REAL TIME EYE TRACKING FOR HUMAN COMPUTER INTERFACES

Subramanya Amarnag, Raghunandan S. Kumaran and John N. Gowdy.

Department of Electrical and Computer Engineering
Clemson University, Clemson, SC 29634, USA.
Email: {asubram, ksampat, jgowdy}@clemson.edu

Abstract

In recent years considerable interest has developed in real time eye tracking for various applications. An approach that has received a lot of attention is the use of infrared technology for purposes of eye tracking. In this paper we propose a technique that does not rely on the use of infrared devices for eye tracking. Instead, our eye tracker makes use of a binary classifier with a dynamic training strategy and an unsupervised clustering stage in order to efficiently track the pupil (eyeball) in real time. The dynamic training strategy makes the algorithm subject (speaker) and lighting condition invariant. Our algorithm does not make any assumption regarding the position of the speaker's face in the field of view of the camera, nor does it restrict the 'natural' motion of the speaker in the field of view of the camera. Experimental results from a real time implementation show that this algorithm is robust and able to detect the pupils under various illumination conditions.

1. Introduction

The eyes and lips are two promising facial organs that can be used to create an interface between man and machine. Tracking these organs may be considered as a first step in creating such an interface. Hence, the development of real time eye tracking and (or) lip tracking algorithms forms an important step in the design of efficient Human Computer Interfaces (HCI). The approaches used thus far for eye tracking may be grouped under two broad categories. The first approach makes use of IR devices for the purposes of pupil tracking [1-3]. Eye tracking based on active IR illumination utilizes the special bright pupil effect. This is a simple and very accurate approach to pupil detection using the differential infrared lighting scheme. However, this method requires the use of additional resources in the form of infrared sources and infrared sensors.

The second approach attempts to track eyes with the aid of an 'ordinary' camera [4,6,8]. Baluja et al. [4] have used Artificial Neural Networks for eye tracking. One major drawback of this system is the number of eye images that are required to train the neural network. Also each frame is treated independently which results in processing of redundant data. Smith et al. [6] use skin color predicates to locate the face before locating the eyes. Since skin color is highly variable form person to person this method is not very robust. Sobottka and Pitas [8] use color predicates to locate the face and the eyes are detected using minima analysis. Since minima analysis in not invariant to lighting conditions, this method is not very robust. An inherent advantage (or disadvantage!) in these approaches is the absence of any kind of infrared devices.

Before we further pursue this approach to eye tracking let us examine the merits and demerits of eye tracking over lip tracking. Although there exist a number of lip tracking algorithms, they are bound by a number of constraints for their successful application. For example, a number of these algorithms assume that the speaker's face is always positioned in the center of the field of view of the camera and the head is 'relatively' stationary. Another difficulty in implementing practical lip tracking algorithms is that they have to deal with the inherent motion of the lips when the speaker is 'talking'. It has been shown that the relative motion of the pupil is small when compared to the motion of the lips when the speaker is talking. Also, the pupils are 'round' in shape and have a unique color that distinguishes them from other components of the human face. Hence, eye tracking appears to be a reasonable solution to the lip tracking problem, as a fix on the speakers eyes will give us a rough estimate on the position of the lips.

One of problems associated with the tracking of eyes is the voluntarily closure of eyes by the speaker. It is a well-known fact that we tend to blink involuntarily; the average interval of such a blink lasts for not more than a frame period (30 Hz). However voluntary closure of eyes would result in the algorithm's failing to track the eyes. Hence, before we go any further we need to make an assumption that the speaker does not voluntarily close his eyes. Another inherent assumption is that the speaker's eyes are always in the field of view of the camera. However, we do not need to make any assumption on the position of the speaker's eyes in the field of view of the camera.

2. Eye-Tracking

Our eye-tracking algorithm essentially consists of four stages, which include pre-processing, classification, clustering and post processing. At the start of the algorithm we assume that all the pixels in the frame are candidate eye pixels, and at each stage we reduce the number of such pixels.

