Robust Eye Gaze Estimation

Joanna Wiśniewska[1], Mahdi Rezaei[2], and Reinhard Klette[2]

[1] Warsaw University of Technology, Plac Politechniki 1, 00-661 Warsaw, Poland,
✉ J.Wisniewska@stud.elka.pw.edu.pl
[2] The University of Auckland, Computer Science, Auckland, New Zealand
{m.rezaei, r.klette}@auckland.ac.nz

Abstract. Eye gaze detection under challenging lighting conditions is a non-trivial task. Pixel intensity and the shades around the eye region may change depending on the time of day, location, or due to artificial lighting. This paper introduces a lighting-adaptive solution for robust eye gaze detection. First, we propose a binarization and cropping technique to limit our region of interest. Then we develop a gradient-based method for eye-pupil detection; and finally, we introduce an adaptive eye-corner detection technique that altogether lead to robust eye gaze estimation. Experimental results show the outperformance of the proposed method compared with related techniques.

1 Introduction and Related Work

Eye gaze estimation is an active research topic in computer vision. The development of such systems depends on the quality of the target camera, for example a high-resolution one which focuses only on the eye region, or a simple low-resolution webcam. The first option supports very accurate results for the detection of eye features. Due to cost limitations or varying head poses it may not be applicable. Images captured by a webcam typically contain the whole face and some background, and may require noise removal or image enhancement. A webcam is a widely used device due to its low cost.

High-resolution and infrared cameras were used, for example, in [9, 21], with a focus on high resolution images of the eye-region which simplifies eye-features detection. Both papers use infrared lighting to ease eye-region detection. The Starburst algorithm is often used for detecting the pupil, see [9, 11, 13, 14]. The algorithm applies simple thresholding, fits an ellipse to the pupil, and then refines the ellipse by consideration of a pupil's "edge points". When testing along the basic idea of this algorithm, we noticed that it cannot be applied to noisy and low-resolution images.

Some other researchers use a Hough circle transform for detecting the position of a pupil [17]. When dealing with low resolution images, the first issue is the localization of the eye center. Different methods are proposed to achieve this goal. Typically, methods are based on the usage of an edge detector. Then, a Hough circle transform is suggested to detect the iris or pupil [6, 8]. Published research typically uses test images recorded under ideal lighting conditions where the eye region is not dark, e.g., due to the shadows. The curvature of isophotes

(i.e. points of equal intensity) is used in [19] to design a voting scheme for the localization of the pupil. Next, they extend the method by using SIFT features for each pupil candidate and match such features with examples from a database before obtaining a final decision. The approach in [12] is based on an ensemble of randomized regression trees.

An extension to Logarithmic Type Image Processing (LTIP) is presented in [5]. The parametric extensions of LTIP models were used to built a reliable auto-focus mechanism for cameras working under extreme light conditions. According to facts presented by the authors, LTIP resembles properties of the human visual system. It could be used to enhance images acquired under extreme lighting.

This paper provides an eye gaze detection method for using a low resolution camera, especially also for challenging lighting conditions. Recorded images are assumed to contain an entire face region. The proposed algorithm consists of four major steps. At first, we apply a face-detection algorithm using Haar-features. Then we detect the eye region followed by a processing step to find the eye-pupil. Inspired by the work in [18] we implement pupil localization by analyzing gradients in the eye region. The method defines a function which takes a maximum at a location where many of the gradients intersect. This location is expected to correspond to the position of the pupil. Finally, in order to estimate eye-gaze, we detect the eye corner. Eye-corner detection in [23] proposes a method based on spatial filtering and the application of corner-shape masks. More recent work, e.g. [4, 22], uses Harris corner detection with additional improvements such as active shape models (ASM). The authors of [7] also follow [23] and use additionally weighted variance-projection functions for determining approximately the corner area. In this research we use FAST features [16] with added adaptive thresholds for eye-corner detection.

We evaluate our eye-gaze detection algorithm based on experimental results performed on the BioID database [3] and video sequences recorded from different subjects under various lighting conditions.

2 Proposed Method

The method is structured into five steps: face and eyes detection, adaptive binarization, pupil detection, eye corner detection and gaze estimation.

