Title

Rethinking Model-Based Gaze Estimation

Journal

Proceedings of the ACM on Computer Graphics and Interactive Techniques, 5(2)

ISSN

2577-6193

Authors

Kaur, Harsimran

Jindal, Swati

Manduchi, Roberto

Publication Date

2022-05-17

DOI

10.1145/3530797

Peer reviewed

Rethinking Model-Based Gaze Estimation

HARSIMRAN KAUR, SWATI JINDAL, and ROBERTO MANDUCHI, University of California, Santa Cruz

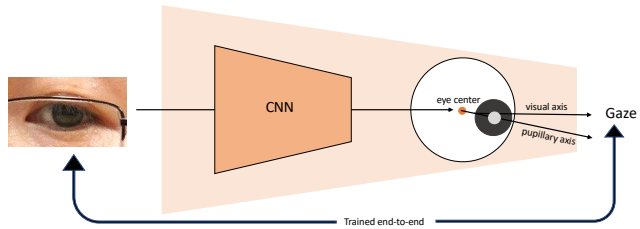


Fig. 1. End-to-End Geometric Gaze estimation

Over the past several years, a number of data-driven gaze tracking algorithms have been proposed, which have been shown to outperform classic model-based methods in terms of gaze direction accuracy. These algorithms leverage the recent development of sophisticated CNN architectures, as well as the availability of large gaze datasets captured under various conditions. One shortcoming of black-box, end-to-end methods, though, is that any unexpected behaviors are difficult to explain. In addition, there is always the risk that a system trained with a certain dataset may not perform well when tested on data from a different source (the "domain gap" problem.) In this work, we propose a novel method to embed eye geometry information in an end-to-end gaze estimation network by means of a "geometric layer". Our experimental results show that our system outperforms other state-of-the-art methods in cross-dataset evaluation, while producing competitive performance over within dataset tests. In addition, the proposed system is able to extrapolate gaze angles outside the range of those considered in the training data.

CCS Concepts: • **Computing Methodologies** → **Computer Vision**.

Additional Key Words and Phrases: Gaze Estimation, Neural Networks, Eye Geometry

ACM Reference Format:
Harsimran Kaur, Swati Jindal, and Roberto Manduchi. 2022. Rethinking Model-Based Gaze Estimation. *Proc. ACM Comput. Graph. Interact. Tech.* 5, 2, Article etra-fp1050 (June 2022), 17 pages. https://doi.org/10.1145/3530797

1 INTRODUCTION

Measuring gaze direction from an image of the viewer (taken, for example, by a laptop camera) has proven a challenging task. To date, the most successful approaches have been based on deep network models [Cheng et al. 2021]. These systems take the whole image, or a cropped portion thereof containing one or both eyes, together with some information about the viewer's head pose,

Authors' address: Harsimran Kaur, hkaur14@ucsc.edu; Swati Jindal, swjindal@ucsc.edu; Roberto Manduchi, manduchi@soe.ucsc.edu, University of California, Santa Cruz.

2577-6193/2022/6-ARTetra-fp1050
https://doi.org/10.1145/3530797

to produce the gaze direction in a suitable reference frame. While providing generally accurate results, this mechanism has a few well-known shortcomings, such as its dependence on the dataset used to train the network (the "domain gap" problem), and its lack of explainability (vaguely defined as the ability to justify unexpected results).

A different strategy, which predates purely machine learning approaches, is to leverage the known mechanical and optical characteristics of the human eyeball. For example, IR gaze trackers [Guestrin and Eizenman 2006] use active illumination to measure the center of curvature of the cornea as well as the pupil center, from which they compute the pupillary axis [Atchison et al. 2000]. While this mechanism cannot be replicated using regular webcams, it is reasonable to assume that the ability to measure specific anatomical properties of the eye could be beneficial for gaze estimation. Consider for example measurement of the pupillary axis, which can be used as a proxy for the visual axis (modulo the κ angles, which describes the relative position of the two axes when looking straight ahead). The pupillary axis[1] connects the pupil center with the anterior corneal center of curvature. It is situated close to what could be considered in first approximation the "eye center", or center of rotation of the eyeball, a point that moves only minimally with eye rotation. Hence, pupillary axis estimation boils down to the measurement of the location of the pupil center (a relatively simple task), thanks to its location in the center of the iris) and of the eye center, which can be considered a function only of the (measurable) head pose.

A common criticism of this class of algorithms is that their results are highly sensitive to errors in the estimation of the physical parameters considered (in the example above, pupil and eye center location). This is certainly true, and this error sensitivity can be quantified (Sec. 5). However, the fact that this type of error modeling is not available for "black box" neural networks, does not mean that these systems are immune to errors (as due to, e.g., image noise). Indeed, the ability to describe the cause of errors, and thus to predict the quality of the results and to "explain" possible malfunctioning, can be considered an asset, rather than a shortcoming.

One well-known problem of model-based approaches to gaze estimation is that some of the parameters to be measured (such as the eye center) are not directly observable. This complicates the use of machine learning algorithms for estimating such parameters. An ingenious solution [Park et al. 2018b; Wood et al. 2016] is to generate synthetic realistic eye images from a physical model, whose parameters are known in advance. Unfortunately, this synthetic data may not be fully representative of real-world images. Some researchers [Kaur and Manduchi 2020] have attempted to employ domain-transfer techniques to generate domain-specific images with known gaze direction. An as yet unexplored direction could be to use an IR gaze tracker to obtain the location of the eye center of training images, provided that the geometric calibration between gaze tracker and webcam is available.

Rather than attempting to create labels for unobservable quantities of interest, we train a network to find the location of a "pseudo eye center", **PEC** (or, more precisely, of its projection on the image plane) by defining an inductive loss that utilizes the annotated gaze direction and the location of the pupil, which, as mentioned earlier, can be obtained fairly reliably from the image. Note that the line joining **PEC** to the pupil center ("pseudo pupillary axis", **PPA**) is not guaranteed to coincide with the real pupillary axis (whose exact location is not available in the training data). Rather, training aims to ensure that the relative location of **PPA** and of visual axis (as determined by the "pseudo κ angles") is constant for a given individual. At deployment, our system computes the image projection of the **PEC** and of the pupil center. Then, the **PPA** is obtained by backprojecting

[1]Note that in the literature (e.g. [Guestrin and Eizenman 2006]), the pupillary axis is often misnamed as the "optic axis", the latter being the line of best fit through the centers of curvature of the different refracting surface within the eyeball [Atchison et al. 2000].

these two points, and is then rotated according to the pseudo-κ values, which are regressed using a prior standard procedure with the viewer fixating at a number of calibration targets on the screen. To properly train the network, we found it beneficial to add a second branch (Figure 2b) that starts from a common image embedding then directly regresses the **PPA**, with the loss function accounting for the gaze estimation error from both branches. This second branch (which is not used at deployment) effectively conditions the training of of the convolutional network that generates the image embedding.

2 RELATED WORK

The gaze estimation techniques can primarily be categorized as appearance-based and model-based, which can be further be person-independent or personalized methods.

2.1 Appearance-based methods

Earlier appearance-based methods used classical machine learning techniques - linear regression [Lu et al. 2014], Support Vector Regression [Martinez et al. 2012], Random Forests [Sugano et al. 2014] to regress gaze from eye images. The accuracy of the models improved considerably with the use of CNN based models that used either eye as input [Zhang et al. 2015] or the entire face [Cheng et al. 2018, 2020; D and Biswas 2021; Fischer et al. 2018; Krafka et al. 2016; Zhang et al. 2020, 2017]. Other approaches included use of visual saliency information [Alnajar et al. 2013; Chen and Ji 2011; Park et al. 2020; Sugano and Bulling 2015; Sugano et al. 2010], Bayesian neural networks [Wang et al. 2019] and unsupervised learning [Sun et al. 2021; Yu and Odobez 2020].

