Eye Gaze Tracking Using an Active Stereo Head

David Beymer and Myron Flickner
IBM Almaden Research Center, San Jose, CA 95120
{beymer, flick}@almaden.ibm.com

Abstract

In the eye gaze tracking problem, the goal is to determine where on a monitor screen a computer user is looking – the gaze point. Existing systems generally have one of two limitations: either the head must remain fixed in front of a stationary camera, or, to allow for head motion, the user must wear an obtrusive device. We introduce a 3D eye tracking system where head motion is allowed without the need for markers or worn devices. We use a pair of stereo systems: a wide angle stereo system detects the face and steers an active narrow FOV stereo system to track the eye at high resolution. For high resolution tracking, the eye is modeled in 3D, including the corneal ball, pupil and fovea. In this paper, we discuss the calibration of the stereo systems, the eye model, eye detection and tracking, and we close with an evaluation of the accuracy of the estimated gaze point on the monitor.

1 Introduction

The problem of gaze tracking is estimating where on the computer monitor a user is looking – the gaze point. A number of interesting applications drives interest in this technology. In HCI, eye tracking could be used to pre-fetch web sites or make large movements of the cursor across the screen [1, 2]. Other applications include interfaces for the disabled, psychological experiments on human eye motion, car driver attentiveness, and evaluating the effectiveness of web site design. To date, however, eye tracking systems have been too expensive, cumbersome, and limited in tracking range to make inroads on these applications.

The higher accuracy eye tracking techniques acquire high resolution imagery of the eye. This high resolution video is acquired with either (1) a stationary camera with narrow field of view [3], or (2) a camera on a head-mounted device that captures a close-up shot of the eye [4]. Both of these approaches have their challenges, though, with solution (1) limiting the head motion of the user being tracked, and (2) requiring the user to wear a cumbersome device. There are eye gaze tracking techniques that use wider angle lenses [5, 6], but the accuracy of the gaze direction is about 2°. The appearance based technique of [7] reports an accuracy of 0.38° on three users, but their appearance manifold seems tuned for a specific head-to-monitor geometry.

In this paper, we propose using an active stereo head with narrow FOV cameras to actively track a user's gaze. Our active stereo head pans, tilts, and adjusts its focus, so the user has a range of head motion in all directions. A wide angle stereo system detects the users head and steers the narrow FOV system onto the user's right eye. This allows for freedom of head motion without the need for wearing camera devices.

We propose a 3D tracking approach that models the 3D anatomy of the eye. Our model includes the corneal ball, the pupil, and the angular offset of the fovea from the optical axis. The FreeGaze system [8] also proposes using a 3D model for tracking, but it only models the corneal ball as a sphere (we explore using an ellipsoid), and the fovea is not explicitly modeled in 3D. Our model is the most detailed 3D model of the eye used for tracking in the computer vision community.

Using this model, we track the eye using a 3D alignment approach. For a hypothesized pose, the model eye pupil can be refracted though the cornea and projected into the image. These projected model features can then be matched to extracted image features, driving our tracking approach.

In this paper, we first describe the stereo hardware and our 3D eye model. Then we address calibrating the system in section 4 and in section 5 we describe how the system detects and tracks the eye. Finally the eye gaze direction is intersected with the monitor plane to estimate the gaze point, and we close with an evaluation of gaze point accuracy.

2 Hardware setup

At the top level, the system is divided into wide angle stereo for face detection and narrow FOV stereo for eye tracking. The wide angle system, shown in Fig. 1, is located just below the center bottom of the monitor screen. The stereo baseline is oriented vertically since this optimizes stereo matching on the central facial features, which are predominantly horizontal edges. We use a commercially available

Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)
1063-6919/03 $17.00 © 2003 IEEE

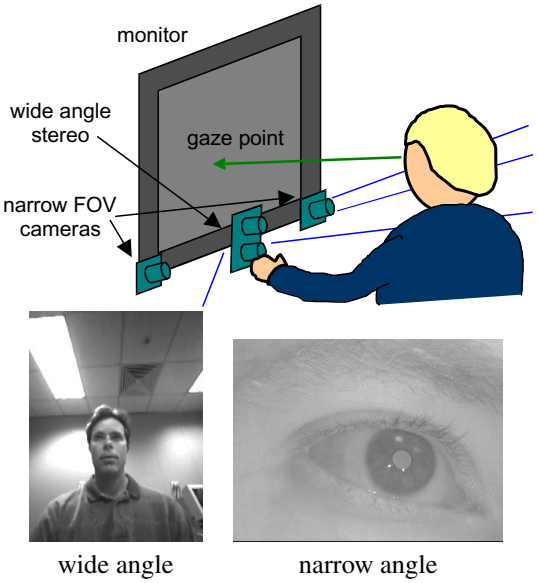

Figure 1: Our gaze tracking system combines wide angle stereo for head detection and narrow FOV stereo for high resolution eye tracking.

stereo system, Videre Design's MEGA-D stereo head [15]. With 4.8 mm lenses, it has a horizontal FOV of 73°, wide enough to cover close to 1 m of horizontal head motion when the head is 600 mm away from the camera. As we will explain in section 5, heads are detected using a combination of stereo and intensity-based processing. The wide angle system steers the narrow FOV cameras when a new head is acquired for tracking.