2.1 Face and Eyes Detection

A fundamental requirement for any eye-tracking system is accurate face and eye socket detection. This can be achieved by Haar-like object detectors which were proposed in [20] and improved in [10]. Under non-ideal lighting conditions it is possible that one half of the face is darker than the other half. By applying face partitioning into two halves, and by changing the detector parameters for each half, we obtained better results under non-ideal lighting conditions than when using a standard detection approach. This idea was inspired by [15].

To improve accuracy, a hard-coded method was also used to estimate locations where eye sockets are expected to be. Even if a Haar-like face detector temporarily fails, non-rotated eye regions can be expected to be almost at their

previous locations. As a result, it is possible to estimate their positions depending on the face width and height. Our experiments show that the eye-region width can be assumed to be around 33% of the face's width, and the eye-region height is around 25% of the face's height. The position of the left eye begins at about 13% from the left border of the face, and at about 27% from the top of the face.

2.2 Adaptive Binarization

We apply four hierarchical steps to limit the eye-socket region where we search for the pupil: binarization, edge (or contour) detection, determining the longest contour, and image cropping.

We tested various thresholding techniques [1], e.g. a basic *inverted* thresholding based on the minimum pixel value T of the eye region:

$$D(x,y) = \begin{cases} 0 & \text{if } S(x,y) > T \\ 255 & \text{otherwise} \end{cases} \tag{1}$$

This proved to be too unstable and worked only reasonable under very ideal lighting conditions. Otsu's adaptive thresholding had the same problems; results were also not adaptive to lighting conditions.

Due to the failures of the above mentioned thresholding methods, we decided to use *p-tile thresholding* as described in [2]. An adaptation to the light conditions according to pixel intensities was added to the basic technique. The first step of this method is the creation of a grey-level histogram of the image. The choice of threshold value T depends then on the chosen percentage parameter *p-value*. This parameter is a percentage value by which the size of the vector is multiplied. By taking the total value of the multiplication result, the position in a vector is obtained. The value hidden under this position is the query threshold value T. Using the calculated T, we applied basic thresholding as described in Equ. (1). In order to find an appropriate p-value, we calculate *mean* and *mode* M of pixel intensities. By combining these two parameters we identify a parameter I_p which determines the appropriate p-value. We describe I_p as follows:

$$I_p(\alpha) = \alpha * M_P + \frac{1-\alpha}{n} \cdot \sum_{i=1}^{n} P(x_i, y_i) \tag{2}$$

where P is the considered input window having n pixels. Following [15], this process (i.e. calculation of p-value and binarization) is performed independently for both eye regions. An adjustable parameter α is used to manipulate the influence of mean and mode values. Experimentally we identified $\alpha = 0.6$ as the mode intensity which led to better evaluations of the brightness of the pixels.

The next step is to perform a dilation in the binarized image and to find the smaller eye region by cropping the window for the largest region. Results of these steps are illustrated in Fig. 1.

2.3 Pupil Detection

We use the gradient method [18] for pupil detection due to its efficiency. Applying the method on the BioID database [3] we obtained very reasonable results. The

method may fail in cases when the image is too blurry or the eyelashes are very long and dark. Among all the tested methods the method led to a smaller number of false positives. Figure 2 illustrates samples of positive and negative results on the BioID database [3].

In a video sequence, errors can be reduced by applying some averaging over the last few frames. At the start of the program, we spend a few seconds to calibrate the gradient method. During calibration, the recorded subject is expected to look straight into the camera. After 20 frames, the averaging process starts for subsequent pupil detection.

2.4 Eye Corner Detection

Pupil detection is followed by localizing eye corners. For corner detection we use FAST features [16] with added adaptive thresholds. Parameters for the FAST algorithm are adjusted depending on pixel intensities, by following the same idea of using mean and mode values of the eye region as already discussed for the binarization process. The eye region is divided into two parts according to the location of the pupil, and two independent feature detection algorithms are applied on both halves.

Some of the detected FAST features could be false positives (e.g. at eyelashes, borders of glasses, or iris). To reject such FPs we assume that eye corners are maximally 50% of eye width and 30% of eye height away from the pupil. All features which satisfy this constraint are stored in a vector and averaging defines the final feature position. For more accurate results, we find the features that are most extreme to the left or right within the defined border.

