POINTING WITH THE EYES: GAZE ESTIMATION USING A STATIC/ACTIVE CAMERA SYSTEM AND 3D IRIS DISK MODEL

Michael Reale, Terry Hung, and Lijun Yin

Department of Computer Science, State University of New York at Binghamton
aragorn358@verizon.net, hungkf@corning.com, lijun@cs.binghamton.edu

ABSTRACT

The ability to capture the direction the eyes point in while the subject is a distance away from the camera offers the potential for intuitive human-computer interfaces, allowing for a greater interactivity, more intelligent HCI behavior, and increased flexibility. In this paper, we present a two-camera system that detects the face from a fixed, wide-angle camera, estimates a rough location for the eye region using an eye detector based on topographic features, and directs another active pan-tilt-zoom camera to focus in on this eye region. We also propose a novel eye gaze estimation approach for point-of-regard (PoG) tracking on a large viewing screen. To allow for greater head pose freedom, we developed a new calibration approach to find the 3D eyeball location, eyeball radius, and fovea position. Moreover, we map both the iris center and iris contour points to the eyeball sphere (creating a 3D iris disk) to get the optical axis; we then rotate the fovea accordingly and compute our final, visual axis gaze direction. We intend to integrate this gaze estimation approach with our two-camera system, permitting natural, non-intrusive, pose-invariant PoG estimation in distance and allowing user translational freedom without resorting to infrared or complex hardware setups such as stereo-cameras or "smart rooms."

Keywords— Eye tracking, eye gaze estimation, static/active dual-camera system

1. INTRODUCTION

The ideal human-computer interaction system should function robustly with as few constraints as those found in human-to-human interaction. One of the most effective means of interaction is through the behavior of the eye. Specifically, knowledge of the viewing direction and thus the area of regard offers insight into the user's intention and mental focus, and consequently this information is vital for the next generation of user interfaces [21][22]. Applications in this vein can be found in the fields of HCI, security, advertising, psychology, and many others [5][21].

As such, there has been intensive research on eye tracking and gaze estimation for the past 30 years [5]. However, with the conventional single-camera system, either the user must keep his or her head locked in place (so that it remains within the narrow field of view), the camera must be strapped to the subject's head (a common approach for eye detection), or some other marker (like a small infrared light) must be worn by the subject [2]. The resulting situation can be prodigiously inconvenient and uncomfortable for the user. Because of the unique reflective properties of the pupil to infrared (IR) light, some systems have opted to detect the face by first finding the eyes [2][10][11]. While robust to visible light conditions, these methods have issues with changes in head pose, reflections off glasses, and even decreased reflectivity of the retina from contact lenses [9]. An alternative approach is to use stereo cameras [2][11]; while robust, these systems generally require substantial calibration time. Some complex systems (like "smart rooms" [1]) require expensive and/or non-portable setups. Even existing, non-stereo, two-camera systems [16] often restrict the user to a preset location in the room.

Therefore, we propose a system that first robustly locates the face from a wide-angle camera view (as well as a rough approximation of the eye positions) and uses this information to guide a pan-tilt-zoom camera to focus in on the eye area of the detected face. This allows the user translational freedom relative to the camera, and a simple adjustment of parameters also permits varied movement in depth freedom. Our system differs from the majority of other systems in this vein in that we do not use infrared, stereo-cameras, specially-constructed hardware (like Noureddin et al. [10]), or specific room setups (i.e. "smart rooms"). We intend to leverage this system for gaze and PoG (Point-of-Regard) estimation.

The overwhelming majority of gaze estimation approaches rely on glints (the reflection of light off the cornea) to construct 2D or 3D gaze models [5]. Alternatively, eye gaze may be determined from the pupil or iris contours [18] using ellipse fitting approaches [3][15]. One can also leverage the estimated iris center directly and use its distance from some reference point (e.g., the eye corners) for gaze estimation [12][19]. Indeed, the entire eye

978-1-4244-7493-6/10/$26.00 ©2010 IEEE

region may be segmented into the iris, sclera (white of the eye), and the surrounding skin; the resulting regions can then be matched pixel-wise with 3D rendered eyeball models (with different parameters) [17][20]. However, different subjects, head pose changes, and lighting conditions could significantly diminish the quality of the segmentation [20].

