and presbyopia. *Vision research*, 103:20–31, 2014.

[44] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99, 2015.

[45] T. Schneider, B. Schauerte, and R. Stiefelhagen. Manifold alignment for person independent appearance-based gaze estimation. In *2014 22nd International Conference on Pattern Recognition*, pp. 1167–1172. IEEE, 2014.

[46] J. Schwiegerling. Arizona eye model. In *Field Guide to Visual and Ophthalmic Optics*. International Society for Optics and Photonics, 2004.

[47] L. Sesma, A. Villanueva, and R. Cabeza. Evaluation of pupil center-eye corner vector for gaze estimation using a web cam. In *Proceedings of the symposium on eye tracking research and applications*, pp. 217–220. ACM, 2012.

[48] T. Shibata, J. Kim, D. M. Hoffman, and M. S. Banks. The zone of comfort: Predicting visual discomfort with stereo displays. *Journal of vision*, 11(8):11–11, 2011.

[49] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2107–2116, 2017.

[50] J. Sigut and S.-A. Sidha. Iris center corneal reflection method for gaze tracking using visible light. *IEEE Transactions on Biomedical Engineering*, 58(2):411–419, 2010.

[51] B. A. Smith, Q. Yin, S. K. Feiner, and S. K. Nayar. Gaze locking: passive eye contact detection for human-object interaction. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*, pp. 271–280, 2013.

[52] Y. Sugano, Y. Matsushita, and Y. Sato. Learning-by-synthesis for appearance-based 3d gaze estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1821–1828, 2014.

[53] L. Świrski, A. Bulling, and N. A. Dodgson. Robust real-time pupil tracking in highly off-axis images. In *Proceedings of ETRA*, Mar. 2012.

[54] L. Swirski and N. Dodgson. A fully-automatic, temporal approach to single camera, glint-free 3d eye model fitting. *Proc. PETMEI*, 2013.

[55] V. Tanriverdi and R. J. Jacob. Interacting with eye movements in virtual environments. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 265–272, 2000.

[56] M. Tonsen, J. Steil, Y. Sugano, and A. Bulling. Invisibleeye: Mobile eye tracking using multiple low-resolution cameras and learning-based gaze estimation. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3):106, 2017.

[57] M. Tonsen, X. Zhang, Y. Sugano, and A. Bulling. Labelled pupils in the wild: a dataset for studying pupil detection in unconstrained environments. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, pp. 139–142, 2016.

[58] D. Torricelli, S. Conforto, M. Schmid, and T. D'Alessio. A neural-based remote eye gaze tracker under natural head motion. *Computer methods and programs in biomedicine*, 92(1):66–78, 2008.

[59] K. Wang and Q. Ji. Real time eye gaze tracking with 3d deformable eye-face model. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1003–1011, 2017.

[60] K. Wang, R. Zhao, and Q. Ji. A hierarchical generative model for eye image synthesis and eye gaze estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 440–448, 2018.

[61] E. Wood, T. Baltrušaitis, L.-P. Morency, P. Robinson, and A. Bulling. A 3d morphable eye region model for gaze estimation. In *European Conference on Computer Vision*, pp. 297–313. Springer, 2016.

[62] E. Wood, T. Baltrušaitis, L.-P. Morency, P. Robinson, and A. Bulling. Learning an appearance-based gaze estimator from one million synthesised images. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, pp. 131–138. ACM, 2016.

[63] E. Wood, T. Baltrusaitis, X. Zhang, Y. Sugano, P. Robinson, and A. Bulling. Rendering of eyes for eye-shape registration and gaze estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3756–3764, 2015.

[64] Z. Wu, S. Rajendran, T. van As, J. Zimmermann, V. Badrinarayanan, and A. Rabinovich. Eyenet: A multi-task network for off-axis eye gaze estimation and user understanding. *arXiv preprint arXiv:1908.09060*, 2019.

[65] Y.-H. Yiu, M. Aboulatta, T. Raiser, L. Ophey, V. L. Flanagin, P. zu Eulenburg, and S.-A. Ahmadi. Deepvog: Open-source pupil segmentation and gaze estimation in neuroscience using deep learning. *Journal of neuroscience methods*, 324:108307, 2019.

[66] X. Zhang, Y. Sugano, and A. Bulling. Evaluation of appearance-based methods and implications for gaze-based applications. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–13, 2019.

[67] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling. Appearance-based gaze estimation in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4511–4520, 2015.

[68] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling. Mpiigaze: Real-world dataset and deep appearance-based gaze estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(1):162–175, 2017.

[69] G. Zoulinakis, J. J. Esteve-Taboada, T. Ferrer-Blasco, D. Madrid-Costa, and R. Montés-Micó. Accommodation in human eye models: a comparison between the optical designs of navarro, arizona and liou-brennan. *International journal of ophthalmology*, 10(1):43, 2017.