UC Santa Cruz

UC Santa Cruz Electronic Theses and Dissertations

Title

It’s All in Your Eyes: Gaze Tracking, Synthesis, and Redirection

Permalink

https://escholarship.org/uc/item/9v40k78h

Author

Kaur, Harsimran

Publication Date

2021

Copyright Information

This work is made available under the terms of a Creative Commons Attribution License, availalbe at https://creativecommons.org/licenses/by/4.0/

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA
SANTA CRUZ

IT'S ALL IN YOUR EYES: GAZE TRACKING, SYNTHESIS, AND REDIRECTION

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

by

Harsimran Kaur

December 2021

The Dissertation of Harsimran Kaur
is approved:

Dr. Roberto Manduchi, Chair

Dr. James Davis

Dr. Yang Liu

Peter F. Biehl
Vice Provost and Dean of Graduate Studies

Copyright © by

Harsimran Kaur

2021

Table of Contents

Abstract

It's all in your eyes: Gaze tracking, synthesis, and redirection

by

Harsimran Kaur

The human eye manifests remarkable optical and mechanical characteristics that can be exploited to determine where a person is looking. While the IR-based devices can closely model such attributes, the webcam-based geometrical methods for determining gaze often suffer from low accuracy due to their sensitivity to estimated physical parameters. 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 data set 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 an "analytic 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.

For many gaze-related tasks, such as eye landmarks detection, obtaining manual annotations for deep learning models is labor-intensive and prone to errors.

Consequently, such models can be trained on the synthetic datasets, rendered using computer-graphics techniques. However, a model trained on these datasets might not perform well on the eye images captured in the real world due to significant photometric differences. We propose a domain adaptation algorithm to translate the images in the synthetic domain to the target domain of real-world images using an intermediate segmentation mask while preserving the annotations from the synthetic domain. Our method outperformed the previous domain adaptation techniques in maintaining the annotations, which is critical for training deep learning models for downstream tasks.

We further augment this approach to control the gaze and attributes of the generated eye image. We cast this problem as a style-based eye image synthesis and separately train a gaze redirector network to manipulate the gaze of the segmentation mask. The eye image with the target gaze is thus obtained by altering the gaze of the corresponding mask and then generating the eye image from the modified mask, while preserving the style. Since the segmentation masks are domain-independent, the whole pipeline does not require gaze labeled real-world data for training, while still showing competitive performance with the previous state-of-the-art algorithms trained on real-world gaze annotated datasets.

Acknowledgments

I am incredibly grateful to my advisor Prof Roberto Manduchi for giving me the opportunity to work on an important research problem. I appreciate you regularly taking out time for discussions and feedback. It helped build my thought process and shaped the ideas which finally went into the thesis. Your critical yet insightful advice and constant encouragement through numerous setbacks helped me materialize my Ph.D.

I am grateful to Prof James Davis for his valuable feedback on my proposal and for encouraging me to broaden the scope of my thesis. I am also thankful to Prof Yang Liu and Prof Mircea Teodorescu for reviewing my dissertation.

Thanks to Computer Vision Lab members, especially Swati, Brigit, and Andy, for the invaluable discussions and brainstorming sessions. A big thanks to Dr. Jixu Chen, my intern advisor at Facebook, for his mentorship. Interning with his team was an immense learning experience for me.

I am grateful to my father and brother for their continuous encouragement and my mother for her unwavering love. Thanks to the lovely companionship of my husband Tarun, the Ph.D. journey never felt that long and daunting.

And a special thanks to Aasra for lightening up the final year with her coos, smiles, and laughter. Raising my daughter along with grad school would not have been possible without the help and support of my mother-in-law. I am also thankful to Mrs. Satwinder Kaur for taking care of my daughter as hers.

The text of this dissertation includes reprints of the following previously published material:

- Kaur, Harsimran and Roberto Manduchi. "Subject Guided Eye Image Synthesis with Application to Gaze Redirection." 2021 IEEE Winter Conference on Applications of Computer Vision (WACV) (2021): 11-20.

- Kaur, Harsimran and Roberto Manduchi. "EyeGAN: Gaze–Preserving, Mask–Mediated Eye Image Synthesis." 2020 IEEE Winter Conference on Applications of Computer Vision (WACV) (2020): 299-308.

The co-author listed in this publication directed and supervised the research which forms the basis for the dissertation.

List of Figures

List of Tables

Chapter 1

Introduction

> The eyes can do a thousand things that the fingers can't.
>
> *Iranian Proverb*

The eyes are the fascinating sense organs that empower us to perceive the outside world. They not just enable us to navigate through our day-to-day lives but also help us collect non-verbal cues from people around us to recognize their mood, intention, and state of mind. Such visual cues are useful for computer vision problems too. Enabling a computer system to detect a person's gaze can open up a whole new dimension of human-computer interaction.

The knowledge of the user's gaze drives a host of applications in AR/VR systems. For example, foveated rendering in VR that helps to save real-time computations by selectively rendering in high definition only that portion of the scene where the user is looking, and rendering the rest in lower resolution. In Assisted Driving Systems, eye-tracking helps to alert the driver when their focus shifts away from the road. Eye-tracking also finds applications in consumer behavior analysis, market research as well as assistive technologies.

This research is inspired by the potential use of gaze tracking in a gaze-

contingent screen magnification system. Screen magnification is a well-established, popular technology for access to onscreen content. It is particularly helpful to people with low vision for reading documents. One of its main shortcomings is that it requires the user to continuously control the location of the focus of magnification with the mouse or a track-pad, to ensure that the magnified content of interest is within the screen view-port. This is a tedious process that may be time-consuming and ineffective. Rather than using a touchpad or a mouse, it would be a seamless experience for users to interact with the system using their gaze.

Determining where a person is looking is a challenging problem. IR-based gaze trackers can model the optics of the eye with high fidelty and can exhibit high accuracy with angular error close to 1°. Such devices use the refractions and reflections of the light from the eye surface to obtain the center of curvature of the corneal surface, a key point to compute the pupillary axis that is related to the gaze direction. However, the use of specialized hardware, limit their widespread adoption. To make eye-tracking accessible to more users, it is imperative to study webcam based solutions. Several geometrical methods based on eye images have been proposed for estimating gaze, but they often suffer from low accuracy ($> 7^\circ$ angular error). Part of the problem is that it is challenging to compute some eye parameters which are not perceivable from the eye images. The rough approximations can be made for such features, but the accuracy of the model is highly sensitive to these parameters. On the other hand, machine learning based methods do not require the explicit computations of eye features and map the appearance of the eye directly to the gaze direction. They can be trained end-to-end on the gaze-labeled eye image datasets and have shown to perform much better ($4^\circ - 5^\circ$), owing to the development of effective CNN-based architectures and collection of large gaze datasets.

Even so, their performance is limited to the characteristcs of the datasets on which they are trained and do not generalize well to the the samples obtained outside of the distribution, unlike geometrical models.

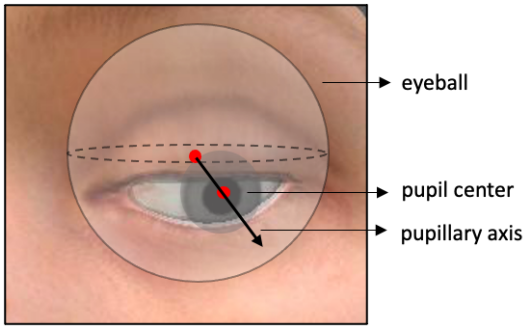
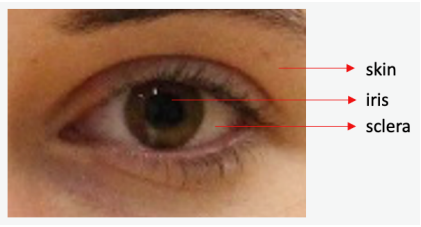


Figure 1.1: Human gaze

Consequently, we tackle the problem of webcam based gaze estimation using a hybrid approach. We combine the data-driven approach with the geometrical model for determining the gaze. We do so by embedding an analytical layer into the neural network, and train the network end-to-end on the gaze loss. Our method exploits the advantages of CNN based models, making it robust to illumination and appearance changes, while having the geometrical layer at the end enables the system to generalize better to novel gaze angles.

Another approach for improving the gaze-estimation systems is by automatically synthesizing realistic eye images with predetermined gaze direction. It enables us to generate annotated data without manual efforts, for training the data-driven models involving gaze estimation and intermediate tasks of pupil detection, iris segmentation, etc. There are computer graphics methods for rendering the eye images using underlying eye geometry and are automatically labeled with gaze and other eye features. However, they often lack realism and exhibit a domain gap between the generated images and those sampled from the real world. Conditional GAN-based domain adaptation methods can be applied to translate the image from one domain to another. Previous domain adaptation methods could not faithfully preserve the gaze and eye landmark anno-

tations between the source and the target domain. Our approach to alleviate this issue is to generate the eye images from the ternary segmentation mask. The segmentation mask consists of three regions: the iris, the white sclera and the skin area around the sclera. There are multiple advantages to deploying the mask for synthesizing eye images. Firstly, the shape of the mask represents the gaze direction and thus can guide the generation of eye images with specified gaze direction. Secondly, it can also ensure that the eye region boundaries are preserved between the mask and the synthesized image, if we train using the paired mask-image data.



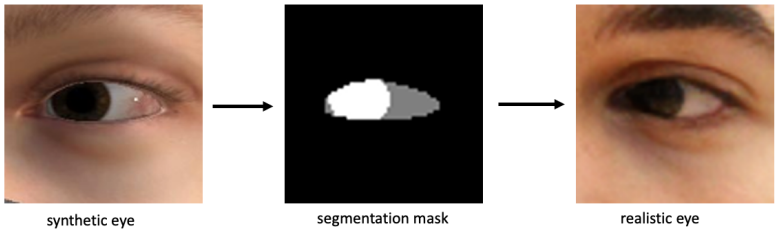


Figure 1.2: Mask-mediated eye image synthesis

The major challenge in the segmentation mask based approach is to procure the masks corresponding to the real eye images. This would require manual annotations of boundaries that can be tedious and time consuming. Instead, we make use of the segmentation mask automatically obtained from the simulated eye image. We alternately train a segmenter using the masks corresponding to the synthetic eye images, to extract noisy masks from real eye images, and a generator that generates realistic eye images from the noisy segmented masks, progressively improving the segmentation of the real eye images. Once we have the stable generator trained to generate realistic eye images from the corresponding masks, any

amount of labelled data could be generated corresponding to the ternary masks that can by automatically obtained from the purely synthetic eye images.

Interestingly, since the segmentation masks control the gaze direction, they can be manipulated to redirect the gaze in the synthesized images. A very interesting and desired use-case for redirecting gaze is in video conferencing. We naturally prefer to make an eye contact while interacting with other people. However in virtual interactions, it is difficult to look straight to the camera to simulate an eye contact while simultaneously being able to see the other person's video stream on the screen. For such scenarios, it is desirable to have a system to automatically correct the gaze to the desired direction (normally towards the camera), even if the user prefers to look at the screen instead. To redirect the gaze in the eye

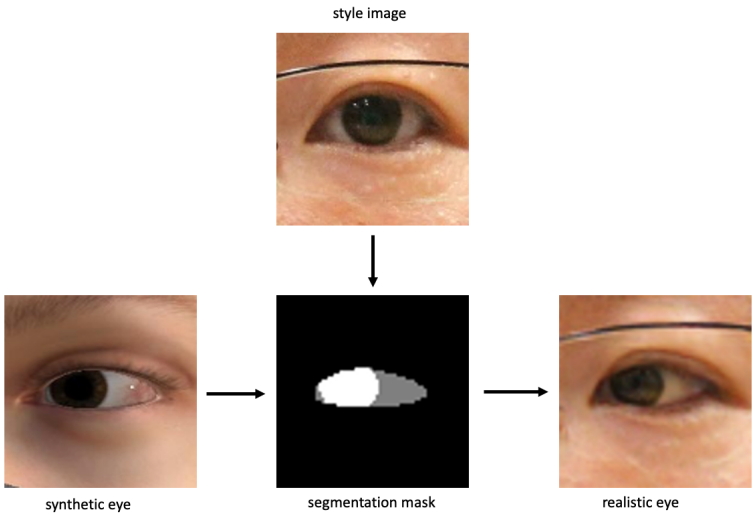


Figure 1.3: Style-based eye image synthesis

image, from a segmentation mask, we ought to have a control over the attributes of the generated image. Precisely, the generator should be able to preserve the identity of the source eye image for which the gaze is manipulated. To achieve this, the generator network is provided with the reference style image along with the segmentation mask with the desired gaze angle. While the segmentation mask

provides a proxy to the target gaze direction, the additional eye image input guides the generation of attributes like skin color and texture and the iris color of the synthesized image. We call this as style-based eye image synthesis. The task of gaze manipulation is limited to redirecting the mask, which can be achieved by training a separate network using gaze labels corresponding to the masks extracted from synthetic eye images. Importantly, since the segmentation masks are domain independent, we do not need gaze-annotated real eye images.

The thesis is organized as follows: Chapter 2 has two sections. Section 2.1 provides an overview of the gaze estimation problem, along with some useful concepts pertinent to understanding the problem itself. Section 2.2 describes the proposed deep geometrical model for estimating gaze. We explain the motivation of the approach followed by the details of the method. The proposed model is evaluated on multiple datasets and compared to both the geometrical and appearance-based techniques. The next chapters of the thesis are related to eye image synthesis. Chapter 3 provides the related work on the same. Chapter 4 describes the proposed method of eye image synthesis from segmentation masks. We also provide the experiments on different downstream tasks that are driven by the generated eye data. Chapter 5 provides the details on the proposed style-based eye image synthesis along with its application to gaze redirection.

Chapter 2

Gaze Estimation

> Of all the senses, sight must be the most delightful.
>
> *Helen Keller*

2.1 Background

We are interested in monocular gaze estimation from images taken from a webcam. A typical setting involves a user sitting in front of a laptop or a desktop, looking at the screen at distance of approximately 600 mm. We require a camera with can either be embedded in the laptop or could be placed near the screen, such that the person's face is in its field of view. The gaze ray or the line of sight is estimated by computing the 3d gaze origin and the gaze direction. We use camera coordinate system as the reference frame and all quantities are measured with respect to the camera. The screen which is usually not in the camera's FOV is calibrated using a mirror-based camera-screen calibration [86] method. The screen pose information is necessary to compute the gaze target from the intersection of gaze ray and the screen plane.

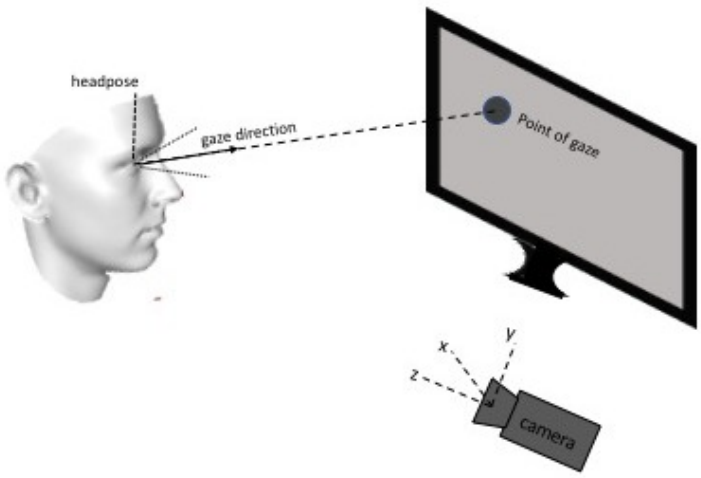
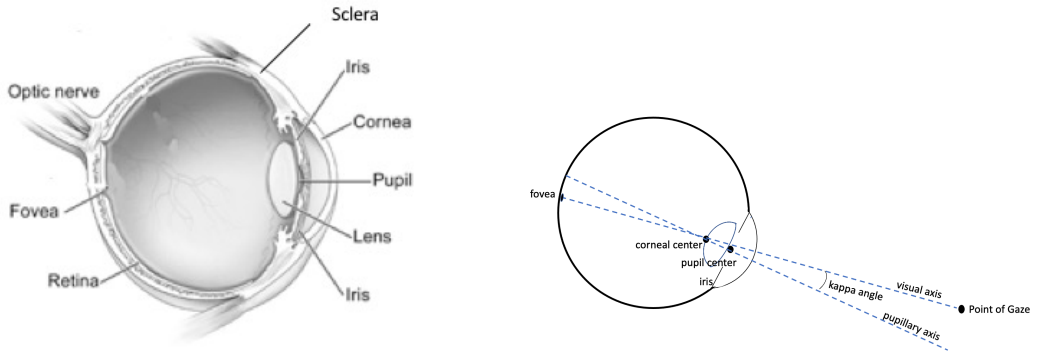


Figure 2.1: Gaze estimation system

2.1.1 Eye Geometry

For understanding the eye gaze, we need to understand the eye geometry. The human eye can be considered roughly spherical with a diameter of 24mm. The eye ball is covered by an opaque layer of tissue called sclera except at the anterior part. The anterior part is covered by a transparent layer called cornea. The cornea lies in front of the iris and has a radius of about 7.8mm [4]. At the center of the iris there is a small opening called pupil through which light enters the eye. The pupil regulates the amount of light entering the eye by continuously expanding and contracting. Fig 2.2a shows a labelled diagram of the human eye. The pupil, the iris, and the sclera are the only visible portions of the eye. The border between the iris and the sclera is called limbus. The center of the anterior corneal surface is called corneal center. The line passing through the corneal center and the pupil center is called the pupillary axis. The gaze direction is modelled as the visual axis, which is the line joining the point of gaze, the corneal center and the fovea. The fovea is a small region in the center of the retina which has the highest



(a) Diagram of the human eye [2] **(b)** Eye Schematic

Figure 2.2: The eye schematic

concentration of cones. These cones are responsible for perceiving fine details in the human vision. Since we have full acuity in this small region, while reading a text we have to move the eyes so that the light from the text falls on the fovea. Since fovea lies at the offset of the line joining the the corneal center and the pupil center, this leads to an angular difference between the pupillary axis and the visual axis. These are called the kappa angles (κ). They are the Euler angles of the rotation bringing the pupillary axis to the visual axis. The kappa angles varies from person to person.