In a video sequence, the errors can be reduced using temporally averaged positions of eye corners. We describe our algorithm by the following equations. First, we define the mean feature position $A(x, y)$ in one iteration as follows:

$$A(x,y) = \sum_{i=1}^{n} f(x_i, y_i) \quad \text{if} \quad P_y - b_y + c \leq f_{y_i} \leq P_y + b_y \tag{3}$$

where f is the set of detected features, P_y is a pupil location, b_y is an off-set for y-values, and c is an adjusting parameter. In our experiments we consider c as equals to 20% of the height of the eye region. The new corner position can be described as follows:

$$C(x,y) = \begin{cases} \frac{E(x,y)+A(x,y)}{2} & \text{if } (P_y - b_y + c \leq E_y \leq P_y + b_y) \wedge \\ & \quad (P_x \pm b_x \leq E_x \leq P_x \pm b_x + d) \\ E(x,y) & \text{if } A_x < P_x - 0.3 \cdot b_x \\ A(x,y) & \text{otherwise} \end{cases} \tag{4}$$

Fig. 1. Steps for extracting smaller eye regions. *Left to right:* Input image, *p-tile thresholding*, dilation result, and the final cropped eye region.

Fig. 2. Examples of pupil detection on BioID images. Both images on the left supported correct detections, and the other two led to incorrect detections.

where $E(x, y)$ is the most extreme (towards left or right) feature, b_y is again a y-offset and b_x is an offset for x values, $P(x, y)$ is the pupil location, and d is an adjusting parameter, which is experimentally set to an optimum value of 24% of the width of the eye region for the inner corner, and to 15% for the outer corner. The sign in front of b_x depends upon whether we calculate a new position for the inner or outer corner.

Finally we need to check whether our new corner candidates are not too far away from the averaged corner position of the last few frames, described as

$$A_f(x, y) = \frac{1}{20} \sum_{i=1}^{20} C'(x_i, y_i). \tag{5}$$

after having already the first 20 frames (before this time the average corner position is not yet calculated for avoiding false results). The final corner position is as follows:

$$C'(x, y) = \begin{cases} C(x, y) & \text{if } (A_{f_x} - 10 < C_x < A_{f_x} + 10) \wedge (A_{f_y} - 10 < C_y < A_{f_y} + 10) \\ A_f(x, y) & \text{otherwise} \end{cases}$$

2.5 Gaze Estimation

We consider the distance between the pupil and the eye corners for estimating the horizontal direction. We also use the distance between the center of an eye region and the pupil for estimating the vertical direction.

First, we define a line segment which connects both eye corners. Then we calculate the distance from the pupil to the line as follows:

$$d = \frac{(P_x - S_x) \cdot (D_y - S_y) - (P_y - S_y) \cdot (D_x - S_x)}{\sqrt{(D_x - S_x)^2 + (D_y - S_y)^2}} \tag{6}$$

where $P(x, y)$ is the pupil location, $S(x, y)$ is the one corner position, and $D(x, y)$ is the other one. We also calculate the distances between x-values of the pupil and both corners. Then we check the values of these three distances.

The information from both eyes is used in order to avoid estimation errors. For example, for our camera parameters, if the distance from the pupil to the line is less than -2 pixels, we assumed that the person is looking downward; if it is less than 3.5 and more than -1.8 we assume that the person is looking straight forward; if it is more than 3.5 we consider that the person is looking upward.

To determine whether a person is looking to the right or to the left, first, we check whether the distance between pupil and the outer left corner is less than

18 pixels. At the same time we check whether the distance between pupil and inner right corner is less than 20. To reduce errors, we also check whether the distance from the inner left corner and the outer right corner is more than 28. If 'yes' then we assume that the person is looking to the right. Similar conditions are checked for looking to the left. If none of the conditions are fulfilled then we assume that the person is looking straight forward.

3 Experimental Results

First, we performed face and eye-socket detection on the BioID database [3]. A combination of Haar-feature based eye detection and a hard-coded method for detecting eye sockets provided good results; only 0.15% of correctly detected faces had incorrectly detected eyes. Adaptive binarization also provided very good results: above 99.5% of images had a smaller eye region cropped correctly.