Two narrow field cameras, positioned near the lower monitor corners (Fig. 1), capture high resolution images of the eye for gaze tracking. Due to the narrow field of view, quick head motions would outpace pan-tilt heads. Thus, pan and tilt are controlled using rotating mirrors on high performance galvos that can rotate and settle in 2 ms (Fig. 2). The motion of the galvos is synchronized with the refresh cycle of the cameras, so galvo motion causes no image blur (head and eye motion still does, however). The camera has a 49 mm lens and is a standard NTSC CCD camera. The long focal length substantially reduces the depth of focus, so focusing is an issue. Thus, the lens is translated along the camera's optical axis by a motor; using a thin lens model will be discussed in section 4 to calibrate and control focusing. Finally, as in Morimoto, *et. al.* [11], a ring of IR LEDs is placed around the lens. However, this is not used in their differential lighting scheme to locate pupils, but to produce corneal glints for constraints in tracking.

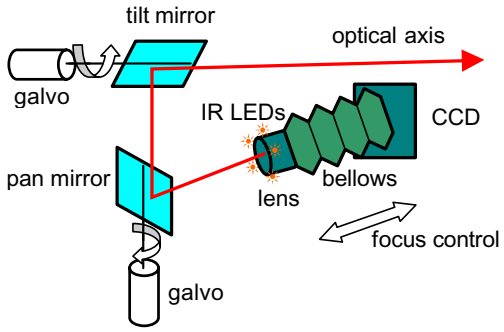

Figure 2: Our actively controlled narrow FOV camera. Pan and tilt control rotates the galvo mirrors, and focus control translates the lens along the camera's optical axis.

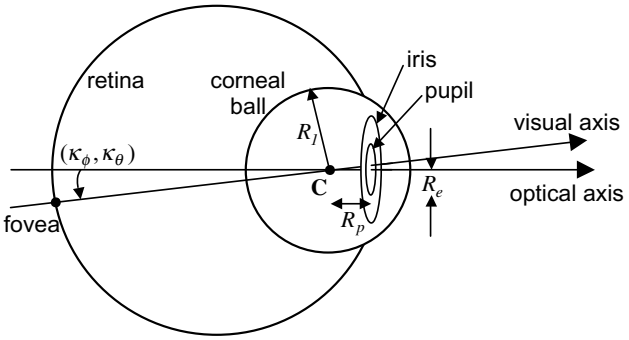

Figure 3: Our 3D eye model. Please refer to the text for details.

3 Eye modeling

Our tracking system uses a parameterized 3D model of the eye. Tracking basically consists of simultaneously fitting projected model features to extracted image features in the two narrow FOV cameras. The main features being tracked are the pupil edges and corneal glints, or reflections, of the camera LEDs, so our model should be sufficiently detailed to predict these features. Our 3D eye model is outlined in Fig. 3.

The cornea is modeled as either a sphere or an ellipsoid with center **C**. Modeling the cornea is necessary since the IR glints are reflected off of it and the pupil edges are refracted through it. For refraction, we use an index $n = 1.34$ [9] at the cornea/air boundary. Our earlier modeling attempts began with using sphere to model the cornea, using a radius R_1. However, the cornea is not an exact sphere; for instance, one of the causes of myopia is a bulging or steepening of curvature radius of the cornea. Thus, we have also investigated modeling the eye as an ellipsoid with radius R_1 along the optical axis and R_2 along the remaining

2 directions. Thus, for a subject with myopia, one would expect $R_1 > R_2$.

The pupil is modeled as a circle at a distance R_p along the optical axis from $\mathbf{C}$ and with a radius of R_e. The locations of pupil edges are refracted though the cornea and projected into the image.

The gaze direction as indicated by the pupil (the so-called *pupillary axis*, which we assume is equal to the optical axis for our model) is not the same as the *visual axis*. The visual axis passes through the fovea, the densest area of photoreceptors in the retina, and is a true measure of where an eye is looking. Thus, the pupillary axis estimated from our model is rotated by spherical angles $(\kappa_\phi, \kappa_\theta)$ about $\mathbf{C}$ to estimate the visual axis.

