Detecting Computer Vision Syndrome Using Eye Blink – An Experimental Evaluation

Qurban A Memon¹, Mohammad Y Hassan²
¹Associate Professor, EE Department;
²MSc student, EE Department, UAE University, Al Ain, 15551, UAE;
quurban.memon@uaeu.ac.ae

Abstract: Nowadays, people spend more time using computers, laptops, and smartphones, etc., that causes decrease in eye blink frequency. Since eye blink spreads tears on eye cornea to moisture, the reduction in blink rate results in eye redness and dryness. In this work, eye blink detector is reported that is utilized to prevent eye dryness and redness. A number of existing eye blink detection schemes are surveyed and then a solution is proposed based on analysis. The proposed solution is based on histogram back projection. The proposed approach detects opening and closing state of the eyes and then if blinking rate is determined to be less than normal, it raises flag for computer vision syndrome. The approach is tested on different data sets under various lighting conditions and its performance is compared and analyzed.

1. Introduction

Nowadays, it is commonly known that increase in the number of people using computers is proportional to increase in number of people suffering from symptoms, which are commonly known as Computer Vision Syndrome (CVS). Main symptoms are headache, eyestrain, irritated eyes, dizziness, and difficulty in refocusing the eyes. These are mainly caused by use of personal electronic devices such smart phones, tablets, computers, etc., for longer periods. Other reasons are environmental, which include improper lighting conditions or air passing by the eyes, etc. [1]. One can prevent dry eye by blinking frequently, which in turn spreads tear film to moisten eye surface. The observations collected tell that it is caused by intensive computing work that causes eye redness and redness, sandy-gritty eye irritation and fatigue [2]. According to National Institute for Occupational Safety and Health, the CVS affects about 90% of the people, who use computers continuously for a number of hours [3]. A study [1] conducted on about 800 students in Malaysia reported that students felt headaches and eyestrain, and almost 90% of them experienced some type of syndrome symptoms.

Literature Review: In literature, many research works can be found that report user tracking based on certain features, in hospitals and telecommunications, etc. [4-6]. As an example, the authors in [7] use Graphics Processing Unit (GPU) to facilitate real time computation of eye blink detection. This set up provided speedup of x10 over existing implementations For algorithm, the authors use Scale-Invariant Feature Transform (SIFT) to detect and track important feature points (like eyelids) to estimate eye blinks. However, the performance of this work is affected by vertical head movements or due to gaze lowering.

In another eye blink detection method [8], the authors examine optical flow estimation conducted on GPU for detecting fatigue developing in eyes. This method locates eyes and face position by three
different classifiers, and then feature points are detected by Kanade–Lucas–Tomasi (KLT) tracker. This tracker locates the position, which yields the best match based on spatial intensity information. First, the optical flow estimation is found to be dominant, and then threshold is adapted on data flow to detect eye blinking. The success rate reported was above 90%. Nevertheless, this approach suffers from the situation, when eyes move up and down quickly. Another work in [9] examines eyelid movements using normal flow for cases where normal flow is orthogonal to image gradient. The authors [9] claim computational effectiveness of this approach over the work in [8].

The authors in [10] use Gabor filter to extract contours within eyes [11], which is for detection of open and closed eyes. This filter exploits distance between top and bottom arc of the eye, as it is different for closed and open eyes. For this, a variance map is developed to calculate pixel intensity distribution from image mean value. As intensity distribution within closed and open arc of the eye varies during eye blink, this approach can be used for blink detection, as reported in [12].

Another blink detection approach is discussed in [13], which uses eye contour extraction as a deformable model featured by various landmarks as eye shape. This model learns each landmark as it appears, and then fits it to update eye shape. Finally, the distance is estimated between lower and upper eyelids to represent blink.

Correlation can also be used to measure similarity between closed and open eye because correlation coefficient decreases once someone closes eyes during blink. A work is reported in [14], where process of detecting eye blinks is performed in four steps: (a) detect eye area (b) differentiate between closed and open eyes (c) measure saturation to blink of an eye (d) decide whether it is conscious blink or true blink. Another similar work for eye blink detection analyzes lower and upper part of the eye (involving eyelids) to differentiate between mean intensity distributions during open and closed eyes.

