Appearance-based Eye Gaze Estimation

Kar-Han Tan
Department of CS and
Beckman Institute
University of Illinois
at Urbana-Champaign

David J Kriegman
Department of CS
University of California
at San Diego

Narendra Ahuja
Department of ECE and
Beckman Institute
University of Illinois
at Urbana-Champaign

Abstract

We present a method for estimating eye gaze direction, which represents a departure from conventional eye gaze estimation methods, the majority of which are based on tracking specific optical phenomena like corneal reflection and the Purkinje images. We employ an appearance manifold model, but instead of using a densely sampled spline to perform the nearest manifold point query, we retain the original set of sparse appearance samples and use linear interpolation among a small subset of samples to approximate the nearest manifold point. The advantage of this approach is that since we are only storing a sparse set of samples, each sample can be a high dimensional vector that retains more representational accuracy than short vectors produced with dimensionality reduction methods. The algorithm was tested with a set of eye images labelled with ground truth point-of-regard coordinates. We have found that the algorithm is capable of estimating eye gaze with a mean angular error of 0.38 degrees, which is comparable to that obtained by commercially available eye trackers.

1 Introduction

The ability to *detect* the presence of visual attention from human users, and/or determine what a human user is looking at by *estimating* the direction of eye gaze is useful in many applications. For example if a graphics renderer knows which part of the display the user is looking at, it can adapt itself such that more details are shown where the visual attention is directed. In behavioural studies gaze detection and estimation are also invaluable. For example, the frequency with which a pilot looks at a panel display, and the length of time required to read off the information could be used to measure the effectiveness of the display. Perhaps the application area that would benefit the greatest with the maturation of vision-based gaze estimation techniques would be in human-computer interaction [13, 6, 5]. With the availability of affordable computational power and high-quality imaging hardware, eye trackers may well become standard issue on personal computers in the near future.

2 Previous Work

Not surprisingly, eye tracking has attracted the interest of many researchers, and eye trackers have been commercially available for many years. A comprehensive survey of the earlier works can be found in [16]. Our review shall focus on eye tracking based on computer vision techniques.

A model-based approach to eye image analysis can be seen in Daugman's work [3]. In this application, the machine used for identifying persons needs to seach the input image to localize the iris. The algorithm essentially performs a coarse-to-fine search for a circular contour in the image corresponding to the limbus, and then searches for the pupil. In this case the model is simply a circle and a specificati on of the grayscale variation around the contour. For example, while searching for the pupil (without using the bright pupil method), the algorithm normalizes the grayscale values around the contour so as to bring out the pupil contour.

A more elaborate model for the eye was proposed in Yuille et al.'s deformable template work [17] which explicitly models the limbus as a circle, eye lids as two parabolic sections, the centroids of the two visible portions of the sclera beside the iris, and the two regions of the sclera between the eye lids and below and above the iris. The limbus circle is attracted to contours that are dark on the interior and light on the outside, and the four scleral centroids are attracted to bright portions of the image. Gradient descent is then used to fit the model to images of eyes. Although gaze estimation was not a goal of the paper, one could use the parameters thus derived to estimate the direction of the eye gaze. This template was augmented in [11] to account for eye blinks. This eye model is able to transition between the open and close state according to the given input image to produce the best fit.

Trackers have also been built that look for other facial features, like the corners of the mouth, corners of the eyes, and corners of the eye brows. In [7], the tracking process runs in two stages, where the facial features are first matched with the input video stream, and using the head pose information derived in the first stage, a second stage then estimates the eye gaze using an eye model that is used to track the limbus. A simpler model was also proposed in [4], where only the facial features were used to compute a *facial normal*, which is then used as a substitute for the gaze direction.

In Baluja and Pomerleau's neural network-based method [1], cropped images of the eyes are used as inputs to a neural network. Training data is collected by requiring the user to look at a given point on a computer monitor, and taking a picture of the eye looking at the given point. Thus each training sample consists of the image of the eye and an (x, y) label of the location on the screen that the user is looking at. In their experiments, 2000 training samples were used to train a multilayer neural network, and the authors reported an accuracy of about 1.5 degrees. Their tracker runs at 15Hz, and allows some head movement. A similar method is documented in [14], which also achieved an accuracy of 1.5 degrees, using 3000 training samples.

3 Using Appearance-based Methods for Gaze Estimation

Instead of using explicit geometric features like contours and corners, an alternative approach to object pose estimation is to treat an image as a point in a high-dimensional space. For example, a 20 pixel by 20 pixel intensity image can be considered a 400-component vector, or a point in a 400-dimensional space. Algorithms using this representation are often referred to as being *appearance-based* or *view-based*, and they have been successfully applied to object recognition and detection [2, 8, 12, 15]. View-based techniques have the advantage that they are easy to implement and are generally more robust than feature-based methods. [8, 9] showed that a set of images of an object taken under varying viewing parameters forms a continuous set of points in the high-dimensional space, which they call an *appearance manifold*. For a given image of an object, its viewing parameters can be estimated by finding the point on the object's appearance manifold that is nearest to the given image, and using the parameters for that point as the estimate.