2.2 Model-based methods

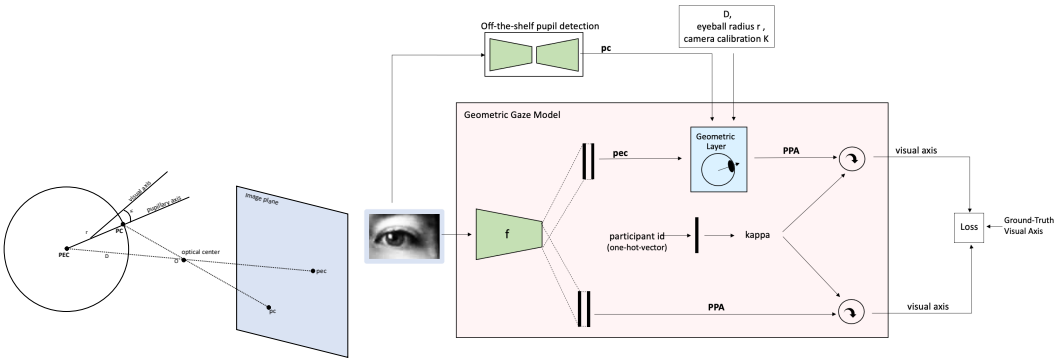
Many of the geometric methods involved computing the 3D eye center by anchoring it a stationary point on the face [Chen and Ji 2008; Sun et al. 2015] or by directly fitting a 3d face-eye deformable model [Wang and Ji 2017]. Some methods fit an ellipse to the detected iris and compute the pose to get the gaze [Wang et al. 2003; Wood and Bulling 2014].

2.3 Hybrid methods

Several hybrid methods have also been proposed that combine the eye geometry with the appearance models. Noticeably, use of intermediate gaze maps computed from eye geometry [Park et al. 2018a] and landmark based models [Ji and Wang 2021; Park et al. 2018b; Yu et al. 2018] demonstrated the effectiveness of using the additional information over appearance based methods.

2.4 Person-specific Gaze Estimation

Personalized models that require subject specific samples can significantly improve the performance of gaze models. This has been achieved by fitting an SVR to the last layer of the CNN model trained on the gaze dataset [Krafka et al. 2016], meta-learning [Park et al. 2019], using mixed-effects neural networks [Xiong et al. 2019], training siamese network with two different gaze samples from the same person [Liu et al. 2018] or learning the person-specific parameters during training [Chen and Shi 2020; Linden et al. 2019]. In [Chen and Shi 2020], the gaze angles are decomposed as pupillary axis and offset, which are added to obtain the visual axis prediction. The offset is learned for each subject during the training itself. This kind of decomposition forces the network to learn a person-independent gaze, which is the function of the eye appearance. During inference, the subjective bias is computed using few calibration samples by calculating the mean of the difference between the predicted gaze and the true gaze. Chen et al. [Chen and Shi 2020] outperforms the previous calibration-based methods. This algorithm (which we will call *End-to-End*) serves as a baseline for our work.



(a) Geometric Model (b) End-to-End geometry embedded CNN model

Fig. 2. Left: Eye geometry model used for our geometric layer. Right: The GeoGaze architecture.

3 METHOD

3.1 Geometric Preliminares

We will denote 3-D points and vectors by uppercase boldface, and 2-D image points by lower case boldface. Matrices are represented using uppercase non-boldface characters.

The task of a (monocular) gaze tracker is to compute the visual axis of each of the viewer's eyes (see Fig. 2). In practice, this is performed by first computing a different axis, such as the pupillary axis in the case of a IR tracker. This is because the visual axis cannot be obtained directly from anatomical measurements. The visual axis goes through the lens' nodal points (whose location cannot be computed in practice [Cui and Lakshminarayanan 2003]) and the preferred retinal location, which, while relatively stable with time, changes from person to person [Kilpeläinen et al. 2021]. It can only be observed through (subjective) fixation tests, whereas other types of axes [Mosquera et al. 2015] can, in principle, be determined from objective measurements. Both the visual axis and the considered axis to be tracked (*measurement axis*) are "attached" to the eyeball, and thus rotate in a similar way when gaze changes (through motion of the eyeball or of the viewer's head). This property can be formalized through the concept of *equivariance*. Specifically, given a sequence U of unit-norm 3-D vectors (*axes*) $(\mathbf{U}_0, \ldots \mathbf{U}_n)$, we will say that the sequence of axes $V = (\mathbf{V}_0, \ldots \mathbf{V}_n)$ is equivariant with U if for any pair of indices (i, j), there exists a rotation matrix $R_{i,j}$ such that $\mathbf{U}_j = R_{i,j}\mathbf{U}_i$ and $\mathbf{V}_j = R_{i,j}\mathbf{V}_i$. Thus, any sequence of measurement axes is equivariant with the associated sequence of visual axes, provided that they come from the same individual.

An axis $\mathbf{U}_i$ can be defined by two angles, e.g. the angular components of its spherical coordinates $[\theta_{\mathbf{U}_i}, \phi_{\mathbf{U}_i}]$ with respect to a given reference frame (such as the camera frame.) This can be thought of as first defining a *canonical* axis (e.g. $\mathbf{U}_0 = [0,\ 0,\ 1]^T$) then rotating the reference frame by an ordered sequence of elementary rotations with associated Euler angles. Using the $Z - X - Y$ ordering: $\mathbf{U}_i = R^Y(\beta)R^X(\alpha)R^Z(\gamma)\mathbf{U}_0$, where, for example: $R^X(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix}$.

Without loss of generality, γ (initial rotation angle around the Z axis) can be set to 0 (since we chose the canonical axis to be oriented along the original Z axis). It is easy to see that, in the final reference frame, the spherical coordinates $(\theta_{\mathbf{U}_i}, \phi_{\mathbf{U}_i})$ of $\mathbf{U}_i$ are related to the Euler angles of rotation as by: $\theta_{\mathbf{U}_i} = \alpha$, $\phi_{\mathbf{U}_i} = -\beta$. Hence, $\mathbf{U}_i = R_{0,i}[0,\ 0,\ 1]^T$ with: $R_{0,i} = R^Y(-\phi_{\mathbf{U}_i})R^X(\theta_{\mathbf{U}_i})$ resulting in

the familiar representation: $\mathbf{U}_i = [\sin\phi_{\mathbf{U}_i} \cdot \cos\theta_{\mathbf{U}_i},\ \sin\theta_{\mathbf{U}_i},\ \cos\phi_{\mathbf{U}_i} \cdot \cos\theta_{\mathbf{U}_i}]^T$. The vector $\mathbf{V}_0$ could similarly be defined in terms of its spherical coordinates $(\theta_{\mathbf{V}_0}, \phi_{\mathbf{V}_0})$. Note that if $\mathbf{U}_0$ represent the pupillary axis of an individual looking straight ahead, and $\mathbf{V}_0$ is its associated visual axis, $(\theta_{\mathbf{V}_0}, \phi_{\mathbf{V}_0})$ represent the κ angles for this individuals. Given the spherical coordinates $(\theta_{\mathbf{U}_i}, \phi_{\mathbf{U}_i})$ of an axis $\mathbf{U}_i$, the associated axis $\mathbf{V}_i$ can be expressed as:

$$\mathbf{V}_i = R_{0,i}\mathbf{V}_0 = R^Y(-\phi_{\mathbf{U}_i})R^X(\theta_{\mathbf{U}_i})R^Y(-\phi_{\mathbf{V}_0})R^X(\theta_{\mathbf{V}_0})[0 \quad 0 \quad 1]^T \tag{1}$$

$$= R^Y(-\phi_{\mathbf{U}_i})R^X(\theta_{\mathbf{U}_i})R^Y(-\phi_{\mathbf{V}_0})R^X(\theta_{\mathbf{V}_0})R^X(-\theta_{\mathbf{U}_i})R^Y(\phi_{\mathbf{U}_i})\mathbf{U}_i \tag{2}$$

