EYE-CONTROLLED COMPUTER INTERFACE WITH NATURAL HEAD MOVEMENTS

Jerry Chi Ling Lam[1], Moshe Eizenman[1,2,3], Parham Aarabi[1]
[1]Department of Electrical and Computer Engineering, [2]Institute of Biomaterials and Biomedical Engineering, [3]Department of Ophthalmology, University of Toronto, Toronto, Canada

INTRODUCTION

Visual aids that use remote gaze estimation systems [1] allow Age-related Macular Degeneration (AMD) patients to perform non-sequential reading tasks on a computer screen in less time and with fewer errors [2]. With gaze-controlled visual aids, AMD patients can position a rectangular region over an area of a global image that they wish to magnify. Patients can then toggle a computer display between the global and magnified images.

Our current visual aid [2] uses a remote camera with a relatively small field of view. To remain within the field-of-view of the camera, subjects are required to limit their head movements or to use a chin-rest. To enhance the current system, it will be desirable to accommodate subjects' natural head movements within a volume of approximately 30x30x30 cubic centimeters. To achieve this goal, the field of view of the camera was increased and new algorithms to locate the right and left eyes in the video images were developed.

This work describes real-time algorithms for the detection and identification of face-regions that contain the right and left eyes. Following the detection of these regions, the algorithms described by Guestrin and Eizenman [1] are used to estimate the point-of-gaze. The eye-detection algorithms are based on the detection of symmetric face features [3]. These algorithms were shown to be more robust to variations in facial features than algorithms that are based on the geometry of specific facial features [4]. Also, an algorithm based on symmetry is insensitive to interferences from eyewear.

FACE SYMMETRY ALGORITHM

Colmenarez and Huang [3] suggested the use of the correlation coefficient as a measure of face symmetry.

Let $x[n], n = 0,1,\cdots,N-1$ represents a row in the image, and x_L and x_R are two neighboring sub-segments of size $W/2$ with $W \le N$, such that

$$x_L[m,k] = x[m-k], k = 1,...,W/2 \tag{1}$$

and

$$x_R[m,k] = x[m+k], k = 1,...,W/2$$

$$x[n] = 0, \forall n < 0 \text{ and } n \ge N$$

Then, the correlation function $C[m]$ is calculated by the following equation:

$$C[m] = \frac{1}{\sigma_L[m]\sigma_R[m]}\sum_{k=1}^{W/2}(x_L[m,k]-\bar{x}_L[m])(x_R[m,k]-\bar{x}_R[m]) \tag{2}$$

where

$$\bar{x}_L[m] = \frac{1}{W/2}\sum_{u=1}^{W/2}x[m-u]$$

$$\bar{x}_R[m] = \frac{1}{W/2}\sum_{u=1}^{W/2}x[m+u]$$

$$\sigma_L[m] = \sqrt{\frac{1}{W/2}\sum_{v=1}^{W/2}(x_L[m,v]-\bar{x}_L[m])^2}$$

$$\sigma_R[m] = \sqrt{\frac{1}{W/2}\sum_{v=1}^{W/2}(x_R[m,v]-\bar{x}_R[m])^2}$$

The symmetry function is then defined by:

$$S[m] = \frac{1}{M}\sum_{i=0}^{M}C_i[m] \tag{3}$$

where

i is a row index and M is the number of rows in the face.

The peak of the symmetry function defines the horizontal position of the line of symmetry (note that the line of symmetry is always vertical). The area to the left of the line of symmetry contains the left eye and the area to the right of this line contains the right eye. The performance of the face symmetry algorithm can be enhanced by using a gradient of the original image. By using a gradient, areas with constant illumination will not bias the symmetry measure. Furthermore, since the human face is dominated by horizontal edges, a gradient in the vertical direction $\Delta(x,y)$ will maintain the integrity of all facial features. The gradient is calculated by the following equation:

$$\Delta(x,y) = [G_{\sigma}(x)\frac{\partial}{\partial y}G_{\sigma}(y)] * I(x,y) \qquad (4)$$

G_{σ} is the Gaussian filter

To detect the symmetry axis for a range of roll head movements, the image is first rotated by a set of discrete angles that span the expected range of roll head movements. Then, the face symmetry algorithm is applied to each rotated image and the axis of symmetry is determined by the line with the highest peak in the symmetry function.