2.1.2 Problem Definition

Formally, let $\mathbf{g_o}$ be the gaze origin, and the gaze direction be given by $\mathbf{v}$. Then gaze ray $\mathbf{g}$ is defined as

$$\mathbf{g} = \mathbf{g_o} + \lambda \mathbf{v} \tag{2.1}$$

Suppose the pose of the screen is given by $\mathbf{p_s}$ and $\mathbf{n_s}$, where $\mathbf{p_s}$ is the point on the screen and $\mathbf{n_s}$ is the normal to the screen plane. The 3d gaze target $\mathbf{g_t}$ can be

computed by intersecting the screen with the 3d gaze ray as

$$\mathbf{g_t} = \mathbf{g_o} + \lambda_s \mathbf{v} \tag{2.2}$$

where

$$\lambda_s = \frac{(\mathbf{p_s} - \mathbf{g_o}) \cdot \mathbf{n_s})}{\mathbf{v} \cdot \mathbf{n_s}} \tag{2.3}$$

2.1.3 Data Preparation and Normalization

The facial region and more importantly the eyes encapsulate most of the information needed to compute gaze. As a necessary first step, the eye/face region is extracted from the images by the camera that could also contain lot of unnecessary background information. Additionally, the gaze origin as well as the direction also depend on the pose of the head which has 6 DOF. The data normalization is performed to reduce the headpose to 2DOF [122]. To achieve this,six facial landmarks (eye corners and mouth corners) are computed using face detection with dlib library [49]. The pose of the head is defined with origin as the mid point of the two eyes. The x-axis is along the eye corners from right eye to the left eye and the z-axis point normally outwards from the face plan. The headpose is computed by fitting a 3d mean face shape model using EPnP algorithm [60]. We extract and

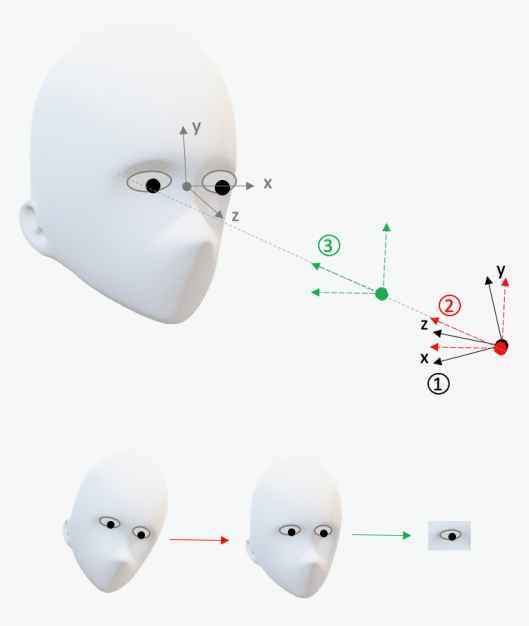


Figure 2.3: Data normalization

normalize the left and the right eye images separately. To normalize the images, the camera is rotated such that the x-axis of head and camera coordinates become coplanar, to make the eyes horizontal. This removes the yaw in the head rotation. The camera is further rotated such that the z-axis is along the back-projected line through the 2d mid point of the corresponding eye. The normalized camera is further translated along the z-axis such that it is at distance of 600 mm from the 3d eye mid point. This reduces the translation of the head to $[0, \quad 0, \quad 600]^T$ and the rotation to 2DoFs accounting for pitch and roll. The images are warped corresponding to the normalized camera and the ground truth gaze target is also converted to the new camera domain.

Mathematically, suppose head pose rotation matrix is given by $\mathbf{R_h} = [\mathbf{r_x}, \mathbf{r_y}, \mathbf{r_z}]$, and the translation is $\mathbf{t_h}$. The unit vector $\hat{\mathbf{t}}_\mathbf{h}$ along $\mathbf{t_h}$ is given by $\frac{\mathbf{t_h}}{\|\mathbf{t_h}\|}$ Then the rotation $\mathbf{R_n}$ from the original camera to the normalized camera is given by $\mathbf{R_n} = [\mathbf{n_x}, \mathbf{n_y}, \mathbf{n_z}]$, where

$$\mathbf{n_z} = \frac{\mathbf{t_h}}{\|\mathbf{t_h}\|} \tag{2.4}$$

$$\mathbf{n_y} = \frac{\mathbf{n_z} \times \mathbf{r_x}}{\|\mathbf{n_z} \times \mathbf{r_x}\|} \tag{2.5}$$

$$\mathbf{n_x} = \frac{\mathbf{n_y} \times \mathbf{n_z}}{\|\mathbf{n_y} \times \mathbf{n_z}\|} \tag{2.6}$$

The scaling matrix resulting from the translation of the camera along the z-axis is given by $\mathbf{S} = \text{diag}(1, 1, \frac{600}{\|\mathbf{t_h}\|})$. The image corresponding to the normalized camera is obtained by warping the original image using the perspective transformation $\mathbf{W}$ given by $\mathbf{K_n S R_n}^T \mathbf{K^{-1}}$ where $\mathbf{K}$ is the original camera matrix and $\mathbf{K_n}$ is the

normalized camera matrix. The corresponding gaze target $\mathbf{g_t}$ will be transformed as $\mathbf{R_n}^T\mathbf{g_t} + \mathbf{t_n}$, where $\mathbf{t_n} = [0, \quad 0, \quad \|\mathbf{t_h}\| - 600]$.

The gaze origin is generally assumed as the mid point of the 3d eye-corners obtained by computing the headpose. The corresponding gaze direction is given by

$$\mathbf{v} = \frac{\mathbf{g_t} - \mathbf{g_o}}{\|\mathbf{g_t} - \mathbf{g_o}\|} \tag{2.7}$$

The unit vector has two degrees of freedom and can be represented in euler angles as pitch θ and yaw ϕ and are given by

$$\theta = \arcsin(-\mathbf{v_y}) \tag{2.8}$$

$$\phi = \arctan(-\mathbf{v_x}, -\mathbf{v_z}) \tag{2.9}$$

The gaze direction is reversed such that the camera look direction angles become (0, 0). In our method, along with the gaze direction, we also compute the gaze origin which we call "pseudo eye center".

2.1.4 Metrics

The most common metric for evaluating gaze methods is the angular error between the gaze direction $\mathbf{n}$ computed from ground truth gaze target and gaze direction $\mathbf{n_p}$ predicted by the underlying algorithm. The angular error is calculated as

$$\theta_{error} = \arccos(\mathbf{n} \cdot \mathbf{n_p}) \tag{2.10}$$

This can be translated to error on the screen at a distance d as $d \tan \theta_{error}$. For a typical distance of 600 mm and an angular error of 4°, this corresponds to 42 mm.

2.2 Deep Model-Based Gaze Estimation

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 [15]. 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, 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 [32] 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 [5]. 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

[1]Note that in the literature (e.g. [32]), 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 [5].

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. 2.2.4). 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 is particularly vexing when trying to use machine learning algorithms for estimating these parameters: lacking labeled data, the network cannot be trained. An ingenious solution [110, 79] 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 [44] 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** and 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 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 2.4b) 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.

2.2.1 Related Work

The gaze estimation methods can primarily be categorized as appearance-based and model-based. The model-based methods apply some underlying eye geometry to compute gaze. On the other hand, the appearance-based approach

involves learning a mapping directly from the eye or face image to a 2D gaze target or 3D gaze direction. Further, there can be person-independent or personalized gaze estimation methods. In this section, we will briefly discuss the work related to these methods.

Appearance-based methods

Earlier appearance-based methods used classical machine learning techniques - linear regression [68], Support Vector Regression [71], Random Forests [95] to regress gaze from eye images. With the recent advances in deep learning architectures, CNN-based gaze estimation gained traction. Zhang et al. [123] trained a shallow LeNet model on the gaze dataset collected in the wild. This was further improved by [124, 53, 23, 121] which used the full-face images to train deeper CNN gaze estimation models. Signals from the head and the two eyes are leveraged in several ways to improve the gaze estimation models, for example, differences in the appearance of the two eyes [14, 16, 20], imposing geometric constraint [21], using attention to fuse the signals [13]. In addition to face/eye image input, visual saliency information has also been exploited [76, 94, 3, 10, 93]. Wang et al. [105] used Bayesian neural networks to output the distribution over the gaze rather than having a point estimate. Unsupervised approaches [117, 98] have also been proposed to learn better representations for gaze systems. Our method has a similar vein to appearance-based methods but we embed an additional geometric layer at the end to improve the gaze direction estimation, by making it more generalizable.

Model-based methods

Many of the geometric methods involve computing the 3D eye center and the pupil center to estimate pupillary axis, which is related to the gaze direction by a rotation. Chen et al. [9] computed the 3D eye center by calculating the intersection of the gaze rays pointing in different directions while keeping the head fixed. Sun et al. [97] instead used a depth camera to compute the offset to the 3D eye center from an anchor point on the face that allowed for free head movements during calibration. Wang et al. [103] fit a 3D face-eye deformable model to compute the 3D eye center directly on the RGB images without needing an anchor point. Other geometric methods have also been proposed - fitting an ellipse to the detected iris and computing the pose of the iris to get the gaze [113] [102].

Hybrid methods

Some works can be considered as combination of geometric and appearance based methods. In GazeML [79], the eye landmark detection model is trained on the synthetic dataset and the gaze is computed by fitting an orthographic geometric model. Park et al. [78] used intermediate gaze maps to estimate gaze. Yu et al. [118] combined landmarks and gaze in a constrained model. Qiang et al. [40] used the Bayesian framework along with the geometric model to compute the gaze.

Person-specific Gaze Estimation

Several personalized gaze models have also been proposed recently to account for the inter-personal bias. These methods require gaze-labeled calibration samples for each user to obtain personalized gaze estimator. Krafka et al. [53] extracted features from the last layer of the AlexNet [54] model trained on the gaze

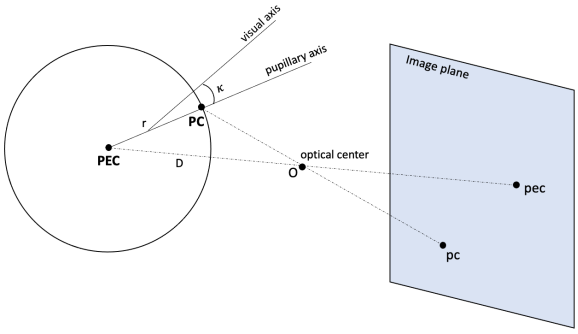
dataset and fit a Support Vector Regression model on the calibration samples. Liu et al. [63] trained a siamese network using two different eye images from the same person to compute the bias, which is then added to any subsequent predictions, while mixed-effects neural networks were used in [115] to develop person-specific models. Meta learning [22] approach was applied in [77] to train the weights in the MLP layers, which could be adapted to a new subject using as few as nine samples. Linden et al. [62] learned the person-specific parameters while training the gaze estimation network. In [11], the gaze angles are decomposed as pupillary axis and the 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. [11] outperforms the previous calibration-based methods, and it serves as the baseline in our work, and we call it as *end-to-end* method because it does not involve the geometric layer.

2.2.2 Method

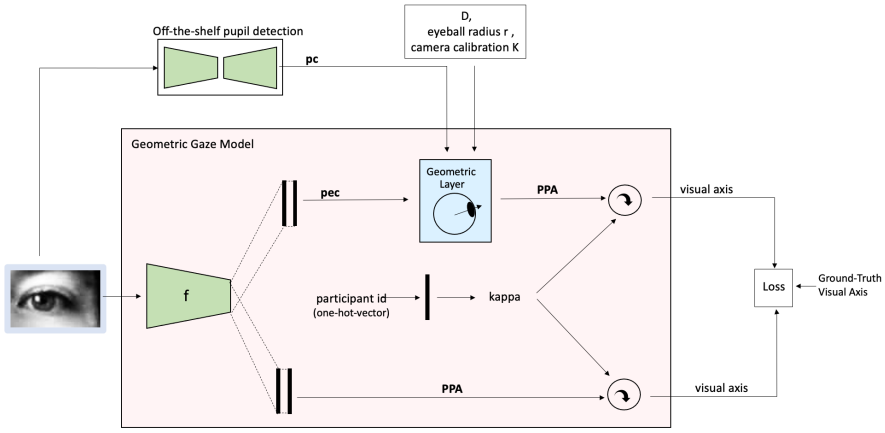
Geometric Preliminares

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. 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



(a) Geometric Model



(b) End-to-End geometry embedded CNN model

Figure 2.4: Proposed Model. On the top, we show the eye geometric model that we embed in our end-to-end CNN based model. On the bottom is shown the overview of the architecture

axis goes through the lens' nodal points (whose location cannot be computed in practice [19]) and the preferred retinal location, which, while relatively stable with time, changes from person to person [47]. It can only be observed through (subjective) fixation tests, whereas other types of axes [74] 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:

$$R^X(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \tag{2.11}$$

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:

$$R_{0,i} = R^Y(-\phi_{\mathbf{U}_i})R^X(\theta_{\mathbf{U}_i}) \tag{2.12}$$

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:

$$\mathbf{V}_i = R_{0,i}\mathbf{V}_0 = R^Y(-\phi_{\mathbf{U}_i})R^X(\theta_{\mathbf{U}_i})R^Y(-\phi_{\mathbf{V}_0})R^X(\theta_{\mathbf{V}_0}) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.13}$$

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

We will denote the relationship between the spherical coordinates of $\mathbf{V}_i$ and those of $\mathbf{U}_i$ as follows:

$$[\theta_{\mathbf{V}_i}, \phi_{\mathbf{V}_i}] = F_{\mathbf{V}_0}[\theta_{\mathbf{U}_i}, \phi_{\mathbf{U}_i}] \tag{2.14}$$

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 (2.13) 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 devi-

ates 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 (2.13), produces an angular error of 0.37°.

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. (2.13)), 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):

$$\mathbf{PEC} = D \cdot K^{-1}\overline{\mathbf{pec}} \tag{2.15}$$

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 [103, 9], 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 $\mathbf{PC} - \mathbf{PEC}$.

It is instructive to compare the GeoGaze algorithm with other model-based algorithms. For example, GazeML [79] 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 [110] 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. 2.2.4, 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 [9, 103], 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 [9], the eye center is determined by means of an offset vector that is computed during an initial per-person calibration phase. The model of [103] 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 $\mathbf{PC} - \mathbf{PEC}$ 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. 2.2.4); (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 [9, 103], 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 [73]. 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.

GeoGaze: Architecture

The architecture of the proposed system is rather straightforward (see Fig. 2.4). An input image, after pose normalization as per [122], 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. 2.4b). This data is passed on to a "geometric layer". Specifically, from **pec**, **pc**, and D, the geometric layer: (1) backprojects **pec** into **PEC** (Eq. (2.15)); (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. (2.13)) 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 [24] 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. 2.4). 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 [118, 89].