To bypass the issue of the diminished accuracy of iris centers from visible light imagery while simultaneously avoiding the problems heir to the potential instability of iris contour extraction, we leverage both to determine eye gaze direction with a 3D eyeball model. We calculate the 3D eyeball centers, radii, and fovea points through a unique calibration approach. During gaze estimation, we map both the estimated iris center and the iris contour points onto the eyeball sphere. After converting these points into vectors starting at the eyeball center, we then use a least-squares approach to find a vector that is at a known angle from the contour vectors while also being approximately in line with the center vector. We then find the current position of the fovea, which is dependent on the rotation of the eye, compute a new iris center from our initial gaze vector, and use the vector from the fovea to this new iris center as our final gaze direction. Our approach differs from the majority of previous techniques, in that we map the 2D contour pixels onto the known 3D eyeball; in contrast, most approaches in this vein project the 3D model into the 2D plane.

We intend to integrate the aforementioned camera system with our gaze and PoG (Point-of-Regard) estimation approach. Section 2 will describe each component of our system individually, Section 3 will present the results from our gaze estimation tests, and Section 4 concludes the paper.

2. SYSTEM OVERVIEW

The complete system consists of five different stages: face detection, eye area detection (wide-angle view), camera control, iris detection (narrow-angle view), and gaze calibration/estimation (narrow-angle view). The first three are part of one program, while the last two are part of another; these two programs run on separate computers. We use an IEEE 1394 camera for the wide-angle view; images from this camera are used for face detection. Once a face and eye area center are detected, the detection program controls a SONY SNC-RZ30N camera, which can pan ±170° and tilt from -90° to +25°. It also has a 25x optical zoom. This camera gives us a close-up view of the face (640 x 480 images at approximately 15 to 30fps), which can then be used by our eye detection program. Figure 1 depicts the system composition, while Figure 2 shows the system in action. In the final integrated system, our gaze estimation work will use the active camera view, and the iris centers and contours will be extracted from the active camera image stream. Then, after gaze calibration, the gaze direction and starting point (i.e. eyeball centers) will be estimated and then mapped to screen coordinates.

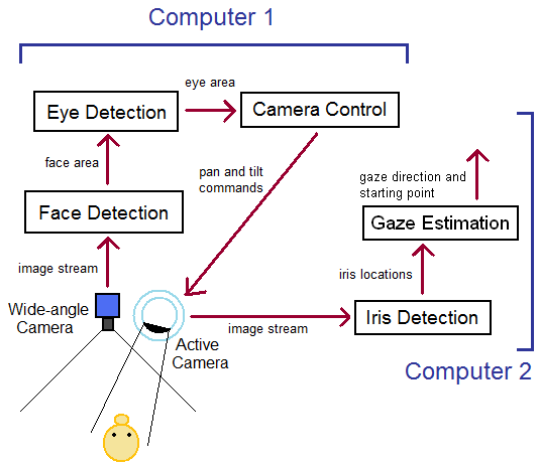

Fig. 1. System overview.

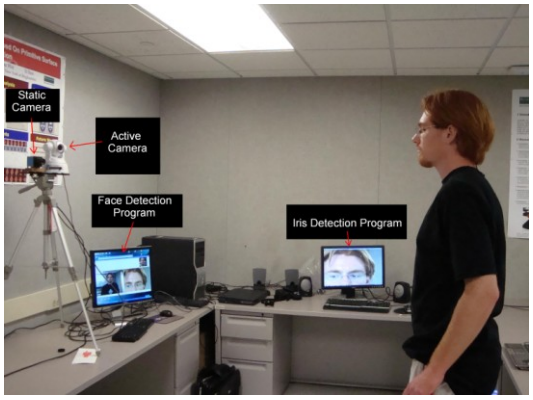

Fig. 2. System in action.

