frontiers in ICT

ORIGINAL RESEARCH
published: 22 March 2018
doi: 10.3389/fict.2018.00004

Mobile-Based Eye-Blink Detection Performance Analysis on Android Platform

*Md. Talal Bin Noman[1] and Md. Atiqur Rahman Ahad[2]**

[1] Department of Computer Science and Engineering, City University, Dhaka, Bangladesh, [2] Department of Electrical and Electronic Engineering, University of Dhaka, Dhaka, Bangladesh

In this article, we develop a real-time mobile phone-based gaze tracking and eye-blink detection system on Android platform. Our eye-blink detection scheme is developed based on the time difference between two open eye states. We develop our system by finding the greatest circle—pupil of an eye. So we combine the both Haar classifier and Normalized Summation of Square of Difference template-matching method. We define the eyeball area that is extracted from the eye region as the region of interest (ROI). The ROI helps to differentiate between the open state and closed state of the eyes. The output waveform of the scheme is analogous to binary trend, which alludes the blink detection distinctly. We categorize short, medium, and long blink, depending on the degree of closure and blink duration. Our analysis is operated on medium blink under 15 frames/s. This combined solution for gaze tracking and eye-blink detection system has high detection accuracy and low time consumption. We obtain 98% accuracy at 0° angles for blink detection from both eyes. The system is also extensively experimented with various environments and setups, including variations in illuminations, subjects, gender, angles, processing speed, RAM capacity, and distance. We found that the system performs satisfactorily under varied conditions in real time for both single eye and two eyes detection. These concepts can be exploited in different applications, e.g., to detect drowsiness of a driver, or to operate the computer cursor to develop an eye-operated mouse for disabled people.

Keywords: gaze tracking, Haar cascade, eye-center localization, template matching, eye-blink detection

OPEN ACCESS

Edited by:
Francesco Ferrise,
Politecnico di Milano, Italy

Reviewed by:
Andrej Košir,
University of Ljubljana, Slovenia
Ilaria Torre,
Università di Genova, Italy

****Correspondence:***
Md. Atiqur Rahman Ahad
atiqahad@du.ac.bd

Specialty section:
This article was submitted to Human-Media Interaction, a section of the journal Frontiers in ICT

Received: *09 August 2017*
Accepted: *16 February 2018*
Published: *22 March 2018*

Citation:
Noman MTB and Ahad MAR (2018) Mobile-Based Eye-Blink Detection Performance Analysis on Android Platform. Front. ICT 5:4. doi: 10.3389/fict.2018.00004

INTRODUCTION

Gaze tracking and eye blinking are informative and very important topics in the field of computer vision and face/emotion analysis to solve various problems. Eye blinking and gaze tracking have a wide range of applications in human–computer interaction (HCI), such as the eye typing applications as in Galab et al. (2014), mouse control for the disabled people (Mohammed and Anwer, 2014), microsleep prevention for safe driving (Danisman et al., 2010; Varma et al., 2012), and in advanced technological gadgets such as Google Glass (Lakhani et al., 2015). Eye tracking is also explored for analysis of autistic children and their behavior (Haque Syeda et al., 2017). This kind of research can be directed to various health-care systems (Ahad et al., 2018). Although significant amount of research on eye detection and tracking has been done in the last three decades, lots of challenges still remain. This is mostly due to the distinctiveness of eyes, the presence of occlusions, and the variability in location, scale, and illumination conditions (Hansen and Ji, 2010).

Explorations on gaze tracking are progressing recently at a good pace. Therefore, we can hope for some resounding development so that the impacts can be profound on human life. Usually, there are few head-mounted devices or costly devices for gaze/eye tracking (Elahi et al., 2013). In this work, we consider a low-cost Android mobile phone as a sensing device. A successful and robust system can have various profitable applications.

In this article, we address the problem of gaze tracking and blink detection using mobile phones on Android platform. Several Android-based approaches have been proposed by different researchers for eye-gaze understanding or gesture recognition based on eye-gaze analysis (Vaitukaitis and Bulling, 2012; Elleuch et al., 2014; Hosek, 2017). Normally, gaze tracker is an eye-attached lens that ensures extremely sensitive recordings of eye movement. But the hardwired lenses or head-mounted tracker proves to be rather intrusive (Elahi et al., 2013). Also, these devices are expensive and cannot be used by everyone. Therefore, we propose a mobile phone-based eye-tracking system on Android platform. It is preferable for both simplicity and cost-effectiveness. A simple Android mobile phone with a frontal camera is the only device we require. To implement the system for better and real-time detection, we have exploited the OpenCV library. We have merged boosted Haar cascade eye tree eyeglasses classifier and Normalized Summation of Square of Difference template-matching method for accurate blink detection and gaze tracking.

The article is organized as follows: in Section "Related Works," we discuss the related works on this topic. Section "Methodology" presents our methodology, followed by the experimental details, results in Section "Experimental Results and Analysis." In Section "Discussion," we analyze and discuss the experimental results and others. The concluding remarks are presented in Section "Conclusion."

RELATED WORKS

In this section, we present important works and aspects, which are related to eye movement, gaze tracking, eye-blink systems and their applications (Hansen and Ji, 2010; Galab et al., 2014; Calistra, 2017). These issues are related to action recognition, emotion analysis, gait recognition, and face recognition (Ahad, 2011, 2012, 2013a,b). Hansen and Ji (2010) extensively covered various approaches of eye detection and tracking in their review work. Chennamma and Yuan (2013) surveyed on eye-gaze tracking in a short manner. Another survey work on eye tracking, head pose, and gaze estimation is presented in Gaur and Jariwala (2014). Although the latter two works only focus on these issues and therefore, a number of other challenges still persist.

Elahi et al. (2013) proposed a webcam-based eye-tracker. It is suitable for a low power device. They used mean of gradient vector algorithm to detect eyeball center position. They resized the eye image to get optimum value position. After detecting the eyeball center, they tracked the movement of the eyeball. Although it provided stability to the system, it failed to detect any eye blink.