The overall loss to be minimize is thus:

$$\mathcal{L} = \mathbb{E}\left(\|[\theta_{\mathbf{VA}}, \phi_{\mathbf{VA}}] - F_{\mathbf{V}_0}[\theta_{\mathbf{PC-PEC}}, \phi_{\mathbf{PC-PEC}}]\| + \lambda_1\|[\theta_{\mathbf{V}_0}, \phi_{\mathbf{V}_0}]\| + \lambda_2\|\mathbf{pc} - \widehat{\mathbf{pc}}\| \right. \tag{2.16}$$
$$\left. + \lambda_3\|[\theta_{\mathbf{VA}}, \phi_{\mathbf{VA}}] - F_{\mathbf{V}_0}[\theta_{\mathbf{PPA}_{E2E}}, \phi_{\mathbf{PPA}_{E2E}}]\|\right)$$

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.

2.2.3 Experiments

Implementation details

We used the dilated-net architecture [12, 11] 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 [50] optimizer, with the learning rate set to 0.001 and batch size of 32. λ_2 is set to 0 after 4 epochs. The pupil center is computed by the GazeML network [79], which uses a hourglass architecture for iris landmarks detection[2]. GazeML produces a heatmap for each landmark, representing the probability of

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

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 [11] 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 consider four different datasets for our experiments. The first data set (UnityEyes [110]) is made by synthetic images, and is used in a toy scenario to highlight some of the features of GeoGaze. We also consider the following real-world datasets: MPIIGaze [123], Columbia [92], and UTMultiview [95]. 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.

Synthetic Data: UnitiEyes

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 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).

Table. 2.1 shows the results for both GeoGaze and End-to-End when tested on the in-distribution and on the out-of-distribution sets. 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°) is substantially larger than for GeoGaze ((3.3°). Clearly, the End-to-End system was unable to "extrapolate" gaze angles that were not seen in the training data. This was less of a problem for GeoGaze, as pupil and pseudo eye center could still be detected for these out-of-distribution images.

Table 2.1: GeoGaze vs End-to-End on UnityEyes – Mean Angular Errors (degrees)

Algorithm	In-distribution	Out-of distribution
End-to-End [11]	**1.0**°	9.2°
GeoGaze	1.3°	**3.3**°

In the next section, we will show that a similar behavior is observed for out-of-distribution images in real-world data.

Real-world Dataset

For these datasets, we normalize the eye images using a technique similar to [122], 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. 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.

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[77], GRS[119], and End-to-End[11] and also the model-based GazeML [79].

Results are shown in Table. 2.2. Note that for this within-dataset experiment, GeoGaze produces results that are comparable with those of other algorithms. Only the End-to-End system consistently produced a lower error.

Table 2.2: Within Dataset Evaluation (MPII) - Mean Angular Error (degrees)

	Number of Calibration Images					
Algorithm	1	5	9	16	32	64
GRS [119]	5.0°	4.2°	4.0°	-	-	-
FAZE [77]	4.7°	4.0°	3.9°	3.8°	3.8°	3.7°
GazeML [79]	8.5°	7.8°	7.2°	7.0°	6.9°	6.7°
End-to-End [11]	**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°

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 published results with other algorithms for this cross-data set evaluation, we only present results from GeoGaze and End-to-End.

In Table. 2.3, 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 two algorithms perform similarly when the pitch angle $< 0°$, GeoGaze has noticeably lower error (5.6°) than End-to-End (6.4°) for images with large pitch. This behavior is similar to what observed in the out-of-distribution experiments of Sec. 2.2.3.

Table 2.3: Cross-dataset Evaluation (MPIIGaze → Columbia) - Mean Angular Error (degrees)

Algorithm	All	Pitch Error (Pitch $> 0°$)	Pitch Error (Pitch $< 0°$)
End-to-End [11]	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 [95] and tested on both the Columbia [92] and the MPIIGaze [123] datasets. For comparison, we also report results from two other model-based algorithms: GazeML [79] and the algorithm of Wang et al. [103]. It is important to note that these two model-based algorithms are not trainable on specific datasets.

Table 2.4: Cross-dataset Evaluations (UTMultiview → Columbia, MPIIGaze) - Mean Angular Error (degrees)

Algorithm	Columbia	MPIIGaze
Wang et al. [103]	7.1°	-
GazeML [79]	7.1°	6.9°
End-to-End [11]	6.7°	8.0°
GeoGaze	**5.6°**	**6.5°**

Table 2.5: Cross-dataset Evaluations (UTMultiview → Columbia, MPIIGaze) - No calibration - Mean Angular Error (degrees)

Algorithm	Columbia	MPIIGaze
GazeML [79]	8.7°	**8.4°**
End-to-End [11]	9.7°	11.2°
GeoGaze	**7.3°**	9.2°

Note from Table. 2.4 that GeoGaze produces consistently smaller errors than the other algorithms considered.

Table. 2.5 shows results obtained without per-person calibration. Note that this type of calibration is not always feasible [125, 96]. 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. [103] were not available for this experiment. Even 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 but worse than GazeML.

Ablation Study Table. 2.6 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 2.6: 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°**	**5.6°**	**6.6°**

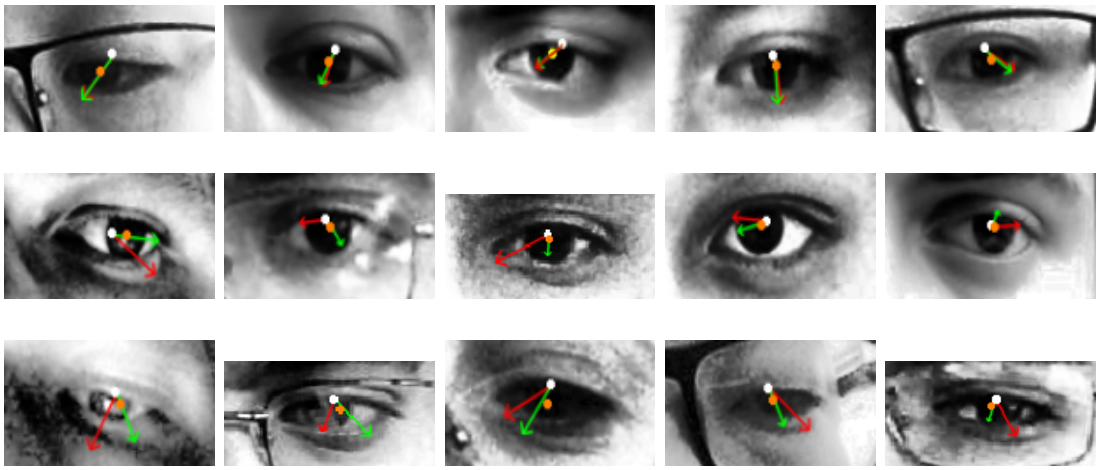


Figure 2.5: Examples of gaze estimation using GeoGaze. White dot: **pec**; Orange dot: **pc**; Green arrow: estimated visual axis; Red arrow: ground-truth visual axis.
Row1: Examples of gaze prediction with error $< 2°$.
Row2: Examples of gaze prediction with error $> 6°$. The pupil center **pc** appears 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 $> 6°$. In these cases, the pupil center **pc** is clearly mislocated, which certainly contributed to the large gaze error.

Qualitative Analysis Fig. 2.5 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 ($> 6°$). 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**.

2.2.4 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 back-projecting **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. 2.4a) subtends an angle α at the optical center with:

$$\tan \alpha = \frac{r \sin \theta}{D - r \cos \theta} \tag{2.17}$$

and that the segment $\mathbf{pc} - \mathbf{pec}$ has horizontal component equal to:

$$\mathbf{pc}_x - \mathbf{pec}_x = f\frac{r\sin\theta}{D - r\cos\theta} \tag{2.18}$$

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.

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:

$$\Delta\theta \approx \Delta(\mathbf{pc}_x - \mathbf{pec}_x)\frac{d\theta}{d(\mathbf{pc}_x - \mathbf{pec}_x)} \tag{2.19}$$

where:

$$\frac{d\theta}{d(\mathbf{pc}_x - \mathbf{pec}_x)} = -\frac{(D - r\cos\theta)^2}{f \cdot r \cdot (r - D\cos\theta)} \tag{2.20}$$

As expected, increasing r reduces the sensitivity of the estimated θ to errors in $\mathbf{pc}$. Fig. 2.6 (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 [79] on the GI4E dataset [90] 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 (2.20), 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. 2.2). We conjecture that, during training, the network, which is tasked with localizing **pec**, may learn to compensate for errors in pupil localization **pc**. Note that the quality of pupil localization is highly dependent on factors such as illumination and gaze direction (see e.g. Fig. 2.5.)

Sensitivity to Errors in D

The sensitivity of estimated gaze direction to errors in the distance D to the user can be computed as:

$$\Delta\theta \approx \frac{\delta\theta}{\delta D}\Delta D \tag{2.21}$$

where

$$\frac{\delta\theta}{\delta D} = -\frac{\sin\theta}{r - D\cos\theta} \tag{2.22}$$

As seen in Fig. 2.6 (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 extreme case shown in Fig. 2.6 (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 $\theta = 30°$.

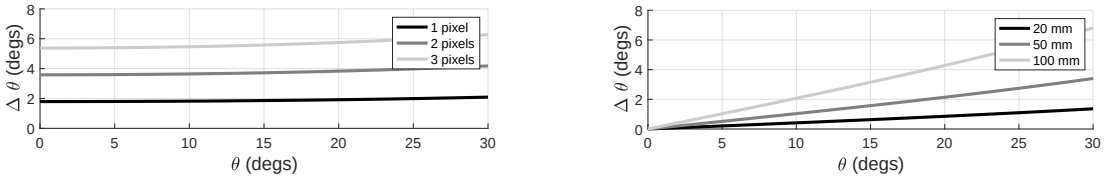


Figure 2.6: Gaze direction error $\Delta\theta$ as a function of gaze direction θ for: (a) different values of error in $\mathbf{pc}_x - \mathbf{pec}_x$ (1, 2, 3 pixels); (b) different values of errors in D (20, 50, 100 mm.)

2.2.5 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 UnityEyes data (Sec. 2.2.3), as well as in cross-dataset evaluation (Sec. 2.2.3), 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 [108]. More in general, it appears that GeoGaze, while producing results that are comparable with other algorithms when trained and tested on data from the same domain, it is particularly robust against the "domain gap"

effect when tested on a different domain. Of course, there are situations in which GeoGaze may fail, such as when, due to poor illumination, the iris is not well visible, thus inhibiting reliable pupil center localization. In these cases, a properly trained end-to-end algorithm that does not depend on pupil localization may still be able to recover gaze.

GeoGaze uses off-the-shelf algorithms for pupil detection and distance measurement, and the parameters for these algorithms are not optimized during training. In future work, we will evaluate whether optimization of these subsystems could result in improved results. Another interesting research direction would pair GeoGaze's geometric layer branch with an end-to-end system in a mixture-of-expert scheme [72]. We have already shown (Sec. 2.2.3) that adding such a branch facilitates training. The ability to soft-switch between these two branches at deployment could help in those situations where GeoGaze fails due, for example, to poor iris visibility.

Chapter 3

Background: Eye Image Synthesis

We are interested in generating realistic images of human eyes with a prescribed gaze direction. A direct practical applications of this technology is gaze redirection for teleconferencing [18] (making people appear as if they were looking at the camera [55]), photo correction [111], and video editing. A more indirect application is the creation of data sets for the training of image-based gaze tracking algorithms. These systems require large amounts of images with specific annotations. While some annotation types (e.g., the location of the pupil center) can be easily obtained via manual labeling, others are more challenging. For example, in order to determine the gaze direction of people visible in the images, data sets are often built by asking human subjects to look at a certain point on a screen [123] or at a calibrated location (such as an object [27]). Then, gaze direction annotations are extrapolated from geometric reasoning, such as by drawing a line from the location on the screen been fixated to the viewer's pupil, whose location in 3-D is assumed known. This is a relatively complex and error-prone procedure. Other features that cannot be obtained by manual labeling (because not observable) include the center of rotation of the subject's eyeball, which is needed to train model-based gaze tracking algorithms [9, 103].

3.1 Related Work

3.1.1 Eye Image Rendering

Due to its relevance in multiple application scenarios, the synthesis of realistic eye images has received considerable attention in the literature. Le et al. [69, 67] captured images under different head poses; eye images for new head poses were then synthesized via warping. Multiple cameras were used in [95] to build a 3D reconstruction of the eye region and to synthesize eye images for novel poses. Wood et al. [112, 110] rendered eye images (via computer graphics) using a 3D geometric eye model and head scans. This tool can be used to build very large data sets of perfectly annotated, high quality eye images. However, these synthetic images may not be representative of specific target domains, for which representative images may be available, but annotations may be difficult or impossible to obtain.

3.1.2 Domain Transfer

An approach to improving the quality of training data, while inheriting existing annotations, is to use a domain transfer algorithm. For example, SimGAN [91] transforms an eye image generated synthetically, with the desired head orientation and gaze direction, into a new image with the style of the target domain. This is accomplished by a GAN[31], trained to minimize an expected loss that includes two terms: the standard minimax adversarial loss (to ensure that the generated images look like samples from the target domain); and a L1 loss that penalizes discrepancies from the input synthetic eye image. This second term is meant to maintain consistency in gaze direction between the input synthetic image and the generated image. SimGAN produces impressive results, yet suffers from the

problem that direct comparison of the generated and of the input image is difficult, as the images are from different domains. Pixel-wise differences between the two images may thus be caused not only by a gaze direction discrepancy, but also by other irrelevant photometric factors (see e.g. Fig. 4.1). In order to mitigate the problem of cross-domain comparison, Lee et al. [59] relied on the CycleGAN training procedure [126]. CycleGAN trains two generators, mapping images from source to the target domain, and vice-versa. A “cyclic loss" is defined (in additional to the standard adversarial loss) that penalizes the L1 norm of the difference between an image I in one domain, and the image obtained by mapping I to the other domain, then mapping the result back to the original domain. Hence, the L1 loss term is computed only between images in the same domain. Yet, this strategy alone cannot ensure that gaze direction is preserved. For example, the generator mapping images from the source to the target domain may still introduce a gaze direction discrepancy, provided that the generator from the target to the source domain learns to remove this discrepancy (that is, to “re-direct" gaze back to the original direction.) While CycleGAN maps source and target domains into separate latent spaces, other algorithms [58, 64] use a shared latent space for domain transfer from unpaired data. The method by Wang et al. [104] combine image synthesis and gaze estimation in a unified model. Our EyeGAN system is directly inspired by the pix2pix algorithm for domain transfer [39]. Pix2pix requires pairs of images for training, where one image is from the source domain, and the other is the associated image in the target domain. A key insight of EyeGAN is that the generator does not need a highly detailed eye image input to produce a target domain image. What is needed is an input image with enough information to guide generation of a target domain image with the prescribed gaze direction. We use ternary mask images for this purpose.

3.1.3 Style-Based Image Generation.

The methods cited above generate new images in the style of the images used in the training data. While this may be acceptable for purposes such as producing a realistic data set, tasks such as gaze redirection call for precise control of the style at run time. Stated differently, a gaze redirection algorithm must ensure that the generated image is consistent with the style of the input image – it must preserve the features that characterize the appearance of the person in the image.

Image style can be modeled as a learned distribution using variational auto-encoders [51]. At inference time, one can sample the style from the learned distribution to generate the image [127] [80]. A method for deterministic generation of images with a specific style using a gram-matrix based style loss was proposed in [29]. The work in [38] showed that style could be transferred from input image to synthesized image using adaptive instance normalization. StyleGAN [43] used GAN based generator with adaptive instance normalization to synthesize novel human face images. The work in [8] also used adaptive normalization with a SPADE [80] generator for synthesizing eye images from segmentation masks consistent with the input style. Wang et al. [106] used style consistent as well style inconsistent pairs as an input to a discriminator, in order to impose the input style in the output image. In our work, we use a cyclic loss to enforce style.

3.1.4 Gaze Manipulation.

Earlier work on gaze manipulation focused correcting gaze direction such that the person appears to be looking at the camera, which is very desirable for applications such as videoconferencing. Some of the proposed approaches required specialized 3D data capture hardware to synthesize novel views of face and eyes [55], [30], [17]. Gaze redirection is a more generic task of manipulating the gaze to

any arbitrary direction. Monocular image-based gaze redirection can be achieved by learning a warping flow field between images with a known correction angle. This flow field can be computed using Random forests [52] or deep networks [28]. In [119], a flow field network is trained on synthetic eye images for gaze redirection, and domain adaptation is applied from synthetic to realistic eye images. This work was primarily focused on improving user-specific gaze estimation by using few-shot learning. Wood et al. [111] used a 3D morphable eye model for gaze manipulation. The work in [34] used GANs to synthesize eye images with redirected gaze using a specific reconstruction loss. Our work is similar to [34], however, rather than guiding synthesis by a gaze angle vector, we use segmentation masks. This mitigates the need for obtaining annotated gaze data for real-world eye images.

