# Eye Gaze Tracking for Assistive Devices

Evan Finnigan

## Abstract

Many webcam based eye gaze tracking systems are intended for use where eye gaze is limited to a computer screen. While these methods can work well for certain applications like improving the usability of webpages, they do not work well when the when gaze tracking is needed in non screen based tasks. In our case, we want to track the user's gaze on a tabletop to allow for eye control of a planar robotic device with just one webcam and no other equipment. To allow for eye gaze tracking in this application, we need to first find the user's visual axis then use the visual axis to infer where the user is looking on the tabletop. This report discusses how we can find a user's visual axis in a way that is flexible to the wide variety of head positions and eye rotations that are present when a user is attempting to naturally complete tasks on a tabletop. The method used here has three steps. First the user's eyeball centers are located using a system that tracks facial landmarks in 3D. Second, the user's pupil centers are located in 2D. Third simple projective geometry is used to find the user's visual axis based on 3D eye location and 2D pupil location. This method is capable of 22° error in the detected visual axis angle which is an improvement over OpenFace, a previous good webcam based eye tracker that works with any head orientation. Using just OpenFace, the visual axis angle error is over 28°.

## Introduction

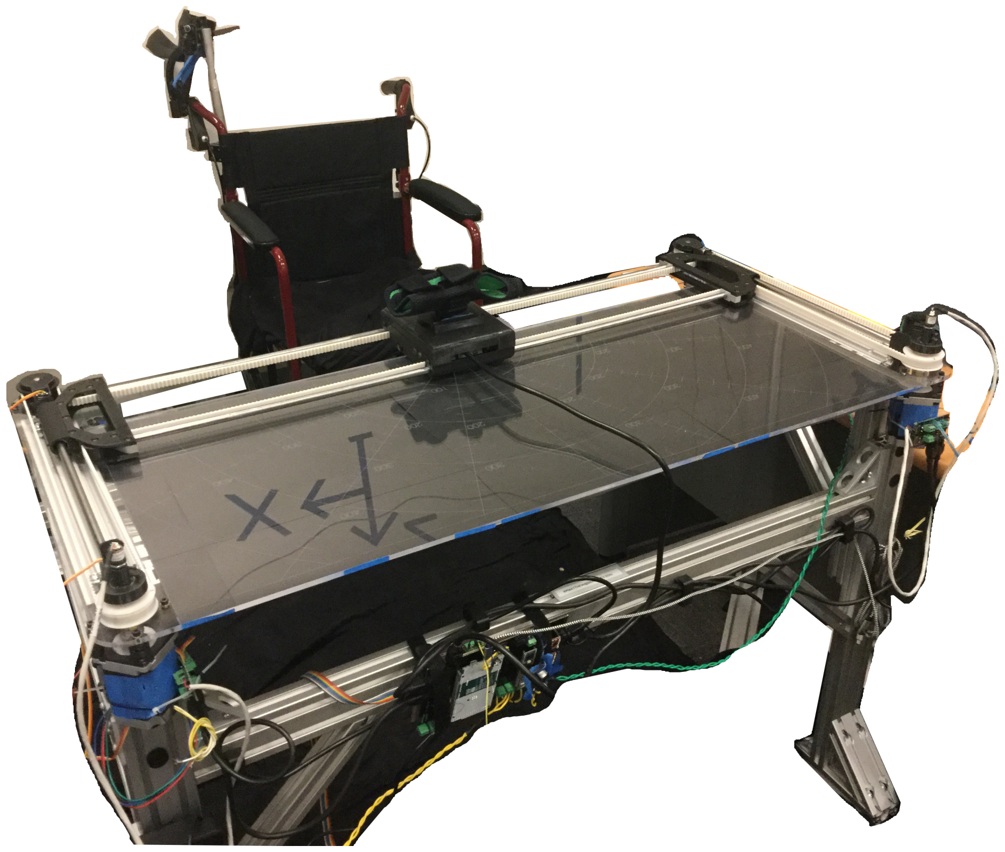


**Figure 1:** The device we are applying eye tracking control to.

Eye Gaze tracking has a wide variety of applications. Several applications are analyzing where the user of a webpage is looking to improve usability, improving the immersiveness of games by allowing users to interact with objects by looking at them and also foveated rendering for VR goggles where only the place where the user is looking is rendered in high resolution. Another application of gaze tracking is in designing interfaces that people with disabilities can use. Eye controlled wheelchairs can allow people unable to use hand controls to operate their wheelchair and eye controlled computer interfaces can allow people with disabilities to use their computers.