Amamag et al. (2003) tracked eyes' center in real time by considering a binary classifier and an unsupervised clustering approach. They achieved 88.3% detection result.

Morris et al. (2002) proposed an eye-blink detection method exploiting spatiotemporal filtering to locate head, and variance maps to find the eye feature points. LK tracker was used to tracking these eye features to find the position of the face region and to extract different eyelid movements. Their method achieved reasonably sound blink detection result.

An eye-gaze estimation method is proposed by Wisniewska et al. (2014) based on gradient-based eye-pupil detection and eye-corner detection. The performance of this approach was good and it is illumination adaptive.

Ince and Yang (2009) introduced an approach to detect eye blobs precisely in sub-pixel level for extracting eye features. The algorithm used OpenCV library. This approach found the eyeball location with its central coordinates. They exploited several examples on simple webcam images to demonstrate the performance of their approach.

Chau and Betke (2005) proposed a blink detection system based on real-time eye tracking with USB camera. They used template-matching approach to detect eyes and analyze eye blinks. They achieved good blink detection accuracy. It can distinguish between two eye states: open state and closed state.

Magee et al. (2004) also implemented in USB camera. In their system, they tracked a face by using multiscale template correlation approach. To detect whether the computer user is looking at the camera, or off to the left or right side, they exploited the symmetry between left and right eyes. The detected eye direction was used to control computer applications (e.g., games, spelling programs, etc.).

Galab et al. (2014) presented a webcam-based approach to detect eye blinks accurately. This system can classify the eye states (i.e., open or closed state) at each video frame automatically. They tested the system with the users who wear glasses and it can perform well without any restriction on the background.

Mohammed and Anwer (2014) proposed a real-time method for eye-blink detection. In this article, they proposed an eye-tracking algorithm by considering the position of the detected face. This approach has explored a smoothing filter to enhance detection rate.

Krolak and Strumillo (2012) proposed an eye-tracking and eye-blink method that is based on template matching. Based on this approach, a user can enter English/Polish text and browse the Internet by having an interface. The system was comprised of a simple web camera and a notebook, under normal illumination.

Hosek (2017) worked on mobile platforms, especially on Android operating system together with the OpenCV library. They used a driver drowsiness detection system using Haar classifier and template-matching approach. But it works on only eye-center localization and tracking. That is not enough for drowsiness detection.

Kroon et al. (2008) analyzed eye localization. They found that eye-localization errors have an influence on face matching. Their eye-localization approach can enhance the performance of face matchers. However, in our work, we stress for a better eye localization and we do not concentrate on face matching context.

Timm and Barth (2011) proposed a method to localize eye center. The approach is based on image gradient vectors. In this approach, a simple objective function *f* is computed based on dot products. Then, the maximum (*f*) denotes the center of an eye, by having the intersecting points of most gradient vectors. It is invariant to changes in pose, scale, light, and contrast.

Grauman et al. (2003) proposed two video-based HCI tools called BlinkLink and EyebrowClicker. The first tool can automatically detect a user's eye blinks and accurately measure their durations. In their approach, any voluntary long blinks are able to trigger mouse clicks. However, it ignored any involuntary short blinks. This system used "blink patterns." But the system is not real time.

Soukupova and Cech (2016) proposed a system to detect eye blinks as a pattern of eye aspect ratio values which characterize the eye opening in each frame. They used support vector machine (SVM) classifier and achieved 90% accuracy rate.

Pauly and Sankar (2016) proposed a method for eye-blink detection based on Histogram of Oriented Gradients features, combined with SVM classifier. They achieved an accuracy of 85.62% on offline mode.

Worah et al. (2017) proposed an algorithm based on convolutional neural network to detect the eye states and to predict the blinks. This algorithm was applied both in still image and video feed dataset. They achieved very good results in all scenarios.

Liu et al. (2017) implemented ocular recognition for eye-blink detection. To select the best feature combinations, sequential forward floating selection (SFFS) was utilized. SFFS is an effective feature-selection approach (Jain and Zongker, 1997). Finally, the non-linear SVM was applied for classification. This method achieved good results for both open eye and blinking eye scenarios.

Ahmad and Borole (2015) proposed two techniques: Gabor filters and Circular Hough Transformation for a method for extracting eye circles from images. The main aim was to create an eye-blink detector.

In this article, we propose a real-time gaze tracking and eye-blink detection system that operates on a simple Android mobile phone having a frontal camera. To implement this system, we exploit the Haar classifier and Normalized Summation of Square of Difference template-matching method for the eye detection and gaze tracking for the first time. Based on our study, this combined approach has not been considered by others.

METHODOLOGY

The proposed system is a real-time system for detecting eye blink and gaze tracking features. This approach does not need any costly hardware system. For this research, a simple Android mobile phone with a frontal camera is the only device we need. In this section, we present the detailed architecture of the system. We explain the principle of each component.

The real-time system can be split into four sections: (A) real-time frame capturing and face detection, (B) eye area extraction, (C) eye-center detection and tracking; and (D) eye-blink detection. Finally, based on the detected eye blinks, an alarm alert implementation is devised for driving safety. **Figure 1** shows the block diagram of the system architecture.

Frame Capture and Face Detection

The first step of the entire process is the capturing frame from a frontal camera of any Android mobile phone. Afterward, the RGB frames are converted into grayscale frames by extracting only the luminance component (Mohammed and Anwer, 2014). The program searches for different facial features (e.g., eyebrows, eye corners, mouth and the tip of the nose, etc.) accurately within the scene. The Haar classifier is used for these facial features detection (Viola and Jones, 2001, 2004a). OpenCV library supported classifier, "Haarcascade_frontalface_alt.xml" is used. It is a feature-based face detection algorithm in cascaded form. It detects key face features from one after another in cascaded form.