Given an estimate of the visual axis, the final step in the gazetracker is to intersect the visual axis with the monitor plane and map to screen coordinates. This step requires the monitor and eye to be in the same coordinate system. In the next section we discuss coordinate systems and we describe the task of calibration.

4 Calibration

Camera calibration is necessary to relate 3D models to their image projections and to relate different 3D coordinate systems via a rigid transform; i.e. from wide angle stereo to narrow FOV stereo. Our overall system has three main coordinate systems:

1. *Narrow FOV stereo.* Calibrating this stereo system adds the complication of rotating mirrors to the normal stereo calibration problem. This serves as the "base" coordinate system that the others are related to.

2. *Wide angle stereo.* This is calibrated using SRI International's Small Vision System software [10].

3. *Monitor coordinates.* This is the local coordinate system of the monitor.

In this section, we explain how the narrow FOV system is calibrated and how rigid transforms relating systems (1) to systems (2) and (3) are estimated. In addition, we explain how the focus calibration and control works.

4.1 Narrow FOV stereo system

While most stereo systems have a fixed baseline between left and right cameras, our active narrow FOV system is constantly in motion, and hence continually changing the transform between left and right cameras. In this subsection, we discuss how this calibration is parameterized in terms of galvo rotation angles. A related class of stereo systems from the robotics community, head-eye systems, often face the same calibration problem. These systems typically build binocular stereo systems from pan-tilt heads, as opposed to our use of galvos and mirrors. But a larger distinguishing factor of our system is its narrow FOV for capturing a high resolution image of the eye. This difficulty is addressed in this section by using a large but highly detailed calibration target, a large number of views, and using 2D image dotcode for corresponding image features to target model features.

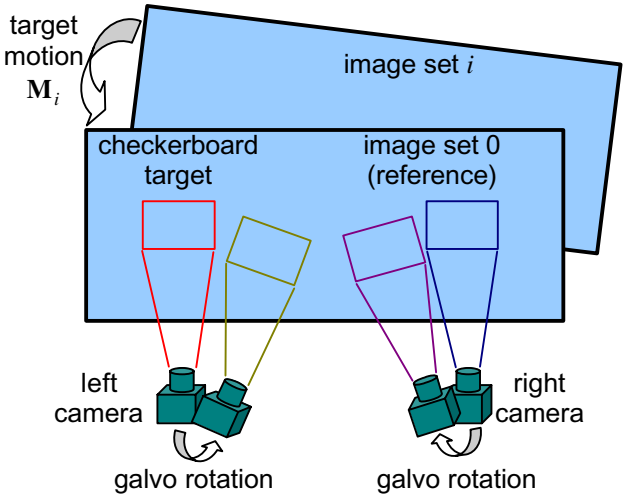

Figure 4: When calibrating the narrow FOV stereo system, multiple image sets are taken of the calibration target. For each image set, the calibration target is fixed and a number of views are taken under different pans and tilts. The pose of the target in the ith image set is defined as a motion $\mathbf{M}_i$ to the target in image set 0, the reference image set.

Our system is a bundle adjustment approach using multiple views of a planar calibration target. Recently planar calibration targets have become popular for monocular [18] and stereo calibration [19]. Our target is a large checkerboard target that is visible by both cameras (Fig. 4). Due to the small FOV, the checkerboard squares are only 3mm on the side. Covering the potential area visible under all pans and tilts requires a large pattern, on the order of 150 by 100 squares. As Fig. 5 shows in the sample target image, only a small portion of the target is visible in any particular view. This creates a feature correspondence problem, for which model features are within the field of view? To address this issue, we add a 2D dotcode (Fig. 5, right) to correspond features to their 3D model features.

Given 2D image features in correspondence with their 3D calibration target features, we would like to construct a projection matrix $\mathbf{P}$ to map the 3D target to images. $\mathbf{P}$ will be parameterized by the galvo rotations θ^L and θ^R; for the

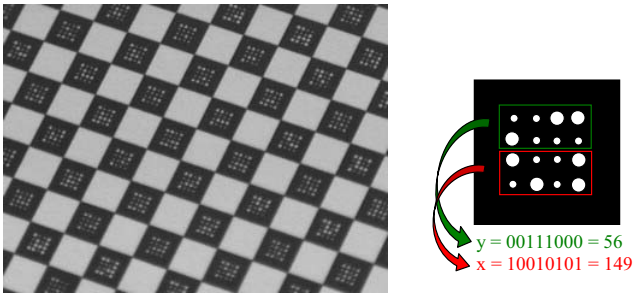

Figure 5: Example image of calibration target and example checkerboard square of our dotcode pattern for (x, y) location within the target.

left camera, we have

$$P(\theta^L) = \mathbf{A}^L \mathbf{T}^L(\theta^L),$$

where $\mathbf{A}^L$ is a 3x3 matrix of intrinsics and $\mathbf{T}^L(\theta^L)$ is a 4x4 matrix of the extrinsics. Fix one view of the target as a reference view and translate all galvo angles such that $\theta^L = \mathbf{0}$ at the reference. Then we have

$$\begin{aligned} \mathbf{P}(\mathbf{0}) &= \mathbf{A}^L \mathbf{T}^L(\mathbf{0}) \\ &= \mathbf{A}^L \mathbf{T}^L \end{aligned}$$

where $\mathbf{T}^L$ is a simple 6 DOF rigid transform.