The problem with gaze tracking however is that high accuracy eye tracking headsets are expensive and even lower cost remote eye trackers do not provide the same level of ubiquity and low cost as webcams. For this reason, it is still worthwhile to investigate how to build systems that perform eye tracking with just a webcam and no speciality hardware.

The use case of webcam based eye tracking in this report will be tracking where a user is looking on a table. This table (pictured in 1) has a system built into it to move the users hand to positions on the table. The point of this system is to allow a user who cannot move their arms to control their arm on a tabletop by looking at targets. Specifically, in this report I will focus on finding the visual axis of a user looking at targets on the table (targets shown in 7).

One of the challenges with detecting a user's visual axis when they are looking at locations on a tabletop is that the user will have use a wide variety of head orientations to be able to naturally look at all the points on the table. Another challenging part of this task is to make sure that gaze tracking will continue to work even if the user is looking at an extreme angle. In other words the scenario that occurs when the user is attempting to look as far to the left or right as possible without moving their head. These two challenges will be a central focus of this report and the main criteria for deciding if an eye tracker is suitable for the task.

## Contributions

The main contribution of this work is to suggest a gaze direction tracking system that is robust to all head poses and works well for the task of tracking where a user is looking on a table. The task of looking at points on a table is somewhat unusual for webcam based eye trackers which usually track where a user is looking on a screen. On a screen the user is primarily looking ahead and only small gaze angles need to be tracked. Therefore, another contribution of this work is finding where some eye trackers fail when tracking large gaze angles and then determining a method that is more capable of tracking eye gaze at these larger angles.

## Overview of Eye-tracking Types

There are a variety of eye-trackers available, some requiring specialized equipment and others that run on a standard webcam.

### Eye Trackers with Specialized Hardware

Systems that require special equipment can be either head mounted such as the Tobii Pro 2 (Figure 2) glasses or remote such as the Tobii 4c (Figure 3). These devices use near-infrared light to produce corneal glints, which are reflections of the near-infrared light off of the cornea [1]. These devices then use the movement between the pupil and the glint position as well as a 3D model of the eye to accurately estimate the users gaze direction. Systems requiring specialized software are quite accurate. For the Tobii Pro 2 glasses, Tobii claims an accuracy of 0.62 degrees for the optimal use case with gaze angle $\leq 15°$.

However this performance is reduced to 3.05° for larger gaze angles and .79° for different lighting conditions.

## Webcam Based Eyetrackers

Eye-tracking systems that do not require specialized hardware are much easier to distribute and two such system, WebGazer and XLabs Chrome plugin, can even be embedded on a webpage. Webgazer [3] uses an eye detector and then performs regularized linear regression to learn a relationship between the low resolution eye patch and the gaze point on the screen. The linear model in trained with data supplied by the user clicking on calibration points, or by the user interacting with a webpage normally and the assumption that a user looks at the location where they are clicking. 2 and 3 show results from a commercial eye tracker and from Webgazer respectively. While WebGazer is less accurate, it certainly still provides usable results. In this case, it is still possible to decide which links the user is primarily looking at. XLabs uses proprietary software and it is not publicly available how it works.