The face symmetry algorithm has several limitations: 1) It fails to find the axis of symmetry when the illumination of the face is non-uniform (e.g. shadows), 2) It is sensitive to asymmetric features (i.e. the distribution of the subject's hair), and 3) It is computationally intensive and cannot be used in real-time gaze-estimation systems.

LOCAL SYMMETRY ALGORITHM

Since face illumination changes very gradually, the illumination of each local area within the face (i.e. left-eye, right-eye) tends to be uniform. Therefore, if the spatial distance between the symmetric features can be reduced the algorithm will become less sensitive to non-uniform face illumination. The peak of the face symmetry algorithm that was described above determines the degree of similarity between the left and right halves of the face. Since the face includes features that are internally symmetric (i.e. each eye, the mouth), it is possible to estimate the degree of similarity of these internally symmetric local features and use their locations to estimate the axis of symmetry of the face. By using the symmetry of local features within the face, the algorithm becomes less sensitive to non-uniform illumination and to asymmetric projections on the image plane of the eye-tracker's camera.

The local symmetry algorithm has two stages: 1) face detection, and 2) local symmetry detection.

Face Detection

The remote gaze estimation system uses Infra Red (IR) light sources to illuminate the face. Due to this illumination, the brightest regions in the image are associated with reflections from the subject's face. The face detection algorithm uses a relatively high threshold to locate regions with brightness that is above this threshold. Figure 1b shows a binary image that illustrates these regions. Each bright region is then dilated using a flood-fill technique [5] and connected component analysis, similar to [6], is performed on the resulting blobs to form a face blob (Figure 1c). Figure 1d shows the bounding-box for the face blob. The width of the bounding box determines the value of W in Equations 1 and 2.

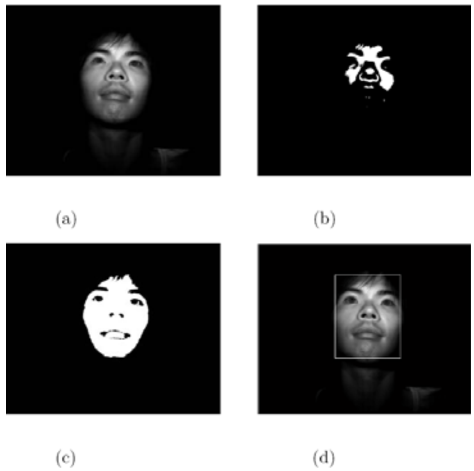


Figure 1 a) Original Image, b) Skin Blobs, c) Face Blob, d) Face Bounding Box

Local Symmetry Detector

The face symmetry algorithm can be easily extended to search for local symmetric features. This is done by reducing the length of the two sub-segments (x_L, x_R Equation 1). As the width of the two sub-segments decreases, the shape of the symmetry function changes from a function with a single dominant peak (figure 2a) to a function with multiple peaks (figure 2b). The positions of the peaks mark the locations of local symmetric features in the face. When the width of each sub-segment is set to be approximately 10% of the total width of the bounding box (W), the peaks in Figure 2b represent the left-eye the right-eye and the nose-mouth complex.

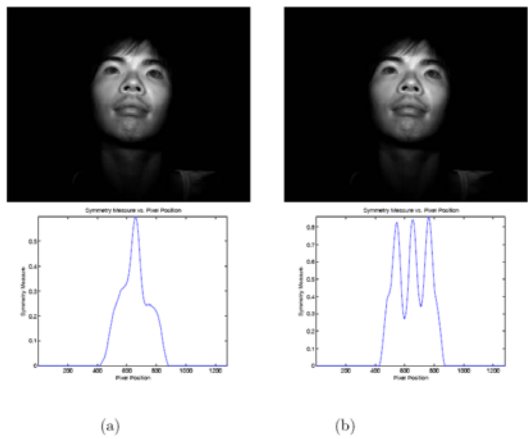
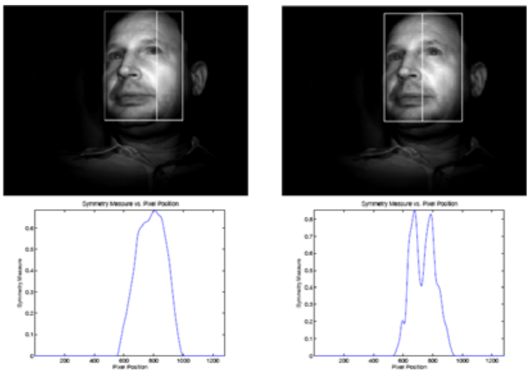