Chapter 4

EyeGAN: Gaze-Preserving, Mask-Mediated Eye Image Synthesis

Several methods have been proposed and demonstrated for the generation of realistic eye images, with generative adversarial networks (GAN [31]) arguably producing the best results. Controlling the gaze direction of the generated images, though, has proven more elusive. Part of the problem is that assessing gaze direction from an image, or at least determining whether it is congruent with that of another image, is difficult. Consider, for example, SimGAN [91], a popular algorithm that casts the synthesis problem as one of domain transfer. Starting from purely synthetic images, created using computer graphics from a model of the human eye, with prescribed gaze direction and head orientation, SimGAN generates realistic images sampled from a specific *target* domain. Gaze direction is controlled by adding to the adversarial loss a term that measures the L1 norm of the pixel-wise difference between the generated and the input image. Unfortunately,

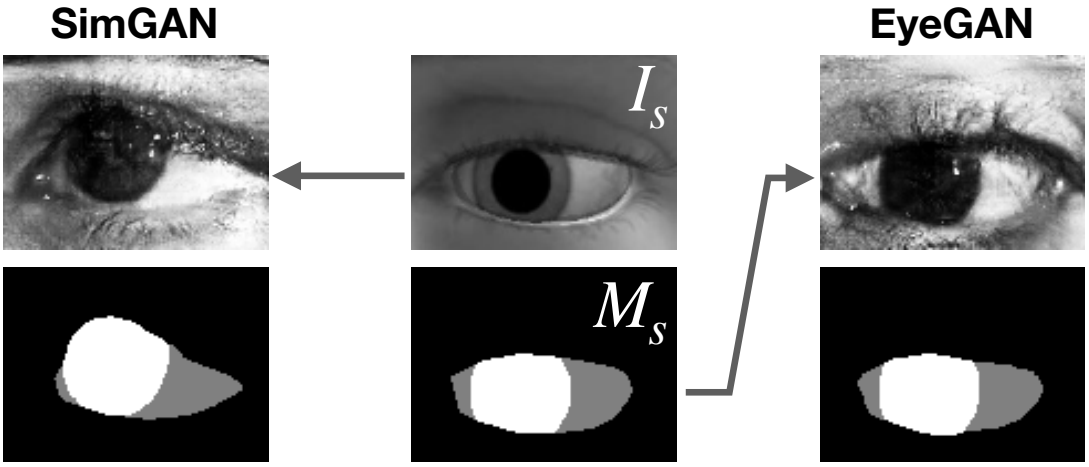


Figure 4.1: Center: Image–mask (I_s, M_s) pair synthesized by UnityEyes [110]. Left: The image generated by SimGAN [91] using I_s as input. Right: The image generated by our EyeGAN system using M_s as input. Both generated images are shown with the associated mask, computed by the segmenter trained with EyeGAN.

substantial photometric differences between the images in the two domains tend to bias this simple measure of gaze discrepancy, especially for larger image sizes. This is shown in the example of Fig. 4.1, wherein an image generated by SimGAN appears to look in a different direction than in the synthetic image provided in input.

Our approach to controlling the gaze direction of the generated images is inspired by the intuition that important information about gaze direction is revealed by a *segmentation mask* of the eye image. A well-formed segmentation mask describes three main components of an eye image: the iris, the white sclera, and the skin area surrounding the sclera (see Fig. 4.1, lower row.) It is well known that, for a fixed head pose, the iris eccentricity (relative location of the iris within the white sclera) determines the perceived gaze direction [99], and that the amount of visible sclera depends on the head orientation [84]. It thus stands to reason that such a ternary mask could be used to represent gaze direction and head ori-

entation. It is also conceivable that a well-designed segmenter should be able to extract acceptable segmentation masks from real eye images. Based on these observations, we decided to experiment with masks as *domain-independent proxies* for gaze direction. Closely related to our work is the Cycada algorithm [36], which used CycleGAN[126] for domain transfer, then segmented the resulting images using a fully convolutional network (FCN) [66] (symultaneously trained), where the FCN loss is fed back into the GAN to ensure correctness of the inherited annotations. Differently from Cycada, our EyeGAN algorithm directly starts from a segmentation mask.

Our proposed system takes in input a ternary mask produced by the UnityEyes graphic engine [110] with the desired head orientation and gaze direction, and generates an eye image with the same gaze direction in the “style" of the desired domain (Fig. 4.1, right column). The network is trained using a conditional GAN under the pix-to-pix paradigm [39]. Specifically, each training sample is formed by a pair (image, ternary mask) from the target domain. Whereas only the ternary mask is fed into the generator, the associated image is used for two purposes: to facilitate the job of the discriminator, and to enforce faithfulness of appearance by means of a L1 loss term that penalizes the difference between the input and the output images. Herein lies a critical difference with SimGAN: we never directly compare images from different domains, thus sidestepping the risk of bias from effects that are independent of gaze orientation.

A subtle but important characteristic of our algorithm is that the actual angles of gaze direction or head orientation are not needed at training time. We only use images from the target domain during training, and don’t assume that these images have been annotated (as mentioned earlier, obtaining the required type of annotation can be challenging). Head pose and gaze direction information is

embedded in the ternary masks, which are computed from the images themselves. When the generator is used to synthesize new images for a desired gaze direction, it takes in input a proper ternary mask, produced, for example, by UnityEyes.

For this system to work, it is critical that good quality ternary masks be available for images in the target domain. Standard segmentation algorithms can be used for this purpose, provided that enough labeled data is available for their training. Manual labeling (by drawing the iris and visible sclera regions in each image) is a conceivable option, albeit a time-consuming and error-prone one We decided instead to experiment with a training procedure that only uses the ternary masks automatically generated by UnityEyes along with the synthetic eye images. The segmenter is trained in parallel with the generator in an iterative fashion. This scheme is shown to produce excellent results after just a few iterations.

Our proposed EyeGAN system was evaluated comparatively in two different ways. First, we looked at the consistency of gaze direction by comparing the ternary masks computed on the generated images with the masks that were given as input. If the two masks agree, it can be expected that the perceived gaze direction of the generated images is congruent with the prescribed gaze direction, which was used to create the synthetic input. Second, we used EyeGAN to generate image data sets in target domains, while inheriting original annotations, and used this data to train networks for specific tasks: image region segmentation, pupil localization, and gaze direction estimation. These are applications of great interest for biometrics [75], medical diagnostic [100], and eye gaze tracking [79]. In many situations, annotating this type of data can be difficult or impossible, hence the interest in domain transfer methods for network training. The results of our experiments show that EyeGAN compares favorably with other state of the art domain transfer algorithms under the metrics considered.

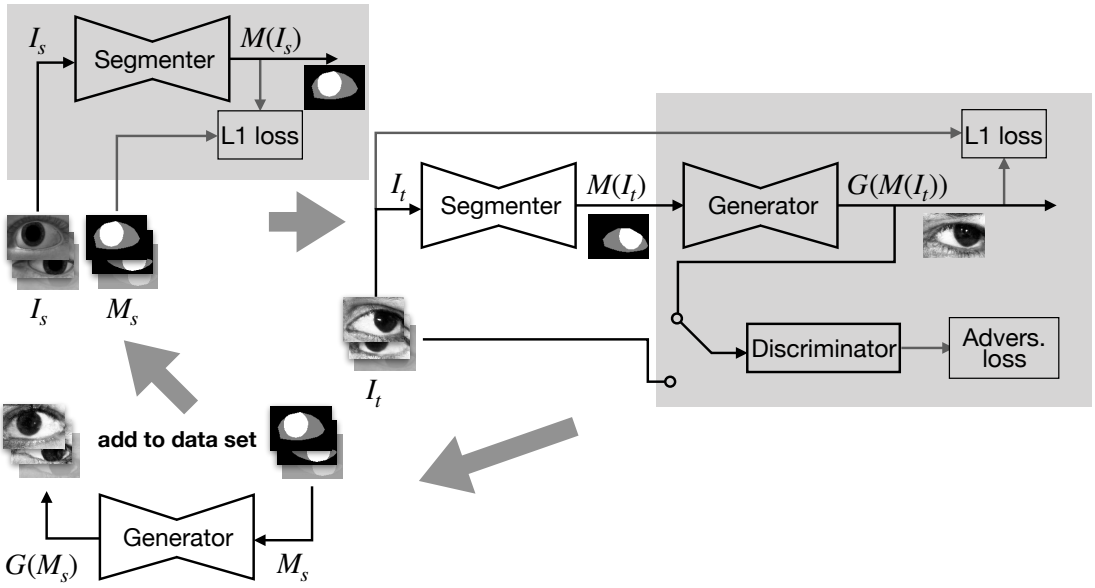


Figure 4.2: The overall training scheme of EyeGAN. At each step, the modules being trained are shown on a grey background.

4.1 Method

Our system generates eye images with a desired style, as represented by a set of (un-annotated) images taken in a particular target domain. Head pose and gaze direction for the generated images are controlled by means of ternary masks which, as discussed in the Introduction, function as proxies for the desired pose and gaze direction. Specifically, we assume that a data set of synthetic images I_s (where the subscript $_s$ stands for "source") is available, along with associated masks M_s, also synthetically produced. (Alternatively, real images with manual mask annotations could be used.) At run time, the generator, implemented as a convolutional network, takes one such mask in input, and produces an image in the desired style. Note that, unlike similar algorithms such as SimGAN or CycleGAN, we do not use synthetically generated eye images as input, but only the associated masks.

The generator is trained according to two criteria: *(1) Realism:* the generated

images must look realistic (as if they were actual samples from the target domain); *(2) Consistency:* the perceived gaze direction and head orientation of a generated image must conform to the prescribed values, in the sense that the associated mask should look similar to the mask fed into the generator. To generate realistic images, we follow the same conditional GAN strategy as pix2pix [39]. Specifically, the training data is formed by pairs mask–image in the target domain, of which only the mask is fed into the generator. The network is trained using a minimax adversarial loss, to which a L1 loss term is added to ensure that the generated image looks similar to the image associated with the input mask. This loss term enforces consistency: if the output image is similar (in L1 norm) to the image associated with the input mask, then the mask associated with the output image can be expected to be similar to the input mask. Critically, the L1 loss component is computed from two images that can be assumed to be from the same domain (unlike SimGAN). We used quadratic loss for the adversarial component, as it was shown to be superior to log loss in terms of training stability [70]. The overall loss function is thus:

$$\mathcal{L}(G,D) = \mathbb{E}_{I_t}[D(I_t)-1]^2 + \mathbb{E}_{M(I_t)}[D(G(M(I_t)))]^2 + \lambda\,\mathbb{E}_{I_t}\|I_t - G(M(I_t))\|_1 \quad (4.1)$$

This training scheme requires availability of images I_t in the target domain along with associated masks M_t. Unfortunately, such masks are normally not available, and their production via manual labeling can be exceedingly time consuming. Instead, we create the required masks from target domain images using a properly trained semantic segmentation algorithm (such as FCN [66]) that takes in an image I_t to produce a mask $M(I_t)$ (note the overloaded use of the symbol M). Still, the problem remains: in order to train the segmenter, we need image–mask pairs. We tackle this problem by leveraging the pairs (I_s, M_s) available in the synthetic eye

image data set. Intuitively, a segmenter trained on this data should be able to produce a recognizable, albeit probably not accurate, masks when applied to a target domain image. An example is shown in Fig. 4.3.

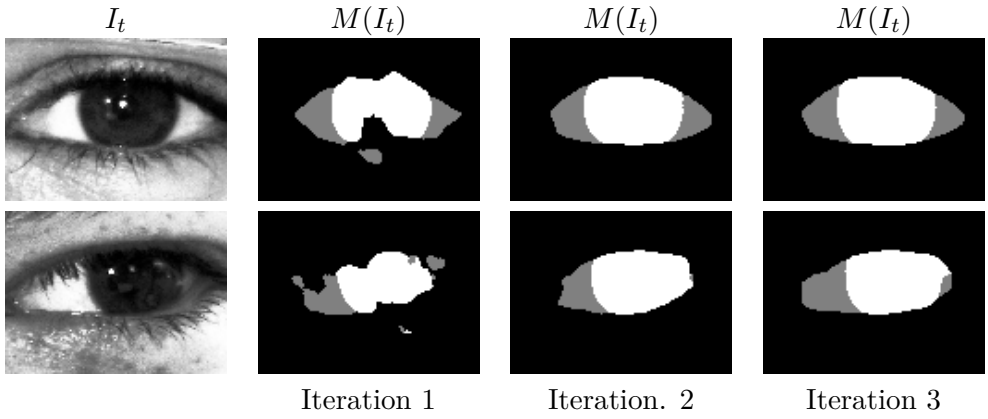


Figure 4.3: Two examples of segmentation of target domain images I_t. At Iteration 1, the segmenter was only trained using synthetic images and masks (I_s, M_s). In further iterations, pairs $(G(M_s), M_s)$ were added to the data set.

In order to improve the quality of segmentation, we augment the training data set for the segmenter (Fig. 4.2, top left) with image–mask pairs from the target domain. Of course, no improvement should be expected by simply adding to the training set pairs $(I_t, M(I_t))$, where the masks $M(I_t)$ were obtained using a suboptimal segmenter. Rather, we add pairs image-mask of the form $(G(M_s), M_s)$, where the masks M_s come from the synthetic data set (and thus are of perfect quality), and the associated images $G(M_s)$ are created by the generator, which, as explained earlier, was trained using pairs $(I_t, M(I_t))$. While not "real", these images can be considered to be samples from the distribution of the target domain (thanks to adversarial training.) We have observed that, after retraining the segmenter with this augmented data set, its performance on the target domain images improve noticeably (see Fig. 4.3). The process is then repeated. After 2–3 iterations, the segmenter produces satisfactory results, leading to good quality target domain image-mask pairs, which are used to re-train the generator. Fig. 4.2

shows the overall training scheme. Note that, at run time, the generator is only fed with synthetic mask M_s.

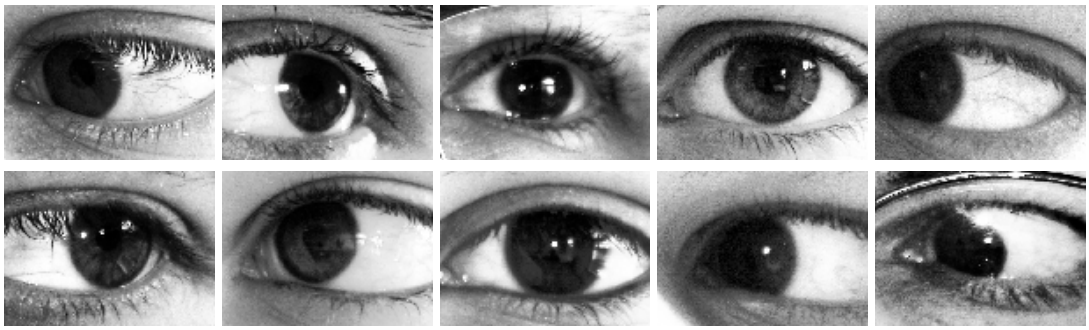


Figure 4.4: Sample images from the UBIRIS data set [83] (selected from those taken at a distance of 4 meters.) The images were histogram equalized.

4.1.1 Implementation Details

In all of our experiments, source eye images and masks were created using the UnityEyes tool [110]. The images were cropped to only include the eye region, and resized to 120×88 pixels. The ternary masks were obtained from the landmark points provided to indicate the boundary the sclera and of the iris regions. A set of 25,000 synthetic images and masks was thus generated.

The segmenter was implemented using the FCN-8s architecture [66]. The learning rate was set to 0.001, with batch size of 8. The network was optimized using Adam [50]. A pytorch implementation[1] of the pix2pix scheme [39] was used to train the generator mapping masks to target domain images. The architecture of the generator was similar to that of [42, 126] with six Resnet [33] blocks. The discriminator used the same PatchGAN architecture of [126]. The balancing coefficient λ was set to 40.

[1] `https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix`

4.2 Experiments

4.2.1 Gaze Direction Validation

A simple way to evaluate whether a generated eye image in the target domain (*target image* for short) is consistent with a desired gaze direction, is to compare its associated ternary mask (obtained via segmentation) with the mask fed into the generator, which was synthetically created according to the prescribed gaze direction. If the two masks are identical, we may assume that gaze direction is maintained (more precisely, the gaze direction as perceived when observing the image coincides with the gaze direction represented by the input mask). A target image $I_t = G(M_s)$ whose mask $M(I_t)$ (as computed by the segmenter) is dissimilar from the input mask M_s, is unlikely to be judged to have the same gaze direction. We measure the similarity S of two equally-sized ternary masks M_1, M_2 by the number of pixels in which the masks agree, divided by the total number of pixels in each masks. The number $S(M_1, M_2)$ takes values between 0 and 1, and is equal to 1 only when the masks are identical. When considering the similarity of two masks, one synthetically produced[2] for gaze direction θ (denoted by M_s^θ), the other obtained by segmentation of the target image generated with input mask M_s^ϕ ($M(I_t^\phi)$, where $I_t^\phi \equiv G(M_s^\phi)$), we will use the shorthand $S_{s,t}(\theta, \phi) \equiv S(M_s^\theta, M(I_t^\phi))$.