The WebGazer system does not take head orientation into account when determining

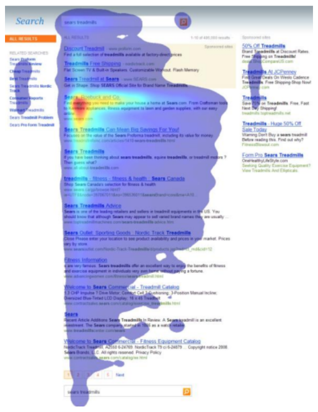
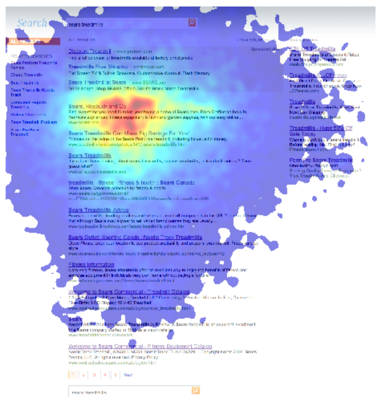


**Figure 2:** Result from a commerical eye tracker



(b) SearchGazer

**Figure 3:** Result from SearchGazer which uses WebGazer eyetracking

gaze point and needs to be recalibrated when the user moves their head.

Another system for eye tracking that does not require specialized hardware is OpenFace [9]. OpenFace tracks the head orientation as well as visual axis for both eyes. This makes it possible to account for different head orientations in determining where the user is looking.

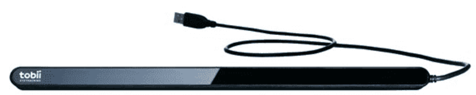


**Figure 4:** Tobii 4c remote eye tracking bar

**Figure 5:** Tobii Pro 2 eye tracking glasses

OpenFace is based on a Point Distribution Model to parameterize and regularize the detected face shape and a patch expert that detects the landmarks that make up the overall face shape.

## Needed Improvements

In preliminary versions of the system we used the WebGazer system to find gaze points. This is adequate for classifying which point on the table the user is looking at. However, it performs poorly for inferring a gaze point anywhere on the tabletop because the estimates of gaze point inferred from the eye patches with linear regression are very noisy. Additionally it requires that the user keep their head still.

OpenFace is superior to WebGazer because it tracks the head as well as the direction that the eyes are looking. This makes it possible to track gaze point at any head orientation. However OpenFace has poor eye tracking performance for large gaze angles. A possible explanation for this is that OpenFace is based on a Point Distribution Model (PDM) that was trained on the MPIIGaze [11] dataset. The MPIIGaze dataset only contains images of people looking at a computer screens. The variety of visual axis angles required to look at positions on a computer screen is far smaller than the variety of visual axis angles required to perform tasks on a tabletop. 8 shows the distribution of gaze angles in radians. It displays variation of less than 30 degrees of yaw to the left and right. Points that the user could be looking at on the tabletop have variation of 60 degrees to the left and right 7. The PDM in OpenFace both defines what variation is possible in eye landmark shapes and regularizes the shape. This means that shapes of landmark locations that are not heavily represented in the data the PDM is trained on will not be detected properly. This can be seen to be a problem in 6 where the ground truth eye shape should have the iris further to the right. The blue circle and smaller concentric circle should be around the iris and pupil respectively. The fact that they are not properly aligned will damage the final visual axis estimation. In order to improve upon this system, I decided to use the face tracking features of OpenFace and investigate new ways to find gaze direction.