As a next experiment, we tested our method on six different subjects under different lighting conditions, including varying colors, from natural light condition to different artificial lights, obtained from various light sources. All the video sequences had a resolution of 640×48 pixels. Test subjects were at various age and of different gender. We also used two different webcams, one external and the other one integrated in a laptop. We used the first 20 frames for system calibration and for estimating potential pupil and eye-corner positions, in order to calculate mean positions for these eye features. Then the subjects were asked to look several times in four basically different directions (up, down, left, and right), and in four additional directions (e.g. upper left and right, bottom left and right). All results are recorded, frames are analyzed, and error rates are calculated. The evaluation of results has its limits; for example, with a naked eye, it is not feasible to determine the exact direction of the subject's eye gaze in dark, noisy, and low-resolution images. In order to evaluate the results it was assumed that a false detection of eye corners or of the pupil causes incorrect eye-gaze estimation. This might not be exactly true always. In some cases, pupil or eye corners are detected incorrectly but the detections are not too far away from the correct position, and it is still possible to have a correct final result for gaze

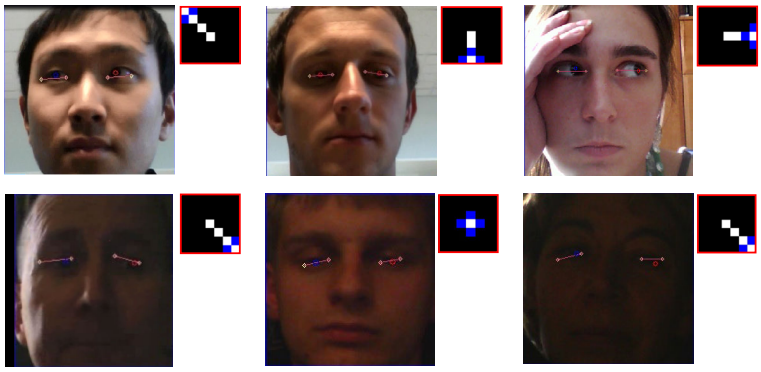

Fig. 3. Samples of eye-gaze detection under various lighting conditions.

Table 1. Percentages of correct eye-feature detections, separately for both eyes.

Number of test	1	2	3	4	5	6	7	8	9	10	11	12	13
Left eye [True positive (%)]	100	100	99	96	98	98	97	95	100	94	98	87	78
Right eye [True positive (%)]	98	100	99	100	93	96	100	95	100	77	64	57	59

estimation. Figure 3 presents results of eye-feature detection obtained for six different subjects, where the circles are symbolizing pupil and corner positions and a line between the corners.

Table 1 presents the final results obtained from 13 performed tests. The first ten were performed using an external webcam with higher image quality, and the last three are obtained using an integrated webcam in a laptop with poorer image quality. Also, the last three tests were performed for very dark faces and challenging lighting conditions (in shadows), as also illustrated in Fig. 3. The table confirms that the less noisy images support better results.

We realized that the most common reason of failures are dark spots near the eye regions (e.g. a very thick and dark frame of eyeglasses). Another problem appears if there are multiple dark and bright areas in the same eye; then the algorithm misses the eye-corner position and replaces it with a false-positive. We believe that this kind of problem could be avoided by adding an eye tracking solution into the algorithm. This could improve the stability of the gaze detection. When the eyes are closed then a (non-existing) false pupil might be detected as well; however, in most cases the eye corners are detected correctly. Our adaptive binarization and cropping algorithm worked efficiently in all the tested cases.

4 Conclusions

The detection of face features under non-ideal lighting conditions and in low-resolution images is a challenging issue. We had to face various complications for detecting points of interest accurately. The proposed and implemented adaptive algorithm ensures less vulnerability due to changes in lighting conditions. The paper outlined a robust solution for eye-gaze detection and suggests in particular a method for minimizing the region of interest. By using light-adapting *p-tile thresholding*, the method provides robustness and efficiency over a comprehensive range of experimental tests. For future work, the algorithm might be more stable by adding a tracking module and eye state detection. This can further reduce errors generated when the eyes are closed.