We frame gaze orientation validation in probabilistic terms by defining a probability density function on the gaze direction θ perceived upon observation of a target image generated under prescribed gaze direction ϕ: $p(\theta|I_t^\phi)$. This means that, upon observing the target image I_t^ϕ, with probability $p(\theta|I_t^\phi)d\theta$ the perceived gaze direction angle is within an interval $d\theta$ around θ.

[2]For simplicity of exposition, we only consider here one angle, instead of two, of gaze direction, and conflate head orientation with gaze direction.

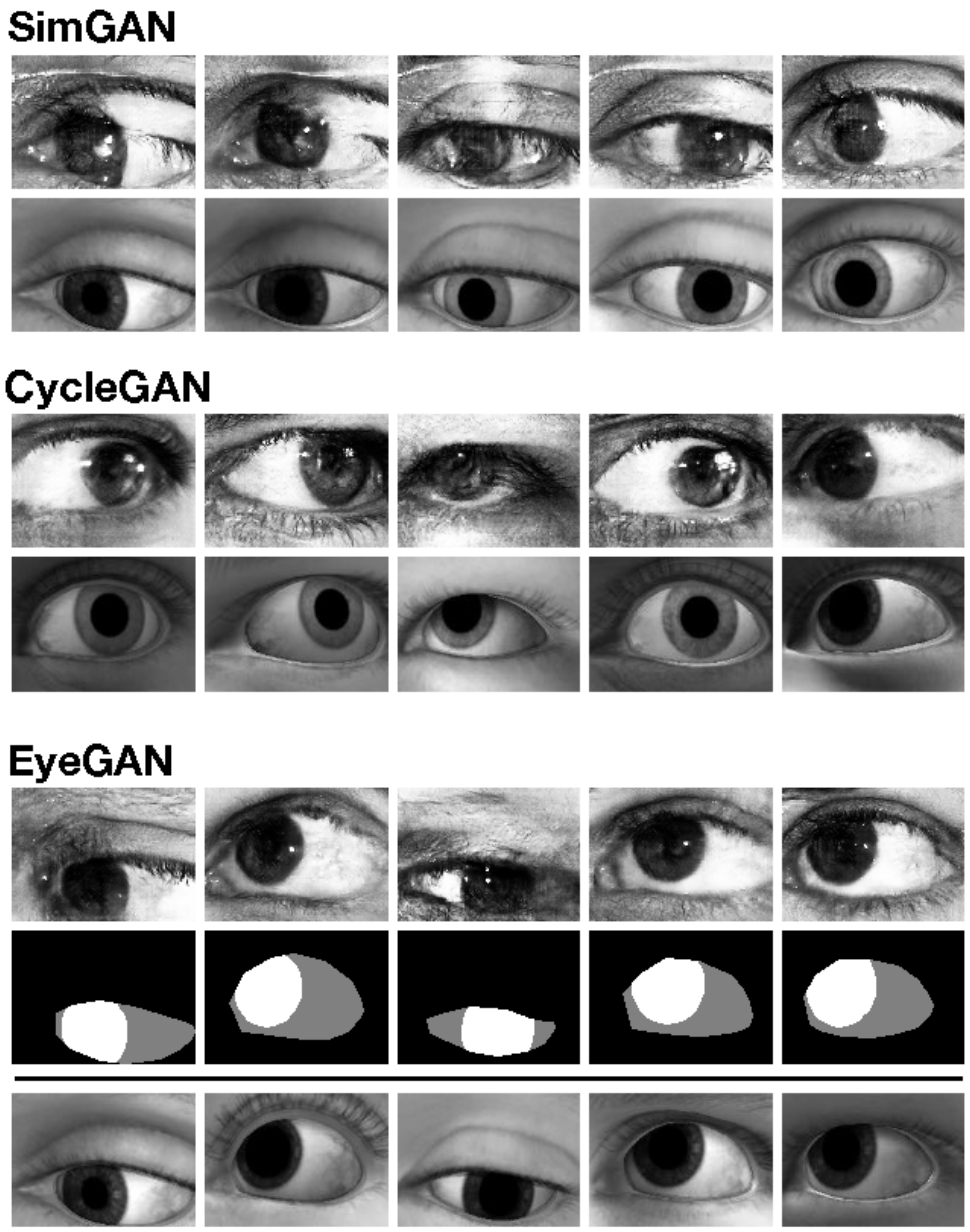


Figure 4.5: The five generated eye images with the lowest similarity score $S_{s,t}(\phi, \phi)$ for each method. Each image is shown with the synthetic image I_s or mask M_s that was fed into the corresponding generator. For reference, we also show the synthetic images I_s corresponding to the masks M_s for the EyeGAN case in the last row, even though only the masks M_s were fed into the generator in this case.

We will make the assumption that $p(\theta|I_t^\phi)$ is a function of the similarity between the mask $M(I_t^\phi)$, and the "ideal" mask for gaze direction θ, which is M_s^θ. Formally:

$$p(\theta|I_t^\phi) = K(\phi) f(S_{s,t}(\theta, \phi)) \tag{4.2}$$

where $f(S) = \exp(-\alpha \cdot (1 - S))$. α is a parameter that controls the dispersion of the density $p(\theta|I_t^\phi)$ (we set α=10 in our experiments.) $K(\phi)$ is a normalization constant that can be estimated as follows. We sample N gaze directions $\{\theta_i\}$ uniformly within the angular interval Θ in which θ can take values, and compute the mean $\bar{f}_\phi = \sum_{i=1}^N f(S_{s,t}(\theta_i, \phi))/N$:

$$\bar{f}_\phi \approx \frac{1}{K(\phi)} E_{\theta \sim \mathcal{U}(\Theta)}[p(\theta|I_t^\phi)] = \frac{1}{K(\phi)\|\Theta\|} \tag{4.3}$$

from which we obtain:

$$K(\phi) = 1/(\bar{f}_\phi \cdot \|\Theta\|) \tag{4.4}$$

Given a target image generated for gaze angle ϕ, the probability that the perceived gaze direction coincides with ϕ with tolerance $d\phi$ is $p(\phi|I_t^\phi)d\phi$. Hence, the probability that the perceived gaze direction for a generic target image is "correct" (coinciding with the prescribed gaze direction, which is assumed to be uniformly distributed) within tolerance $d\phi$, is $p(C_{s,t})d\phi$, with:

$$p(C_{s,t}) = E_{\phi \sim \mathcal{U}(\Theta)}[p(\phi|I_t^\phi)] \tag{4.5}$$

$$\approx \frac{1}{N} \sum_{j=1}^N p(\phi_j|I_t(\phi_j)) = \frac{1}{N} \sum_{j=1}^N K(\phi_j) f(S_{s,t}(\phi_j, \phi_j))$$

$$\approx \frac{1}{\|\Theta\|} \sum_{j=1}^N \frac{f(S_{s,t}(\phi_j, \phi_j))}{\sum_{i=1}^N f(S_{s,t}(\theta_i, \phi_j))}$$

$$= \frac{1}{\|\Theta\|} \sum_{j=1}^{N} \frac{e^{-\alpha(1-S_{s,t}(\phi_j,\phi_j))}}{\sum_{i=1}^{N} e^{-\alpha(1-S_{s,t}(\theta_i,\phi_j))}}$$

where the prescribed gaze directions $\{\phi_j\}$ are sampled uniformly within Θ.

The relative effectiveness of different eye image synthesis methods at preserving gaze direction can be quantified by comparing $p(C_{s,t})$, computed for each method, with the same quantity computed in the "ideal" case, where $M(I_t)$ is substituted by M_s (the resulting value is denoted by $p(C_{s,s})$). The ratio $p(C_{s,t})/p(C_{s,s})$ (termed *gaze consistency index*) is shown for the SimGAN, CycleGAN, and EyeGAN methods in Tab. 1 (note that term $\|\Theta\|$ disappears in the ratio.) We used the segmenter designed as part of the EyeGAN training process to extract masks from the target images in all three methods. These results were obtained using Eq. (4.5) on $N = 81$ input masks M_s (or associated synthetic images I_s in the case of SimGAN and CycleGAN), sampled uniformly in terms of gaze direction and head orientation. Target domain images were culled from the UBIRIS data set [83], selecting those taken at a distance of 4 meters. The images were resized to 120×80 pixels and histogram equalized. The results show that EyeGAN produces a substantially higher gaze consistency index than the other methods.

$\mathbf{p(C_{s,t})/p(C_{s,s})}$		
EyeGAN	CycleGAN	SimGAN
0.89	0.36	0.48

Table 4.1: Gaze consistency indices for the methods considered.

Examples of generated images for the three methods considered are shown in Fig. 4.5. For each method we selected the five images I_t with the lowest similarity score $S_{s,t}(\phi, \phi)$. Each image is shown next to the source image I_s or mask M_s (for EyeGAN) that was fed into the generator. We noted that EyeGAN generally produces images with better overall quality than the other two methods. Some

examples of poor quality images generated by EyeGAN are shown in Fig. 4.6.

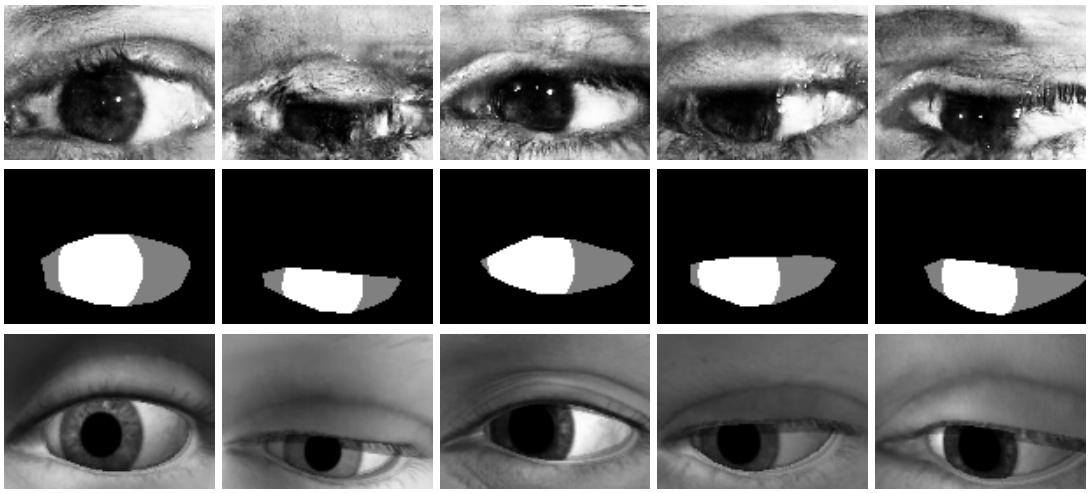


Figure 4.6: Examples of poor quality images generated by EyeGAN.

4.2.2 Eye Region Segmentation

Segmentation of various ocular regions as well as of the periocular region is instrumental for ocular biometric applications [75, 7]. Eye segmentation is also useful for animation of eyes and eyebrows of avatars for virtual reality [114]. Generation of training images via manual eye region segmentation and labeling, however, can be time consuming and thus expensive, and possibly error-prone. Domain transfer techniques can be used to generate large data sets with inherited annotations from labeled source domains. We comparatively evaluated our EyeGAN network as a tool to generate annotated training data for a segmenter tasked with extracting specific regions in eye images.

We considered two available labeled data sets for these experiment. The first data set (UBIRIS, already considered in Sec. 4.2.1) has manual annotations of the iris region. The second data set (SBVPI [88, 87]) contains 1822 eye images of 55 subjects looking towards four different directions (images were resized to 120×88

pixels.) SBVPI contains iris and pupil annotation for only a small number of subjects, but sclera and periocular masks are available for all subjects. Since both sclera and iris lie within the periocular region, the iris mask can be easily obtained as the area inside the periocular region that is not part of the sclera. Thus, for images in the SBVPI, we are able to access ground-truth ternary masks. Note that binary (for UBIRIS) and ternary (for SBVPI) masks were only used for validation (not during training).

Two subsets were culled from each data set, by partitioning the set of subjects associated with the images (i.e., any two eye pictures of the same subject were assigned to the same subset.) The first subset (1,750 images for UBIRIS, 1092 images for SBVPI) was used to train the domain–transfer network $G(M_s)$, while the remaining images in the considered data set were used to validate the segmenter, which was trained on images synthesized by EyeGAN using the synthetic masks.

Four different FCN segmenters were trained, where in all cases the labels were represented by synthetic masks M_s. Note that when experimenting with the UBIRIS data set, the ternary synthetic masks generated using UnityEyes were transformed into binary by conflating the sclera and background into one region. We first considered a baseline scenario, with the segmenter trained using the synthetic images I_s associated with the synthetic masks M_s, then tested on the real images. We then re-trained the segmenter using the same masks M_s as labels, but with domain–transferred images in input. These training images were generated starting from synthetic images ($G(I_s)$) using SimGAN and CycleGAN, and from synthetic masks ($G(M_s)$) for EyeGAN. All four segmenters were trained on 25,000 synthetic or domain–transferred images.

We used the metrics considered in [66] (Tab. 2 and Tab. 3) to evaluate the quality of segmentation. We also show the results when the segmenter is trained

directly on the real labels available in the training data ("train on target", or TT.) For both data sets, training the segmenter using target domain data produced by EyeGAN with inherited annotation from UnityEyes gave the best results. In fact, for the UBIRIS data set, the results using EyEGAN images for training are better than when the segmenter is trained on the real labels (TT). This can be justified by the fact that many more EyeGAN images (with inherited annotations) were available for training than real target images.

	Baseline	EyeGAN	CycleGAN	SimGAN	TT
IoU:Skin	0.95	**0.98**	0.93	0.96	0.97
IoU:Iris	0.77	**0.90**	0.68	0.80	0.87
mean IoU	0.86	**0.94**	0.81	0.88	0.92
f.w. IoU	0.92	**0.96**	0.89	0.93	0.96
pix. acc.	0.96	**0.98**	0.94	0.96	0.98
mean pix. acc.	0.90	**0.97**	0.88	0.93	0.94

Table 4.2: Comparison of segmentation into sclera and iris produced by the different algorithms considered for the UBIRIS data set, using standard metrics for multi-class segmentation [66], and specifically: IoU for each class; mean IoU; frequency weighted (f.w.) IoU; pixel accuracy; and mean pixel accuracy. The last column shows the "trained on target" (TT) results.

	Baseline	EyeGAN	CycleGAN	SimGAN	TT
IoU:Skin	0.86	**0.94**	0.84	0.83	0.96
IoU:Sclera	0.44	**0.78**	0.34	0.35	0.86
IoU:Iris	0.68	**0.84**	0.45	0.49	0.89
mean IoU	0.66	**0.85**	0.54	0.56	0.91
f.w. IoU	0.78	**0.91**	0.72	0.72	0.94
pix. acc.	0.87	**0.95**	0.82	0.82	0.97
mean pix. acc.	0.73	**0.92**	0.71	0.72	0.94

Table 4.3: Comparison of segmentation into skin, sclera, and iris produced by the different algorithms considered for the SBVPI data set. (See caption of Tab. 2.)

4.2.3 Pupil Localization

Another feature of interest in eye images is the location of the pupil center. High accuracy is needed for applications such as model–based gaze tracking [9]. We conducted an experiment similar to the one described in the previous section, where in this case the output of the network is a pair of numbers, representing the normalized coordinates of the estimated pupil center location. For this purpose, we used a DenseNet [37] architecture, with the last softmax layer replaced by a linear layer producing the coordinates vector. L2 loss was used for training. Specifically, we used the compact variant DenseNet-BC with the following configuration: L (number of layers) =100; k (growth rate of feature maps in each layer) =12; four dense blocks. The learning rate was set to 0.001 and the network parameters were optimized using Adam [50].