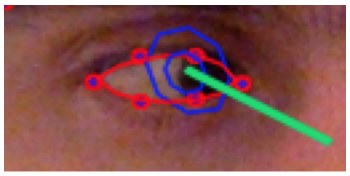


**Figure 6:** Demonstration of poor tracking at extreme angles.

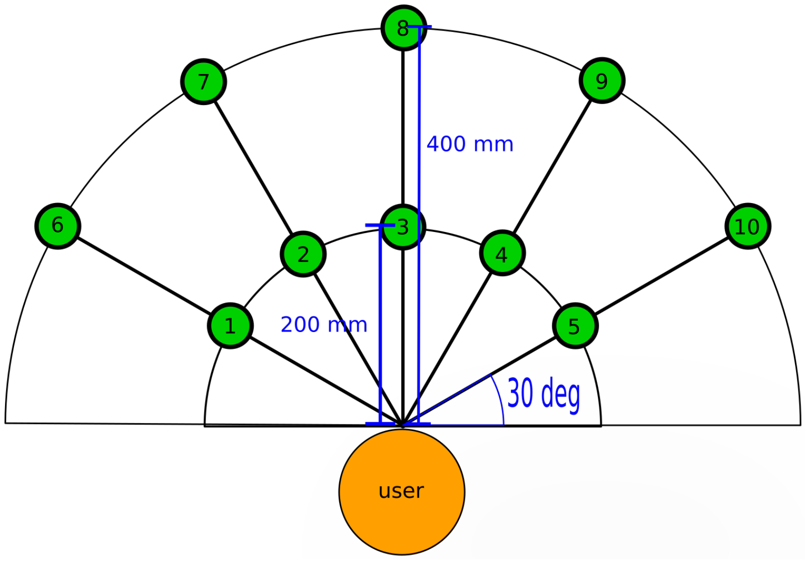


**Figure 7:** Gaze targets on the working table area. The user sits at the bottom middle.

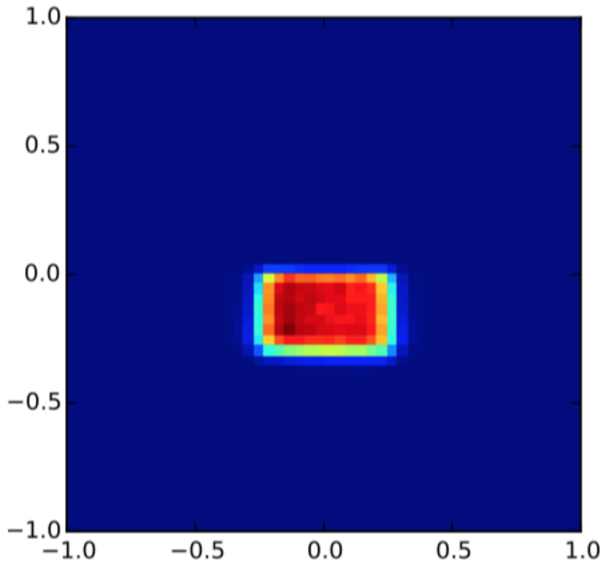


**Figure 8:** Graph displaying the distribution of pitch and yaw angles in radians. From [11].

## Related Work

### CNN Based Method on a Large Dataset

Zhang et al. [10] presents a method to learn to predict gaze direction with a CNN. They collected 213,659 images and corresponding gaze targets from users' regular laptop use. These images then go through a normalization procedure where the eye patches are extracted and warped so that the eyeball center is in the center of the extracted image and the rotation caused by different head angles is removed. The gaze vector from the eyeball to the gaze target is also rotated to bring it into the normalized camera frame. Then, a CNN is trained to regress from normalized eye patch images to gaze vectors. This method obtains errors of less than 15 degrees for visual angle tracking.

### Latent Space Gaussian Process Gaze-Tracking

Wu et al. [6] presents a way to track gaze using Gaussian Processes, particularly a GP-LVM (Gaussian Process Latent Variable Model) [2]. A GPLVM is a method of nonlinear dimensionality. The method presented in [6] starts by using calibration images to build a 2-dimensional latent feature space using GP-LVM. Quite impressively, the latent variable space has semantic meaning where the axis represent the pitch and yaw of the gaze vector (Figure 6).

To detect where a user is looking they first find the nearest neighbors of the user's newly detected eye patch in the 2-dimensional latent space. The initial estimate of where

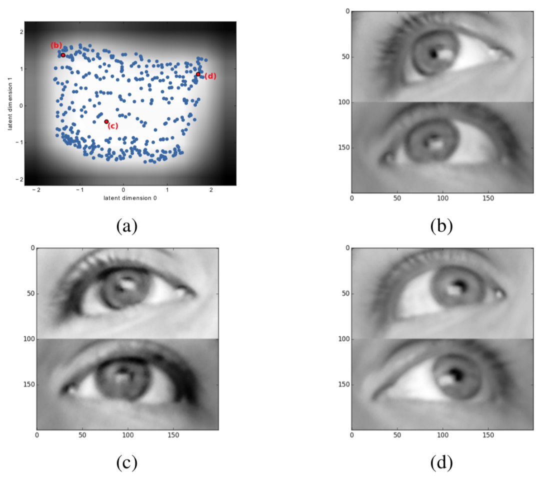


**Figure 9:** Visualization of latent space created by GPLVM and corresponding generated eye patches. From [6]]

the user is looking on the screen is the average gaze point of the neighbors. As GPLVM is a generative model, it is possible to generate an eye patch that would match a certain gaze point. Using this generated eye patch the gaze estimate is then improved by solving an optimization problem to maximize the normalized cross correlation between the user's detected eye patch and the generated eye patch.