2.1. Pre-Processing

It can be seen from Figure 1 that the pre-processing stage forms the second step in eye tracking algorithm. However, for reasons of clarity we will discuss the first step of the algorithm in

a later section. At this point it suffices to say that the frame search region stage gives us a rough estimate of the position of the eye in the present frame based on previous 'knowledge' of the eye position. In the pre-processing stage we are attempting to determine the candidate eye pixels using the color properties of the eye. It is a well-known fact that the regions of the pupil are regions of high color saturation (saturation value would depend on the color of the pupil). In this stage we rely on the fact that

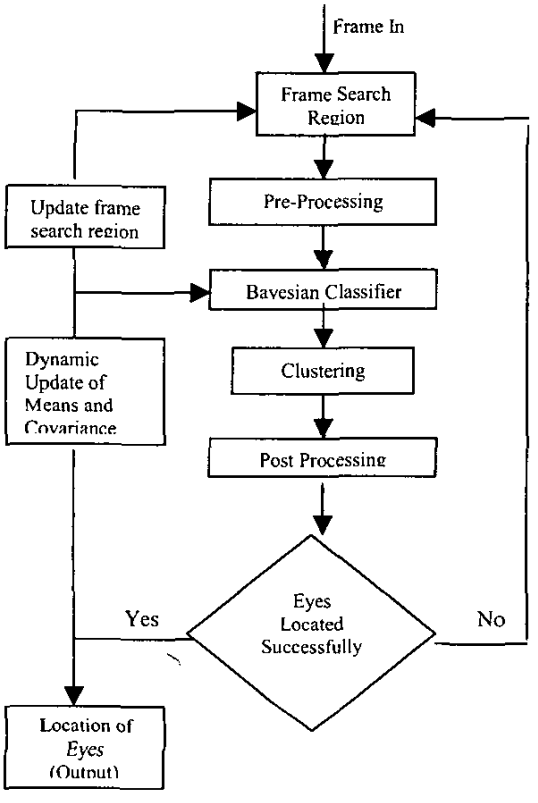

Figure 1: Eye Tracking Algorithm

'dark' objects are poor reflectors of light. Hence, the intensity of any pixel can be used as a first step in eliminating the number of candidate eye pixels. Therefore if the intensity of any pixel is greater than a particular threshold, it cannot be classified as belonging to the eye. Here the intensity of any pixel is computed as shown below

$$i = 0.33*(r+g+b) \tag{1}$$

where r, g, b are the red, green and blue intensities of the pixel under consideration. In our experiments we have found a threshold of 0.27 works well with most types of lighting conditions. This stage tends to make the algorithm more robust to lighting conditions.

2.2. Bayesian Classifier

In this step we are trying to identify the candidate eye pixels among the pixels that were output by the pre-processing stage, by the use of a Bayesian classifier. We treat the problem at hand as a two-class problem, i.e., every candidate pixel is classified as belonging either to the eye or not. The Bayesian classifier has long been considered as one of the best existing binary classifiers [5]. We have used a vector consisting of the Hue, Saturation and Intensity values for each of the candidate pixels as the feature vectors for the Bayesian classifier. The conversion to the 'HSI' color model from other models (e.g. 'RGB) is fairly straightforward. The equations for computing the mean and covariance are

$$\hat{\mu}_i = (1/N_i)*\sum_{j=1}^{N_i} X_j^i \tag{2}$$

$$\hat{\Sigma}_i = (1/(N_i-1))*\sum_{j=1}^{N_i}(X_j^i-\mu_i)(X_j^i-\mu_i)' \tag{3}$$