The projection matrix of other views will be created by modifying the reference. Given a new view where only the galvo mirrors θ^L have changed, we have

$$\mathbf{P}(\theta^L) = \mathbf{A}^L \mathbf{G}^L(\theta^L) \mathbf{T}^L,$$

where $\mathbf{G}(\theta^L)$ represents the mirror rotation and is described the appendix. When focus control translates the lens to maintain focus (to be described in section 4.4), this z translation $\mathbf{F}^L$ is inserted as

$$\mathbf{P}(\theta^L) = \mathbf{A}^L \mathbf{G}^L(\theta^L) \mathbf{F}^L \mathbf{T}^L.$$

Finally, views are grouped into *image sets*, views of the target at a particular pose. The reference views are from image set 0, the reference image set. As shown in Fig. 4, the pose of the target in the ith image set is represented as a motion $\mathbf{M}_i$ from the target to its pose in the reference image set. Thus, the complete projection matrix is

$$\mathbf{P}^L(\theta^L, i) = \mathbf{A}^L \mathbf{G}^L(\theta^L) \mathbf{F}^L \mathbf{T}^L \mathbf{M}_i, \qquad (1)$$

where $\mathbf{M}_0 = \mathbf{I}$.

In bundle adjustment, for each image set i, we have a set of views j, and each view j has a number of features k. For 2D image feature $f_{i,j,k}$ in correspondence with 3D target feature $Q_{i,j,k}$, first project the model feature

$$\mathbf{q}_{i,j,k} = P\left[\mathbf{A}^L \mathbf{G}^L(\theta^L_{i,j}) \mathbf{F}^L \mathbf{T}^L \mathbf{M}_i \mathbf{Q}_{i,j,k}\right]$$

where P is homogeneous divide for projection, and then $\mathbf{f}_{i,j,k} \approx \mathbf{q}_{i,j,k}$. Our calibration procedure uses a nonlinear optimization on the error function

$$\sum_i \sum_j \sum_k \|\mathbf{f}_{i,j,k} - \mathbf{q}_{i,j,k}\|$$

taken over both the left and right camera views. Our calibration uses 3 image sets containing a total of 71 left views and 70 right views. Feature correspondence is completely automatic, using an X junction detector to find the square corners and the dotcode to place them within the 3D planar target.

In summary, given galvo angles θ^L and θ^R, we can compute projection matrices $\mathbf{P}^L$ and $\mathbf{P}^R$ for the target poses in the image sets. In the runtime system, we simply use the reference image set $i = 0$ as the narrow FOV coordinate system. As with a normal stereo system, the computed left and right projection matrices can be used for tasks such as 3D reconstruction, computing the fundamental matrix, or computing a rectifiying homography. For 3D reconstruction, this reconstructs points in the coordinate system of the reference pose of the calibration target, but this *is rigidly attached* to the galvo system, so this is fine.

4.2 Wide angle stereo system

As previously mentioned, the wide angle system is calibrated using SRI's Small Vision System [10], so the remaining task is to find a rigid transform $\mathbf{R}_w$ from the wide angle system to the narrow FOV system. Computing $\mathbf{R}_w$ is straightforward since both the wide and narrow stereo systems are calibrated. We simply track a point feature (manual initialization) in both 3D systems and record a set of corresponding reconstructed 3D points. Then the best rigid transform $\mathbf{R}_w$ aligning the set of points in 3D is found by minimizing the 3D distance between the narrow FOV points and $\mathbf{R}_w$ applied to the wide angle points. We have performed this 3D alignment a few times on a dataset of around 10 points, and the residual alignment error is on the order of a few mm. The transform $\mathbf{R}_w$ will be used in the tracking system to map wide angle face detections to the narrow FOV system and steer the galvos.