References

1. Klette, R.: Concise Computer Vision. Springer, London (2014)
2. Al-amri S. S., Kalyankar N. V.: Image segmentation by using threshold techniques. arXiv preprint arXiv:1005.4020 (2010)
3. BioID database, `www.bioid.com/index.php?q=downloads/software/bioid-face-database.html` (2014)

4. Bengoechea, J. J., Cerrolaza, J. J., Villanueva, A., Cabeza, R.: Evaluation of accurate eye corner detection methods for gaze estimation. Int. Worksh. Pervasive Eye Tracking Mobile Eye-Based Interaction (2013)
5. Florea, C., Florea, L.: Parametric logarithmic type image processing for contrast based auto-focus in extreme lighting conditions. Int. J. Applied Mathematics Computer Science, 23:637–648 (2013)
6. Fu, B., Yang, R.: Display control based on eye gaze estimation. IEEE Int. Conf. Image Signal Processing, Vol. 1, pp. 399–403 (2011)
7. Haiying, X., Guoping, Y.: A novel method for eye corner detection based on weighted variance projection function. IEEE Int. Conf. Image Signal Processing, 1–4 (2009)
8. Khilari, R.: Iris tracking and blink detection for human-computer interaction using a low resolution webcam. Indian Conf. Computer Vision Graphics Image Processing, pp. 456–463, ACM (2010)
9. Li, D., Winfield, D., Parkhurst, D.J.: Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. CVPR Workshops, volume 3, page 79–86 (2005)
10. Lienhart R., Maydt J.: An extended set of Haar-like features for rapid object detection. IEEE Conf. ICIP 2002, Vol. 1, pp. 900–903 (2002)
11. Lupu, R. G., Ungureanu, F., Siriteanu, V.: Eye tracking mouse for human computer interaction. E-Health and Bioengineering Conference (EHB), pp. 1–4, IEEE (2013)
12. Markuš, N., Frljak, M., Pandžič, I. S., Ahlberg, J., Forchheimer, R.: Eye pupil localization with an ensemble of randomized trees. Pattern Rec., 47:578–587 (2014)
13. McMurrough, C. D., Metsis, V., Kosmopoulos, D., Maglogiannis, I., Makedon, F.: A dataset for point of gaze detection using head poses and eye images. J. Multimodal User Interfaces, 7:207–215 (2013)
14. Nagel, J. A., Kutschker, C., Beck, C., Gengenbach, U., Guth, H., Bretthauer, G.: Comparison of different algorithms for robust detection of pupils in real-time eye-tracking experiments. Biomed Tech, 58, 1 (2013)
15. Rezaei, M., Klette, R.: Adaptive Haar-like classifier for eye status detection under non-ideal lighting conditions. Int. Conf. Image Vision Computing New Zealand, pp. 521–526, ACM (2012)
16. Rosten E., Porter R., Drummond T.: Faster and better: a machine learning approach to corner detection IEEE Trans. PAMI, vol 32:105–119 (2010)
17. Schwarz, L., Gamba, H. R., Pacheco, F. C., Ramos, R. B., Sovierzoski, M. A.: Pupil and iris detection in dynamic pupillometry using the OpenCV library. IEEE Int. Conf. Image Signal Processing, pp. 211–215 (2012)
18. Timm, F., Barth, E.: Accurate eye centre localisation by means of gradients. Int. Conf. Computer Vision Theory Applications, Vol. 1, pp. 125–130 (2011)
19. Valenti, R., Gevers, T.: Accurate eye center location through invariant isocentric patterns. IEEE Trans. PAMI 34:1785–1798 (2012)
20. Viola P., Jones M. J.: Rapid object detection using a boosted cascade of simple features. IEEE Conf. CVPR (2001)
21. Zhang, W., Zhang, T-N., Chang, S-J.: Eye gaze estimation from the elliptical features of one iris. Optical Engineering 50.4: 047003-047003 (2011)
22. Zhou, R., He, Q., Wu, J., Hu, C., Meng, Q. H.: Inner and outer eye corners detection for facial features extraction based on ctgf algorithm. Applied Mechanics Materials, 58:1966–1971 (2011)
23. Zhu, J., Yang, L.: Subpixel eye gaze tracking. IEEE Int. Conf. Automatic Face Gesture Recognition, pp. 124–129 (2002)