Where $i = 1 or 2$, since we are dealing with a two-class problem, N_i = Number of Vectors in each class, X= (h, s, i)^t is the column vector consisting of the hue, saturation and intensity values for the various pixels, $\hat{\mu}_i$ is the mean for the class i, $\hat{\Sigma}_i$ is the covariance of class i. The probability that a given feature vector x belongs to a class i is given as

$$p(x \mid w_i) = K*\exp(-0.5*(x-\mu_i)(\Sigma_i)^{-1}(x-\mu_i)') \tag{4}$$

Where $K=(1/((2\pi)^{n/2}*(|\Sigma_i|)^{1/2})$, n = dimension of the vector (in our case n=3). If $p(x \mid w_1) > p(x \mid w_2)$ we classify the vector x as belonging to class 1 and if the converse is true the vector would be classified as belonging to class 2. It should be noted here that the variables are assumed to have a Gaussian PDF. One of the foremost drawbacks of the Bayesian classifier (with respect to implementing them real time) is the computational complexity of the mean and the covariance. Hence, we propose to compute the mean and covariance off-line, and then use a dynamic update strategy to include on-line information in the mean and covariance.

The off-line mean and covariance were hand trained by taking samples of images of various speakers. We also propose a dynamic update of mean and covariance, i.e. during the execution of the algorithm; the results of each frame are used to update the mean and the covariance of the eye and non-eye classes for the next iteration. This step ensures that the algorithm adapts to various positions of the speaker and also makes it subject invariant.

2.3 Clustering Algorithm

In this stage we treat the problem from a geometrical perspective. The reason for performing a clustering at this stage is that the Bayesian classifier outputs sometimes include pixels that do not belong to the eye. It was noticed that it classified certain pixels that had a 'similar color information' when compared to the speaker's eye (e.g. hair) as eye pixels. Hence, in this stage we create an image in which the pixels, which are candidate eye pixels (as returned by Bayesian classifier), are turned on. We are trying to find clusters in this image by using an unsupervised clustering algorithm. We have not made any assumption regarding the number of clusters in the image nor their means. The algorithm is as shown in Figure 2, 'noe' is the number of clusters and the exemplar of any cluster is calculated by a simple

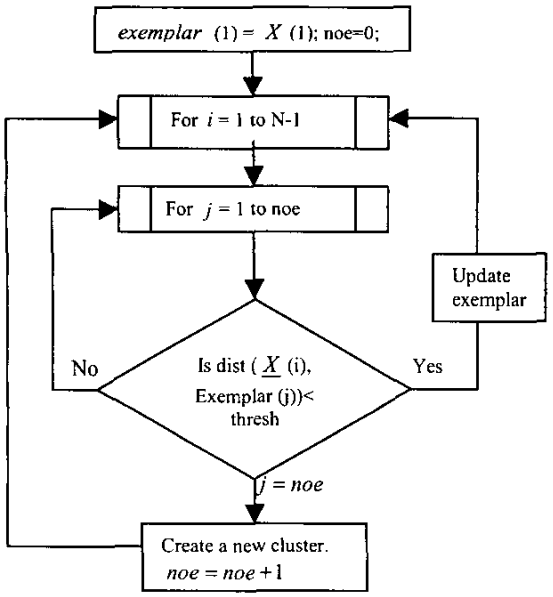

Figure 2: Clustering Algorithm

averaging computation. We have used the Euclidean distance in the above algorithm for distance computation. We have found that a threshold of the range of '20' works well with many images.

3. Post Processing

In the post-processing stage we once again rely on the geometrical features of the eye. Since the output of the clustering algorithm we have used depends on the order in which the vectors are presented to it, there exist circumstances when clusters whose distance is below the threshold are classified as separate clusters. Hence, in the first step of the post processing stage we combine all clusters whose distance is less than a particular threshold (distance between any two clusters is the distance between their exemplars). The threshold may be chosen using the empirical formula, to be any values less than $((1/72)*H)$, where H is the camera resolution in the horizontal direction. The above formula was derived using the relation between the eye separation and image resolution.