4.3 Monitor

In monitor calibration, the goal is to determine the pose of the monitor plane in the narrow FOV coordinate system. At first this seems challenging since the monitor is not visible from the narrow field cameras, but we improvised by placing a mirror to reflect the monitor image into the narrow field cameras. The same checkerboard pattern with dotcodes is taped to the monitor plane and it's "virtual" pose is computed in the reflected image. To extract the "real"

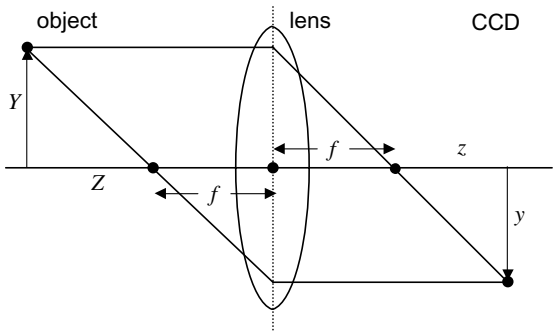

Figure 6: The thin lens equation can be used to relate focal length f, distance to object Z, and the back focal distance z.

pose from the virtual one, the plane of the mirror needs to be estimated, and then a reflection transform can be composed with the virtual pose (see appendix A). The mirror plane is estimated by placing the planar calibration target at the same location of the mirror. In the eye tracking system, the 3D foveal axis is intersected with the monitor plane to estimate the eye gaze point.

4.4 Focus

One technique for auto focusing is to vary focus while maximizing high frequency content of the image. Since we have 3D information available to us, however, we can use the thin lens equation to relate object depth Z to the focal length f and the back focal distance z (see Fig. 6). Using similar triangles, one can show that

$$y = \frac{fY}{Z}, \qquad \text{and} \qquad f^2 = zZ.$$

The latter equation is of use in focus control. In the eye tracking system, Z is supplied from the wide angle tracking, and f is estimated during calibration, making it straightforward to estimate z at run time. (During calibration, Z is estimated from the 3D calibration target model.) To calibrate the focus control, we estimate a linear mapping from z to the focus motor ticks from two example focused images at known depths Z.

5 Detection and Tracking

Our eye gaze tracking system detects the computer user's face using the wide angle stereo system, and then the narrow FOV is steered onto the user's right eye to begin high resolution eye tracking and gaze point estimation. In this section, we discuss face detection, eye tracking, and some quantitative experiments on the accuracy of the gaze point estimates.

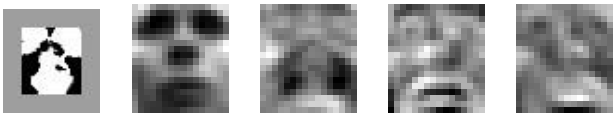

Figure 7: Stereo template used to detect face shapes in the disparity map (left), and eigenspace (right) for tracking and localizing faces in the intensity image.

5.1 Wide Angle Detection

The goal of the wide angle stereo processing is to detect the face, estimate the 3D location of the user's right eye, and pass this information on to the narrow FOV system. It is advantageous to use stereo for this face detection/eye localization task, as opposed to competing techniques such as color segmentation, background subtraction, and motion. While all these techniques can quickly segment out the face from the background, only stereo can estimate the location of the eye in 3D, a requirement in steering the narrow FOV system.

Our wide angle tracking system uses eigenfeatures[12] and is very close to eigentracking[13]. Before the face is detected, the range map is scanned at a coarse spatial resolution (i.e. coarse pyramid level) for shape blobs that resemble the interior features of the face. This is performed in a scanning operation in depth: stereo disparities are thresholded in a small depth range, say 10 cm, and then the binary mask is correlated with the stereo mask in Fig. 7. Regions scoring high in this correlation test are further examined with an eigentracking technique. The face region being tracked is extracted using an affine transform and matched against an eigenface model shown on the right in Fig. 7. Face matching is done in a coarse-to-fine manner, producing the final tracking results in Fig. 8.

Once in tracking mode, the eigen model is the primary tracking mechanism. Stereo is used for robustness, effectively monitoring the tracking process and eliminating the track if it slips off the face. For in this case, the face region in the thresholded range map will no longer match the stereo mask initially used for detection.

Given the 3D location $\mathbf{P}_{wide}$ of the right eye in the wide angle system, this is mapped to the narrow system using $\mathbf{P}_{narrow} = \mathbf{R}_w \mathbf{P}_{wide}$. Steering angles θ^L and θ^R are estimated using nonlinear optimization such that the projection of $\mathbf{P}_{narrow}$ is in the image center of both narrow field cameras.