We will denote the relationship between the spherical coordinates of $\mathbf{V}_i$ and those of $\mathbf{U}_i$ as follows: $[\theta_{\mathbf{V}_i}, \phi_{\mathbf{V}_i}] = F_{\mathbf{V}_0}[\theta_{\mathbf{U}_i}, \phi_{\mathbf{U}_i}]$ This relationship is, in general, complex. If, however, both $\theta_{\mathbf{U}_i}$ and $\theta_{\mathbf{V}_0}$ are small in magnitude, the product of the rotation matrices in (1) approximately commutes, in which case one may write: $\theta_{\mathbf{V}_i} \approx \theta_{\mathbf{U}_i} + \theta_{\mathbf{V}_0}$, $\phi_{\mathbf{V}_i} \approx \phi_{\mathbf{U}_i} + \phi_{\mathbf{V}_0}$. Note that this oft-used approximation may generate non-negligible errors when gaze deviates substantially from the canonical axis $[0,\ 0,\ 1]^T$. For example, for κ angles of $\theta_{\mathbf{V}_0} = 2°$, $\phi_{\mathbf{V}_0} = 6°$ and a pupillary axis of $\theta_{\mathbf{U}_i} = 10°$, $\phi_{\mathbf{U}_i} = 30°$, computing the visual axis by simply adding the κ angles to the angular spherical coordinates of the pupillary axis, rather than using (1), produces an angular error of 0.37°.

In order to build a data set of images with associated ground-truth visual axis $\mathbf{V}_i$, a standard approach [Park et al. 2018b; Sugano et al. 2014; Zhang et al. 2015] is to ask participants to fixate specific points on a computer screen. Assuming that the camera's pose with respect to the screen is known (this can be computed using standard methods, e.g. [Rodrigues et al. 2010]), the location of each fixation point on the screen can be converted to the camera's reference frame. A deformable 3-D face model is used to compute the participant's face pose from the image (defined in the camera frame) and to define a "gaze origin" point on this model. Specifically, the gaze origin is taken to be the center of the two eye corners in the deformable model. The visual axis is then defined as the line joining the gaze origin to the gaze point on the screen.

3.2 GeoGaze: Algorithm

During training, our system is provided with a sequence of images of different viewers, along with the ground-truth visual axis for each eye in each image. It is tasked with determining, for each image and each eye, a *pseudo pupillary axis* **PPA**, such that the sequence of **PPA**s computed for all images of the same viewer is equivariant with the sequence of visual axes associated with the same images. The *pseudo* κ angles relating the **PPA**s with the visual axes are computed for each user during training. At deployment, the system computes the **PPA** for each image, then rotates it using the *pseudo* κ angles (Eq. (1)), obtained for each viewer through a standard calibration procedure.

The main novelty of our approach is in the way **PPA**s are represented and computed. Rather than directly regressing **PPA**s (e.g. in terms of their spherical coordinates), we identify a **PPA** by two 3-D points: the pupil center **PC** and the *pseudo eye center* **PEC**. Both points are first computed as 2-D points (**pc** and **pec** respectively) in the image. The pseudo eye center is then backprojected in space, using a value of distance D obtained from the head pose (itself computed from the image):

$$\mathbf{PEC} = D \cdot K^{-1}\overline{\mathbf{pec}} \tag{3}$$

where K is the intrinsic camera matrix and $\overline{\mathbf{pec}}$ is the augmented vector (with a 1 appended as third entry). **PEC** is assumed to be close to the center of rotation of the eyeball, therefore it is reasonable to think that, in first approximation, its depth should only be a function of the head pose, and not of the direction of gaze.

The image point **pc** (projection of the pupil center) is also backprojected to a point in space **PC** that is at distance of r=12 mm from **PEC**. Similarly to [Chen and Ji 2008; Wang and Ji 2017], we compute the points where the camera ray through **pc** (expressed as $d \cdot K^1\overline{\mathbf{pc}}$ for a generic scalar

d) intersects the sphere of radius r centered at **PEC**, and choose the intersection point at shorter distance from the camera. If the camera ray defined by **pc** does not intersect this sphere, then **PC** is assigned to the point in this camera ray that is closest to **PEC** (resulting in an axis **PPA** that is orthogonal to this camera ray.) The pseudo pupillary axis is then given by **PC** − **PEC**.

It is instructive to compare the GeoGaze algorithm with other model-based algorithms. For example, GazeML [Park et al. 2018b] also estimates the location of the eye center **pec** from the image. The GazeML network is trained on ground-truth eye center data. Since the eye center is not directly observable, GazeML uses synthetic data from the UnityEyes dataset [Wood et al. 2016] for training. Reliance on synthetic data (which may not be fully representative of real-world conditions) is a main limitation of this approach. Differently from GazeML, we do not assume ground-truth knowledge of the eye center **PEC** during training. In fact, **PEC** does not necessarily correspond to a specific anatomical feature (such as the corneal center of curvature), which would be difficult or impossible to annotate manually. Instead, **PEC** is implicitly defined by the constraint that it should be located at a distance of r from **PC**, and that the resulting sequence of **PPA**s (i.e. axes through **PEC** and **PC**) should be equivariant with the ground-truth visual axes **VA**s. This allows us to train the system with any images for which the visual axis is available. The underlying assumption is that the pupil location **pc** in the image, as well as the distance D to the user, can be reliably estimated from each image. In Sec. 5, we provide an analytical derivation of the the sensitivity of the measured **PPA** to errors in the estimation of **pc** and of D.

A different approach was taken in other model-based algorithms [Chen and Ji 2008; Wang and Ji 2017], which also estimate the location of the eye center despite missing ground-truth annotations. In both such algorithms, the eye center is computed in relation to a set of facial features, detected in the image and backprojected at an appropriate distance. In the case of [Chen and Ji 2008], the eye center is determined by means of an offset vector that is computed during an initial per-person calibration phase. The model of [Wang and Ji 2017] matches a deformable face model to the backprojected features. The location of the eye center is included in the face model, which is adapted to the viewer through an initial calibration phase with a 3-D camera. In both cases, the per-person calibration procedure requires a stable head pose. Unlike these methods, GeoGaze does not rely on facial features, but estimates the pseudo eye center directly from the image. Because of this, GeoGaze can be trained end-to-end, and requires no personalization procedure besides what is needed for the determination of κ angles.

We should also emphasize that the choice of the distance r between the pseudo eye center **PEC** and the pupil enter **PC** is not critical. Indeed, the only constraint on **PEC** is that it should be "attached" to the eyeball, such that any generated sequence of axes **PC** − **PEC** is equivariant with the associated visual axes. Our choice for r=12 mm (the average eyeball radius) was based on two observations: (1) if r is too small, the computed pseudo pupillary axis **PPA** becomes very sensitive to localization errors for the projected **pec** (see Sec. 5); (2) a too large r results in the location of **PEC** (and thus of **pec**) that may vary widely with gaze, even when the head pose is fixed. Intuitively, it would be desirable that the location of **PPE** be only dependent on head pose and not on gaze, as this would remove one cause of variability. Indeed, this is the main assumption of classic model-based algorithms such as [Chen and Ji 2008; Wang and Ji 2017], that compute the eye center location based on facial features that supposedly do not move with gaze. In practice, though, the eye center *does* move with gaze, by as much as 0.7 mm [Moon et al. 2020]. Neglecting this eye center displacement would result in a gaze estimation error as high as 3°. We experimented with leaving the distance r as a variable to be estimated by the network, but did not find any significant difference with respect to using a fixed value.