2.1. Face detection

We applied an appearance-based technique based on the work by Viola and Jones [14] for face detection. To improve the robustness of the face detector, we have developed a novel, linear-time mechanism to make the system invariant to variable lighting conditions: all features are scaled by a ratio of the average gray-level intensity of the training samples over the average gray-level intensity of the current search region. We use the integral image to efficiently compute feature block intensity levels. The best Haar-like features are selected with Adaboost; we then used Information Gain in the Gain Ratio as a metric of usefulness. The details of the developed face detector can be found in [7].

2.2. Eye detection (from the static camera view)

Once the face is detected, we also perform eye detection using the so-called topographic features of eyes presented in our previous work [15]. The basic idea is to create a terrain map from the gray-scale image (effectively treating it like a continuous 3D surface) and extracting "pit" locations as

pupil candidates. The actual eye pair is chosen using a Gaussian Mixture Model (GMM) and certain heuristics (e.g. the eye candidates cannot be vertical). The approximate center of the two detected eye candidates is used as the focus of the pan-tilt-zoom camera. Note that we do not use the mutual information tracking approach presented in [15] and instead detect the eyes once every time the face is detected in the wide-angle view.

2.3. Iris detection (from the active camera view)

The above stage provides an approximate localization of eyes, allowing the active camera to zoom into the region of interest. To detect the accurate positions of irises, we propose a fine detection approach as follows.

We work on the red channel of the image, since the iris (unlike the surrounding skin areas) generally has very low red intensity values [13]. We search for "highlight positions" on the eye (corresponding to specular reflections off the iris); these are regions containing enough dark pixels to be considered part of the iris while still having a high enough standard deviation to contain specular reflection points. To remove distracting points on glasses, we eliminate specular reflections that are long and thin (unlike iris highlights, which are generally equal in width and height). Unlike Vezhnevets and Degtiareva [13], we equalize the histogram within the eye region to increase contrast. To determine the proper thresholds and parameters, we extract information from 7 video sequences we collected using the active camera. Finally, we refine our initial iris estimate using a circular template in a very small area around the initial estimate.

2.4. Gaze calibration (from the active camera view)

Our calibration procedure has two stages: the first acquires an estimate for the 3D eyeball center and radius, and the second iteratively refines our initial estimate while also extracting the position of the fovea on the eyeball surface. Note that each eyeball is handled independently during calibration.

2.4.1. Stage 1 calibration (initial estimate)

At this point, we can get the 2D iris positions, and we assume the camera's focal length is known; given a 3D eyeball radius r, we want to find an estimate for the 3D eyeball center E.

Let us consider the camera center as the origin of a 3D coordinate system and the camera view direction as the z axis. Given a 2D pixel position and the camera's focal length, we can get a 3D vector from the camera center through every 3D position that maps to our given 2D pixel point. This can be done simply by constructing a "world image plane," a 3D plane which is perpendicular to the camera's view axis and is far away enough such that 3D points in the plane have the same x and y coordinates as their projected 2D points. A 2D pixel point can then be converted to a 3D vector by setting its z coordinate to be the z depth of the world image plane.

We operate under the assumption that, if the user is looking straight into the camera, the eyeball center must be somewhere along a 3D vector starting at the camera center and going through the 2D iris center; that is, the gaze direction goes from the 3D eyeball center E to the 3D iris center E' (this vector is known as the "optical axis"). Granted, this is not strictly true, since the correct gaze direction (called the "visual axis") is a vector from the fovea (on the back of the eye) through the pupil [5], and this visual axis is offset from the optical axis by approximately 4-8° [4]. However, we will correct for this in Stage 2.

The user first looks into the camera and then at a calibration point P whose 3D location is known. This gives us 2D iris locations $m1$ and $m2$, respectively. These points are then converted into normalized vectors A and B. We want to find two points, E (the eyeball center) along vector A and E' (the iris center) along vector B, such that the distance between them is r (the radius of the eyeball) and the vector between them points at P. Figure 3 illustrates this idea.