In this work, we use experimental approach to determine success rate of histogram approach to estimate skin color in differentiating between open and closed areas of the eye, while blinking. In the next section, proposed approach is presented in detail, followed by experimental results in section 3. The conclusions are summarized in section 4.

2. Proposed approach

As the computer screen is monitoring the user face, the higher recognition rate within efficient computation becomes possible. This is accomplished by histogram back projection, where user skin color is represented by histogram. The probability map generated for the map using histogram back projection tells how well the image pixels represent distribution. This means, higher percentage of skin color pixels means eyes are closed, otherwise open. Obviously, a threshold is required to help in this decision. This is pictorially illustrated in Figure 1.

2.1 Preprocessing – Getting ready

In order to get ready, a camera needs to be connected to the system. For simplicity in our approach, a built-in camera ‘webcam’ was used to simulate computer vision syndrome environment. The format of the camera used was “MPEG_640x480”. Once webcam turns on, a frame is captured within a default frame rate of the camera. This is shown in Figure 1a.

2.2 Eye region detection

This phase requires the detection of face to provide eye region for processing. This is accomplished using Viola-Jones algorithm [15], which detect human upper body features such as face, mouth, nose, or eyes. Motivated by face detection features, this algorithm provides competitive object detection rates in real-time. The framework used in [15] exploits a variant of (AdaBoost) learning algorithm to select features and then train feature-based classifiers. This algorithm constructs a robust classifier as a combination of simple classifiers, and uses the following equation:

\[
C(x) = \text{sign} \left( \sum_{j=1}^{N} \beta_j c_j(x) \right)
\]  

(1)
where $\beta_j$ are coefficients and each weak classifier $c_j(x)$ is a threshold function based on the feature $F_j$, and is defined as:

$$c_j(x) = \begin{cases} -y_j & \text{if } F_j < T_j \\ y_j & \text{otherwise} \end{cases}$$

(2)

The threshold value $T_j$, coefficients $\beta_j$ and the polarity $y_j \in \pm 1$ are calculated during training [15]. It should be noted here that the object detected and shown using this classifier is little bit enlarged down to cover more skin color area to compensate eyebrow area.

Finally, a rectangle is drawn around the detected eyes region. This is illustrated in Figure 2.

**Figure 1a**: Preprocessing

- Getting ready
- Interface a camera
- Get seated in front of camera
- Capture a frame
- Detecet Eye region
- Crop Eye region
- Color Enhancement
- Convert to Grayscale
- Contrast Enhancement
- Convert to Binary scale
- Skin color percentage calculation
- Blink frequency updated
- Detect blink

**Figure 1b**: Blink Detection Algorithm

**Figure 2**: Eye detection

2.3 *Crop eyes region*

The purpose here is to separate eye region from remaining area of the image, as the other parts are not important. This is done through cropping. The output of this function is displayed in the Figure 3.
2.4 Color enhancement

After cropping, the eye region is enhanced to overcome effects of (dim) lighting, makeup effects, etc. Effectively, this step brightens the color intensity pixels, which brings up the brightness of essentially the same colors using the power of gamma (γ), where

\[ γ = \begin{cases} 
1 - \alpha, & \alpha > 0 \\
\frac{1}{1 + \alpha}, & \alpha \leq 0
\end{cases} \]

where 0 < \( \alpha < 1 \). The result of color enhancement is displayed in Figure 4.

\[
\text{Detected eyes} \quad \text{Brightened Detected eyes}
\]

\[\text{Figure 3. Cropped eye region} \quad \text{Figure 4. Brightened eye region}\]

2.5 Grayscale Conversion

The image enhanced in previous step is then mapped to grayscale by retaining only luminance component while dropping hue and saturation information. The following image in Figure 5 is the result of this step.

\[
\text{Detected eyes in gray} \quad \text{EQ detected eyes in gray}
\]

\[\text{Figure 5. Brightened eye region in grayscale} \quad \text{Figure 6. Enhanced eye region in grayscale}\]

2.6 Contrast Enhancement

The result in previous step (as shown in Figure 5) needs to be contrast increased before conversion to black and white for the purpose of skin color estimation. During this, the grayscale intensity values are mapped to new values such that only 1% of data is saturated at low and high intensities. The output of this step is shown in the image displayed in Figure 6.