3.3 GeoGaze: Architecture

The architecture of the proposed system is rather straightforward (see Fig. 2b). An input image, after pose normalization as per [Zhang et al. 2018], is cropped to only contain one eye, and fed to a convolutional neural network (CNN) that produces a 256-dimensional embedding. Then, a fully connected layer computes the estimated **pec** location in the image. In parallel, the distance D to the user and the pupil center **pc** are also computed from the image, using off-the-shelf algorithms (see Sec. 2b). This data is passed on to a "geometric layer". Specifically, from **pec**, **pc**, and D, the geometric layer: (1) backprojects **pec** into **PEC** (Eq. (3)); (2) backprojects **pc** into **PC** as explained in the previous section; (3) computes the pupillary axis $\mathbf{PPA} = \mathbf{PC} - \mathbf{PEC}$; (4) applies an appropriate rotation (Eq. (1)) to **PPA**s using person-specific pseudo κ angles (i.e., the spherical coordinates of $\mathbf{V}_0$).

The goal of training is to optimize the parameters of the CNN and of the fully connected layer, as well as to determine the optimal pseudo κ angles for each person in the training set. The loss to be minimized is a linear combination of two components: the average discrepancy between the reconstructed and the real visual axes (as measured by the Euclidean distance of the spherical coordinates of the two axes); and the average norm of the person-specific pseudo κ angles. This second regularization term is necessary, as the same loss could otherwise be achieved by infinite equivalent solutions with different pseudo κ angles. Note that, as mentioned above, if the estimated **PEC** is at a distance larger than r from the camera ray defined by **pc**, no anatomically consistent solution can be found, and thus a fail-safe mechanism is invoked (by setting **PPA** to be orthogonal to this camera ray). While this situation occurs only sporadically once the network is properly trained, we noticed that this may happen relatively often during the initial phase of training, potentially driving convergence towards local plateaus of the loss function. This is because the angular error is relatively unaffected by changes in the **pec** location in these situations (the error effectively "saturates"). To reduce this risk, we consider one additional loss component at the beginning of training. Specifically, we add a vector of length r, oriented along the ground-truth visual axis **VA**, to the estimated **PEC**. We use this as a proxy for the actual pupil location **PC**, under the assumption that the visual axis is relatively close to the pseudo eye center. This point is then projected onto the image in location $\widehat{\mathbf{pc}} = EN[K(\mathbf{PEC} + r\mathbf{VA})]$, where EN is the operator that computes the Euclidean normalization [Förstner and Wrobel 2016] of a homogeneous vector (divides its entries by the last one) and then removes the homogeneous part (last entry). Then, we add to the overall loss the Euclidean distance (in the image plane) between **pc** and $\widehat{\mathbf{pc}}$. Note that this loss component never saturates, even when the **pec** is grossly wrong. We found that this loss component is no longer necessary after the initial training phase.

We have also experimented with implementing a second "end to end" (E2E) branch during training (see Fig. 2). This branch feeds off the output of the CNN, and directly generates predictions of the spherical coordinates $[\theta_{\mathbf{PPA}_{E2E}}, \phi_{\mathbf{PPA}_{E2E}}]$ of the pseudo pupillary axis **PPA** axis through a fully connected layer. This branch is only considered during training, with the purpose to improve optimization of the initial CNN. This can be seen as a form of multi-task optimization [Ruder 2017; Yu et al. 2018].

The overall loss to be minimize is thus:

$$\mathcal{L} = \mathbb{E}\left(\|[\theta_{\mathbf{VA}}, \phi_{\mathbf{VA}}] - F_{\mathbf{V}_0}[\theta_{\mathbf{PC-PEC}}, \phi_{\mathbf{PC-PEC}}]\| + \lambda_1\|[\theta_{\mathbf{V}_0}, \phi_{\mathbf{V}_0}]\| + \lambda_2\|\mathbf{pc} - \widehat{\mathbf{pc}}\| \right. \\ \left. + \lambda_3\|[\theta_{\mathbf{VA}}, \phi_{\mathbf{VA}}] - F_{\mathbf{V}_0}[\theta_{\mathbf{PPA}_{E2E}}, \phi_{\mathbf{PPA}_{E2E}}]\|\right) \tag{4}$$

Note that, during training, the system generates one value of **PEC** per image and one pair $[\theta_{\mathbf{V}_0}, \phi_{\mathbf{V}_0}]$ per person. In practice, this is obtained by feeding in input a one-hot vector representing the

identity of the person for each image. The weight λ_2 is set to 1 during the initial phase of training, 0 afterwards. λ_1 and λ_3 are both set to 1.

3.4 Calibration

A user-specific calibration procedure is required to estimate the kappa angles associated with each user. This procedure requires taking a number of images of the user with associated ground-truth visual axes (see Sec.3.1) as the user looks in different direction. (In practice, we sample a number of images of each participant within the considered data sets.) After computing the pseudo-pupillary axis for each image and for each eye of the user, we use the Orthogonal Procrustes algorithm [Schönemann 1966] to find the optimal rotation matrix that transforms the set of ground-truth visual axes to match the associated pseudo-pupillary axes, as computed with our algorithm. We then obtain the Euler angles α, β, γ associated with this rotation matrix (using the $Z - X - Y$ ordering). The two kappa angles for this user are then obtained as $(\theta_{\mathbf{V}_0}, \phi_{\mathbf{V}_0}) = (\alpha, -\beta)$ (see Sec. 3.1).

4 EXPERIMENTS

4.1 Implementation details

We used the dilated-net architecture [Chen and Shi 2020, 2018] for the the CNN backbone followed by two MLP branches. Each MLP branch consists of two fully connected layers with 256 neurons each followed by the output layer of dimension 2. Training uses the Adam [Kingma and Ba 2014] optimizer, with the learning rate set to 0.001 and batch size of 64. λ_2 is set to 0 after 4 epochs. The pupil center is computed by the GazeML network [Park et al. 2018b], which uses a hourglass architecture for iris landmarks detection[2]. GazeML produces a heatmap for each landmark, representing the probability of each pixel being the landmark location. We computed the pupil center as the mean of the iris landmarks. Errors are produced in terms of the angle between estimated and ground-truth visual axis. In our experiments, we always compare the results of GeoGaze against those of a system (*End-to-End*) that is identical to GeoGaze, but does not have the final geometric layer. End-to-End [Chen and Shi 2020] directly computes the spherical coordinates of the **PPA**, along with the pseudo κ angles for each person in the training data. It does not require the external pupil detection module. We compared our results to another baseline where within End-to-End system, we concatenate the 2-d pupil center to the 256-d feature vector before feeding it to the fully connected layer. We call this baseline End-to-End+pupil.

We consider five different datasets for our experiments. The two data sets (UnityEyes [Wood et al. 2016], U2Eyes [Porta et al. 2019]) are made up of synthetic images, and are used in a toy scenario to highlight some of the features of GeoGaze. We also consider the following real-world datasets: MPIIGaze [Zhang et al. 2015], Columbia [Smith et al. 2013], and UTMultiview [Sugano et al. 2014]. MPIIGaze was collected "in the wild" from 15 participants, with no constraints on head and eye movements. The Columbia dataset (56 participants) and UTMultiview (50 participants) were collected in a laboratory environment, with multiple cameras (5 and 8, respectively) used to reliably compute ground-truth head poses.

4.2 Synthetic Data: UnityEyes and U2Eyes

Unity Eyes: We generated 20,000 images for training, with yaw and pitch angles uniformly distributed between $-15°$ and $15°$. We used these images to train the GazeML model used for pupil detection (using the available ground-truth annotated landmark), as well as the GeoGaze and the End-to-End networks. We then generated two test sets, with 20,000 images each: one with the same gaze distribution as the training set (*in-distribution*), and the other with yaw and pitch sampled

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

uniformly between −30° and −15° and between 15° and 30° (*out-of-distribution*). For unity images, the visual axes were set to be identical to the pupillary axes, and the κ angles were forced to 0 for both algorithms, and used the geometric function specific to the unity eyes model (which uses orthographic projection model without camera calibration parameters).