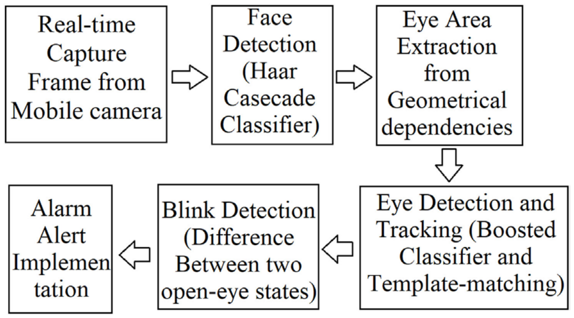
Real-time Capture Frame from Mobile camera
Face Detection (Haar Casecade Classifier)
Eye Area Extraction from Geometrical dependencies
Eye Detection and Tracking (Boosted Classifier and Template-matching)
Blink Detection (Difference Between two open-eye states)
Alarm Alert Implemen tation

FIGURE 1 | General flow diagram of the system.

If one fails to detect, it does not advance to the next feature. Haar classifier rapidly detects any object, based on detected features (not pixel values), similar to facial features. Haar features are computationally efficient (Hansen and Ji, 2010). In this approach, the detected area is partitioned and false positives are removed. Thereby, the face is detected (Mohammed and Anwer, 2014). (**Figure 2** demonstrates face detection. The detected area is enclosed by a rectangle.)

Eye Area Extraction

The second step is to extract the eye area in a face for better further computation. If a face is found in the image, its area of interest is reduced by region—region of interest (ROI) and the probable eye area. If the face is within a ratio, eyes are found in a certain position of the face. For example, let us assume that the face part is taken without hair or other distraction. It is a convention that, a face can be split into approximate six portions (few partitions are given in figure). The eyes are located about 0.4 of the way from the top of the head to the eyes (Kroon et al., 2008) (**Figure 3** depicts this concept, where "*h*" represents *height*). The ROI is now concentrated on the main eye area. From this area then extract the separate areas for the right and left eye, see **Figure 4** for further processing.

Eye Detection and Tracking

The fourth step is eye-center detection and tracking. Eye detection or localization can be done by various approaches, e.g., by finding intensity minima, by linear filtering of the face image and the Fisher Linear Discriminant classifier (Kroon et al., 2008), by analyzing the gradient flow which resulted from the dark iris surrounded by the sclera (sclera is the white of the eye or the outer layer of the eyeball) (Kothari and Mitchell, 1996), by detecting pupil using infrared or other special lighting (Haro et al., 2000; Morimoto and Flickner, 2000), by detecting sclera using color-based Bayes decision thresholds (Betke et al., 2000), etc. Haro et al. (2000) studied the physiological properties of eyes. They also studied the head and eye motion dynamics. To model eye or head dynamics, Kalman trackers are used. Moreover, a probabilistic-based appearance model was used to represent eye appearance. Another gradient-based method is proposed by Wisniewska et al. (2014) for eye-pupil detection. Initially, eyeball is extracted. It can be considered as a circular element with approximation, or semi-circular entity (Elahi et al., 2013).

FIGURE 2 | Face detection (experimental). Written and informed consent was obtained from the depicted individual for the publication of their identifiable image.

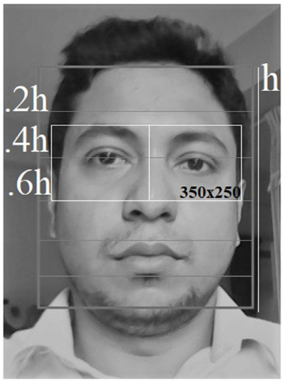

FIGURE 3 | The human face is split into few partitions, and eyes are located for further processing. Written and informed consent was obtained from the depicted individual for the publication of their identifiable image.

Therefore, we use Haar cascade eye tree eyeglasses classifier and template-matching approach. Only Haar classifier is time consuming and template matching has inadequate precision. We create a new approach combination of both solutions has excellent detection outcomes and also time-conservative. We find the eye in the desired ROI by OpenCV supported Haar cascade eye tree eyeglasses classifier. This is boosting classifier has a tree-based 20 × 20 frontal eye detector. This Tree-based AdaBoost is a machine learning boosting algorithm classifier consists of a linear combination of weak classifiers (Freund and Schapire, 1997). A weak classifier which correctly classifies a little bit more than half the cases. A weak classifier is mathematically described as follows:

$$h(x, f, \theta) = \begin{cases} 1 \text{ if } f(x) > \theta \\ 0 \quad \text{otherwise} \end{cases},$$

where x is a 20 × 20 pixel sub-window, f is the applied feature, and θ the threshold that decides whether x should be classified as a positive (eye) or a negative (non-eye).

It has a better handling of eyeglasses region than other Haar classifiers (Freund and Schapire, 1997; Caporarello et al., 2014; Hameed, 2017). In this region, the darkest point is searched for as a pupil representative. This point is as the center of the future eye pattern. The feature allows you to resize created templates depending on the size of the detected face.

Now, to create a run-in pattern or template (**Figure 5**), we select the location of the larger of the two components. In this case, we decide to keep the larger component, as it provides higher brightness cue. We select the larger element because the size of the template is proportional to the size of the element. Through this selection process, we can achieve higher tracking accuracy and more correlation scores (Grauman et al., 2003; Chau and Betke, 2005). **Figure 5** demonstrates the basic steps of a template creation for an eye. We want to produce the template

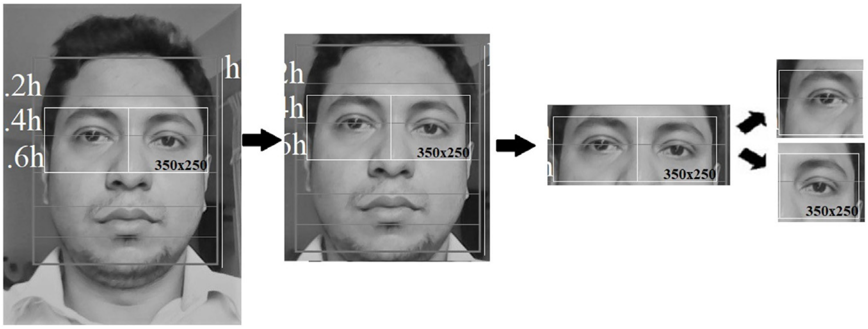

FIGURE 4 | Eye extraction and region of interest determination. Written and informed consent was obtained from the depicted individual for the publication of their identifiable image.