This method is interesting because it is uses an unsupervised learning technique and is therefore less dependent on available labeled datasets that may provide a wide enough variety of eye types and gaze directions.

### A Hierarchical Generative Model for Eye Image Synthesis and Eye Gaze Estimation

A more modern semi-supervised approach is presented in [5]. The authors of [5] use a two step procedure that first converts eye images to 2D eye shape using a biderectional GAN then converts eye shape to gaze direction using a hierarchical generative shape model (HGSM) which the authors introduce. The HGSM is a probabilistic model trained on the Unity Eyes dataset [7] and is called hierarchical because it first determines the user's personal eye parameters then uses a user agnostic model to determine gaze direction.

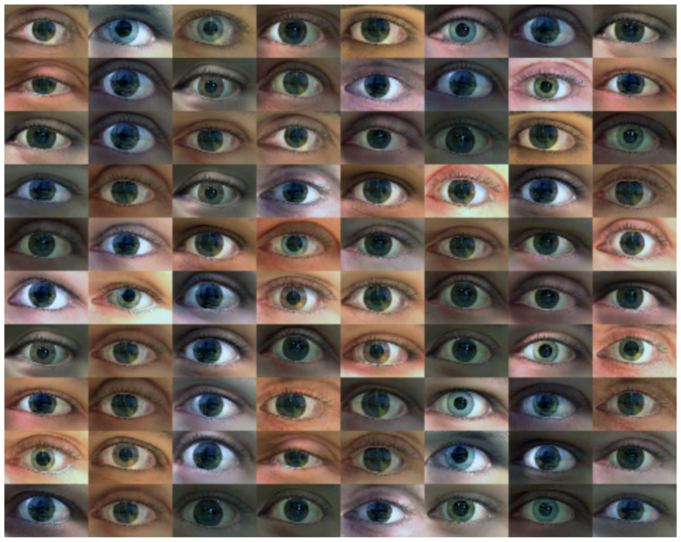


**Figure 10:** Examples of eyes from the Unity Eyes dataset. These are computer generated eye patch images, which is convenient because the gaze angle is already known and the eyes do not need to be labeled.

## Pupil Center Tracking by Means of Gradients

In contrast to the previous methods, the method presented in [4] does not use learning. Instead, their method solves an optimization problem to maximize the alignment between a displacement vector from the proposed pupil center to each point in the image and the image gradient at each point. This optimization problem is given mathematically as

$$c^* = \underset{c}{\operatorname{argmax}} \left\{ \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{d}_i^T \boldsymbol{g}_i)^2 \right\} \tag{1}$$

$$d_i = \frac{\boldsymbol{x}_i - \boldsymbol{c}}{||\boldsymbol{x}_i - \boldsymbol{c}||_2}, \forall i : ||\boldsymbol{g}_i||_2 = 1 \tag{2}$$

where the $d_i$ is a displacement vector from c to point $i$ and $\boldsymbol{g}_i$ is the gradient at point $i$. The eye patch is small enough so that all of the pixels in the eye patch can be checked and the optimal c found. The idea behind this algorithm is that the point $c^*$ from the optimization problem 1 will be at the center of the most prominent circle. This is because the displacement vector from the center of a circle to any point on the edge of the circle is aligned with the gradient at the point on the edge of the circle. 11 shows aligned and non-aligned gradient and displacement vectors.