U2Eyes, based on UnityEyes, contains binocular eye images, with kappa angles introduced. Data is available for 20 participants, with 5875 images per participant distributed across different head pose and gaze directions. The pitch and yaw angles are roughly distributed between −20° to 20°. For each participant, we divided the data in two sets. Set 1 contains samples with pitch and yaw between −10° to 10° (in-distribution), while Set 2 contains the rest of the data (out of distribution). The left eye images were cropped using the eye corners information provided. We performed five-fold cross-validation (training on 16 subjects and testing on the remaining 4), training on Set 1 and testing on either Set 1 (in-distribution) or Set 2 (out-of-distribution). 16 samples per participant were used for user-specific calibration.

Table. 1 shows the results for both GeoGaze and End-to-End (with and without pupil) when tested on the in-distribution (i-d) and on the out-of-distribution (o-o-d) sets for both UnityEyes and U2Eyes. Note that while the results are comparable for the in-distribution data, the average error of End-to-End for out-of-distribution data (9.2°) in case of UnityEyes and (5.3°) in case of U2Eyes is substantially larger than for GeoGaze (3.3° and 1.9° respectively). Clearly, the End-to-End system was unable to "extrapolate" gaze angles that were not seen in the training data. With pupil information added, the model was able to perform better than simple End-to-End. However providing the pupil information in the geometric layer proved much better for out-of-distribution images.

Table 1. GeoGaze vs End-to-End baselines on UnityEyes and U2Eyes – Mean Angular Errors (degrees)

Algorithm	i-d	o-o-d
End-to-End+pupil	**0.9°**	7.4°
End-to-End [Chen and Shi 2020]	1.0°	9.2°
GeoGaze	1.3°	**3.3°**

(a) UnityEyes

Algorithm	i-d	o-o-d
End-to-End+pupil	**1.4°**	3.4°
End-to-End [Chen and Shi 2020]	1.7°	5.3°
GeoGaze	1.5°	**1.9°**

(b) U2Eyes

Since U2Eyes provides ground-truth annotations for the kappa angles (along with pupil center, eye center, and pupillary axis), we performed a simple experiment to evaluate the ability of our system to perform user-specific calibration, as well as the effect of calibration on the predicted visual axis. We randomly chose one participant, then trained our system on all other participants on data from Set 1 and tested on data from both Set 1 and Set 2. We obtained an average error of 0.14 pixels in pupil center prediction and 2.13 pixels in eye center prediction, for an average pupillary axis angular error of 3.99°. (See Sec. 5.1 for a model of the effect of pupil center prediction errors on the predicted pupillary axis.) Note that the pseudo eye center **PEC** and pseudo pupillary axis **PPA** computed by our system need not coincide with the actual eye center and pupillary axis, as explained in Sec. 3.3. This discrepancy is evident in Fig. 3, which shows the predicted **PPA** vs. ground-truth pupillary axis pitch and yaw angles for different gaze directions of the same participant. It is clear from the figure that the predicted **PPA** follows the ground-truth pupillary axis fairly accurately, but with a fixed bias. This bias is compensated by an opposite bias in the estimated kappa angles, obtained via user-specific calibration (Sec. 3.4), which returned kappa angles of $(-1.63°, 6.96°)$, against ground-truth values of $(2.50°, 8.00°)$. As a result of this compensation, the

predicted visual axis (after kappa angles compensation) has a much smaller average angular error (equal to 1.30°) than the pupillary axis.

Fig. 3. Distribution of pitch and yaw angles corresponding to predicted pseudo pupillary axis and ground-truth pupillary axis for U2Eyes. The 45° line represents perfect matching.

4.3 Real-world Dataset

For these datasets, we normalize the eye images using a technique similar to [Zhang et al. 2018], which applies a homography to the image equivalent to rotating the camera such that its optical axis points at the mid-point of the eye. Note that in our case we used the 2-D corners detected in the image, rather than the 3-D eye corners, to determine this mid point. This reversible normalization procedure requires estimation of the head pose, which could be inaccurate due to various reasons. Unlike other model-based methods (e.g., [Sun et al. 2015], [Wang and Ji 2017]) that define the gaze direction in reference to head orientation, our algorithm computes all quantities in the camera frame of reference, and is therefore not affected by errors in the rotational component of head pose. However, incorrect estimation of the head distance may generate errors in gaze direction; this effect is modeled in Sec. 5.2. The image was resized to correspond to a normalized distance of 600 mm. We cropped a patch of 128x64 pixels around each eye. Right eye images were flipped, along with the corresponding yaw and the horizontal coordinate of the iris landmarks. The κ angles were computed separately for the left and the right eye through a standard initial calibration procedure. For the UTMultiview dataset, we only used those images for which a full face is visible.

4.3.1 Within Dataset Evaluation. Our first experiment shows results on the MPIIGaze dataset, with the system trained and tested on the same data set, We performed leave-one-person-out evaluation with a variable number of images used for per-person calibration. It has approx 75,000 images for left and right eyes of 15 subjects. We compared the results of GeoGaze against published results from other competing algorithms: FAZE[Park et al. 2019], GRS[Yu et al. 2019], End-to-End[Chen and Shi 2020], as well as GazeML [Park et al. 2018b], which is model based.

Results are shown in Table. 2. Note that for this within-dataset experiment, GeoGaze produces results that are comparable with those of other algorithms. Only the End-to-End systems along with the one where pupil is added, consistently produced a lower error.

4.3.2 Cross-Dataset Evaluation. Next, we trained our algorithms on the MPII dataset, leaving one person out for validation. The models thus trained were tested on a different dataset (Columbia). Calibration was computed from 20 randomly sampled images for each participant. It is important to note that the Columbia dataset contains image with gaze direction with positive value of pitch (i.e., looking upwards), while pitch values are always negative (or very small positive value) in the MPII images. This represents an out-of-distribution challenge for our algorithms. Since there are no

Table 2. Within Dataset Evaluation (MPII) - Mean Angular Error (degrees)

	Number of Calibration Images					
Algorithm	1	5	9	16	32	64
GRS [Yu et al. 2019]	5.0°	4.2°	4.0°	-	-	-
FAZE [Park et al. 2019]	4.7°	4.0°	3.9°	3.8°	3.8°	3.7°
GazeML [Park et al. 2018b]	8.5°	7.8°	7.2°	7.0°	6.9°	6.7°
End-to-End+pupil	**4.6°**	**3.5°**	**3.4°**	**3.3°**	**3.3°**	**3.3°**
End-to-End [Chen and Shi 2020]	**4.6°**	3.6°	**3.4°**	3.4°	**3.3°**	**3.3°**
GeoGaze	5.0°	4.0°	3.8°	3.7°	3.5°	3.5°

published results with other algorithms for this cross-data set evaluation, we only present results from GeoGaze and End-to-End (with and without pupil center).

In Table. 3, we show the resulting average angular errors for all images in the Columbia dataset. In addition, we show the pitch angle error when pitch angles are higher or lower than 0°. While the algorithms perform similarly when the pitch angle < 0°, GeoGaze has noticeably lower error (5.6°) than End-to-End (6.4°) and End-to-End+pupil (6.6°)for images with large pitch. This behavior is similar to what observed in the out-of-distribution experiments of Sec. 4.2.