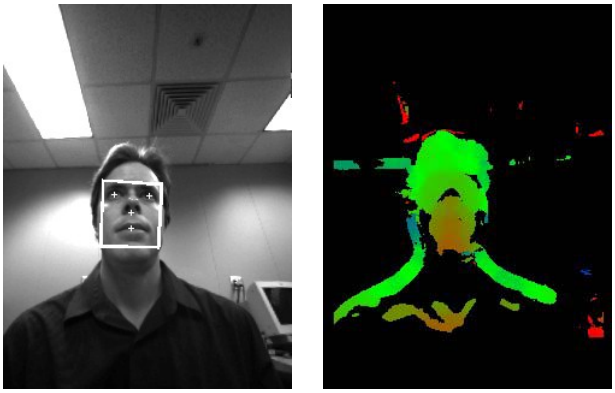

Figure 8: Results of wide angle tracking (left) and color-coded disparity map (right).

5.2 Narrow FOV Tracking

The wide angle system has steered the narrow FOV cameras onto the eye, and the focusing mechanism from section 4.4 has brought the left and right input images into focus. The goal of the narrow field tracking system is to fit projected model features from our 3D eye model to detected image features. Then the visual axis is intersected with the monitor plane and the gaze point reported.

The parameters from the 3D eye model in section 3 can be divided into three groups for narrow FOV tracking:

1. *Eye extrinsics.* 3D eye position $\mathbf{C}$ and optical axis $\mathbf{N}$. $\mathbf{N}$ is a direction and is parameterized using 2D spherical coordinates.

2. *Fixed eye intrinsics.* These parameters of the eye should remain fixed during a tracking session, but may change slowly over the years. They include corneal ball radii R_1 and R_2, foveal offsets κ_ϕ and κ_θ, and the pupil radial offset R_p.

3. *Variable eye parameters.* These parameters change the shape of the eye model and include the pupil radius R_e.

Under normal tracking conditions (after the user's eye intrinsics have been calibrated), only the parameters in (1) and (3) would be allowed to vary during the 3D model fit. User calibration is required to estimate the parameters in (2), and thus far we have divided the estimation of the intrinsics into two stages. The first stage estimates parameters R_1, R_2, and R_p solely from a set of input images – the model fit drives the parameter estimation. However, the foveal offsets are not observable from the input images, so estimating these parameters requires feedback from the user. We describe this step in more detail in the experiments section below.

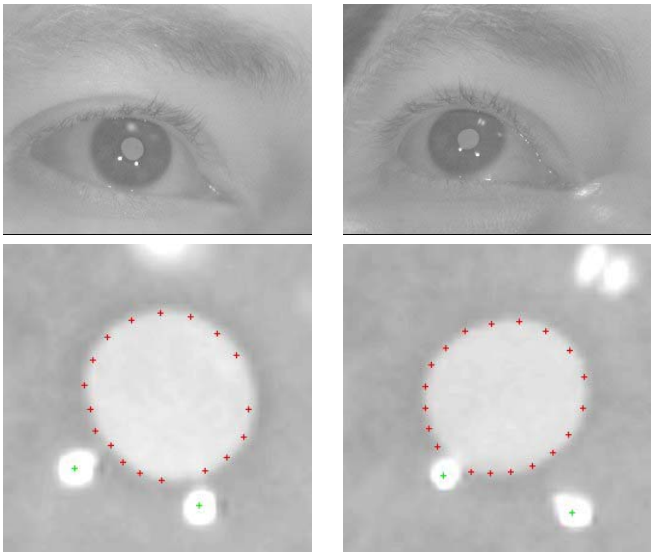

Figure 9: Feature detection of glints (green) and pupil edges (red) in narrow FOV tracker, left and right views of right eye.

The image features used for matching to the 3D model include the glints of the IR LEDs and the pupil edges. To detect them in the narrow field images, first we coarsely estimate the center of the pupil. This is done using an iterative affine registration technique that registers a synthetic sclera/iris/pupil template to the input images. It operates in a coarse-to-fine manner and uses motion templates [14] to update the affine transform. This estimates the pupil center within a few pixels, which is sufficiently accurate for the next step that collects edge pixels for the pupil and glint boundaries.

For the glint boundaries, we search the iris/pupil region using vector correlation with a gradient template matched to the expected circular, inward-pointing glint gradient. Once the glints are roughly located, their location is refined by (a) radially searching for the glint edges, (b) estimating them to subpixel resolution, (c) fitting an ellipse to the edge pixels [17], and (d) reporting the center of the ellipse. The detected glint edges are masked out of the image before the pupil edges are located. The pupil edges are found in a similar manner, but the ellipse fitting step in (c) is made more robust by using a leave-n-out strategy. A number of ellipses are fit to the data; for each ellipse, four of the potential pupil points are left out as potential noise. The best fitting ellipse and its nearby supporting points are reported as the pupil edges. Fig. 9 shows the estimated glint centers and pupil points for a sample stereo pair.