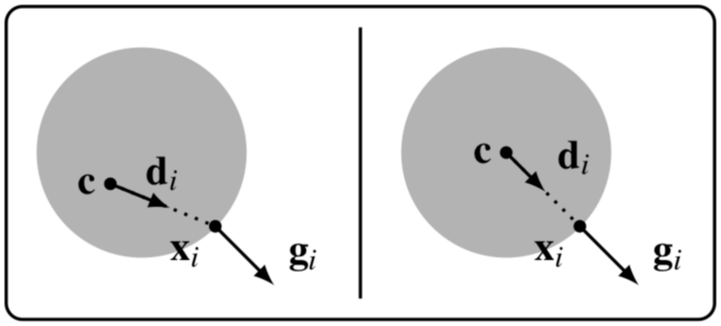


**Figure 11:** Examples of un-aligned and aligned pupil center and iris edge gradient vectors (left and right respectiveley).

## Method

The method proposed here has three steps. First an eye patch is located for the eye we are finding the gaze axis for using OpenFace. Then the pupil center is localized in the found eye patch using the algorithm from [4]. Finally an unprojection of the pupils centers and eyeball geometry are used to find the gaze axis. This last step is a commonly used way to infer gaze direction from 2D pupil locations and 3D eye geometry. This method only assumes that the pupil is visible to be tracked and is not occluded by any structures such as frames on glasses. Both eyes need to be visible. The method does not make any assumptions on head pose or require that the user maintain the same head pose throughout tracking.

### Eye Patch Localization

The OpenFace toolkit [9] provides a stable estimate of the eye patch location that is flexible to a wide variety of head orientations. This is in contrast to the pupil tracking performance of OpenFace that uses the same algorithm but has low performance because it was trained on a dataset with a limited variety of eye gaze angles.

OpenFace is based on Convolutional Experts Constrained Local Model (CE-CLM) which is described in Zadeh et al. [8]. CE-CLM has a CNN based patch expert component and a Point Distribution Model (PDM) component. The patch expert component finds a distribution of probabilities for the location of each landmark in the vicinity of the current estimated position of the landmark. Then, the parameters of the PDM are optimized taking into account the responses of the patch experts as well as the probability of landmarks being in a certain orientation in relation to each other.

After facial landmarks are found with CE-CLM, the eye patch for each eye can be extracted by finding a bounding box around the eye landmarks.

## Pupil Localization

Pupil localization is achieved by using the last method presented in the Related Work section, pupil center tracking by means of gradients. This algorithm is applied to the eye patch found by the CE-CLM algorithm. This method was chosen for its simplicity and the fact that unlike learning based methods it is less sensitive to changes in lighting and users not well represented in the training set.

## 3D Geometry to find Gaze Axis

Once a pupil center and eye landmarks have been obtained the visual axis vector can be found. First, the pupil center is unprojected. This consists of rewriting the 2 dimensional pupil center location in homogenous coordinates as $\boldsymbol{x} = [x_p, y_p, 1]$ where $x_p$ and $y_p$ are the coordinates of the pupil center in the image.

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix} \tag{3}$$

Equation 3 is the perspective projection formula relating $\boldsymbol{x}$ and $\boldsymbol{p} = [X_p, Y_p, Z_p]$, the position of the pupil in 3D space. s is a scaling factor, therefore at this point only the direction towards the pupil from the camera is known. To proceed, the intersection of the ray starting at the origin of the camera coordinate system and pointing in the direction of $\boldsymbol{x}$ and a sphere centered at the center of the eyeball, $\boldsymbol{e}$ of radius 12 mm is found. This is the estimate for $\boldsymbol{p}$ the pupil center in 3D space. The center of the eyeball can be found as the average of the 3D face landmarks corresponding to the eye corners (lateral canthus and medial canthus). A sphere of radius 12mm is used because the average human eyeball has a radius of 12mm. Finally, the gaze vector $\boldsymbol{g}$ is determined as $\boldsymbol{g} = \boldsymbol{e} - \boldsymbol{p}$.