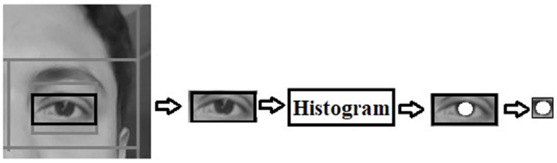

FIGURE 5 | Creating a run-in pattern or template of an open eye. Written and informed consent was obtained from the depicted individual for the publication of their identifiable image.

only when the eye is not blinking. The reason is that we want to make a system that will track a person's open eye state. Therefore, when the eye is considered to be located, we trigger a timer. For this system, we need to pass by the first few frames. This initial period is required for the eye to be open after any involuntary eye blinking. Afterward, we create the template of the user's open eye state. Note that, we assume that, initially a user blinks at a normal rate of one involuntary blink.

It is to be mentioned that we create only online templates. There is no requirement to have any prior offline templates for this purpose. Moreover, the produced online template is totally unrelated to any former templates, which may have been produced (Chau and Betke, 2005). We create a rectangle, which is centered in the pupil of the eye. This latest eye template is matched in the eye's rectangle area. Template matching is done here on a frame by frame basis. We try to match between an image patch and an input image by getting the summation of square of difference between the template and the source image. We find the position of the template in an image where the value is lowest. It is defined as follows:

$$R(x,y)=\sum_{x',y'}\left(T(x',y')-I(x+x',y+y')\right)^2,$$

where, $I(x, y)$ denotes image, $T(x, y)$ is the template, $R(x, y)$ is the result, $x' = 0, \ldots, w - 1$ and $y' = 0, \ldots, h - 1$. At each location, a metric is calculated. So it represents how similar the patch is to that particular area of the source image. For each location of $T(x, y)$ over $I(x, y)$, it stores the metric in the result matrix $R(x, y)$. Each location (x, y) in R contains the match metric.

We use in normalized form to divide the value of sum with the following expression:

$$R(x,y)=\frac{\Sigma_{x',y'}\left(T(x',y')-I(x+x',y+y')\right)^2}{\sqrt{\Sigma_{x',y'}T(x',y')^2\cdot\Sigma_{x',y'}I(x+x',y+y')^2}}$$

This operation makes the tracking feature more versatile. In normalized form, we can understand more about the percentage of matching.

Eye-Blink Detection

Blinking is considered as any rapid closing of the eyelid. In our eye-blink detection system is developed based on the time between two open eye states. The detection of blinking is based solely on the observation of the previous step by using the online template of the user's eye. In this approach, the open eyelid is taken as a template for detecting eye blink. As the user's eye opens during the process of a blink, we try to match the open eye template in the eye region rectangle area frame by frame. We use the normalized summation of square of difference method to find the minimum values in a given area.

When it gets the minimum values, it detects the center of the eye (pupil) and it draws a circle within a rectangular shape. It indicates that the eye is in an open state. Similarly, when the eyelid is closed, the template eye cannot match within the ROI. Therefore, the normalized form of the summation of square of difference method cannot find the minimum values in the ROI that indicates eye is in the closed state (**Figure 6**).

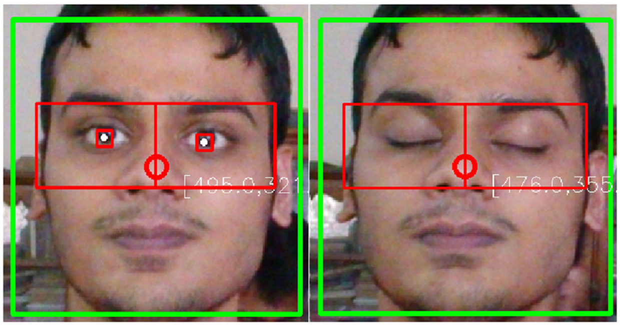

FIGURE 6 | Eye-blink detection (open and closed eye). Written and informed consent was obtained from the depicted individual for the publication of their identifiable image.

TABLE 1 | Test configuration of the system.

System	Android version 6.0
OpenCV	Version 2.4.6
CPU	Quad core 1.3 GHz
RAM	2.0 GB
Resolution	1,280 × 720 pixels
Resolution of front camera	2 MP
Average frame processing speed of the system	15 frames/s

EXPERIMENTAL RESULTS AND ANALYSIS

This section provides the necessary analysis using previously discussed method of gaze tracking and eye-blink detection. For researching gaze tracking and blink detection, we develop a dataset in different conditions, for example, different angles, different gender, the light source at different directions, and different Android mobiles with their different processors, RAM capacities and varied front camera pixels. We used four Android phones that were used for experiments. The basic configuration of an Android phone is provided in **Table 1**, which was considered for most of the dataset preparation.

The flowchart of the system is given in **Figure 7**.