3.1 Spline-based Representation

In theory, the appearance manifold is continuous. However, in practice often only a discrete set of manifold samples are available. Clearly if we estimate object pose parameters with a nearest-neighbour search, the precision is directly related to how densely the manifold is sampled. As such, a spline representation has been proposed in [8, 9], where a spline is fitted to the sparse set of samples and then used to approximate the continuous manifold. It was then suggested that the spline can be densely sampled and the resulting set of points be used in the nearest-neighbour search, thus allowing the manifold to support queries at arbitrary precisions. Of course, this can result in a large number of sample points, and consequently a large computational cost in the nearest-neighbour search. Dimensionality reduction techniques such as PCA can be employed to lessen the impact, and also clever data structures can be used to speed up the search process [9].

3.2 Linear Representation

An alternative approach for representing the appearance manifold is to assume that points on the manifold can be approximated by linear combinations of neighbouring samples. Thus for a given test image, we find a set of manifold sample points that are near to the test image, and interpolate these neighbouring sample points to give an estimate for the point on the manifold which is closest to the test image. This approach is similar to that proposed in [10], in which the k-nearest sample points from the manifold are used for interpolation. However this does not make use of the topological information available in a parametrically sampled manifold in the sense that the k-nearest sample points may not actually be from a single neighbourhood on the manifold. A better way would be to ensure that the picked sample point are truly 'geo desic' neighbours on the manifold, as shown in Figure 1. This constraint is critical when using an appearance manifold for parameter estimation.

3.3 Nearest manifold point querying

Having discussed the idea and motivation behind our technique for parameter estimation, we are now ready to present our algorithm, which consists of three steps:

1. Given a test point, find the closest set of sample points on the manifold forming a neighborhood.
2. Find a set of weights that interpolate these points to give an estimated point on the manifold.
3. Use the same weights to interpolate the parameters for the sample points to obtain the estimated parameters for the test point.

Details for each of the steps are as follows:

Choose closest manifold points. In order to ensure that the samples chosen form a neighbourhood on the manifold,

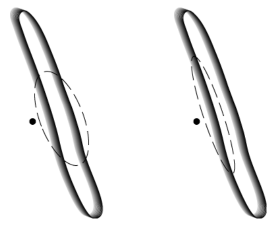


Figure 1. Choosing samples from a manifold (shown as an elongated loop) that are near to a given point (shown as a black dot). The chosen manifold points are enclosed within the dashed ellipses. (a) Without topological constraint. It can be seen that two disjoint patches on the manifold are chosen. (b) With topological constraint. The chosen set of points form a neighbourhood

we first need a way to represent the topology of the manifold. In our case, each manifold point is an image of an eye, labeled with a 2D coordinate of a point on a display device which the eye is fixating upon when the image was taken. In this case we can use a Delaunay triangulation of the set of 2D coordinates to represent the topology: if there is an edge in the Delaunay triangulation between two of the points, then the two corresponding eye images can be considered neighbours on the manifold. Given a test image, we then choose the set of samples by first ranking all manifold points by their distance to the test image. We then start from the nearest point and go down the sorted list. Choose the first three points encountered that form a triangle. We also add the set of points that are adjacent to these three points to the chosen set.

Compute interpolating weights. Once we have the set of sample manifold points, we can compute the weights by solving a constrained least squares problem. Let x be the test image and $s_1 \dots s_k$ be the manifold samples chosen. We want to find a set of weights$w_1 \dots w_k$ that minimizes the error

$$\epsilon = |x - \sum_i w_i s_i|$$

subject to the constraint

$$\sum_i w_i = 1$$

. Which can be solved by the procedure described in [10].

Estimate parameters. Once we have the weights w_i, we can estimate the parameters for the point x by interpolating the sample parameters with the same weights,

$$Q_x = \sum_i w_i Q_{s_i}$$

where Q_y is the parameter vector for a point y.