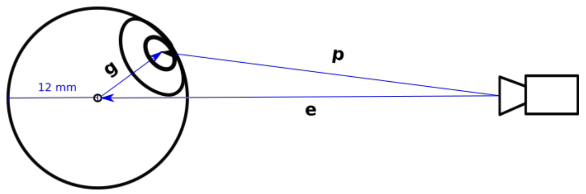


**Figure 12:** Diagram of the method of finding gaze vector $\boldsymbol{g}$ from $\boldsymbol{p}$ and $\boldsymbol{e}$

This method exploits the spherical symmetry of the eyeball to remove the dependence of the estimated gaze vector on the orientation of the head.

## Results

10 experiments of looking at the top 5 targets (6, 7, 8, 9, 10) on the table were run to demonstrate that the method proposed in this report can improve accuracy for large gaze angles. The method in this report increases the accuracy especially at large gaze angles, however the precision is decreased.

| Accuracies (over 10 trials) | | | | |
|---|---|---|---|---|
| Method | Head Pose Fixed | Yaw Error (degrees) | Pitch Error (degrees) | Total Error (degrees) |
| Original Method | no | 28.64 | 2.641 | 28.76 |
| Augmented Method | no | 21.82 | 4.179 | 22.21 |
| Original Method | yes | 31.16 | 2.52 | 31.26 |
| Augmented Method | yes | 23.52 | 4.06 | 23.87 |

The data from which these accuracies are generated is taken with the user looking at each of the targets on the outer row of the table. The user is seated at the center of the arc of targets. The angles recorded should ideally be $-60°$, $-30°$, $0°$, $30°$, $60°$.

These results show that the augmented method presented in this report can slightly but noticeably reduces the error for the yaw angle. The error for the pitch angle increases because the new eye gaze estimate is less stable than the original method. The important factor however is that the augmented method does allow a broader range of gaze angles to be tracked.

Additionally, whether the head is free to move or must remain strictly in a forward facing pose does not have a large effect on the accuracy of the system.

## Discussion

While the method proposed in this paper provides some improvement over the original method (OpenFace) that finds eye landmarks to track gaze direction, it is still not particularly accurate. The error is still above $20°$ which is substantial compared to the $120°$ range of possible locations to look at on the table. A big contributing factor is that computing the gaze vector based on the 2D pupil location and 3D eyeball center is heavily reliant on the 3D eyeball center location being correct. A next step to improve accuracy would be to remove the reliance of the gaze vector direction from the location of the eyeball center. Head mounted eye trackers detect gaze direction by performing polynomial regression on the location of the pupil, but head mounted eye trackers only need to deal with eye images in one orientation. If this method could be generalized to predict gaze direction based on pupil location within an eye patch it could greatly improve the accuracy of the system.