We exploit a combined solution of tree-based boosted Haar classifier and Template-matching approach for a better result. So, we tested on two merged solutions; (i) Haar cascade left eye 2 splits classifier with template-matching approach and (ii) Haar cascade eye tree eye glasses classifier with template-matching approach. From **Table 2**, we clearly state that solution (ii) has low time-consumption than the solution (i) and two other single classifiers. Although only Haar cascade left eye 2 splits, classifier has better speed than Haar cascade eye tree eyeglasses classifier, and is used for detecting both open and close eye (Caporarello et al., 2014). It is a tree-based 20 × 20 left eye detector and trained by 6,665 positive samples from FERET, VALID, and BioID face databases. But solution (ii) achieves great speed than others because Haar cascade eye tree eye glasses classifier works only open eye (also on glass) (Caporarello et al., 2014) that is our main concern. So, template-matching approach with this classifier establishes low detection time.

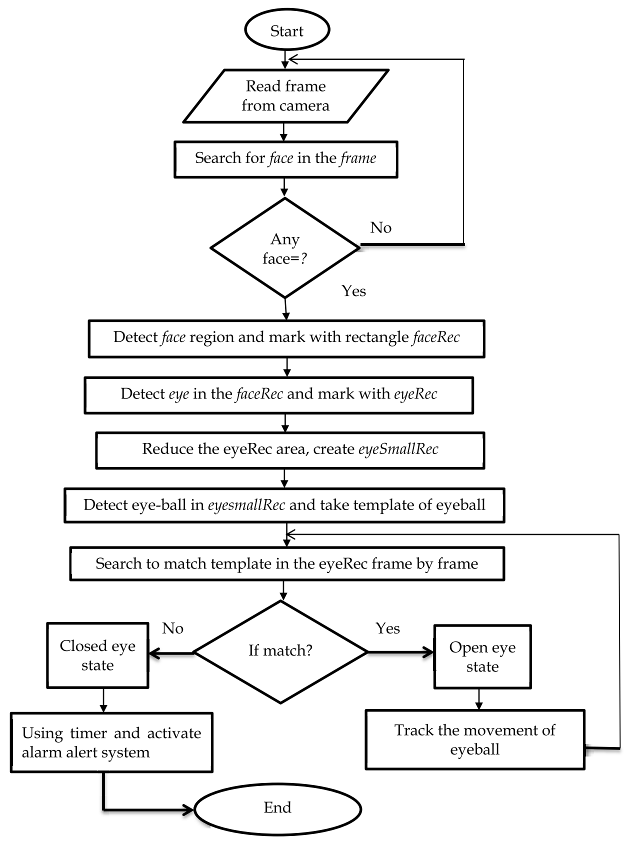
Start
Read frame from camera
Search for *face* in the *frame*
Any face=?
No
Yes
Detect *face* region and mark with rectangle *faceRec*
Detect *eye* in the *faceRec* and mark with *eyeRec*
Reduce the eyeRec area, create *eyeSmallRec*
Detect eye-ball in *eyesmallRec* and take template of eyeball
Search to match template in the eyeRec frame by frame
No
If match?
Yes
Closed eye state
Open eye state
Using timer and activate alarm alert system
Track the movement of eyeball
End

FIGURE 7 | Flowchart of the proposed system.

TABLE 2 | Performance comparison for different methods.

Method for detecting open eye	Average detection time in per frame
Haar cascade left eye 2 splits	8 ms
Haar cascade eye tree eye glasses classifier	9 ms
Haar cascade left eye 2 splits classifier with template-matching approach	5 ms
Our approach: Haar cascade eye tree eyeglasses classifier with template-matching approach	3 ms

The analyses aroused from different test subjects with a different number of samples (as shown in **Table 3**). In this experiment, total 100 subjects are considered, having 64 males and 36 females. All subjects are graduate and post-graduate students, ages ranging from 19 to 25 years. As we test on different cameras from different phone sets, the frame sizes are flexible too. However, the total number of samples taken under each condition has been the same. But experimental data are taken by mobile configuration in **Table 1**. In this research, the experimentations are carried out by using a low-cost and simple Android mobile phone having a frontal camera.

We analyze a 14 s captured online video frame by frame in milliseconds, the output pattern (from 7,000 to 14,000 ms) of correlation scores resembles as a binary trend (**Figure 8**) (Grauman et al., 2003; Chau and Betke, 2005). Small rectangle box with a white circle in each frame scores 1.0 is in open state, and the complement of the system scores 0 is in the close state. Based

TABLE 3 | Statistics of experimental data output for true detection of the eye blink based on the time between two open eye states using solution (ii).

Conditions	Total sample	Total error	% of Accuracy
0° (2 eyes), Average light from the front	100	2	98
0° (1 eye), Average light from the front	100	1	99
0° (2 eyes), Light from the side	100	24	76
0° (1 eye), Light from the side	100	8	92
0° (2 eyes), Light from the backside	100	80	20
0° (1 eye), Light from the backside	100	70	30
37° Left (2 eyes), light from the front	100	8	92
37° Left (1 eye), light from the front	100	3	97
37° Right (2 eyes), light from the front	100	9	91
37° Right (1 eye), light from the front	100	4	96
Male (2 eyes, light from the front, 0°, 37° left, 37° right)	192	7	96.35
Female (2 eyes, light from the front, 0°, 37° left, 37° right)	108	12	88.9
Male (1 eye, light from the front, 0°, 37° left, 37° right)	192	4	97.9
Female (1 eye, light from the front, 0°, 37° left, 37° right)	108	4	96.3

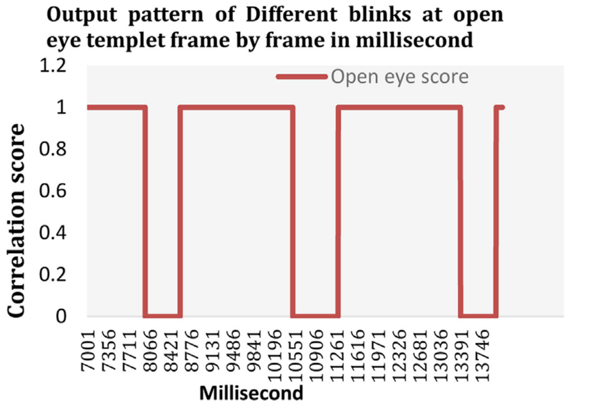

FIGURE 8 | Output waveform of different blinks is neighboring as Grauman et al. (2003), distinct clearly two open and close level like binary trend. The experiment is operated under 15 frames per second.