Finally, the 3D model is fitted to the detected features using a nonlinear estimation technique, a version of the

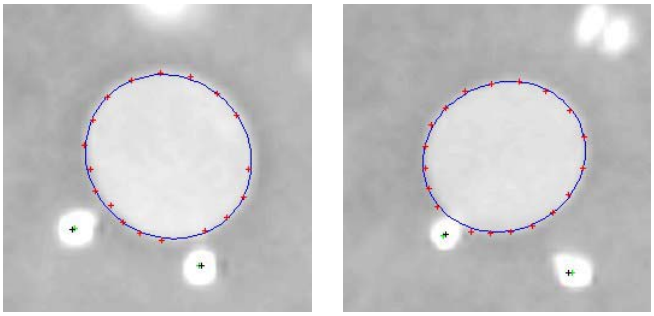

Figure 10: 3D eye model fit displayed with features from Fig. 9. The pupil ellipse is shown in blue and the glint projections are in black.

Levenberg-Marquardt algorithm from MINPACK. When the face and eye are first detected, the initial conditions for the model center **C** are set from the wide angle tracking, and the initial conditions for the optical axis is pointing along the z-axis. When in tracking mode, the initial conditions are simply the previous model state; future work includes incorporating a Kalman filter framework into the tracker. The constraints for the MINPACK routine include (a) the image distance between model and feature glints, and (b) the distance from each pupil edge feature to the model ellipse projected into the image. Fig. 10 shows the model fit for the input image in Fig. 9.

In terms of processing speed, both the wide and narrow FOV tracking system run at a frame rate of roughly 10 Hz on a dual processor Pentium III at 933 MHz. We are in the process of upgrading to a 2.8 GHz Xeon dual processor machine, and we anticipate a frame rate of 20 Hz on the new machine. This is fast enough for our current target study area of evaulating readers/learners on the web, as humans fixate on words for roughly 200-250 ms when reading[20].

5.3 Experiments

5.3.1 User Calibration - R_1, R_2, and R_p

In the 3D fitting process, the instrinsic parameters R_1, R_2 and R_p can be included as variables and estimated along with the eye extrinsics. However, since these parameters are fixed, one can increase their accuracy by fitting a set of eye stereo pairs simultaneously. We took a set of 18 stereo pairs of author DB's right eye, where DB gazed at 9 points on the monitor screen from two different head positions. The 9 points were distributed mostly along the edges of the monitor where one would expect fitting error to be higher. The intrinsics were estimated as follows: R_1 = 9.08 mm, R_2 = 9.02 mm and R_p = 5.5 mm, with an average image error of 1.6 pixels. Note that the user does not need to look at specific screen locations to perform this intrinsic calibration.

5.3.2 User Calibration - foveal angles

The foveal angles are not observable from the input images, so one needs to use feedback from the user to estimate these parameters. To estimate the foveal angles, the user is asked to gaze at specific monitor points to collect stereo pairs with ground truth on the gaze point. The error metric in the nonlinear fitting code is then switched to measure error between the estimated gaze point and ground truth data. In addition to estimating foveal angles κ_ϕ and κ_θ, we also include a refinement of the monitor pose. The foveal angles were estimated for author DB, and the angle κ_ϕ estimated between the optical and visual axes is about 3.6°, which close to the expected value of 5°. The other angle κ_θ seems to be less stable with higher variance on its estimation; stabilizing this parameter is the subject of future work.

5.3.3 Gaze point accuracy

To estimate gaze point accuracy, we took a test data set of author DB looking at 22 known locations on the monitor and evaluated the error between ground truth and reported gaze. Overall the average fit error in the eye model dropped to 1.1 pixels, probably because more of the test points were in the central area of the monitor. The average monitor error is 18.8 pixels = 6.6 mm for our monitor. Person DB was estimated to be 622 mm away from the monitor, so this translates into 0.6° accuracy in gaze direction.

6 Summary

In this paper we have developed an eye gaze tracking system that allows for freedom of head motion without the need for wearing any camera devices. The user's head is acquired by a wide angle stereo system that detects the face, and the detected eye location is used to steer an active narrow FOV stereo system onto the eye. The eye is modeled with a 3D model that includes the corneal ball, pupil, and fovea – the most complete eye model used in the computer vision community. After fitting this model to the two stereo narrow field views, the visual axis is intersected with the monitor plane to estimate the gaze point. Experiments have been performed to calibrate the instrinsic parameters of the eye, and the resulting system achieved a gaze point accuracy of 0.6° on a set of 22 stereo pairs.

References

[1] S. Zhai, C. Morimoto, and S. Ihde. Manual And Gaze Input Cascaded (MAGIC) Pointing. In Proc. CHI99: ACM Conference on Human Factors in Computing Systems. Pittsburgh, May, 1999, pp. 246-253.