Table 3. Cross-dataset Evaluation (MPIIGaze → Columbia) - Mean Angular Error (degrees)

Algorithm	All	Pitch Error (Pitch > 0°)	Pitch Error (Pitch < 0°)
End-to-End+pupil	**6.6°**	6.6°	**3.3°**
End-to-End [Chen and Shi 2020]	6.8°	6.4°	3.6°
GeoGaze	6.7°	**5.6°**	3.4°

We also considered a second cross-dataset experiment, with the GeoGaze and End-to-End algorithms trained on the UTMultiview dataset [Sugano et al. 2014] and tested on both the Columbia [Smith et al. 2013] and the MPIIGaze [Zhang et al. 2015] datasets. For comparison, we also report results from two other model-based algorithms: GazeML [Park et al. 2018b] and the algorithm of Wang et al. [Wang and Ji 2017]. It is important to note that these two model-based algorithms are not trainable on specific datasets. Note from Table. 4a that GeoGaze produces

(a) Cross-dataset Evaluations (UTMultiview → Columbia, MPIIGaze) - Mean Angular Error (degrees)

Algorithm	Columbia	MPIIGaze
[Wang and Ji 2017]	7.1°	-
GazeML [Park et al. 2018b]	7.1°	6.9°
End-to-End+pupil	6.7°	8.3°
End-to-End [Chen and Shi 2020]	7.2°	7.4°
GeoGaze	**5.4°**	**6.6°**

(b) Cross-dataset Evaluations (UTMultiview → Columbia, MPIIGaze) - No calibration - Mean Angular Error (degrees)

Algorithm	Columbia	MPIIGaze
GazeML [Park et al. 2018b]	8.7°	**8.4°**
End-to-End+pupil	8.8°	10.8°
End-to-End [Chen and Shi 2020]	8.2°	9.2°
GeoGaze	**7.1°**	8.8°

consistently smaller errors than the other algorithms considered.

Table. 4b shows results obtained without per-person calibration. Note that this type of calibration is not always feasible [Sugano et al. 2016; Zhang et al. 2013]. In this case, results are biased due to the fact that the κ angles are not considered (they were forced to 0 in our experiments). Note that results from the algorithm of Wang et al. [Wang and Ji 2017] were not available for this experiment. Even

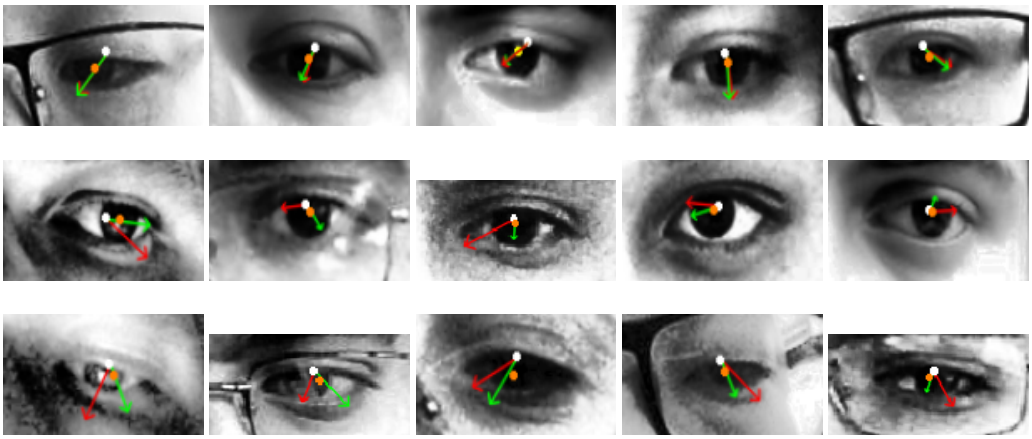


Fig. 4. Examples of gaze estimation using GeoGaze. White dot: **pec**; Orange dot: **pc**; Green arrow: estimated visual axis; Red arrow: ground-truth visual axis. Row1: Examples of gaze prediction with error $< 2^\circ$. Row2: Examples of gaze prediction with error $> 6^\circ$. The pupil center **pc** appears to be correctly localized, suggesting that the problem could be with the localization of the pseudo eye center **pec**. Row3: Examples of gaze prediction with error $> 6^\circ$. In these cases, the pupil center **pc** is clearly mislocated, which certainly contributed to the large gaze error.

in this highly biased case, GeoGaze performed comparatively better than GazeML and End-to-End for the Columbia dataset. For MPIIGaze, the performance was better than End-to-End with and without pupil center but worse than GazeML.

4.3.3 Ablation Study. Table. 5 compares previously shown results for the GeoGaze algorithm, against those obtained when the second branch (without the geometric layer) is removed during training. It is seen that the second branch (which is not used during deployment) facilitates training, and that its use leads to improved results.

Table 5. Ablation Study - Mean Angular Error (degrees)

Algorithm	MPIIGaze → MPIIGaze	MPIIGaze → Columbia	UTMultiview → Columbia	UTMultiview → MPIIGaze
GeoGaze (2nd branch removed)	4.0°	7.2°	6.2°	7.0°
GeoGaze	**3.8°**	**6.7°**	**5.4°**	**6.6°**

4.3.4 Qualitative Analysis. Fig. 4 contains examples of gaze detection using GeoGaze (green arrow), shown along with the ground-truth visual axis (red arrow). The second and third row in the figure contain images with substantial gaze error ($> 6^\circ$). In the third row, the pupil center **pc** is poorly located, which may justify the large error. For the images in the second row, the pupil center appears to be correctly located. In this case, the error is likely to be due to a poorly located pseudo eye center **pec**.

5 GEOMETRIC MODEL: ERROR SENSITIVITY

One of the advantages of using a geometric model for gaze estimation is that it makes it possible, to some extent, to measure the sensitivity of the system to errors in specific components, such as

those that measure relevant features. This can be useful for multiple reasons. It may help explain unexpected errors, by tracking the problem down to faults in individual components. It may allow identification of the weakest links in the chain, which may require particular attention in terms of design or training. In some cases, it may also be possible to predict when a specific component may fail based on observable data (e.g., image blur), which may provide a means to determine the reliability of the produced result.

GeoGaze computes the location of two feature points in the image (**pec** and **pc**) as well as the distance D of the eye center to the camera. It then generates the pseudo pupil axis **PPA** by backprojecting **pec** to distance D, and by backprojecting **pc** to a point **PC** at distance r to **PEC**. The pseudo pupillary axis $\mathbf{PPC} = \mathbf{PC} - \mathbf{PEC}$ is then rotated according to the κ angles. Barring errors in estimation of the κ angles, any errors in the direction of **PPC** are due to errors in **pec**, **pc**, or D. In the following, we will derive the angular error as a function of errors in the computation of **pc** and D. Note that errors in **pec** have an identical effect on the estimated direction of **PPC** as errors in **pc**.

To simplify our analysis, we will assume that both κ angles are $0°$, and will use a simplified eye model with a fixed eye center, located along the camera's optical axis at a distance D from the camera's optical center. We will consider an eye rotation by θ around a fixed vertical axis. Thus, the eye center projects onto the principal point, and the pupil center projects onto a horizontal line with eye rotation. Note that the segment joining the eye center and the pupil center (Fig. 2a) subtends an angle α at the optical center with $\tan\alpha = r\sin\theta/(D - r\cos\theta)$, and that the segment $\mathbf{pc} - \mathbf{pec}$ has horizontal component equal to $f \cdot r\sin\theta/(D - r\cos\theta)$, where f is the camera's focal length (remember that $\mathbf{pc} - \mathbf{pec}$ has vertical component equal to 0.) In the following, we will assume the following representative values for the considered quantities: f=1300 pixels; D=500 mm. As mentioned earlier, we set r=12 mm.