Let t_a and t_b represent the lengths such that $At_a = E$ and $Bt_b = E'$; we can then express our first constraint as follows:

$$\left\| At_a - Bt_b \right\| = r \tag{1}$$

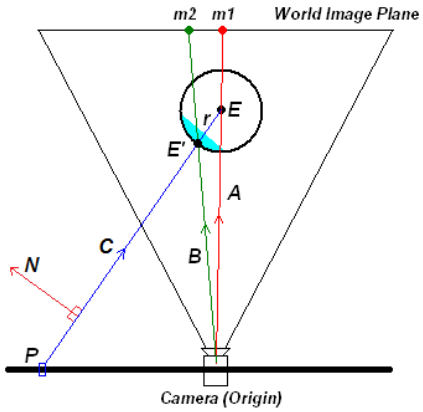

Fig. 3. Stage 1 calibration.

Point E' (or Bt_b) is the point of intersection between $C = At_a - P$ and B. This is equivalent to the point of intersection between B and a plane determined by point P and normal $N = (A \times B \times C)$. So, our second constraint is as follows:

$$t_b = \frac{N \bullet P}{N \bullet B} \tag{2}$$

Given above equations (1) and (2) with two unknowns t_a and t_b, we can derive the following quartic formula and solve for t_a:

$$St_a^4 + Tt_a^3 + Ut_a^2 + Vt_a + W = 0 \tag{3}$$

where:

$$S = Y^2 \quad (4)$$

$$T = -2YZ - 2(A \bullet B)XY \quad (5)$$

$$U = Z^2 + 2(A \bullet B)XZ - r^2Y^2 + X^2 \quad (6)$$

$$V = 2r^2YZ \quad (7)$$

$$W = -r^2Z^2 \quad (8)$$

and:

$$X = (A \times B \times A) \bullet P \quad (9)$$

$$Y = (A \times B \times A) \bullet B \quad (10)$$

$$Z = (A \times B \times P) \bullet B \quad (11)$$

Once we have t_a, we can scale A by this value and get a 3D eyeball center estimate for a given radius. Although we could use a standard eyeball radius, we instead cycle through a range of 1.1cm to 1.4cm (in 1/10th millimeter increments) and get an eyeball estimate for each radius size. This corresponds approximately to the natural range of eyeball radius sizes found in humans (1.2cm to 1.3cm) [5].

During the calibration procedure, we have the user look into the camera first, and then look at four calibration points. The first calibration point is used to get our B vector. For each eyeball estimate (and corresponding radius) and each of the four calibration points, we use our gaze estimation approach to determine the estimated point of regard and get its distance from the true gaze (calibration) point. These error distances are added together, and the eyeball estimate (and radius) with the smallest error is chosen.

Note that this approach assumes that eyeball center (E), 3D iris center (E'), and 3D gaze point (P) are all coplanar, which may not be true if the user's head moves during the calibration procedure; thus, we must adjust our vectors for head rotation and translation before performing our calibration calculations.

2.4.2. Stage 2 calibration (refinement and fovea location)

Due to the optical/visual axis offset (as well as user error during calibration), our initial estimates can be off slightly from their true position. Therefore, we perform a secondary, iterative refinement procedure that uses a binary search to find the best eyeball center within ±5cm of our initial estimate (to a precision of 1/100th of a millimeter). We also refine our radius search simultaneously, using the same range as before but also with 1/100th millimeter precision.

Finally, we perform another binary search, this time moving the eyeball position up and down (within 2.5mm of the current estimate). The fovea is assumed to lie along the A vector, and we get its position by intersecting A with the sphere centered at our final eyeball center estimate.

We now have the 3D eyeball center (which can be rotated and translated with head position and pose information), the 3D eyeball radius, and the fovea point (stored as an offset from the eyeball center). Note that we use 5 sample frames for each calibration point, including the look-into-camera stage.