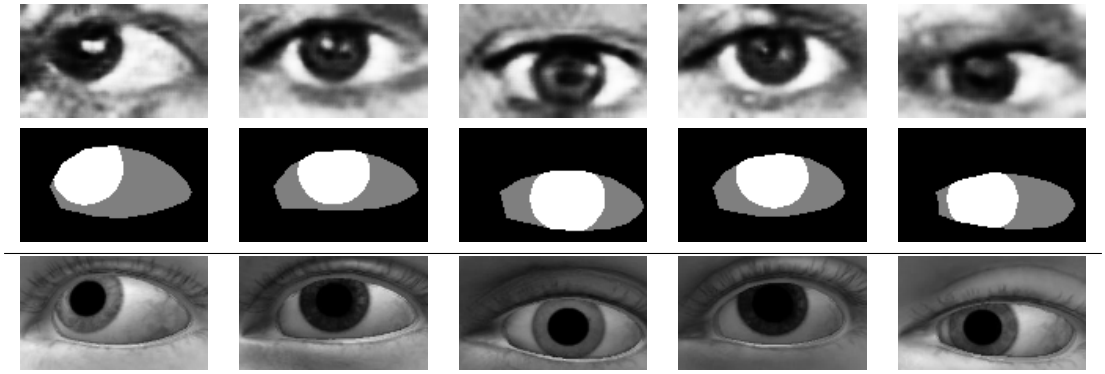


Figure 4.7: Examples of eye images generated by EyeGAN in the style of the BioID data set (top row), shown together with the UnityEye masks that were fed to the generator (middle row). For reference, we also show the synthetic images I_s corresponding to the masks M_s for the EyeGAN case in the last row.

As in the previous section, we trained a baseline regressor, using solely synthetic images generated by UnityEyes, as well as three regressors trained on domain–transferred images, inheriting annotations from UnityEyes. The target domain distribution was represented by the BioID data set [1], which contains

1521 grayscale images of 23 subjects taken at different head orientations. The images were resized to 72×120 pixels and histogram equalized. The location of the pupil center was available for each image; this information was only used in the final evaluation. Samples of the images produced by EyeGAN in the style of the BioID data set, generated starting from UnityEyes masks, are shown in Fig. 4.7. Fig. 4.8 shows the cumulative distribution function (CDF) of the Euclidean norm

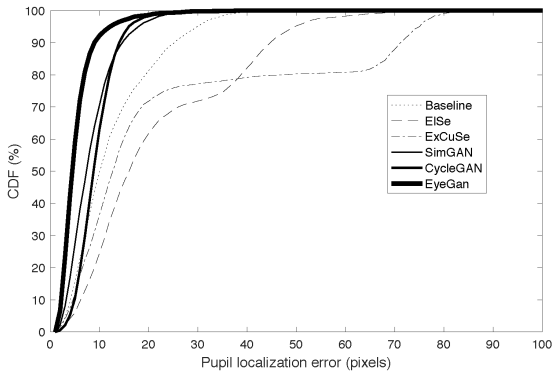


Figure 4.8: The cumulative distribution functions (CDFs) of the Euclidean norm of pupil localization error for the different algorithms considered (Sec. 4.2.3.)

of the localization error for the different methods considered (where the CDF for a certain error value e represents the portion of images with error smaller than e.) For comparison, we also showed results using two well–known existing algorithms for pupil localization: ExCuSe [25] and ElSe [26], both based on fast elliptical fitting of the pupil region. Note that training with EyeGAN gave the best results.

4.2.4 Gaze Estimation

Appearance-based gaze estimation algorithms compute the direction of gaze directly from images of the user taken by a camera, without resorting to geometrical models of gaze formation. Training a network for appearance-based gaze estimation requires availability of data with precise annotation of gaze direction for each image. This can only be obtained indirectly, i.e. by asking the user to

look at a certain point on the screen, and then inferring gaze direction from the known location of the user's head location. The ability to generate realistic images with inherited annotation is highly desirable, as it would enable construction of larger and more diverse training data sets.

We used NVGaze [48], a data set that contains both real eye images captured under IR illumination from a wearable headset device, as well as synthetic eye images from a similar viewpoint. The real eye images are annotated with gaze direction; the synthetic images have both gaze and segmentation mask annotation. The goal of this experiment was to learn a mapping from real images to gaze direction, but without making use of any available ground-truth labels for these images during training. We decided to use an intermediate representation, formed by a set of 25 feature points extracted from the segmentation masks (12 points uniformly distributed around the edge of the periocular region, 12 points around the iris, and one point in the center of the iris.) We trained a fully connected neural network (with two hidden layers of size 500 each) to learn a mapping from feature points extracted from the synthetic masks to gaze direction. We then trained another network to predict the location of feature points from images generated with EyeGAN. During training, each EyeGAN image was associated with the feature points extracted from the synthetic mask used to generate the same image. This network used the same DenseNet [37] architecture described in Sec. 4.2.3, this time with a 50-dimensional (25×2) output (using L2 loss.) We then tested our system on real eye images using a cascade of the two networks just described: for each image, we first predicted the associated feature points, and then, from these feature points, the gaze direction. The synthetic image portion of NVGaze contains two million images, while real eye images are collected for 35 subjects take at a high frame rate. For synthetic data, we sampled 50000 images. For real

world data, we randomly selected one subject with 76000 images containing 50 gaze directions. We sampled 128 images, ensuring that all gaze directions were covered. When testing the gaze detector on the real eye images, we reserved 22 such images for subject calibration [48]. This procedure, akin in spirit to subject calibration for standard IR gaze tracker, is designed to remove individual bias (as due, for example, to the kappa angle between the visual and the pupillary axes [57]. Specifically, we computed a quadratic regression from the predicted gaze directions to the ground-truth gaze directions over these 22 images; we then applied the same quadratic function on the predicted gaze for the remaining images, before computing the angular error with respect to the ground-truth gaze direction. The results are shown in Tab. 4, where “Baseline" represents the case in which the predictor for the feature points was trained entirely on synthetic data. This experiment once more shows that training the network (in this case, the feature points predictor) on EyeGAN-generated images with inherited annotations results in the best performance.

Baseline	EyeGAN	CycleGAN	SimGAN
23°	**5.3**°	20°	16°

Table 4.4: Mean gaze angular errors for the experiment described in Sec. 4.2.4.

4.3 Conclusion

We have introduced a new algorithm, EyeGAN, for the generation of eye images with a prescribed gaze direction in the style of a desired target domain. Like similar techniques, EyeGAN operates within the framework of domain transfer: starting from synthetically generated data, it produces an image that can be considered as a sample from the target domain distribution. The key differ-

ence between EyeGAN and other competing algorithms is in the way it enforces consistency of gaze direction. Our experiments have shown that ternary masks, which are easy to generate, contain enough information to “guide" the generation process into producing realistic images with the desired gaze direction. Comparative tests with different tasks and different target domains have shown that the images produced by EyeGAN lead to better results when used as training data with inherited annotations.

Chapter 5

Subject Guided Eye Image Synthesis for Gaze Redirection

> The Eyes are the window to your soul
>
> *William Shakespeare*

In this chapter we address the specific problem of *gaze redirection*: given an image of a person's face, we want to generate a new image that is identical to the first one, except for this person's gaze, which should be consistent with a certain direction. We cast the task of gaze redirection as one of image synthesis with a pre-determined style. By *style* we mean, for example, the appearance of a certain person's face under a certain illumination. In this context, the *content* to be manipulated is gaze direction. Following EyeGAN, we use ternary *segmentation masks* to characterize gaze direction. Masks act as style-independent proxies for gaze. In previous chapter, the EyeGAN algorithm was introduced that takes a synthetic mask for a prescribed gaze direction as input, and generates an image under content (gaze direction) consistent with the input mask and some random style. In this chapter, we push this idea forth, and introduce a new cyclic mecha-

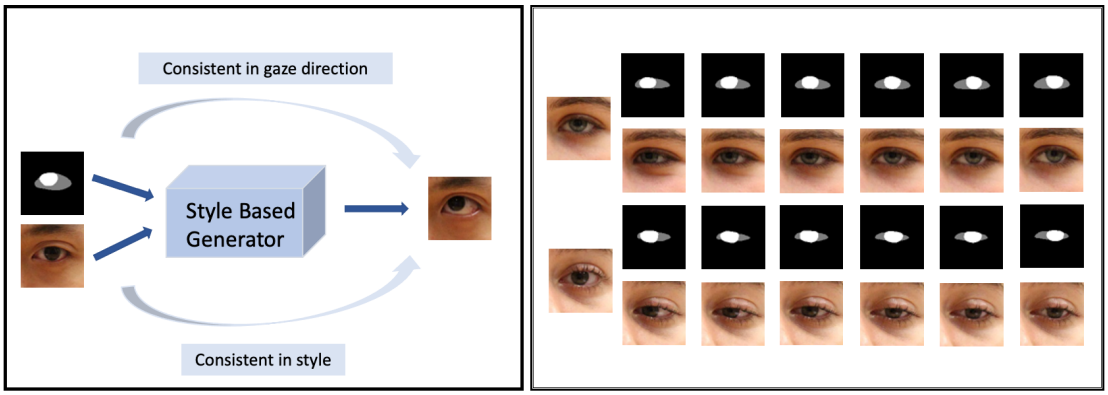


Figure 5.1: Overview of our Style-Based Eye Image Generation. The generator receives in input a segmentation mask and a style image. It synthesizes an image which is consistent in gaze with the segmentation mask, with generated features similar to the style image.

nism to ensure consistency of both style and content of the generated image. In addition, we introduce an algorithm that redirects one's gaze without relying on model-based synthetic mask generation. A ternary mask is extracted from the input image, and redirected (using a trained network) to the desired gaze direction. This new mask is then used to control style-preserving gaze redirection. Remarkably, our algorithms *do not* require gaze annotated real-world images for training.

5.1 Method

5.1.1 Style–Based Eye Image Synthesis

The goal of this module is to generate a realistic image with a certain style and a prescribed gaze direction. Style is guided by means of an eye image from a domain $\mathcal{E}$. This image is given in input to the network, along with a ternary segmentation mask (from the mask domain $\mathcal{M}$), which characterizes the prescribed gaze direction. Thus, our system is a mapping G from $\mathcal{M} \times \mathcal{E}$ to $\mathcal{E}$. Fig 5.1 shows

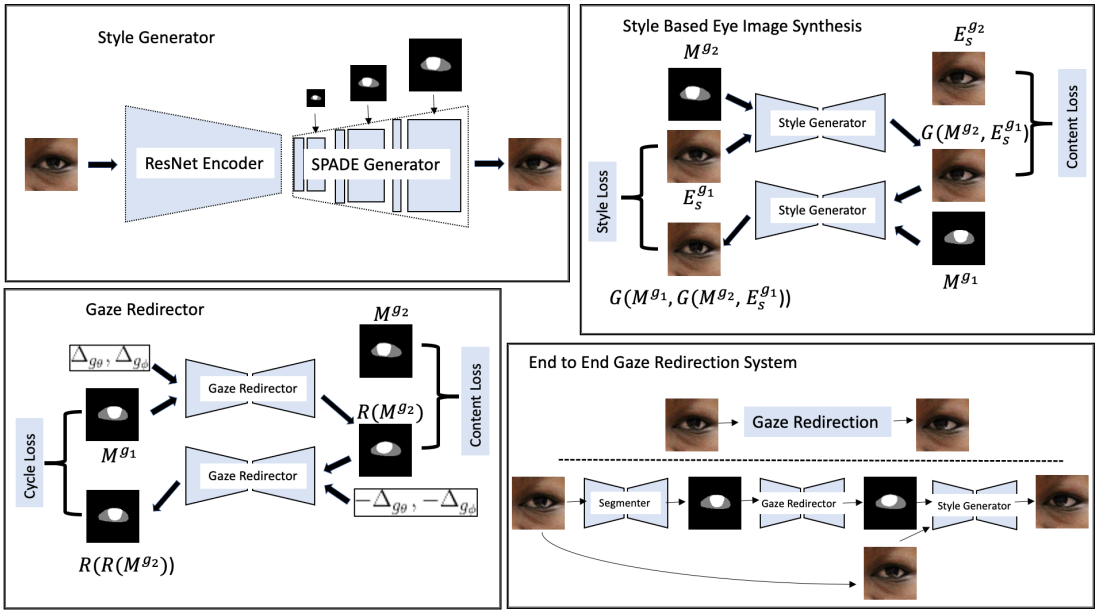


Figure 5.2: Top Left: SPADE [80] based Generator architecture. The ResNet Encoder encodes the style image. The style encoding is input to the SPADE generator which also receives segmentation mask input at different scales. Top Right: Our Style-Based Eye Image Synthesis flow. The generator synthesizes the eye image from the mask and style image. During training, the generated image is fed back to the generator along with mask corresponding to the style image. Bottom Left: Gaze Redirection model trained on the segmentation masks. Bottom Right: End to End Gaze redirection system involving segmentation mask generator, gaze redirector and style generator.

an overview of our style–based eye image synthesis.

Let $E_s^{g_1} \in \mathcal{E}$ be the input eye image, where the subscript $_s$ indicates the style, and the superscript ^{g_1} indicates the gaze direction in the image. Note that "style" is not a directly measurable quantity – it expresses the appearance of the eye image, in terms, for example, of iris color, skin color, skin texture, illumination. Gaze is quantifiable, but we don't need to know, nor make use of, the gaze direction g_1. The goal of the generator G is to synthesize an eye image $E_s^{g_2*} \in \mathcal{E}$ with same style s as the input, but with gaze direction g_2 (the superscript * indicates that this is a synthesized image.) The algorithm uses a synthetically generated ternary

mask, M^{g_2} as a proxy for the prescribed direction g_2.

The network is trained using samples consisting of four images each: $\{E_s^{g_1}$, $M(E_s^{g_1})$, $E_s^{g_2}$, $M(E_s^{g_2})\}$. Here, $E_s^{g_1}$ and $E_s^{g_2}$ are images of the same individual with different gaze directions. $M(\cdot)$ is a function that extracts a ternary mask from an eye image [44].

Synthesis Loss. The *synthesis loss* term ensures that the generated image, $E_s^{g_2*}$, is similar to the desired one. For this purpose, we use a perceptual loss function [41], [34]:

$$\mathcal{L}_{syn}(G) = \sum_j w_j * \frac{1}{N_j} ||f_j(E_s^{g_2*}) - f_j(E_s^{g_2})||_2^2 \tag{5.1}$$

$$= \sum_j w_j * \frac{1}{N_j} ||f_j(G(M^{g_2}, E_s^{g_1})) - f_j(E_s^{g_2})||_2^2 \tag{5.2}$$

Here, $f_j(\cdot)$ is the feature map of size $N_j = C_j \times H_j \times W_j$, extracted from j^{th} convolutional layer in the VGG-16 network, pre-trained on the ImageNet data set.

Re-synthesis Loss. As an additional device to ensure style consistency between the input $E_s^{g_1}$ and the synthesized image $E_s^{g_2*}$, the latter is taken as input to the generator during training along with the segmentation mask M^{g_1} from the input image, to generate a new image $E_s^{g_1*} = G(M^{g_1}, E_s^{g_2*})$. A second loss component is added as follows:

$$\mathcal{L}_{resyn}(G) = ||E_s^{g_1*} - E_s^{g_1}||_1 \tag{5.3}$$

$$= ||G(M^{g_1}, G(M^{g_2}, E_s^{g_1})) - E_s^{g_1}||_1 \tag{5.4}$$

This idea borrows from the CycleGAN scheme [59], with the difference that CycleGAN maps an image from a domain to a different domain and back, while we map an image to the same domain (and back), but with an additional "content guidance" image (the ternary segmentation representing gaze direction.)

Adversarial Loss. In order to improve the quality of image generation, we also consider an adversarial loss [31]. A discriminator network $D(\cdot)$ is shown an image, either synthesized ($E_s^{g_2*}$) or real ($E_s^{g_2}$), and is tasked with determining whether the input image is real or synthesized. This loss term penalizes the discriminator when the determination is incorrect, and the generator when it is correct:

$$\mathcal{L}_{adv}(G, D) = \log(D(E_s^{g_2})) + \log(1 - D(E_s^{g_2*})) \tag{5.5}$$

$$= \log(D(E_s^{g_2})) + \log(1 - D(G(M^{g_2}, E_s^{g_1}))) \tag{5.6}$$

Along with the adversarial loss, we consider a discriminator-based feature matching loss [107]. Features are extracted from intermediate layers of the discriminator network for both a real ($E_s^{g_2}$) and synthesized ($E_s^{g_2*}$) image. This loss penalizes discrepancy between the features for the two images.

$$\mathcal{L}_{feat}(G) = \sum_j \frac{1}{N_j} ||D_j(E_s^{g_2*}) - D_j(E_s^{g_2})||_2^2 \tag{5.7}$$

$$= \sum_j \frac{1}{N_j} ||D_j(G(M^{g_2}, E_s^{g_1}) - D_j(E_s^{g_2})||_2^2 \tag{5.8}$$

Here D_j represents the feature map extracted from the j^{th} layer of the discriminator network. The size of the feature map is given by $N_j = C_j \times H_j \times W_j$.