5.1 Sensitivity to Errors in pc

The error $\Delta\theta$ in gaze direction can be related to the error $\Delta(\mathbf{pc}_x - \mathbf{pec}_x)$ as by:

$$\Delta\theta \approx \Delta(\mathbf{pc}_x - \mathbf{pec}_x)\frac{d\theta}{d(\mathbf{pc}_x - \mathbf{pec}_x)} \text{ with: } \frac{d\theta}{d(\mathbf{pc}_x - \mathbf{pec}_x)} = -\frac{(D - r\cos\theta)^2}{f \cdot r \cdot (r - D\cos\theta)} \tag{5}$$

As expected, increasing r reduces the sensitivity of the estimated θ to errors in **pc**. Fig. 5 (a) shows the error $\Delta\theta$ as a function of θ for errors in $\mathbf{pc}_x - \mathbf{pec}_x$ ranging from 1 to 3 pixels. While $\Delta\theta$ is relatively constant with θ, it is seen that just 1 pixel of error in the location of pupil center or if eye center can lead to a $2°$ gaze direction error. This shows the importance of accurate pupil and eye center localization with this type of algorithms. For context, we note that the pupil localization errors reported using the GazeML algorithm [Park et al. 2018b] on the GI4E dataset [Sesma et al. 2012] have a median value of approximately 3% of the palpebral fissure width. Given an average palpebral fissure width value of 30 mm, this corresponds to an error of about 1 mm. With the focal length and distance parameters considered here, this maps to a 2.6 pixel error in the localization of **pc**. Following this reasoning, and using (5), we should expect a median error of about $4.7°$ due to incorrect localization of the pupil center. This number is consistent with the errors we measured in our experiments; in fact, we typically obtain lower errors for within-dataset evaluations (see Table. 2). Note that the quality of pupil localization is highly dependent on factors such as illumination and gaze direction (see e.g. Fig. 4.)

5.2 Sensitivity to Errors in D

The sensitivity of gaze direction to errors in the distance D to the user can be computed as $\Delta\theta \approx \frac{\delta\theta}{\delta D}\Delta D$, with $\frac{\delta\theta}{\delta D} = -\sin\theta/(r - D\cos\theta)$. As seen in Fig. 5 (b), this type of error becomes significant for large gaze angles. Note that, while large errors of D (e.g., 100 mm, which is the

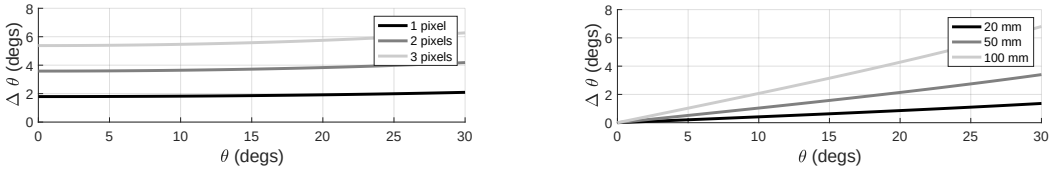


Fig. 5. Gaze direction error $\Delta\theta$ as a function of gaze direction θ for: (a) different values of error in $\mathbf{pc}_x - \mathbf{pec}_x$ (1, 2, 3 pixels); (b) different values of errors in D (20, 50, 100 mm.)

extreme case shown in Fig. 5 (b)) are unlikely, it is reasonable to assume that a generic face model that simply detects visible features such as palpebral fissure corners may be unable to measure the actual distance to the eye center, and errors of 10 or 20 mm may be expected. Our analysis shows that these errors may result in a gaze direction error of about 0.6° for $\theta = 15°$, and of 1.4° for $\theta = 30°$.

6 CONCLUSION

We have presented a new algorithm for gaze estimation that uses a model of the eye, embedded in a deep neural network. From the architectural standpoint, we simply constrain the flow of computation by means of a final “geometric layer". Compared to prior model-based algorithm, our GeoGaze system (1) does not require ground-truth eye center annotations, (2) is trained end-to-end, and (3) does not require a personalization phase with fixed head pose. As with any other gaze tracking systems, a personalization phase is needed to estimate the κ angles, which vary from individual to individual.

Our experimental results highlighted some interesting properties of the GeoGaze algorithm. In the toy example with synthetic datasets (Sec. 4.2), as well as in cross-dataset evaluation (Sec. 4.3.2), we showed that, when compared to a black box end-to-end system, GeoGaze was able to “extrapolate" gaze directions that were unseen during training. This is not surprising: under the assumption that the pseudo eye center **PEC** is relatively independent of gaze, previously unseen gaze angles are easily measured if the pupil position can be correctly measured. In contrast, extrapolation is known to be a difficult problem for neural networks [Webb et al. 2020].

ACKNOWLEDGMENTS

Research reported in this publication was supported by the National Eye Institute of the National Institutes of Health under award number R01EY030952-01A1. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. The material in this publication is directly taken from the chapter 2 of the published thesis [Kaur 2021].

REFERENCES

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

David A Atchison, George Smith, and George Smith. 2000. *Optics of the human eye*. Vol. 2. Butterworth-Heinemann Oxford.

Jixu Chen and Qiang Ji. 2008. 3D gaze estimation with a single camera without IR illumination. In *2008 19th International Conference on Pattern Recognition*. IEEE, 1–4.

Jixu Chen and Qiang Ji. 2011. Probabilistic gaze estimation without active personal calibration. In *CVPR 2011*. 609–616. https://doi.org/10.1109/CVPR.2011.5995675

Zhaokang Chen and Bertram Shi. 2020. Offset Calibration for Appearance-Based Gaze Estimation via Gaze Decomposition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Zhaokang Chen and Bertram E. Shi. 2018. Appearance-Based Gaze Estimation Using Dilated-Convolutions. In *ACCV*.

Yihua Cheng, Feng Lu, and Xucong Zhang. 2018. Appearance-Based Gaze Estimation via Evaluation-Guided Asymmetric Regression. In *ECCV*.

Yihua Cheng, Haofei Wang, Yiwei Bao, and Feng Lu. 2021. Appearance-based Gaze Estimation With Deep Learning: A Review and Benchmark. *ArXiv* abs/2104.12668 (2021).

Yihua Cheng, Xucong Zhang, Feng Lu, and Yoichi Sato. 2020. Gaze Estimation by Exploring Two-Eye Asymmetry. *IEEE Transactions on Image Processing* 29 (2020), 5259–5272. https://doi.org/10.1109/TIP.2020.2982828

C Cui and V Lakshminarayanan. 2003. The reference axis in corneal refractive surgeries: visual axis or the line of sight? *Journal of Modern Optics* 50, 11 (2003), 1743–1749.

Murthy L R D and Pradipta Biswas. 2021. Appearance-Based Gaze Estimation Using Attention and Difference Mechanism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 3143–3152.

Tobias Fischer, Hyung Jin Chang, and Yiannis Demiris. 2018. RT-GENE: Real-Time Eye Gaze Estimation in Natural Environments. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Wolfgang Förstner and Bernhard P Wrobel. 2016. *Photogrammetric computer vision*. Springer.

Elias Guestrin and Moshe Eizenman. 2006. General Theory of Remote Gaze Estimation Using the Pupil Center and Corneal Reflections. *Biomedical Engineering, IEEE Transactions on* 53 (07 2006), 1124 – 1133. https://doi.org/10.1109/TBME.2005.863952

Qiang Ji and Kang Wang. 2021. Bayesian Eye Tracking. *CoRR* abs/2106.13387 (2021). arXiv:2106.13387 https://arxiv.org/abs/2106.13387

Harsimran Kaur. 2021. *It's All in Your Eyes: Gaze Tracking, Synthesis, and Redirection*. Ph. D. Dissertation. https://www.proquest.com/dissertations-theses/s-all-your-eyes-gaze-tracking-synthesis/docview/2629428271/se-2?accountid=14523 Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2022-02-28.