on the duration of closure time, we can categorize the eye blinks into three classes: long, medium and short blink. If the closure time is less than 500 ms, it is referred to as short blink. However, if it is around 1 s, is can be considered as medium blink. And a closure time greater than 1 s is treated as long blink. Long and medium blink detection accuracies are satisfactory. All experimental result analysis is conducted on medium blink under 15 frames per second. We obtain 98% accuracy for both eyes and 99% accuracy for single eye detection at 0° angle. Medium and long blink can be exercised as clicking cursor. On the other hand, detecting the driver's drowsiness can be favored by long blink. However, frequent short blinks are difficult to detect in low processing speed of RAM and processor. This accuracy partially can be improved by enhancing the speed of RAM and processor.

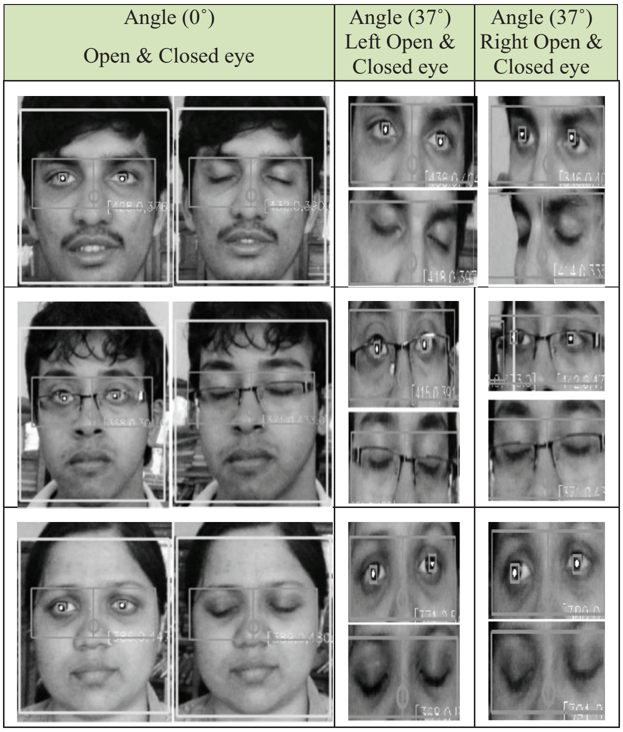
Angle (0˚)
Open & Closed eye
Angle (37˚) Left Open & Closed eye
Angle (37˚) Right Open & Closed eye

FIGURE 9 | Detection of the open eye state and the closed eye state at different angles. Written and informed consent was obtained from the depicted individual for the publication of their identifiable image.

DISCUSSION

In our experiment, we used OpenCV 2.4.6 and Android version 5.1.1–6.0.1. The developed system performs very well to detect the position of an eye. Face detection part is also very accurate in most of the cases. However, the tracking of an eyeball position fails in few instances. This tracking process depends on the quality of the camera and on the illumination conditions. We notice that in a poor lighting condition, the eyeball and the eye region become dark. In that condition, getting the gradient is a daunting task. Hence, getting the center positions demonstrate some errors. Moreover, the backside light creates a higher percentage of error in detecting a face and also locating any eye-center position. Under poor lighting condition, the performance degrades to some extends, especially when the light source is behind the face. Moreover, if multiple persons are on the scene, it detects multiple faces in the frame. This poses a problem. But most of the time, it detects the biggest face in the scene and works with it. It also detects the partial face. If the part of a face is not in the frame, it can detect the face and the eye-center position and works with it.

We have experimented to detect eye-blinks at different angles, e.g., angles at 0°, 37° left, and 37° right (see **Figure 9**). The experimental result shows that the angle at 0° has the minimum

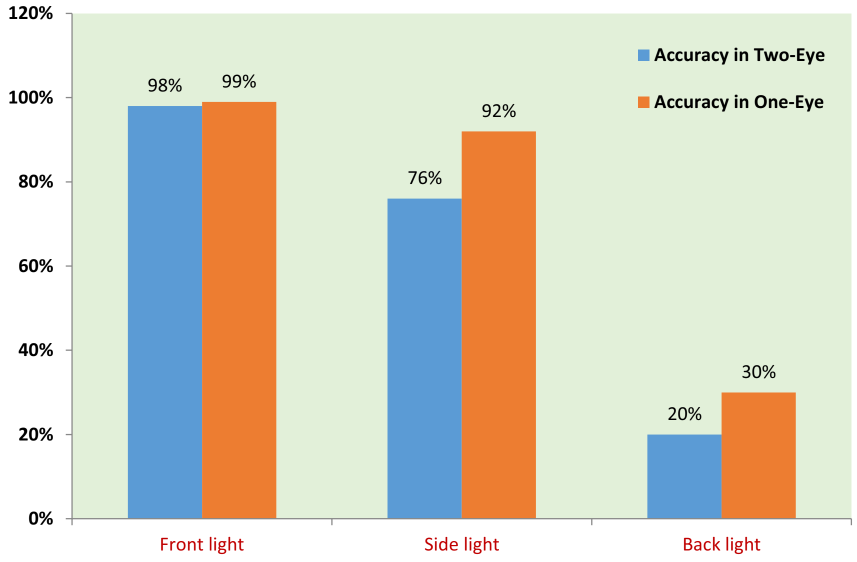

FIGURE 10 | Experimental detection accuracy in different light conditions (at 0° angle).

error for both one eye and two eyes detection. From our experiments, we determine a key feature, i.e., poor illumination level can demonstrate poor recognition. Based on our experimental results and analysis, we can find that the presence of backlight in a scene deteriorates the detection accuracy. So far, we obtain the detection accuracy of 20% for both eyes and 30% for a single eye. Still, the detection accuracy is prevalent 98% for both eyes and 99% for the single eye when the average light comes from the front (see **Figure 10**).