[2] P. Maglio, R. Barrett, C. Campbell, and T. Selker. SUITOR: An attentive information system. In H. Lieberman, editor, IUI2000: International Conference on Intelligent User Interfaces, ACM, New York, 2000, pp. 169-176.

[3] LC Technologies, Inc. The Eyegaze Development System. www.eyegaze.com

[4] SensoMotoric Instruments. EyeLink Gaze Tracking. www.smi.de

[5] Seeing Machines. faceLAB. www.seeingmachines.com

[6] R. Stiefelhagen, J. Yang, and A. Waibel. Tracking Eyes and Monitoring Eye Gaze. In *Proc. Workshop on Perceptual User Interfaces (PUI'97)*. Alberta, Canada. 1997, pp.98-100.

[7] K.-H. Tan, D. Kriegman, and H. Ahuja. Appearance-based Eye Gaze Estimation. In *Proc. IEEE Workshop on Applications of Computer Vision (WACV'02)*. pp. 191-195.

[8] T. Ohno, N. Mukawa, A. Yoshikawa. FreeGaze: A Gaze Tracking System for Everyday Gaze Interaction. In *ACM Eye Tracking Research and Applications*, New Orleans, LA, March, 2002, pp. 125-132.

[9] R. Rabbetts. Clinical Visual Optics. Butterworth-Heinemann Ltd, Oxford, UK.

[10] SRI Small Vision System, Calibration Supplement, July, 2001. *www.videredesign.com/calibrate.pdf*

[11] C. Morimoto, D. Koons, A. Amir, and M. Flickner. Pupil detection and tracking using multiple light sources. *Image and Vision Computing*, 18(4):331-335, 2000.

[12] A. Pentland, B. Moghaddam, and T. Starner. View-Based and Modular Eigenspaces for Face Recognition. In *Proc. CVPR*, Seattle, WA, 1994, pp. 84-91.

[13] M. J. Black and A. Jepson. EigenTracking: Robust matching and tracking of articulated objects using a view-based representation. In *Proc. ECCV*, April, 1996, pp. 329-342.

[14] G. Hager and P. Belhumeur. Real-Time Tracking of Image Regions with Changes in Geometry and Illumination. In *Proc. CVPR*, June, 1996, pp. 403-410.

[15] Videre Design. MEGA-D Stereo Camera. www.videredesign.com

[16] K. Konolige. Small Vision Systems: Hardware and Implementation. Eighth International Symposium on Robotics Research, Japan, October 1997.

[17] A. Fitzgibbon, M. Pilu, and R. Fisher. Direct Least Squares Fitting of Ellipses. *IEEE PAMI*, vol 21, no 5, pp. 476-480, 1999.

[18] Z. Zhang. A Flexible New Technique for Camera Calibration. *IEEE PAMI*, vol 22, no 11, pp. 1330-1334, 2000.

[19] Jean-Yves Bouguet. Camera Calibration Toolbox for Matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/

[20] K. Rayner and A. Pollatsek. The Psychology of Reading. Lawrence Erlbaum Associates, Publishers, Hillsdale, NJ.

A Galvo Rotations

Given galvo rotations θ^L, how does one construct the transformation matrix $\mathbf{G}^L(\theta^L)$ representing the rotation of the galvo mirrors used in equation (1)? First, consider the transformation of a reflection in a planar mirror. Let the equation of the mirror plane be

$$\mathbf{n} \cdot (x, y, z)^T + d = 0$$

where $\mathbf{n}$ is the plane normal and d its distance to the origin. The mirror reflection transform $\mathbf{D}$ is

$$\mathbf{D} = \begin{bmatrix} \mathbf{I} - 2\mathbf{n}\mathbf{n}^T & 2d\mathbf{n} \\ \mathbf{0} & 1 \end{bmatrix}.$$

In the galvo, the mirror plane is rotating an angle θ about the galvo's 3D axis. Thus, $\mathbf{n}(\theta)$ and $d(\theta)$ can be written as a nominal $\mathbf{n}_0$ and d_0 rotated about the galvo axis, and likewise, $\mathbf{D}$ is a function of θ, $\mathbf{D}(\theta)$.

The overall transform $\mathbf{G}$ is the composition of two reflections for the pan and tilt mirrors. Since the reference transform $\mathbf{T}^L$ is a "virtual" camera position, we first apply the reflection at $\theta = 0$ to "undo" the reflection at the virtual position. Then we reapply the reflection at the given θ. This yields

$$\mathbf{G}(\theta^L) = \mathbf{D}_0(\theta_0^L)\mathbf{D}_0(0)\mathbf{D}_1(\theta_1^L)\mathbf{D}_1(0)$$

where subscripts 0 and 1 are for pan and tilt, respectively.