Figure 2 Symmetry Function a) Face Symmetry Algorithm, b) Local Symmetry Algorithm

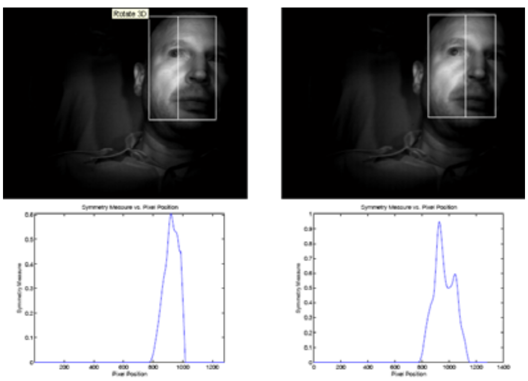
The line that separates the left eye region from the right eye region is calculated by averaging the locations of the peaks in the symmetry function. The local symmetry algorithm attempts to detect up to three peaks in the symmetry function. If the height of a peak is less than 50% of the highest peak, it is not used in the averaging process (i.e. these peaks do not provide information with respect to local symmetries).

The following examples demonstrate the differences between the face symmetry algorithm and the local symmetry algorithm. Figure 3 shows the ability of the local symmetry algorithm to locate the left and right eye windows when the eye-projections on the image plane are asymmetric. In Figure 3b, the locations of the left eye and the nose-mouth complex are detected by the local symmetry detector. Using these locations the detector is able to find a line that separates the left and right eye regions.



(a) (b)

Figure 3 Asymmetric Projection a) Face Symmetry Algorithm, b) Local Symmetry Algorithm



(a) (b)

Figure 4 Non-Uniform Illumination a) Face Symmetry Algorithm, b) Local Symmetry Algorithm

Figure 4 shows that even when one side of the face is poorly illuminated, the local symmetry algorithm is capable of finding the proper regions for the left and right eyes.

Figure 5 demonstrates the ability of the local symmetry algorithm to detect the left and right eye regions when the subject's head is tilted in the roll direction. Note that the local symmetry algorithm does not require image rotation prior to the determination of the line that separates the left and right eye windows. Therefore, it is computationally much more efficient than the face symmetry algorithm. For a typical image, the local symmetry algorithm is at least 50 times faster than the face symmetry algorithm.

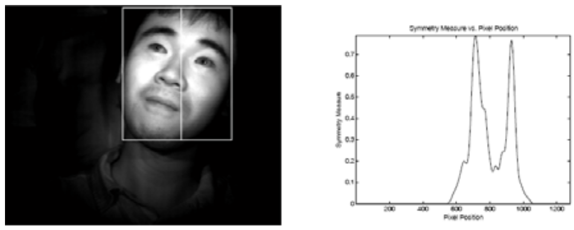


Figure 5 Local Symmetry Algorithm Applied on Tilted Head Image

EXPERIMENTAL RESULTS

The performance of the face symmetry algorithm and the local symmetry algorithm were tested with three subjects[1]. For each subject, the ranges of head movements (relative to the subject's initial position) for which the algorithms were able to detect the left and right eye regions were determined. Head movements were measured by a head tracker [7]. The results are summarized in tables 1 and 2.

Table 1: Experimental Results: Face Symmetry Algorithm

Subject	X(cm)	Y(cm)	Z(cm)	Yaw°	Pitch°	Roll°
S1	±21.5	±8	±7	±11	±25	±40
S2	±21	±6.5	±6.5	±8	±46	±45
S3	±19	±5	±10.5	±9	±55	±43

Table 2: Experimental results: Local Symmetry algorithm

Subject	X(cm)	Y(cm)	Z(cm)	Yaw°	Pitch°	Roll°
S1	±21.5	±11.5	±7	±29	±25	±32
S2	±21	±9.5	±6.5	±24	±46	±45
S3	±19	±7.5	±10.5	±33	±55	±31

[1] All subjects shown in this paper have signed a consent form to allow the usage of their face images in academic publications

For the data in Tables 1 and 2 the plane formed by the Y-Z axes was parallel to the computer screen with the Z-axis pointing upwards. Figure 6 shows head movements in the X (Figure 6a and 6b), Y (Figure 6c and 6d) and Z (Figure 6e and 6f) directions using the local symmetry algorithm. Figure 7 shows head movements in the roll (Figure 7a and 7b), yaw (Figure 7c and 7d) and pitch (Figure 7e and 7f) directions using the local symmetry algorithm.