Contrary to the above statement on poor illumination, we find that the performance demonstrates fairly constant accuracy during steady indoor light status or under varied daylight conditions. However, if the camera is cheap (therefore, if the camera has less sensitivity), then the even indoor light condition cannot guarantee the decay of performance. On the other hand, if the camera sensitivity is better, then there is hardly any change in the lighting (whether it is indoor or outdoor). The light either only from the left side or the right side declines the detection accuracy,

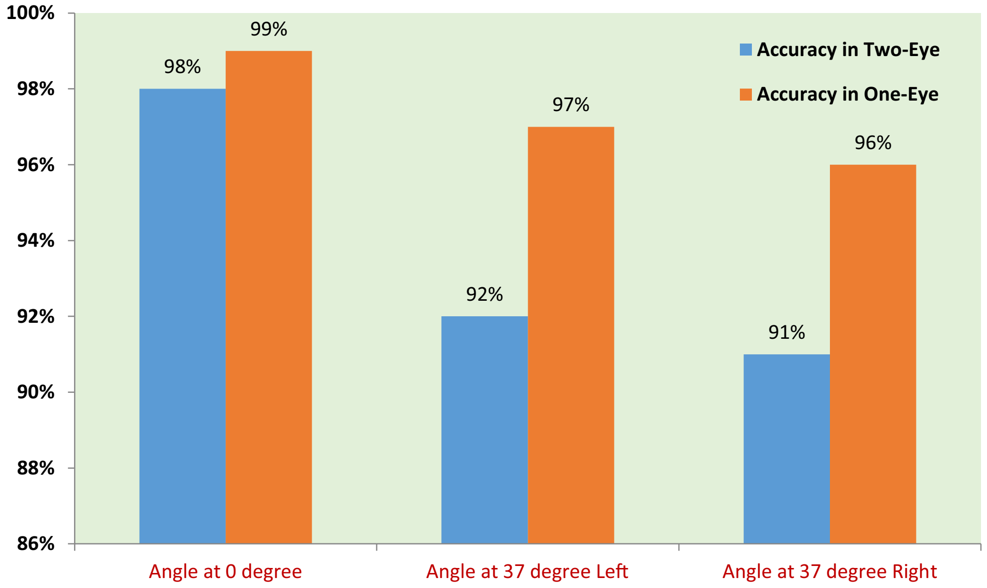

FIGURE 11 | Experimental detection accuracy for different angles (front light).

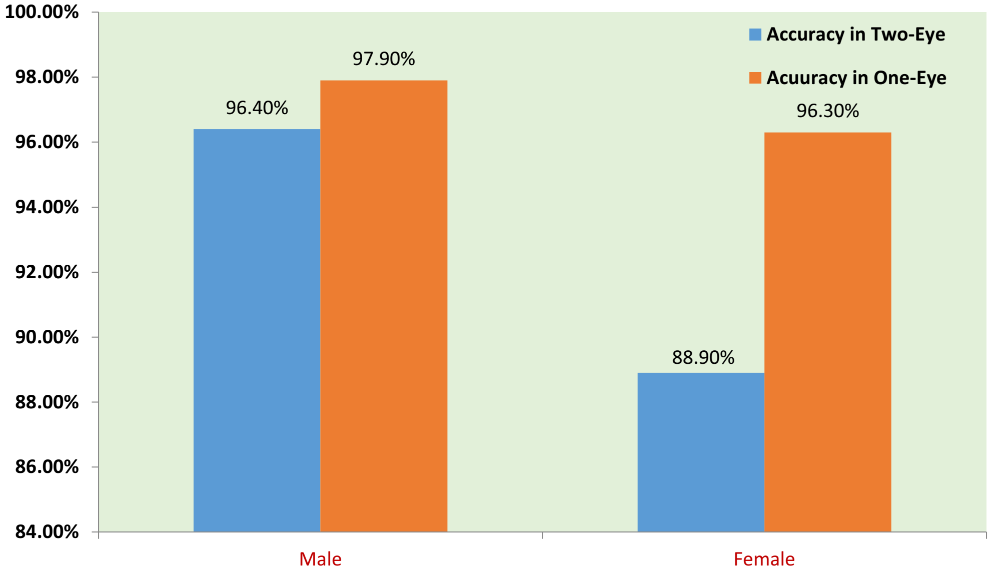

FIGURE 12 | Experimental detection accuracy under different genders.

and we obtain 76% accuracy for both eyes and 92% accuracy for the single eye.

Even though both eyes position's detection algorithms are the same, the detection of the right eye-center position shows little fluctuation and error than the left eye. The reason is that the mobile phone's front camera is always situated in the left side of the mobile phone at landscape position. And the distance of two eyes from the camera are not the same. The right eye distance from the camera is greater than the left eye distance. This is responsible for creating the errors (detection exactness: 92% for both eyes and 97% for the single eye at 37° left; 91% for both eyes and 96% for the single eye at 37° right). However, the percentage of error is insignificant and most of the time, it demonstrates accurate results. Therefore, we can neglect the errors easily (see **Figure 11**). It also drives that the single eye detection accuracy (shown in orange-colored bar in **Figure 11**) is always greater than the both eyes detection at any circumstance (see **Figures 10–12**).

We have also analyzed the performance on gender variations. Outputs vary due to the fact that female's long hairs mostly float around and it may partially occlude an eye to detect the eye properly. The presence of human hair, poor illumination condition, or narrow eyebrow can degrade the system performance (**Figure 11** demonstrates the result).