4 Experimental Setup

To test the performance of our algorithm in eye gaze estimation, we constructed an image capture system to collect a data set consisting of images of an eye that are labeled with the coordinates of a point that the eye is looking at. The system consists of a computer with a monitor, a monochrome camera, and a framegrabber. The image capture process is as follows: the computer displays a crosshair marker on its screen, and the user is asked to align (by moving a mouse) a cursor (also displayed as a crosshair) with the marker shown, and click on the marker. The next marker is then shown in a different location, and the user repeats the task. Since the user' s eye will be looking at the position of the marker on screen when he/she is trying to align the marker and the cursor, images of the user' s eye are captured when the cursor and marker are almost aligned. Note that the image capture takes place before the user clicks on the marker, since it is likely that upon completing the task the eye may start looking away in anticipation of the next marker. To further ensure that the user is indeed looking at the marker positions displayed, we pick the positions of the markers randomly while ensuring that at some point the extreme corner coordinates are chosen. If the marker positions can be reliably predicted by the user, it may be possible for the user to align the markers without looking at the markers directly, which would result in erroneous labels for the eye images.

We also mounted an infrared illuminator next to the camera, which has a sensor that is sensitive to infrared as well as visible light. This not only ensures that the eye is well lit, it also provides a certain amount of control over the illumination. Typical images can be seen in Figure 2(a). It can be seen that the facial region around the eye appears very bright. One should realize, however, that since the illumination was with infrared light, the user does not see a bright light, which would be very distracting. It should also be noted that the pupil does not exhibit the “red eye” effect since the infrared light was not close to the optical axis of the camera. In fact, the eye turns out to be much darker than the brightly illuminated facial region. We use this effect to find the eye in the image by thresholding the intensity image and identifying the connected dark region that is closest to a predetermined size. For the images shown, the intensity threshold is set at 200 (out of 255), and the region size parameter is chosen to be 4000 pixels. The eye image is then cropped from a rectangular region centered at the spatial

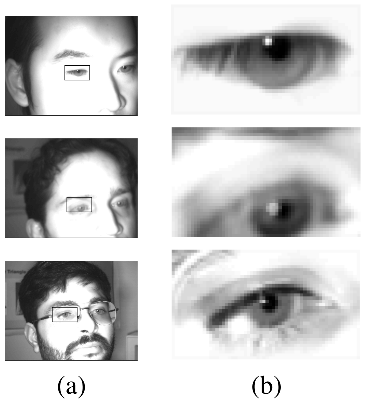


Figure 2. Samples of labeled data set collected from three subjects (from top) X, Y, and Z, who are at distances 18, 20, and 24 inches away from the display respectively. (a) Raw captured image. Cropped eye region is shown as an overlaid rectangle. (b) Cropped and scaled image of eye used as appearance sample.

centroid of this dark region. Examples of the cropped image is shown in Figure 2(b). It can be seen that the cropping is different for the three users shown, in that the cropping algorithm also included the eye brows for the second user, and did not include the eye completely in the image. It turns out that this is still good enough for gaze estimation.

4.1 Measuring Accuracy

We used a set of 252 images each from three users, and evaluated our system by "Leaving one out": using each one of the eye images as the test image, and the rest of the set to form the appearance manifold. The mean error is computed as

$$\text{Mean error} = \frac{1}{n}\sum_{i=1}^{n}|P_e - P_t|$$

where

$$\begin{aligned} P_e &= \text{estimated position} \\ P_t &= \text{true position} \\ n &= \text{number of test samples} \end{aligned}$$

In this experiment, the eye is 18-24 inches away from the monitor, which displays the markers in an approximately planar, rectangular region of about 12 inches wide and 9 inches tall. We also assume that the line of sight is perpendicular to the planar region when the eye is looking at its center. We can estimate the mean angular error as follows:

$$\text{Mean angular error} = \tan^{-1}\frac{\text{Mean error}}{\text{Distance from Screen}}$$

Figure 3(c) shows the results for Subject X with the entire set of 252 images. The crosses show the real marker positions, and the round dots show the estimated positions, with a line segment joining corresponding crosses and dots. From the scatter plot, we can see that there were some large errors around the periphery. This is to be expected, as the positions were estimated by interpolating marker positions from the appearance manifold, and the test samples on the periphery cannot be expected to be approximated by extrapolating points that are near the periphery. We can identify the peripheral test samples as those whose true marker positions form the convex hull of the complete set of marker positions. We also discard the convex hull vertices of the set of points left after removing the first convex hull. For the data set shown, it amounts to removing the set of points on the rectangular border. The average angular errors for the three subjects are tabulated in Figure 3(a). As can be seen, the angular error averaged over all subjects is 0.3838 degrees.

Figure 3(b) shows that the accuracy achieved is comparable to those reported by commercially available optical trackers, and is superior to the existing appearance-based methods using neural networks. The Dual Purkinje Image eye tracker claims to have an impressive "less than 1 minute of arc resolution" accuracy (http://www.fourward.com). However this is measured with an artificial eye. Vision researchers who have used the Dual Purkinje Image eye tracker reported that its accuracy with human subjects is on the order of 0.5 degree. The same table also shows the number of calibration samples required for each of the methods. While our method still requires considerably more samples than optical trackers like the Eyelink II, it requires dramatically fewer samples than the neural network-based methods while still achieving superior accuracy.