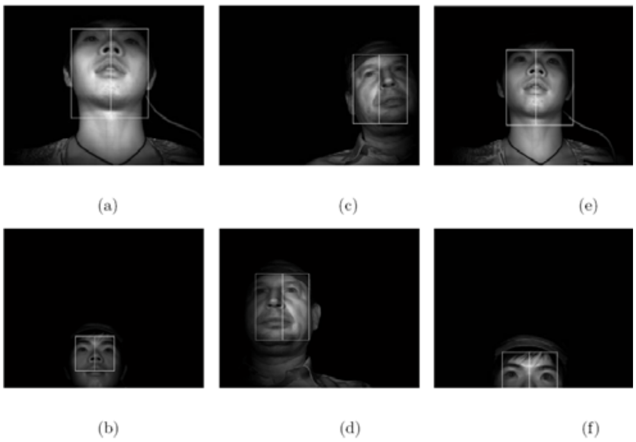


Figure 6 Experimental results: Local Symmetry Algorithm (a) close to monitor, (b) far from monitor, (c) left to the monitor, (d) right to the monitor, e) upward to the monitor, f) downward to the monitor

Figure 7 Experimental results: Local Symmetry Algorithm a) roll to the left, b) roll to the right, c) yaw to the left, d) yaw to the right, e) pitch up, f) pitch down

For head movements in the X, Z and Pitch directions, the two algorithms had similar performance. For head movements in the Y direction, the local symmetry algorithm outperforms the face symmetry algorithm. This is due to the ability of the local symmetry algorithm to handle non-uniform face illumination. The local symmetry algorithm also outperforms the face symmetry algorithm in the yaw direction due to its ability to cope with asymmetric projections of face features. The face symmetry algorithm outperforms the local symmetry algorithm in the roll direction. For the improved performance in the roll direction the face symmetry algorithm requires significantly more processing time. For AMD patients that use gaze-controlled visual aids, head movements in the roll direction are usually limited to less than 20 degrees. Therefore, the performance of the local symmetry algorithm is adequate for the expected range of roll head movements.

CONCLUSIONS

The internal symmetry of local facial features was used by a novel local symmetry algorithm to detect the left and right eye regions. For a visual aid, the local symmetry algorithm has two main advantages over the face symmetry algorithm: a) it performs better for an extended range of head movements in the yaw and Y directions, and b) it requires less processing time, so it can be used in real-time. The local symmetry algorithm performs very well (i.e. probability of false detection less than 1%) for the expected range of head rotations (yaw $\pm 20°$, pitch $\pm 20°$ and roll $\pm 20°$) and head movements in the X direction (± 15 cm). The algorithm does not perform as well for head movements in the Y and Z directions. The current algorithm can detect reliably the left and right eye regions for head movements that span approximately 50% of the desired range in the Y and Z directions. The performance of the local symmetry algorithm can be improved by providing brighter illumination within the desired volume of head movements.

REFERENCES

[1] E.D. Guestrin and M. Eizenman, "General theory of remote gaze estimation using the pupil center and corneal reflections," *IEEE Transactions on Biomedical Engineering*, vol. 53, no.6, pp. 1124–1133, Jun. 2006.

[2] B. E. L. Braymore, "Non-Sequential Reading in People with Age-Related Macular Degeneration," M.A.Sc. Thesis, University of Toronto, 2004.

[3] A. Colmenarez and T. Huang, "Frontal-view face detection," in *Proc. of SPIE,* 1995, vol. 2501, pp. 90–98.

[4] S. Ioannou, M. Wallace, K. Karpouzis, A. Raouzaiou and A. Kollias, "Combination of Multiple Extraction Algorithms in the Detection of Facial Features," in *Proc. of ICIP*, 2005, vol. 2, pp.378-381.

[5] S.V. Burtsev and Y.P. Kuzmin, "An efficient flood-filling algorithm," *Comput. Graph.* , vol. 17, pp. 549-561, 1993.

[6] K. Sobottka and I. Pitas, "Segmentation and Tracking of Faces in Color Images," in *Proc. of the Second International Conference on Automatic Face and Gesture Recognition*, Oct. 1996, pp. 236 – 241.

[7] R.S. Allison, M. Eizenman, and B.S.K Cheung, "Combined head and eye tracking system for dynamic testing of the vestibular system," *IEEE Transactions on Biomedical Engineering*, vol. 43, pp. 1073 – 1082, 1996.