2.7 Black and white Conversion

The purpose of this step is to convert contrast-enhanced image to black and white so that dark areas in previous step such as pupils, eye boundaries, and eyelashes map to black (with intensity zero), and all other area pixels representing skin color to white.

Effectively, the procedure replaces all pixels in the input image with luminance value greater than level=0.25 (equivalent to 64) with value 1 (or 255) and remaining pixels with value less than that with 0. In other words, skin color represents white, while remaining areas (such as pupil, eye boundaries, etc.) map to black. The output of this step is the following image (Figure 7).

\[
\text{Detected eyes in Black & White}
\]

\[\text{Figure 7. Eye region in black and white}\]

2.8 Percentage of skin color calculation

Based on previous step, the percentage of skin color is calculated to estimate, whether eyes are open or closed. This is calculated using skin color in black and white image. Each white and black pixel is counted in separate array to calculate sum of respective pixels. The absence of pupil, and presence of
eyelash cover will likely increase the white pixels, which in turn means closed eyes, otherwise the eye is open. Mathematically, the skin color (white area) is given by:

$$\text{Skin color percentage} = \frac{\text{White Area}}{\text{Total Image Area}} \times 100$$

(4)

As an illustration, this is shown in Figure 8.

![Detected eyes in Black & White](image1)

**Figure 8.** Open and closed eyes

2.9 Blink detection and frequency

The eye blink is related to detection of closed followed by open eyes. This in turn means, as stated previously, detection of skin color. A number of experiments were conducted to check the performance of this approach. Based on experiments, it was determined that when skin color percentage increases beyond 90%, it represents closed eyes. This value was, then selected as a threshold for eye blink detection. A counter was run to estimate blink frequency. In each experiment, when all previous steps were completed, the blinking frequency is incremented. For comparison purposes, the normal blinking frequency of 10 blinks per minute was used. If the counter value is less than this value, then it is estimated that computer vision syndrome is developing, and the user is thus notified to blink eyes frequently, or take a break.

| Trial | Real Status | Detected Status | Room Environment                  | Result    |
|-------|-------------|-----------------|-----------------------------------|-----------|
| 1     | Open        | Open            | Two fluorescent bulbs on top      | Success   |
| 2     | Closed      | Closed          | One fluorescent bulb on top       | Success   |
| 3     | Open        | Open            | One fluorescent bulb on top       | Success   |
| 4     | Open        | Open            | Two fluorescent bulbs on top      | Success   |
| 5     | Open        | Open            | Two fluorescent bulbs on top      | Success   |
| 6     | Closed      | Closed          | Two fluorescent bulbs (one left and right) | Success |
| 7     | Closed      | Closed          | Two fluorescent bulbs (one left and right) | Success |
| 8     | Closed      | Closed          | Two fluorescent bulbs (one left and right) | Success |
| 9     | Open        | Open            | Two fluorescent bulbs on top      | Success   |
| 10    | Closed      | Closed          | Two fluorescent bulbs on top      | Success   |
| 11    | Closed      | Closed          | One fluorescent bulb on top       | Success   |
| 12    | Closed      | Closed          | One fluorescent bulb on top       | Success   |
| 13    | Open        | Open            | One fluorescent bulb on top       | Success   |
| 14    | Closed      | Closed          | Two fluorescent bulbs (one left and right) | Success |
| 15    | Open        | Closed          | Two fluorescent bulbs (one on right only) | Failure  |
| 16    | Closed      | Closed          | One fluorescent bulb on top       | Success   |
| 17    | Open        | Open            | Two fluorescent bulbs on top      | Success   |
| 18    | Closed      | Closed          | Two fluorescent bulbs (one left and right) | Success |
| 19    | Open        | Open            | Two fluorescent bulbs on top      | Success   |
| 20    | Open        | Open            | Two fluorescent bulbs (one left and right) | Success |

Table 1. Experimental results for eye detection
3. Experimental Results
In order to test the approach, a code was written in Matlab and webcam was used to capture frame in a varied lighting environment. A number of trials were run on different persons. The results are displayed in Table 1. The only constraint used in this experiment was the requirement that the user is either reading a document, listening to music, or watching a video, which will force the person to align face to the screen in order for the camera to take a picture. Twenty (20) trials were run, which brought us 19 successes and one failure, with a success rate of 95%. These results are better than reported in [7] and [9]. The main reason behind a failure in one trial was poor lighting resulting in luminance change. The other notable issues observed in experiments were change in gaze direction, smiling, eyelid makeup, etc.