Another geometrical aspect of the eyes is their width and height as seen by the camera. We have used the horizontal and vertical size of the clusters in order to eliminate the ones that do not represent the eye. Any cluster whose size in the vertical direction is greater than the size in the horizontal direction is eliminated as not representing the eye. The reason is that, even though the pupil is round in shape, a camera does not see that entire pupil as a result of the eyelids. Hence, even if we assume we are able to segment the pupil perfectly, the horizontal dimension would be greater than the vertical dimension. As a last step in pre-processing, we eliminate all the clusters whose size is greater than a particular threshold. It should be noted here that the size of a cluster is the number of pixels contained within that cluster. This step eliminates all clusters that have sizes that are greater than the eyes. Finally, we characterize the frame as having successfully located the eyes if we are left with two clusters; else we drop the frame and use the knowledge of the previous frames to fix the position of the eyes.

Once we have located the position of the eyes on the first frame, we may use this knowledge to our advantage in processing the successive frames in time. If we make the assumption, that in practical circumstances the vertical motion of the speakers face is 'small', this would reduce the search region for the successive frames. Hence, the search region on successive frames is limited to plus or minus 30 pixels in vertical direction of the original pixel locations of the eyes in the first frame.

As discussed earlier one of the problems with tracking eyes, is to deal with the frames when the speaker 'blinks'. In most cases the 'blink' of the speaker does not last more than one frame interval, i.e., the blinking time is less than the sampling period in the temporal sense. However, in cases of voluntary 'blinking', it lasts for more than one frame. In such case we drop these frames and use the knowledge of the last frame that was successfully processed. Our experiments indicate this does not pose a major problem unless the speaker voluntarily closes his eyes.

4. Experimental Setup

We have used the Clemson University Audio Visual Experiments (CUAVE) database and the CMU audio-visual dataset [7] developed by Chen *et.al* for our experiments. The CUAVE database consists of 36 speakers (both stationary and in motion). The speakers were asked to speak out a string of digits with no 'rules' regarding the position of the speaker with respect to the camera. The CMU database consists of 10 speakers. The use of two different databases was done to ensure invariance to lighting conditions. The training set consisted of 20 speakers from the CUAVE database and 6 speakers from the CMU database. The remaining speakers were used as the testing set.

5. Results

The system was implemented on an Intel Pentium III 997 MHz machine and achieved a frame rate of 26 fps. Shown in Figure 3(d) is a frame from the CUAVE database. (The frame has been converted to gray scale for illustration purposes.) We have selected this subject for illustration, as the presence of 'facial hair' would tend to make it difficult for eye tracking algorithms to track the subject's eyes. It can be seen from Figure 3(a) that the pre-processing stage was successful in reducing the number of candidate eye pixels. We trained our Bayesian classifier using the results of the first stage. The results of the Bayesian classifier are as shown in Figure 3(b). It can be seen that the classifier successfully improves upon the results of the first stage. This output was subjected to clustering. The result is shown in Figure 3(c). It should be noted that the various clusters are shown in different colors may not be clearly discernible because of the gray scale conversion.

Figure 3(d) shows the final output with a 'boundary' around the speaker's eyes. The boundary was constructed by using the horizontal and vertical dimensions of the eye clusters. It can be seen that the areas of the two boundaries are not equal.

Shown in Figures 4 and 5 are the eye-tracking results. The results in Figure 4 illustrate the performance of the algorithm against complex backgrounds. The results in Figure 5 (c), (d) correspond to the moving speaker case from the CUAVE database. The frames have been zoomed in to present only the speaker's face. It can be seen that in case of Figure 5 (a), (b) and

(c) that the algorithm successfully tracks the eyes. However, in 5 (d) the position of the left eye is not accurate. The system was able to achieve an accuracy of 88.3% on the CMU database and an accuracy of 86.4% and 76.5% on the CUAVE database for the stationary and moving speaker case respectively. The above percentages were computed based on the total number frames processed and the number of frames in which the eyes were successfully located.