Overall Objective The overall objective is given by:

$$G^*, D^* = \min_G \max_D (\mathcal{L}_{adv}(G, D) + \lambda_1 \mathcal{L}_{feat}(G) + \lambda_2 \mathcal{L}_{syn}(G) + \lambda_3 \mathcal{L}_{resyn}(G)) \tag{5.9}$$

5.1.2 Gaze Redirection via Mask Synthesis

In the algorithm described above, gaze redirection is guided by a ternary mask M^{g_2}, which describes the desired gaze direction. This mask would normally be obtained using a computer graphics tool, such as UnityEyes [110]. However, this mask may not be perfectly suited to the considered "style". Due to the variability of facial features, a synthetic mask can be only an approximation of the actual segmentation obtained from a real image. For this reason, a synthetic mask may only be a sub-optimal solution for guiding gaze redirection.

We address this problem with an algorithm that builds on our Style–Based Eye Image Synthesis approach, but that synthesizes the content-guiding ternary mask from the segmentation of the input image. The hope is that this mask may represent a better proxy for our gaze redirection algorithm. Mask synthesis is the job of a *mask redirection* network, which takes in input a ternary segmentation of the input image $M(E_s^{g_1})$, along with the the prescribed variation of gaze angle $(\Delta_\phi, \Delta_\theta) = (g_2^\phi, g_2^\theta) - (g_1^\phi, g_1^\theta)$ [34] [119], to produce a redirected mask M^{g_2*}.

The mask redirection network R is trained with pairs of segmentation masks $(M(E_s^{g_1}), M(E_s^{g_2}))$ from images with known gaze directions (g_1, g_2). A loss function is defined as the sum of two terms: a mask-synthesis loss and a mask-resynthesis loss.

Mask-Synthesis Loss. This is a forward content loss between the gaze redirected

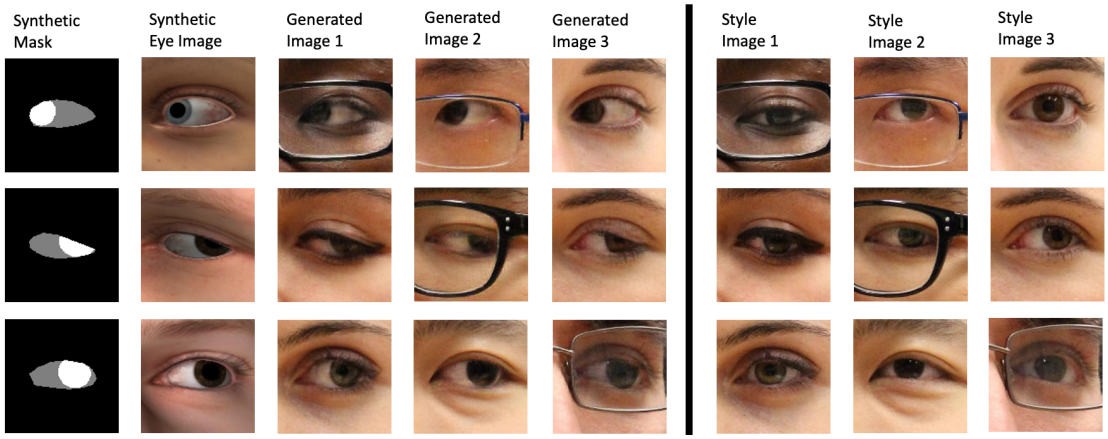


Figure 5.3: Eye Image Synthesis from Unity Masks. First and second column: masks from UnityEyes and corresponding synthetic eye images. Columns 3-5: the generated images from our Style-Based generator for the mask in Column 1. The input style images are shown in last three columns.

mask and the target mask.

$$\mathcal{L}_{m-syn}(R) = \text{CE}[R(M(E_s^{g_1}), (\Delta_\phi, \Delta_\theta)) - M(E_s^{g_2})]$$

Mask-Resynthesis Loss. The redirected mask is fed back to the network with negative gaze direction variation, with the goal to reconstruct the input mask:

$$\mathcal{L}_{m-resyn}(R) = \text{CE}[R(R(M(E_s^{g_1}), (\Delta_\phi, \Delta_\theta)), (-\Delta_\phi, -\Delta_\theta)) - M(E_s^{g_1})]$$

In these equations, CE[·] represents the cross-entropy function.

5.2 Experiments

5.2.1 Style-Based Eye Image Synthesis

Data Set. We trained our Style–Based Eye Image Synthesis network on the Columbia Gaze data set [92]. This data set contains facial images taken of 56 subjects under constrained settings. For each subject, gaze data was collected at 5 horizontal head poses $[0°, \pm 15°, \pm 30°]$. For each head pose, 21 gaze directions (7 horizontal: $\phi \in [0°, \pm 5°, \pm 10°, \pm 15°]$ and 3 vertical $\theta \in [0°, \pm 10°]$) were recorded. We cropped the left and right eye patches from the facial images and resized them to 64×64 as described in [34]. The right eye images were flipped to look like left eye images. Each style category consists of images for one subject with different gaze directions with one head pose and one side (left/ right). In practice, a total of $56 \cdot 5 \cdot 2 = 560$ styles were considered (56 subjects with 5 different head poses and 2 sides). Each style has 21 images (one per gaze direction). We used the first 50 subjects for training and the remaining 6 subjects for testing.

We used EyeGAN [44] to extract segmentation masks for all of the eye images. This method trains a fully convolutional neural net [66] to extract segmentation masks without manual annotations. It uses a set of real eye images along with synthetic eye masks (from the UnityEyes [110] tool.) It alternates training of a segmenter network to extract masks from real eye images, with training of a generator network that synthesizes natural looking eye images from the synthetic masks. Since the mask generated by EyeGAN have a smaller size 48×32, we zero-padded them to fill a 64×64 area.

Implementation Details Our eye image synthesis generator is implemented as an encoder-decoder network. In particular, we use a ResNet [33] encoder with

three residual blocks, and a SPADE [80] decoder for generating images guided by segmentation masks (see Fig. 5.2). In SPADE, the segmentation mask is fed at each block at different scales. Our SPADE decoder consists of four SPADEResNet blocks. Training is performed using the same multi-scale discriminator as in pix2pixHD [107] and SPADE [80] with hinge loss [61], [80]. We used Adam [50] optimizer with $\beta_1 = 0$ and $\beta_2 = 0.9$. Learning rate was set to 0.005 with λ_1, λ_2 and λ_3 as 10, 20 and 20 respectively.

Results We first compare our Style-Based Eye Image Synthesis with three other eye image synthesis algorithms: SimGAN [91], CycleGAN [126], [59] and EyeGAN [44]. These algorithms synthesize an image with the style of the data set on which they are trained (in this case, the Columbia Gaze data set.) In the case of of SimGAN and CycleGAN, the input to the algorithm is a synthetic eye image from UnityEye. EyeGAN takes in input a segmentation mask, also from UnityEye.

Two metrics were considered for comparison: (1) Fréchet Inception Distance (FID) [35]; and (2) mean IoU (mIoU). FID is a metric used to measure similarity between two data sets (in our case, the output of the algorithms and the Columbia Gaze data set.) It captures both the perceptual similarity between generated and real images, and the diversity of generated images (similar data sets have low FID values). The mean IOU is computed between the segmentation of the output, and the mask corresponding to the input image or the mask itself in our case. This segmentation is obtained by a FCN [65] trained on masks corresponding to UnityEyes images and the corresponding eye images generated using EyeGAN since it has been shown to preserve semantic consistency of the generated images. A larger value of mIoU indicates good semantic consistency between the source and the generated image.

Table 5.1 presents quantitative results in terms of the FID and mIoU metrics.

Table 5.1: Comparison of eye image synthesis algorithms using FID score (lower the better) and mIOU (higher the better).

Algorithm	FID	mIoU
EyeGAN [44]	83.9	**0.93**
CycleGAN [126], [59]	39.5	0.61
SimGAN [91]	53.7	0.66
Ours	**8.5**	0.72

Table 5.2: Comparison of Style-Based eye image synthesis using LPIPS metric (lower the better).

(a) LPIPS score on the test data with **supervised** training.

Algorithm	LPIPS
Pix2PixSC [106]	0.104
Seg2Eye [8]	0.077
Ours w/o reconst	0.035
Ours	**0.033**

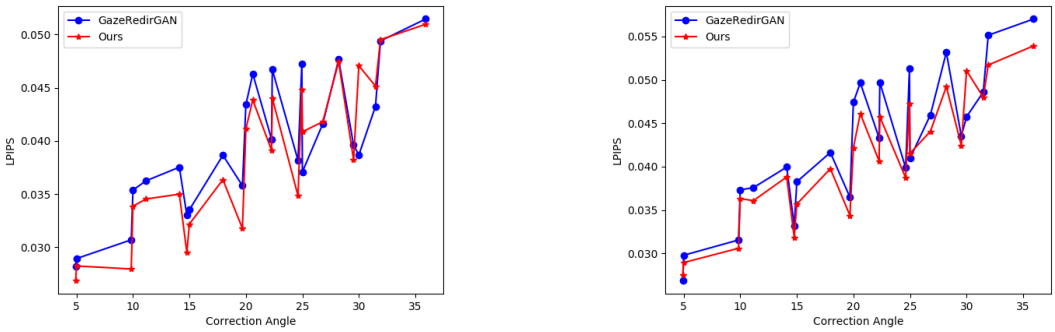
(b) LPIPS score on the test data with **unsupervised** training.

Algorithm	LPIPS
Pix2PixSC [106]	0.125
Seg2Eye [8]	0.124
Ours w/o reconst	0.078
Ours	**0.044**

Our technique achieves the smallest value of FID, and the second highest value of mIOU among the algorithms considered. Samples of eye images generated by our method, along with the “style images" and synthetic masks used in input, are shown in Figure 5.3.

We also compared our algorithm with two other techniques for style-based synthesis: Seg2Eye [8] and Pix2PixSC [106] on Columbia eye data set [92]. Seg2Eye uses the SPADE [80] architecture with adaptive-instance normalization layers for style transfer. We trained Seg2Eye with a single style image per data point. Pix2PixSC uses style consistency adversarial loss with Pix2PixHD [107] as a base architecture.

We used the segmentation mask corresponding to test subject images and generated the style image corresponding to the masks. The generated images are compared with the ground truth images using LPIPS [120] metric. We trained our network with baseline under two kinds of settings, supervised and unsupervised.



(a) Full Dataset. **(b)** Reduced Dataset.

Figure 5.4: LPIPS vs Correction Angle: Quantitative comparison of gaze redirection results for frontal head pose.

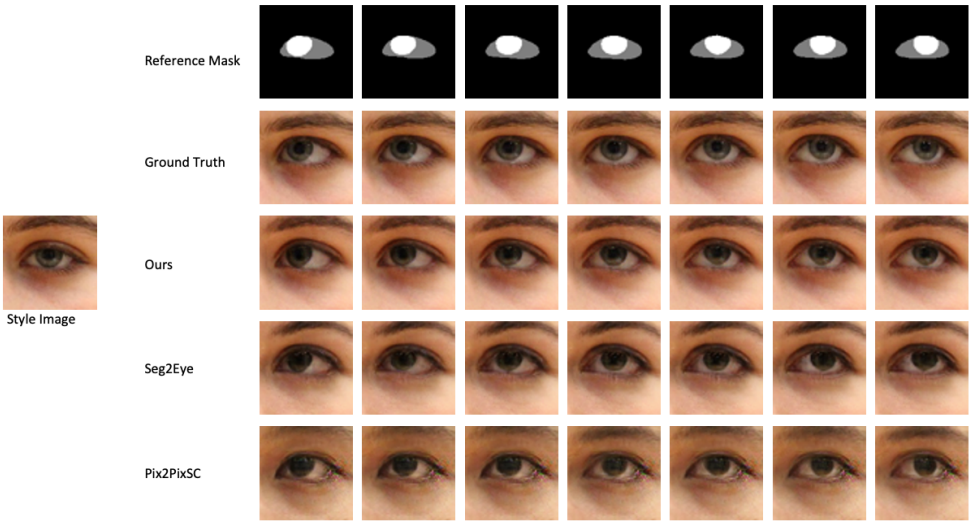


Figure 5.5: Qualitative comparison under supervised setting. We show the results corresponding to different segmentation masks for a test style image. The ground truth images are shown along with the synthesized images for baseline methods.

In supervised setting, the ground truth image corresponding to the input mask has the same style as the input style image. In unsupervised setting the ground truth image has the different style as the input style image. We observed that, with supervised training, SPADE generator architecture with ResNet encoder in itself is good enough to generate Style-Based images, without the need for our

Figure 5.6: Qualitative comparison under unsupervised setting. We show the results corresponding to different segmentation masks for a test style image. The ground truth images are shown along with the synthesized images for baseline methods.

re-synthesis loss. However, our re-synthesis loss proved very useful in case of unsupervised training. We show the LPIPS metric results for the two baselines and our method with and without reconstruction in Table 5.2, for both supervised and unsupervised training. We also show the qualitative results in Figure 5.5 for supervised training and Figure 5.6 for unsupervised training.

5.2.2 Gaze Redirection via Mask Synthesis

Data Set The mask redirection network was trained with synthetic masks corresponding to the eye images from 10 different subjects (different eye parameters) generated for 21 gaze directions in frontal head pose using UnityEyes [110] tool. For each gaze direction, we trained the network to redirect the mask to 20 other gaze directions.

Implementation Details The mask redirection network receives in input a segmentation mask and a gaze direction variation vector. The input segmentation mask is passed through a network with three convolutional layers, followed by five residual blocks and then by three upsampling + convolutional layers. The gaze direction variation vector is input to a Multi-Layer Perceptron, then concatenated with output of the convolutional layers for the input mask, before the residual blocks. The network is trained to minimize per-pixel cross-entropy loss between the generated mask and the ground truth mask from UnitiEyes. We used Adam [50] optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate was set to 0.01.

The output of the mask redirection network is used as input mask in the Style-Based Eye Image Synthesis (Sec. 5.1.1).

Results We compared our algorithm for gaze redirection based on mask synthesis against the method described in [34] (which we dubbed "GazeRedirGAN",) which is shown to give state of the art results. The LPIPS [120] metric was used for comparing generated and ground truth images. The mean LPIPS score is calculated with respect to the correction angle [34], which is the angular difference between target gaze direction and source gaze direction. As shown in Figure 5.4a, our method performs slightly better than GazeRedirGAN, even though we only used the gaze labels corresponding to synthetic masks for redirection.

In another experiment, we removed eye images corresponding to horizontal angles $[\pm 10^\circ, \pm 15^\circ]$. We trained both our Style-Based Eye Image Synthesis algorithm and GazeRedirGAN on this reduced (Columbia) data set, and tested on the gaze directions unseen in the training set. The synthetics masks corresponding to the removed gaze directions were used to train the gaze redirection network. As shown in Figure 5.4b our gaze redirection produces lower LPIPS values. We also show qualitative comparison in Figure 5.7.

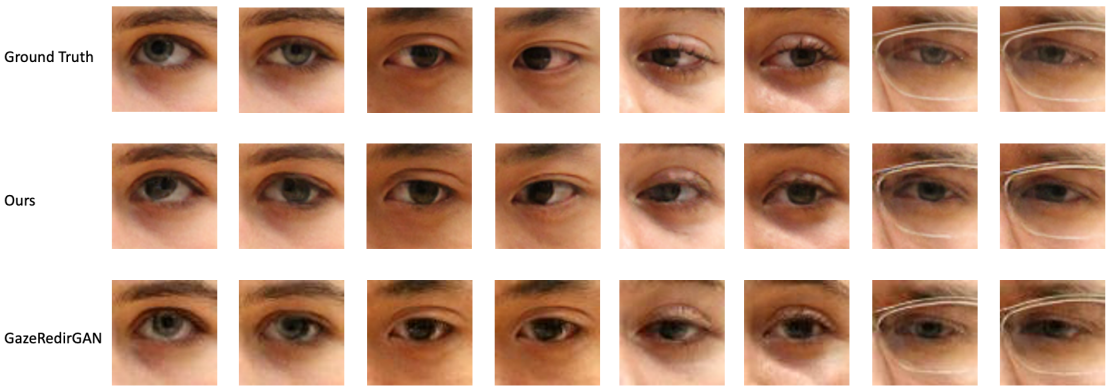


Figure 5.7: Qualitative Comparison of gaze redirection results with models trained on reduced data set.

5.3 Conclusion

We proposed a new method for generating realistic eye images with a prescribed gaze direction. This algorithm takes in input a “style image" as well as a ternary segmentation mask, representing the desired gaze direction. A cyclic training algorithm ensures that the generated image has the desired gaze direction, and that it is in the style of the input image.