2.5. Gaze estimation (from the active camera view)

For a given frame, we adjust the eyeball center position based on the current head pose and position. Using the 2D iris center, we want to find the iris contour points, map them (and the iris center) to the eyeball sphere, determine the optical axis, rotate the fovea point accordingly (since its location is dependent on eye position and rotation only), compute the new (3D) iris center from the optical axis, and finally get the visual axis as the vector from the fovea to the new iris center. To get the iris contour points, we perform the following eight steps: (1) Define a smaller search region around the iris position and blur the grayscale image with a 5x5 Gaussian filter; (2) Extract gradient magnitudes and directions; (3) Given a normalized 2D vector H that goes from our iris center estimate to the 2D eyeball center, shoot rays R_i outwards ±125° around H (one ray per degree); (4) For all pixels (within search region) along each ray, compute $score_{i,j} = m_{i,j} * (dot(H, D_{i,j}) > 0.5)$, where i is the ray index, j is the pixel index, $m_{i,j}$ is the gradient magnitude, and $D_{i,j}$ is the normalized gradient direction vector from dark to light pixels; (5) The highest scoring pixel along each ray is chosen as a potential iris contour point, similar to [8]; (6) Remove duplicate points (caused by overlapping rays), giving us N' iris contour points; (7) To eliminate possible eyelid edges, only keep pixels with gradient angles between [-35°, 35°], [145°, 180°], and [-145°, -180°] (vertical edges only); this gives us our final N iris contour points; (8) Compute confidence value for contour as N/N'.

We map 2D iris contour points to the eyeball sphere by converting them into 3D perspective vectors and intersecting them with the current eyeball sphere. We refer to a normalized vector going from the eyeball center to one of those intersection points as an "iris contour vector" C_i. Before we can perform gaze estimation, we need to estimate the 3D iris radius on the eyeball (or rather, the average dot product between the optical axis and each C_i). We use the information extracted during the look-into-camera phase of the calibration procedure; note that an approximation of the optical gaze vector can be extracted by intersecting the 2D iris center with the eyeball sphere, subtracting the eyeball center, and normalizing the result, giving us vector V. We get the average of the dot products (*aveDot*) between V and each C_i. In this way, we define an "iris disk." See Figure 4.a. for an illustration of this concept. This unique approach of mapping 2D contour points directly onto the 3D eyeball sphere allows for efficient detection of the gaze direction.

For gaze estimation, we first detect the iris center and contours and map these points to the eyeball sphere to ultimately get V and all C_i vectors (respectively). We want

to find the optical axis G such that 1) it is parallel to V and 2) the dot product of G and each C_i is *aveDot*. Thus, we solve the following linear equations:

$$\begin{bmatrix} C_0^X & C_0^Y & C_0^Z \\ \dots & \dots & \dots \\ C_N^X & C_N^Y & C_N^Z \\ V^X & V^Y & V^Z \\ \dots & \dots & \dots \\ V^X & V^Y & V^Z \end{bmatrix} \begin{bmatrix} G^X \\ G^Y \\ G^Z \end{bmatrix} = \begin{bmatrix} aveDot \\ \dots \\ aveDot \\ 1 \\ \dots \\ 1 \end{bmatrix} \quad (12)$$

Note that *aveDot*, V, and the constant 1 are repeated in their respective matrices N times (once for each contour vector).

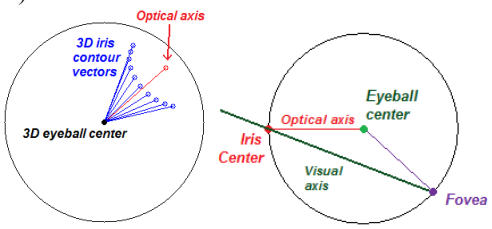

Fig. 4.a. (left) "Iris disk" concept: blue vectors are the "iris contour vectors," while the red vector is the computed (optical axis) gaze direction. **4.b. (right)** Optical vs. visual axis.

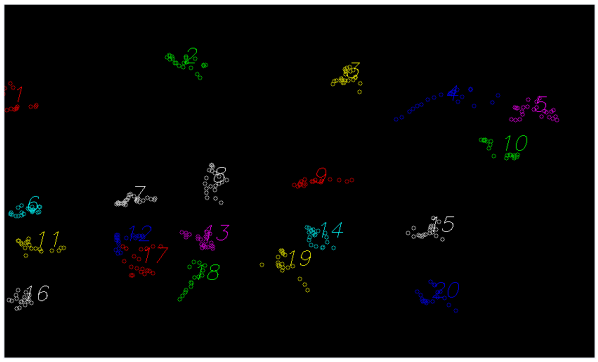

Fig. 5. Gaze test (web camera). The numbers represent each cluster when the user looks at the center of blocks in positions from left to right, row by row from top to bottom.