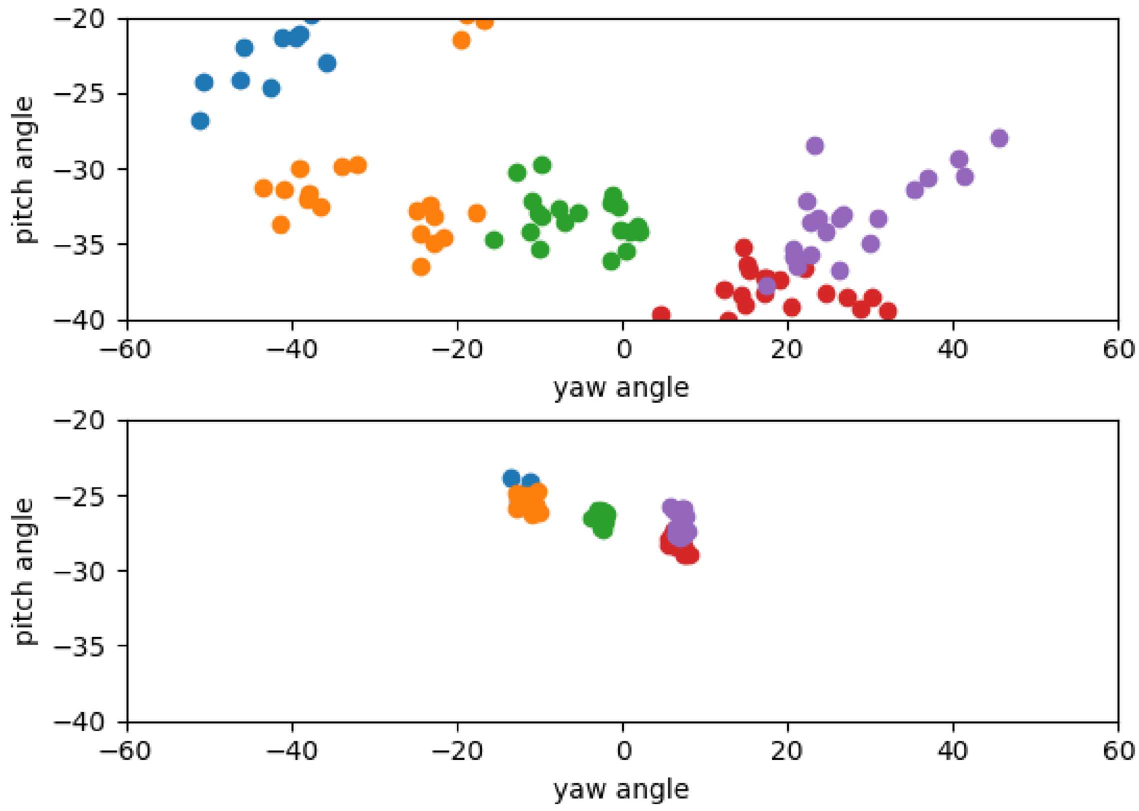


**Figure 13:** Eye gaze yaw vs. pitch generated by looking at the outer row of targets (one trial). Given where the user is sitting, the clusters should all have pitch angle $-30°$ and the yaw angles should be $-60°$, $-30°$, $0°$, $30°$ and $60°$. Augmented method (top) and original method (bottom). The augmented method has higher accuracy which allows gaze tracking on a broader area of the table but has lower precision.

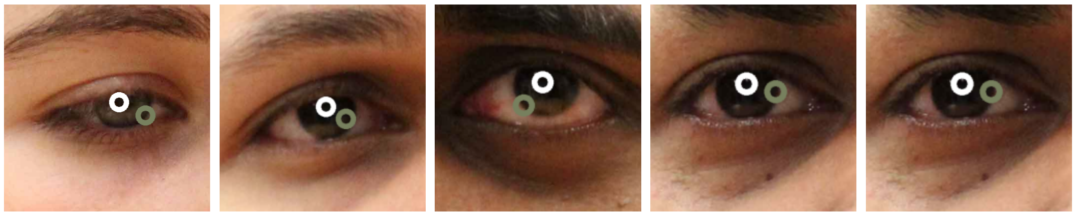


**Figure 14:** Failure cases of the original method that the augmented method is more successful at. White and green circles are for augmented and original methods respectively.

## References

[1] Anuradha Kar and Peter M. Corcoran. A review and analysis of eye-gaze estimation systems, algorithms and performance evaluation methods in consumer platforms. *IEEE Access*, 5:16495–16519, 2017.

[2] Neil D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'03, pages 329–336, Cambridge, MA, USA, 2003. MIT Press.

[3] Alexandra Papoutsaki, Patsorn Sangkloy, James Laskey, Nediyana Daskalova, Jeff Huang, and James Hays. Webgazer: Scalable webcam eye tracking using user interactions. 07 2016.

[4] Fabian Timm and Erhardt Barth. Accurate eye centre localisation by means of gradients. pages 125–130, 01 2011.

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

[6] Nicolai Wojke, Jens Hedrich, and Detlev Droege. Latent space gaussian process gaze-tracking. *OGRW2014*, page 144.

[7] 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 and Applications*, pages 131–138, 2016.

[8] A. Zadeh, T. Baltruaitis, and L. Morency. Convolutional experts constrained local model for facial landmark detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2051–2059, July 2017.

[9] A. Zadeh, Y. C. Lim, T. Baltruaitis, and L. Morency. Convolutional experts constrained local model for 3d facial landmark detection. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2519–2528, Oct 2017.

[10] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling. Appearance-based gaze estimation in the wild. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4511–4520, June 2015.

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