4. Conclusions
It was verified in experiments that Computer Vision Syndrome causes low blinking rate in persons. In the proposed scheme, this syndrome is detected by updating blink frequency using histogram back projection approach in detected eye region. The approach used in this paper provides success rate of 95%. However, the syndrome cannot be eliminated using this method. Nevertheless, at least, its effects can be reduced to minimum by alerting the user to take a break or blink more frequently.

References
[1]. Blehm, C., Vishnu, S., Khattak, A., Mitra, S., Yee, R.W., “Computer Vision Syndrome: A Review,” Survey of Ophthalmology, 2005, vol. 50, no. 3, pp. 253–262
[2]. Jeffrey A., “Computer Vision Syndrome,” Book Chapter in Visual Ergonomics Handbook, ed. Jeffrey A., pp. 23-36, Taylor and Francis, 2005
[3]. Ayudhaya, C., Srinark, T., “A method for a real time eye blink detection and its applications,” Proceedings of the 6th International Joint Conference on Computer Science and Software Engineering (JCSE), 2009, pp. 25–30
[4]. Qurban Memon, “Smarter Healthcare Collaborative Network,” Building Next-Generation Converged Networks: Theory and Practice, CRC Press, 2013.
[5]. Zeenat AlKassim, Qurban Memon, “Designing a Low-Cost Eyeball Tracking Keyboard for Paralyzed People,” Computers and Electrical Engineering, Vol. 58, pp. 20-29, 2017.
[6]. Q Memon, S Khoja, “RFID–based Patient Tracking for Regional Collaborative Healthcare,” International Journal of Computer Applications in Technology, 45 (4), pp. 231-244, 2012
[7]. Lalonde, M., Byrms, D., Gagnon, L., Teasdale, N., Laurendeau, D., “Real-time eye blink detection with GPU-based SIFT tracking,” Proceedings of the Fourth Canadian Conference on Computer and Robot Vision, Washington, DC, 2007, pp. 481–487
[8]. Liu H., Zhang T., Xie H., Chen H., Li F. (2016), Real-Time Driver Fatigue Detection Based on ELM. In: Cao J., Mao K., Wu J., Lendasse A. (eds) Proceedings of ELM-2015 Volume 2. Proceedings in Adaptation, Learning and Optimization, vol 7. Springer, Cham
[9]. Heishman, R., Duric, Z., “Using Image Flow to Detect Eye Blinks in Color Videos,” IEEE Workshop on Applications of Computer Vision, 2007, p. 52
[10]. Weitao, L.; KeZhi, M.; Hong, Z.; Tianyou, C., “Selection of Gabor filters for improved texture feature extraction,” IEEE International Conference on Image Processing (ICIP), 26-29 Sept. 2010.
[11]. Anjith George and Aurobinda Routray, “Fast and Accurate Algorithm for Eye Localization for Gaze Tracking in Low Resolution Images,” IET Computer Vision, 2016.
[12]. Królak, A. & Strumiłło, P., “Eye-blink detection system for human–computer interaction,” Univ Access Inf Soc (2012) 11: 409. https://doi.org/10.1007/s10209-011-0256-6
[13]. Bernard, F., Deuter, CE., Gemmar, P., Schachinger H., “Eyelid contour detection and tracking for startle research related eye-blink measurements from high-speed video records,” Comput Methods Programs Biomed. 2013 Oct;112(1):22-37. doi: 10.1016/j.cmpb.2013.06.003, Epub 2013 July 21.
[14]. Torii I., Ohtani K., Takami S., Ishii N., “Development of Eye-Blink Controlled Application for Physically Handicapped Children,” In: Corchado E., Lozano J.A., Quintián H., Yin H. (eds) Intelligent Data Engineering and Automated Learning, LNCS, Vol. 8669. Springer.
[15]. Viola, Jones., “Robust Real-time Object Detection,” International Journal of Computer Vision, 2001