To get the visual axis, we rotate the fovea offset based on the optical axis G. We intersect the optical axis with the eyeball sphere to get a new estimate for the 3D iris center, and take the normalized vector from the fovea to this new iris center as our final gaze direction (see Figure 4.b).

We have found that it is necessary to average the eyeball centers and the gaze directions over several frames to get stable estimates (our current buffer is for the eyeball is 10 frames, while we use 3 frames for the gaze direction). Moreover, we weigh our gaze direction by the confidence value from the contour extraction.

Once we have our current 3D eyeball position and gaze vector, we must map this information to screen coordinates. We assume that the screen's 3D position, size, and orientation are already known, and so the mapping is a simple ray-plane intersection. We use the average position of the two eyeball centers as the starting point and the average of the two visual axis vectors as our gaze direction.

For all stages of calibration and gaze estimation, we do not use a given sample of data unless its confidence value exceeds 10% (which is rather low but does at least get rid of utterly erroneous values). Moreover, while we compute calibration and gaze values for each eye separately, we do not use a sample unless both eyes have high enough confidence values.

3. EXPERIMENTAL RESULTS

We performed a gaze and PoG estimation experiment wherein the user was asked to look at 20 gaze markers on the screen (effectively, the center of each brick in a 5x4 grid). To ensure that our calibration points were at known locations relative to the camera, we opted to use a Logitech Orbit AF web camera in a special rig with a yardstick attached for the purpose of evaluating our calibration and gaze estimation approaches. Since we would argue that a close-view face image captured from a distance by an active camera (using an optical zoom) is approximately equivalent to the face image captured by a static camera close the subject, we simulated the effect of a zoom lens by positioning the user's head about 30 – 40 cm away from the camera. Camera images were 640 x 480 pixels. Head pose information was extracted using an existing face library [6]. Each marker was gazed at for 2-4 seconds (about 20 gaze

Table 1. Intra- and inter-cluster separation (in degrees)

	Min	Max	Mean	Std Dev
Intra-cluster	0.0684°	4.9158°	1.0889°	0.6710°
Inter-cluster	2.3292°	16.4482°	8.0991°	4.7793°

Fig. 6. Gaze test (web camera). Green circle is eyeball center, white circle is fovea, red circle is iris center, white lines are iris contours, and the large white lines are the gaze directions (visual axis).

samples for each marker on average). Figure 5 shows the estimated gaze positions, while Table 1 lists the variability of the gaze direction for a given gaze point ("intra-cluster"), and the separation (again, in gaze direction angles) between clusters ("inter-cluster"). The latter is computed as the mean angular distance between the average gaze directions of neighbor clusters horizontal and vertically. Finally, Figure 6 shows some samples of the gaze estimation results with the user's head rotated.

The ordering of the gaze points matches the pattern of the grid, and the clusters themselves are fairly compact. We believe that a further corrected calibration between the camera and the screen could improve the results.

We also performed a gaze direction test using our active camera (about 11-feet away from the user's head) with approximately-measured calibration points. The only difference in parameters is that we used 3 samples per calibration point instead of 5. Figure 7 shows a few sample gaze directions.

Fig. 7. Gaze direction test with active camera images.

4. DISCUSSION AND CONCLUSION

In this paper, we have presented a two-camera system that can detect the face and eyes and allows for greater translational movement freedom, without resorting to infrared light, stereo-camera setups, or other complex hardware solutions. Moreover, we have proposed an eyeball center and fovea offset extraction procedure and gaze estimation approach, both of which are fully capable of dealing with head pose adjustments.