We also show how we can use this style synthesis for gaze redirection. Importantly, this algorithm does not require annotation of gaze angle in the training data. Instead, it uses ternary segmentation of the training images, which is much easier to obtain. The gaze labels are required corresponding to the ternary mask which can be generated synthetically.

5.4 Acknowledgment

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 rep-

resent the official views of the National Institutes of Health.

Bibliography

[1] The BioID database. www.bioid.com/facedb/.

[2] Diagram of the Eye. `https://nei.nih.gov/health/eyediagram`.

[3] Fares Alnajar, T. Gevers, Roberto Valenti, and Sennay Ghebreab. Calibration-free gaze estimation using human gaze patterns. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 137–144, 12 2013.

[4] Pablo Artal. Optics of the eye and its impact in vision: a tutorial. *Adv. Opt. Photon.*, 6(3):340–367, Sep 2014.

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

[6] Inessa Bekerman, Paul Gottlieb, and Michael Vaiman. Variations in eyeball diameters of the healthy adults. *Journal of ophthalmology*, 2014:503645, 11 2014.

[7] Kevin W Bowyer, Karen Hollingsworth, and Patrick J Flynn. Image understanding for iris biometrics: a survey. *Computer Vision and Image Understanding*, 110(2):281–307, 2008.

[8] Marcel C. Buehler, Seonwook Park, Shalini De Mello, Xucong Zhang, and Otmar Hilliges. Content-consistent generation of realistic eyes with style. In *International Conference on Computer Vision Workshops (ICCVW)*, 2019.

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

[10] Jixu Chen and Qiang Ji. Probabilistic gaze estimation without active personal calibration. In *CVPR 2011*, pages 609–616, 2011.

[11] Zhaokang Chen and Bertram Shi. Offset calibration for appearance-based gaze estimation via gaze decomposition. In *Proceedings of the IEEE/CVF*

Winter Conference on Applications of Computer Vision (WACV), March 2020.

[12] Zhaokang Chen and Bertram E. Shi. Appearance-based gaze estimation using dilated-convolutions. In *ACCV*, 2018.

[13] Yihua Cheng, Shiyao Huang, Fei Wang, Chen Qian, and Feng Lu. A coarse-to-fine adaptive network for appearance-based gaze estimation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:10623–10630, 04 2020.

[14] Yihua Cheng, Feng Lu, and Xucong Zhang. Appearance-based gaze estimation via evaluation-guided asymmetric regression. In *ECCV*, 2018.

[15] Yihua Cheng, Haofei Wang, Yiwei Bao, and Feng Lu. Appearance-based gaze estimation with deep learning: A review and benchmark. *ArXiv*, abs/2104.12668, 2021.

[16] Yihua Cheng, Xucong Zhang, Feng Lu, and Yoichi Sato. Gaze estimation by exploring two-eye asymmetry. *IEEE Transactions on Image Processing*, 29:5259–5272, 2020.

[17] Criminisi, Shotton, Blake, and Torr. Gaze manipulation for one-to-one teleconferencing. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 191–198 vol.1, 2003.

[18] Antonio Criminisi, Jamie Shotton, Andrew Blake, and Philip HS Torr. Gaze manipulation for one-to-one teleconferencing. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 3, pages 13–16, 2003.

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

[20] Murthy L R D and Pradipta Biswas. Appearance-based gaze estimation using attention and difference mechanism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3143–3152, June 2021.

[21] Haoping Deng and Wangjiang Zhu. Monocular free-head 3d gaze tracking with deep learning and geometry constraints. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3162–3171, 2017.

[22] Chelsea Finn, P. Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

[23] Tobias Fischer, Hyung Jin Chang, and Yiannis Demiris. Rt-gene: Real-time eye gaze estimation in natural environments. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[24] Wolfgang Förstner and Bernhard P Wrobel. *Photogrammetric computer vision*. Springer, 2016.

[25] Wolfgang Fuhl, Thomas Kübler, Katrin Sippel, Wolfgang Rosenstiel, and Enkelejda Kasneci. ExCuSe: Robust pupil detection in real-world scenarios. In *International Conference on Computer Analysis of Images and Patterns*, pages 39–51. Springer, 2015.

[26] Wolfgang Fuhl, Thiago C Santini, Thomas Kübler, and Enkelejda 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*, pages 123–130. ACM, 2016.

[27] Kenneth Alberto Funes Mora, Florent Monay, and Jean-Marc 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*, pages 255–258. ACM, 2014.

[28] Yaroslav Ganin, Daniil Kononenko, Diana Sungatullina, and Victor Lempitsky. Deepwarp: Photorealistic image resynthesis for gaze manipulation. In *European Conference on Computer Vision*, pages 311–326. Springer, 2016.

[29] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423, 2016.

[30] D. Giger, J. Bazin, C. Kuster, T. Popa, and M. Gross. Gaze correction with a single webcam. In *2014 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2014.

[31] I. Goodfellow, J. Pouget-Abadie, M. Mirza, X. Bing, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. 2014.

[32] Elias Guestrin and Moshe Eizenman. General theory of remote gaze estimation using the pupil center and corneal reflections. *Biomedical Engineering, IEEE Transactions on*, 53:1124 – 1133, 07 2006.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[34] Z. He, A. Spurr, X. Zhang, and O. Hilliges. Photo-realistic monocular gaze redirection using generative adversarial networks. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6931–6940, 2019.

[35] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems 30*, pages 6626–6637. 2017.

[36] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018.

[37] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

[38] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1510–1519, 2017.

[39] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[40] Qiang Ji and Kang Wang. Bayesian eye tracking. *CoRR*, abs/2106.13387, 2021.

[41] J. Johnson, A. Alahi, and F. Li. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision – ECCV 2016*, pages 694–711, 2016.

[42] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[43] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019.

[44] H. Kaur and R. Manduchi. Eyegan: Gaze-preserving, mask-mediated eye image synthesis. In *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[45] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik, and Antonio Torralba. Gaze360: Physically unconstrained gaze estimation in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[46] Khan and Lee. Gaze and eye tracking: Techniques and applications in adas. *Sensors*, 19:5540, 12 2019.

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

[48] Joohwan Kim, Michael Stengel, Alexander Majercik, Shalini De Mello, David Dunn, Samuli Laine, Morgan McGuire, and David Luebke. Nvgaze: An anatomically-informed dataset for low-latency, near-eye gaze estimation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, page 550. ACM, 2019.

[49] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[50] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[51] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[52] Daniil Kononenko and Victor Lempitsky. Learning to look up: Realtime monocular gaze correction using machine learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4667–4675, 2015.

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

[54] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges,

L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[55] Claudia Kuster, Tiberiu Popa, Jean-Charles Bazin, Craig Gotsman, and Markus Gross. Gaze correction for home video conferencing. *ACM Trans. Graph. (Proc. of ACM SIGGRAPH ASIA)*, 2012.

[56] Michael F. Land. The human eye: Structure and function. *Nature Medicine*, 5(11):1229–1229, 1999.

[57] Eui Chul Lee and Kang Ryoung Park. A robust eye gaze tracking method based on a virtual eyeball model. *Machine Vision and Applications*, 20(5):319–337, 2009.

[58] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 35–51, 2018.

[59] Kangwook Lee, Hoon Kim, and Changho Suh. Simulated+unsupervised learning with adaptive data generation and bidirectional mappings. In *International Conference on Learning Representations*, 2018.

[60] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81, 02 2009.

[61] Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017.

[62] Erik Linden, Jonas Sjostrand, and Alexandre Proutiere. Learning to personalize in appearance-based gaze tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[63] Gang Liu, Yuechen Yu, Kenneth Alberto Funes Mora, and Jean-Marc Odobez. A differential approach for gaze estimation with calibration. In *BMVC*, 2018.

[64] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in neural information processing systems*, pages 700–708, 2017.

[65] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[66] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[67] Feng Lu, Yusuke Sugano, Takahiro Okabe, and Yoichi Sato. Head pose-free appearance-based gaze sensing via eye image synthesis. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 1008–1011. IEEE, 2012.

[68] Feng Lu, Yusuke Sugano, Takahiro Okabe, and Yoichi Sato. Adaptive linear regression for appearance-based gaze estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(10):2033–2046, 2014.

[69] Feng Lu, Yusuke Sugano, Takahiro Okabe, and Yoichi Sato. Gaze estimation from eye appearance: A head pose-free method via eye image synthesis. *IEEE Transactions on Image Processing*, 24(11):3680–3693, 2015.

[70] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks, 2017.

[71] Francis Martinez, Andrea Carbone, and Edwige Pissaloux. Gaze estimation using local features and non-linear regression. In *2012 19th IEEE International Conference on Image Processing*, pages 1961–1964, 2012.

[72] Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.

[73] Yeji Moon, Won June Lee, Seung Hak Shin, Ji Hong Kim, Ji Young Lee, Sei Yeul Oh, and Han Woong Lim. Positional change of the eyeball during eye movements: Evidence of translatory movement. *Frontiers in Neurology*, 11:1081, 2020.

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

[75] Ishan Nigam, Mayank Vatsa, and Richa Singh. Ocular biometrics: A survey of modalities and fusion approaches. *Information Fusion*, 26:1–35, 2015.

[76] Seonwook Park, Emre Aksan, Xucong Zhang, and Otmar Hilliges. Towards end-to-end video-based eye-tracking. In *European Conference on Computer Vision (ECCV)*, 2020.

[77] Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges, and Jan Kautz. Few-shot adaptive gaze estimation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9367–9376, 2019.

[78] Seonwook Park, Adrian Spurr, and Otmar Hilliges. Deep pictorial gaze estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[79] Seonwook Park, Xucong Zhang, Andreas Bulling, and Otmar Hilliges. 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*, ETRA '18, New York, NY, USA, 2018. Association for Computing Machinery.

[80] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[81] Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Trans. Graph.*, 35(6), November 2016.

[82] Rik Pieters. A review of eye-tracking research in marketing. *Review of Marketing Research*, 4:123–147, 01 2008.

[83] Hugo Proenca, Silvio Filipe, Ricardo Santos, Joao Oliveira, and Luis A Alexandre. The UBIRIS.v2: A database of visible wavelength iris images captured on-the-move and at-a-distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1529–1535, 2009.

[84] Paola Ricciardelli and Jon Driver. Effects of head orientation on gaze perception: How positive congruency effects can be reversed. *The Quarterly Journal of Experimental Psychology*, 61(3):491–504, 2008.

[85] Alexander Richard, Colin Lea, Shugao Ma, Jurgen Gall, Fernando de la Torre, and Yaser Sheikh. Audio- and gaze-driven facial animation of codec avatars. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 41–50, January 2021.

[86] Rui Rodrigues, João P. Barreto, and Urbano Nunes. Camera pose estimation using images of planar mirror reflections. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 382–395, 2010.

[87] P. Rot, M. Vitek, K. Grm, Ž. Emeršič, P. Peer, and V. Štruc. *Handbook of Vascular Biometrics*, chapter Deep Sclera Segmentation and Recognition. Springer, 2019.

[88] Peter Rot, Žiga Emeršič, Vitomir Struc, and Peter Peer. Deep multi-class eye segmentation for ocular biometrics. In *2018 IEEE International Work Conference on Bioinspired Intelligence (IWOBI)*, pages 1–8. IEEE, 2018.

[89] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *ArXiv*, abs/1706.05098, 2017.

[90] Laura Sesma, Arantxa Villanueva, and Rafael 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*, pages 217–220, 2012.

[91] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2107–2116, 2017.

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

[93] Yusuke Sugano and Andreas Bulling. Self-calibrating head-mounted eye trackers using egocentric visual saliency. *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, 2015.

[94] Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. Calibration-free gaze sensing using saliency maps. In *Proceedings of the Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition*, pages 2667–2674, 2010.

[95] Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. Learning-by-synthesis for appearance-based 3d gaze estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[96] Yusuke Sugano, Xucong Zhang, and Andreas Bulling. Aggregaze: Collective estimation of audience attention on public displays. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, page 821–831, New York, NY, USA, 2016. Association for Computing Machinery.

[97] Li Sun, Zicheng Liu, and Ming-Ting Sun. Real time gaze estimation with a consumer depth camera. *Information Sciences*, 320:346–360, 2015.

[98] Yunjia Sun, Jiabei Zeng, Shiguang Shan, and Xilin Chen. Cross-encoder for unsupervised gaze representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3702–3711, October 2021.

[99] Dejan Todorović. Geometrical basis of perception of gaze direction. *Vision research*, 46(21):3549–3562, 2006.

[100] Matteo Tomasi, Shrinivas Pundlik, Kevin Edward Houston, and Gang Luo. Mobile device application for ocular misalignment measurement, February 14 2019. US Patent App. 16/076,592.

[101] Nachiappan Valliappan, Na Dai, Ethan Steinberg, Junfeng He, Kantwon Rogers, Venky Ramachandran, Pingmei Xu, Mina Shojaeizadeh, Li Guo, Kai Kohlhoff, and Vidhya Navalpakkam. Accelerating eye movement research via accurate and affordable smartphone eye tracking. *Nature communications*, 11:4553, 09 2020.

[102] Wang, Sung, and Ronda Venkateswarlu. Eye gaze estimation from a single image of one eye. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 136–143 vol.1, 2003.

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

[104] Kang Wang, Rui Zhao, and Qiang Ji. A hierarchical generative model for eye image synthesis and eye gaze estimation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 440–448, 2018.

[105] Kang Wang, Rui Zhao, Hui Su, and Qiang Ji. Generalizing eye tracking with bayesian adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[106] Miao Wang, Guo-Ye Yang, Ruilong Li, Run-Ze Liang, Song-Hai Zhang, Peter. M Hall, and Shi-Min Hu. Example-guided style-consistent image synthesis from semantic labeling. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[107] T. Wang, M. Liu, J. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8798–8807, 2018.

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

[109] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. A 3d morphable eye region model for gaze estimation. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 297–313, Cham, 2016. Springer International Publishing.

[110] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas 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*, pages 131–138, 2016.

[111] Erroll Wood, Tadas Baltrusaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. Gazedirector: Fully articulated eye gaze redirection in video. *Comput. Graph. Forum*, pages 217–225, 2018.

[112] Erroll Wood, Tadas Baltrusaitis, Xucong Zhang, Yusuke Sugano, Peter Robinson, and Andreas Bulling. Rendering of eyes for eye-shape registration and gaze estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3756–3764, 2015.

[113] Erroll Wood and Andreas Bulling. Eyetab: Model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, page 207–210, New York, NY, USA, 2014. Association for Computing Machinery.

[114] Zhengyang Wu, Srivignesh Rajendran, Tarrence van As, Joelle Zimmermann, Vijay Badrinarayanan, and Andrew Rabinovich. Eyenet: A multi-task network for off-axis eye gaze estimation and user understanding. *arXiv preprint arXiv:1908.09060*, 2019.

[115] Yunyang Xiong, Hyunwoo J. Kim, and Vikas Singh. Mixed effects neural networks (menets) with applications to gaze estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[116] Yunyang Xiong, Hyunwoo J. Kim, and Vikas Singh. Mixed effects neural networks (menets) with applications to gaze estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[117] Yu Yu and Jean-Marc Odobez. Unsupervised representation learning for gaze estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[118] Yuechen Yu, Gang Liu, and Jean-Marc Odobez. Deep multitask gaze estimation with a constrained landmark-gaze model. In *ECCV Workshops*, 2018.

[119] Yuechen Yu, Gang Liu, and Jean-Marc Odobez. Improving few-shot user-specific gaze adaptation via gaze redirection synthesis. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11929–11938, 2019.

[120] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[121] Xucong Zhang, Seonwook Park, Thabo Beeler, Derek Bradley, Siyu Tang, and Otmar Hilliges. Eth-xgaze: A large scale dataset for gaze estimation under extreme head pose and gaze variation. In *European Conference on Computer Vision (ECCV)*, 2020.

[122] Xucong Zhang, Yusuke Sugano, and Andreas Bulling. Revisiting data normalization for appearance-based gaze estimation. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*, ETRA '18, New York, NY, USA, 2018. Association for Computing Machinery.

[123] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. Appearance-based gaze estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[124] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 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*, pages 2299–2308. IEEE, 2017.

[125] Yanxia Zhang, Andreas Bulling, and Hans Gellersen. Sideways: A gaze interface for spontaneous interaction with situated displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, page 851–860, New York, NY, USA, 2013. Association for Computing Machinery.

[126] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In

Proceedings of the IEEE International Conference on Computer Vision, pages 2223–2232, 2017.

[127] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems 30*, pages 465–476. 2017.