Harsimran Kaur and Roberto Manduchi. 2020. EyeGAN: Gaze–Preserving, Mask–Mediated Eye Image Synthesis. In *Winter Conference on Applications of Computer Vision (WACV 2020)*. https://escholarship.org/uc/item/3vk9w8k9

Markku Kilpeläinen, Nicole M Putnam, Kavitha Ratnam, and Austin Roorda. 2021. The retinal and perceived locus of fixation in the human visual system. *Journal of Vision* 21, 11 (2021), 9–9.

Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations* (12 2014).

Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. 2016. Eye Tracking for Everyone. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Erik Linden, Jonas Sjostrand, and Alexandre Proutiere. 2019. Learning to Personalize in Appearance-Based Gaze Tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*.

Gang Liu, Yuechen Yu, Kenneth Alberto Funes Mora, and Jean-Marc Odobez. 2018. A Differential Approach for Gaze Estimation with Calibration. In *BMVC*.

Feng Lu, Yusuke Sugano, Takahiro Okabe, and Yoichi Sato. 2014. Adaptive Linear Regression for Appearance-Based Gaze Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 10 (2014), 2033–2046. https://doi.org/10.1109/TPAMI.2014.2313123

Francis Martinez, Andrea Carbone, and Edwige Pissaloux. 2012. Gaze estimation using local features and non-linear regression. In *2012 19th IEEE International Conference on Image Processing*. 1961–1964. https://doi.org/10.1109/ICIP.2012.6467271

Yeji Moon, Won June Lee, Seung Hak Shin, Ji Hong Kim, Ji Young Lee, Sei Yeul Oh, and Han Woong Lim. 2020. Positional Change of the Eyeball During Eye Movements: Evidence of Translatory Movement. *Frontiers in Neurology* 11 (2020), 1081.

Samuel Arba Mosquera, Shwetabh Verma, and Colm McAlinden. 2015. Centration axis in refractive surgery. *Eye and Vision* 2, 1 (2015), 1–16.

Seonwook Park, Emre Aksan, Xucong Zhang, and Otmar Hilliges. 2020. Towards End-to-end Video-based Eye-Tracking. In *European Conference on Computer Vision (ECCV)*.

Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges, and Jan Kautz. 2019. Few-Shot Adaptive Gaze Estimation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), 9367–9376.

Seonwook Park, Adrian Spurr, and Otmar Hilliges. 2018a. Deep Pictorial Gaze Estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Seonwook Park, Xucong Zhang, Andreas Bulling, and Otmar Hilliges. 2018b. Learning to Find Eye Region Landmarks for Remote Gaze Estimation in Unconstrained Settings. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications* (Warsaw, Poland) *(ETRA '18)*. Association for Computing Machinery, New York, NY, USA, Article 21, 10 pages. https://doi.org/10.1145/3204493.3204545

Sonia Porta, Benoît Bossavit, Rafael Cabeza, Andoni Larumbe-Bergera, Gonzalo Garde, and Arantxa Villanueva. 2019. U2Eyes: A Binocular Dataset for Eye Tracking and Gaze Estimation. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. 3660–3664. https://doi.org/10.1109/ICCVW.2019.00451

Rui Rodrigues, Joao P Barreto, and Urbano Nunes. 2010. Camera pose estimation using images of planar mirror reflections. In *European Conference on Computer Vision*. Springer, 382–395.

Sebastian Ruder. 2017. An Overview of Multi-Task Learning in Deep Neural Networks. *ArXiv* abs/1706.05098 (2017).

Peter H Schönemann. 1966. A generalized solution of the orthogonal procrustes problem. *Psychometrika* 31, 1 (1966), 1–10.

Laura Sesma, Arantxa Villanueva, and Rafael Cabeza. 2012. 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*. 217–220.

B.A. Smith, Q. Yin, S.K. Feiner, and S.K. Nayar. 2013. Gaze Locking: Passive Eye Contact Detection for Human?Object Interaction. In *ACM Symposium on User Interface Software and Technology (UIST)*. 271–280.

Yusuke Sugano and Andreas Bulling. 2015. Self-Calibrating Head-Mounted Eye Trackers Using Egocentric Visual Saliency. *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (2015).

Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. 2010. Calibration-free gaze sensing using saliency maps. In *Proceedings of the Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition*. 2667–2674. https://doi.org/10.1109/CVPR.2010.5539984

Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. 2014. Learning-by-Synthesis for Appearance-based 3D Gaze Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yusuke Sugano, Xucong Zhang, and Andreas Bulling. 2016. AggreGaze: Collective Estimation of Audience Attention on Public Displays. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 821–831. https://doi.org/10.1145/2984511.2984536

Li Sun, Zicheng Liu, and Ming-Ting Sun. 2015. Real time gaze estimation with a consumer depth camera. *Information Sciences* 320 (2015), 346–360. https://doi.org/10.1016/j.ins.2015.02.004

Yunjia Sun, Jiabei Zeng, Shiguang Shan, and Xilin Chen. 2021. Cross-Encoder for Unsupervised Gaze Representation Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 3702–3711.

Wang, Sung, and Ronda Venkateswarlu. 2003. Eye gaze estimation from a single image of one eye. In *Proceedings Ninth IEEE International Conference on Computer Vision*. 136–143 vol.1. https://doi.org/10.1109/ICCV.2003.1238328

Kang Wang and Qiang Ji. 2017. Real time eye gaze tracking with 3D deformable eye-face model. In *Proceedings of the IEEE International Conference on Computer Vision*. 1003–1011.

Kang Wang, Rui Zhao, Hui Su, and Qiang Ji. 2019. Generalizing Eye Tracking With Bayesian Adversarial Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Taylor Webb, Zachary Dulberg, Steven Frankland, Alexander Petrov, Randall O'Reilly, and Jonathan Cohen. 2020. Learning representations that support extrapolation. In *International Conference on Machine Learning*. PMLR, 10136–10146.

Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. 2016. Learning an Appearance-Based Gaze Estimator from One Million Synthesised Images. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*. 131–138.

Erroll Wood and Andreas Bulling. 2014. EyeTab: Model-Based Gaze Estimation on Unmodified Tablet Computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications*. Association for Computing Machinery, New York, NY, USA, 207–210. https://doi.org/10.1145/2578153.2578185

Yunyang Xiong, Hyunwoo J. Kim, and Vikas Singh. 2019. Mixed Effects Neural Networks (MeNets) With Applications to Gaze Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yuechen Yu, Gang Liu, and Jean-Marc Odobez. 2018. Deep Multitask Gaze Estimation with a Constrained Landmark-Gaze Model. In *ECCV Workshops*.

Yuechen Yu, Gang Liu, and Jean-Marc Odobez. 2019. Improving Few-Shot User-Specific Gaze Adaptation via Gaze Redirection Synthesis. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 11929–11938.

Yu Yu and Jean-Marc Odobez. 2020. Unsupervised Representation Learning for Gaze Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xucong Zhang, Seonwook Park, Thabo Beeler, Derek Bradley, Siyu Tang, and Otmar Hilliges. 2020. ETH-XGaze: A Large Scale Dataset for Gaze Estimation under Extreme Head Pose and Gaze Variation. In *European Conference on Computer Vision (ECCV)*.

Xucong Zhang, Yusuke Sugano, and Andreas Bulling. 2018. Revisiting Data Normalization for Appearance-Based Gaze Estimation. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications* (Warsaw, Poland) *(ETRA '18)*. Association for Computing Machinery, New York, NY, USA, Article 12, 9 pages. https://doi.org/10.1145/3204493.3204548

Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2015. Appearance-Based Gaze Estimation in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2017. It's written all over your face: Full-face appearance-based gaze estimation. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*. IEEE, 2299–2308.

Yanxia Zhang, Andreas Bulling, and Hans Gellersen. 2013. SideWays: A Gaze Interface for Spontaneous Interaction with Situated Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 851–860. https://doi.org/10.1145/2470654.2470775