As part of our work towards integrating the two-camera system with our gaze estimation work, we are currently developing a mechanism to locate the screen relative to the camera, and once complete we will use points on the screen itself for calibration. These points will then be adjusted for the rotation of the active camera during calibration.

Other future work will include dealing with larger head pose angles and increasing the speed and robustness of the iris and contour detection phases. We would also like to expand upon the secondary refinement pass to allow for a more accurate estimate of the fovea. Finally, future versions of the dual-camera system will allow for dynamic depth changes while tracking the gaze directions of multiple subjects. (**Acknowledgement:** this material is based upon the work supported by the AFRL at Rome, NY.)

5. REFERENCES

[1] K. Bernardin, H.K. Ekenel, and R. Stiefelhagen, "Multimodal identity tracking in a smart room", Personal and Ubiquitous Computing, vol. 13, pp. 25-31, 2009.

[2] D. Beymer and M. Flickner, "Eye gaze tracking using an active stereo head," CVPR'03, vol. 2, pp. 451-458, 2003.

[3] C. Colombo, D. Comanducci, and A. Del Bimbo, "Robust tracking and remapping of eye appearance with passive computer vision," ACM TOMCCAP, 3:1-20, Dec. 2007.

[4] F.L. Coutinho and C.H. Morimoto, "Free head motion eye gaze tracking using a single camera and multiple light sources," SIBGRAPHI'06, pp. 171-178, Oct. 2006.

[5] D. Hansen and Q. Ji, "In the eye of the beholder: a survey of models for eyes and gaze," IEEE Trans. PAMI, vol. 99, 2009.

[6] http://www.seeingmachines.com/product/faceapi/

[7] Terry K. F. Hung, "Real-time face detection and applications," Master's Thesis, Binghamton University, 2008.

[8] D. Li, D. Winfield, et al, "Starburst: a hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches," IEEE CVPR Workshop on V4HCI, 2005.

[9] K. Nguyen, et al., "Differences in the infrared bright pupil response of human eyes," ACM ETRA'02, 133-138, 2002.

[10] B. Noureddin, P. Lawrence, and C. Man, "A non-contact device for tracking gaze in a human computer interface," CVIU, vol. 98, pp. 52-82, 2005.

[11] T. Ohno and N. Mukawa, "A free-head, simple calibration, gaze tracking system that enables gaze-based interaction," ACM ETRA'04, pp. 115-122, 2004.

[12] E. Pogalin, A. Redert, I. Patras, and E.A. Hendriks, "Gaze tracking by using factorized likelihoods particle filtering and stereo vision," 3DPVT'06, pp. 57-64, June 2006.

[13] V. Vezhnevets and A. Degtiareva, "Robust and accurate eye contour extraction," GraphiCon, pp. 81-84, 2003.

[14] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," IEEE CVPR, 2001.

[15] J. Wang, L. Yin, and J. Moore, "Using geometric property of topographic manifold to detect and track eyes for human computer interaction", ACM TOMCCAP, 3(4):1-19, 2007.

[16] J.G. Wang and E. Sung, "Gaze determination via images of irises," *Image and Vision Computing*, 19:891-911, 2001.

[17] H.Y. Wu et al., "Tracking iris contour with a 3D eye-model for gaze estimation," ACCV'07, p688-697, 2007.

[18] D. Xia and Z. Ruan, "IR image based eye gaze estimation," ACIS-SNPD'07, vol. 1, pp. 220-224, 2007.

[19] J. Xie and X. Lin, "Gaze direction estimation based on natural head movements," IEEE ICIG'07, pp. 672 – 677, Aug. 2007.

[20] H. Yamazoe et al., "Remote and head-motion-free gaze tracking for real environments with automated head-eye model calibrations," CVPR Workshop on CVPR4HB, 2008.

[21] Z. Zeng, M. Pantic, G. Roisman, and T. Huang, "A survey of affect recognition methods: audio, visual, and spontaneous expressions," *IEEE Trans. on PAMI, 31(1):* 39–58, 2009.

[22] S. Zhai, C. Morimoto, and S. Ihde, "Manual and gaze input cascaded (MAGIC) pointing," ACM SIGCHI'99, p246-253.