5 Conclusion

We have presented a new method for estimating eye gaze direction based on appearance-manifolds, and the degree of accuracy achieved is very encouraging. We have also suggested an enhancement to the appearance manifold method by proposing a nearest manifold point search technique that exploits the topological information inherently present in the manifold model. The degree of accuracy achieved in our experiments is comparable to existing eye trackers, indicating that appearance-based parameter estimation is at least a viable solution to the eye gaze estimation problem.

Acknowledgments

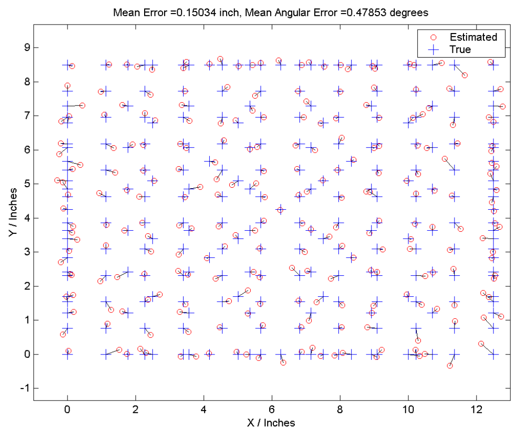
The support of the Office of Naval Research under grant N00014-99-1-0091 is gratefully acknowledged. D. Kriegman was supported in part by NSF ITR IIS 00-85980. The

Subject	Complete	Non-peripheral
X	0.47853	0.44551
Y	0.43685	0.30534
Z	0.47112	0.4004
Mean	0.4622	0.3838

(a)

System	Accuracy (degrees)	Calibration Samples
Xu-Machin-Sheppard [14]	1.5	3000
Baluja-Pomerleau [1]	1.5	2000
ASL Eyelink II	0.5	9
Appearance-based	0.38	252

(b)



(c)

Figure 3. Gaze estimation results. (a) Average angular errors for the three subjects, measured in degrees. (b) Comparing with existing methods. (c) Estimation results for subject X. Crosses are the ground truth marker locations, and the round dots are the estimated location that the eye is fixated upon.

first author is very grateful to Professor George McConkie for his helpful suggestions, and to Dr Hong Hua for her help in locating several of the papers referenced.

References

[1] S. Baluja and D. Pomerleau. Non-intrusive gaze tracking using artificial neural networks. *CMU CS Technical Report*, CMU-CS-94-102, 1994.

[2] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs fisherf aces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[3] J. G. Daugman. High confidence visual recognition of persons by a test of statistical independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1148–1161, 1993.

[4] A. H. Gee and R. Cipolla. Determining the gaze of faces in images. *University of Cambridge Department of Engineering Technical Report*, CUED/F-INFENG/TR 174, 1994.

[5] K. Grauman, M. Betke, J. Gips, and G. R. Bradski. Communication via eye blinks – detection and duration analysis in real time. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000.

[6] R. J. K. Jacob. The future of input devices. *ACM Computing Surveys*, 28(4), 1996.

[7] Y. Matsumoto, T. Ogasawara, and A. Zelinsky. Behavior recognition based on head pose and gaze direction measurement. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2127–2132, 2000.

[8] H. Murase and S. K. Nayar. Visual learning and recognition of 3d objects from appearance. *International Journal of Computer Vision*, 14(1):5–24, 1995.

[9] S. A. Nene and S. K. Nayar. A simple algorithm for nearest neighbor search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):989–1003, 1997.

[10] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(22):2323–2326, 2000.

[11] Y.-L. Tian, T. Kanade, and J. Cohn. Dual-state parametric eye tracking. In *Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition (FG'00)*, pages 110 – 115, 2000.

[12] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 1991.

[13] C. Ware and H. H. Mikaelian. An evaluation of an eye tracker as a device for computer input. In *Proceedings of the ACM Conference on Human Factors in Computing Systems and Graphics Interface (CHI/GI) 1987*, pages 183–188, 1987.

[14] L.-Q. Xu, D. Machin, and P. Sheppard. A novel approach to real-time non-intrusive gaze finding. In*British Machine Vision Conference*, 1998.

[15] M.-H. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

[16] L. R. Young and D. Sheena. Methods and designs: Survey of eye movement recording methods. *Behavior Research Methods and Instrumentation*, 7(5):397–429, 1975.

[17] A. L. Yuille, P. W. Hallinan, and D. S. Cohen. Feature extraction from faces using deformable templaces. *International Journal of Computer Vision*, 8(2):99–111, 1992.