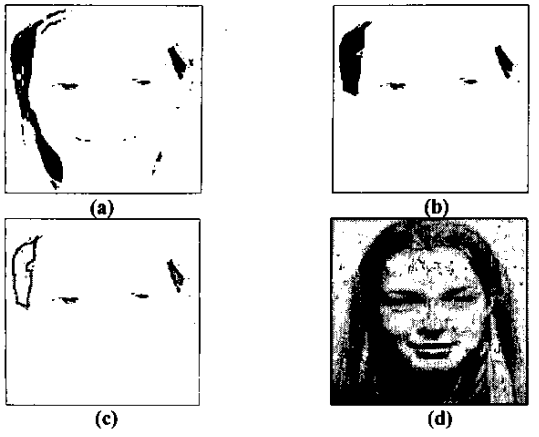

Figure 3: (a) After Pre-Processing. (b) After Bayesian Classifier. (c) Output of Clustering Algorithm. (d) Final Output.

6. Conclusions and Future Work

From the results it can be seen that this algorithm is successful in tracking the pupil. We have tested this algorithm on all speakers in the CUAVE and CMU database and find the results satisfactory. Although the use of IR devices for eye tracking is extremely accurate; algorithms, which do not necessitate the use of IR devices, may be used in applications where the 'high' accuracy is not desired and only an approximate fix on the eyes is essential. We propose to extend this algorithm to lip tracking and subsequently for speech recognition. A fix on the position of the speaker's eyes gives us a fix on the position of the speaker's lips. This algorithm may be easily modified to track speaker's lip for the purposes of speech recognition.

Figure 4: Eye-Tracking results on a frame (with complex background) from the CUAVE database

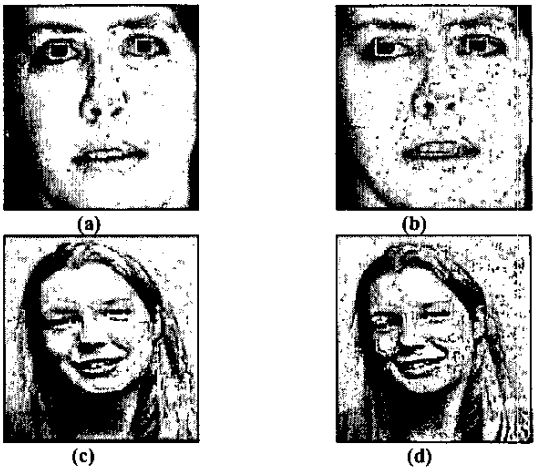

Figure 5: Eye-Tracking results for a sequence of frames from the CMU and CUAVE databases respectively.

7. Acknowledgements

We would like to acknowledge the use of audio-visual data [7] from Advanced Multimedia Processing lab at Carnegie Melon University.

8. References

[1] A. Haro, M. Flicker and I. Essa, "Detection and tracking eyes by using their physiological properties, dynamics and appearance," *Proceedings of IEEE CVPR 2002.*

[2] C. Morimoto and M. Flickner, "Real-Time multiple face detection using active illumination," *Proceedings of the 4th IEEE International Conference on Automatic Face & Gesture Recognition* 2000.

[3] X. Liu, F. Xu and K. Fujimura, "Real time eye-detection and Tracking for Driver Observation under Various lighting conditions," *Proceedings of IEEE Intelligent Vehicle Symposium*, 2002.

[4] S. Baluja and D. Pomerleau, "Non Intrusaive gaze tracking using Artificial Neural Networks," *Technical Report CMU-CS-94-102, Carnegie Mellon University.*

[5] K. Fukunaga, "Introduction to Statistical Pattern Recognition", 2nd Edition, *Academic Press.*

[6] P. Smith, M. Shah, N. Lobo, "Monitoring head/eye motion for driver alertness using one camera," *Proceedings of the International Conference on Pattern Recognition, 2000.*

[7] Advanced Multimedia Processing Lab, CMU, http://amp.ece.emu.edu/projects/AudioVisualSpeechProcessing/.

[8] K. Sobottka, I. Pitas, "A Fully Automatic Approach to Facial Feature Detection and Tracking", Audio Visual Biometric Person Authentication, 1997.