We also examine the performance of distance variations at an average lighting condition and 0° angles for both eyes detection (see **Figure 13**). From the graph, we can clearly sense that from

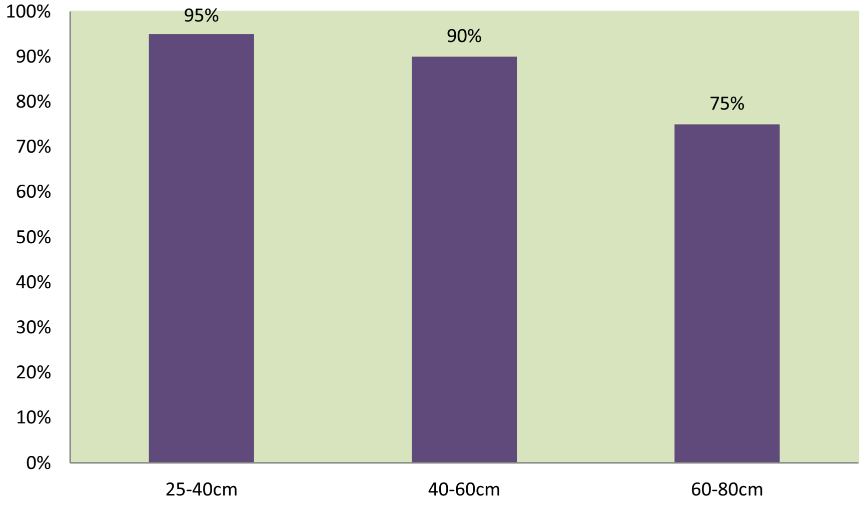

FIGURE 13 | Detection accuracy in different distances from a face (at an average illumination and 0° angles for both eyes).

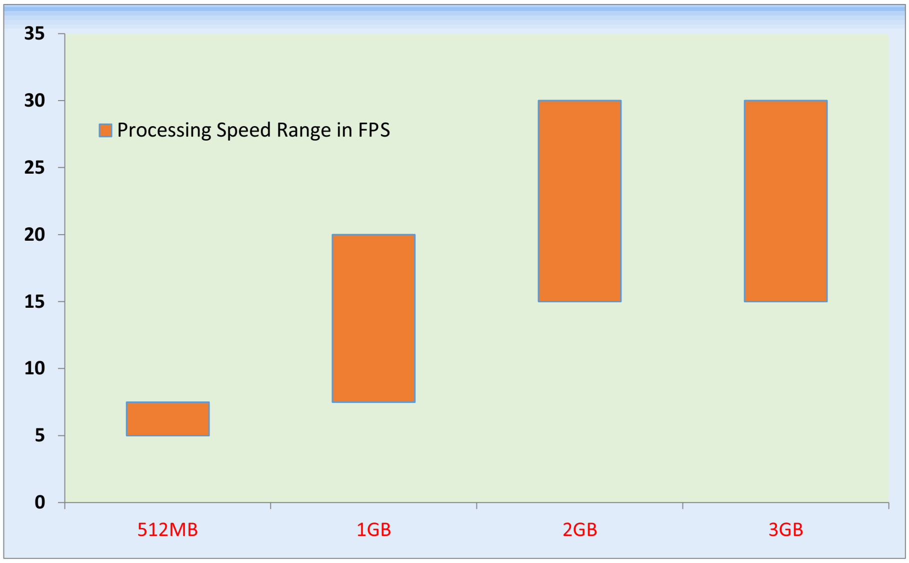

FIGURE 14 | Frame processing speed on different RAM sizes.

25 to 40 cm (we may consider it as the optimum distance), the percentage of detection accuracy is high (about 95%). From 40 to 60 cm, detection accuracy still maintains satisfactory (about 90%). However, if the distance is greater than 60 cm, detection accuracy degrades drastically. The most significant pronouncement is that if a distance from the face is below 25 cm, or above 80 cm, the detection accuracy becomes very poor.

Mobile RAM has great effect on frame processing speed depends on mobile processor, camera pixel and also hardware chipset. Analysis shows that high performance can obtain from high RAM space (see **Figure 14**). In our analysis we have taken processing speed range because same RAM space is used in different mobile processor in the world market.

We develop a real-time application for the purpose of safety-driving by employing the long blink detection. One of the major reasons for vehicle accident is microsleep of a driver on Highway. Therefore, we are trying to solve this problem by implementing microsleep alarm alert system through this real-time application of the gaze tracking and blink detection. We implement the alarm alert system in our application by software, and not on any extra hardware implementation. We exploit a dedicated mobile phone sound system for the alarm.

It works well with some limitations. The system takes more execution time while the face moves abruptly. Hence, the performance of the system degrades and we receive false-alarm. The alarm alert system becomes activated during the execution time between consecutive frames and in short blink (less than 500 ms). This issue can be mitigated by employing a mobile phone having a high-performance processor and implement a timer system that avoids the closure time is less than 500 ms.

CONCLUSION

In this article, we presented a low cost yet effective real-time gaze tracking and blink detection system. Our objective is to develop an easy to make and trivial to program Android mobile phone-based gaze tracker and blink detection system. The proposed system can easily be extended to a simple calibration gaze-based human-mobile interaction system. It only requires a simple Android mobile phone set under normal lighting on the face. We explored different algorithms and finally presented the best-fit approaches for our research. Through our extensive experimental results, we find that using the Haar classifier and the template-matching method has satisfactory detection speed and accuracy. It performs outstandingly in average light condition for single and both eyes detection at 0° angle. Other performance analysis results are also satisfactory with little error, which can be acceptable considering the level of difficulties of this kind of research in the field of computer vision and image processing.

In the future, we look forward to enhancing the performance of the system by testing on a larger number of subjects. Moreover, we can explore an algorithm to make the system illumination-invariant. Apart from these plans, we can evaluate the performance of (a) an eye movement-based mouse control system for disabled people and (b) a microsleep detection system for drivers using the gaze tracker and blink detector presented in this article.

ETHICS STATEMENT

An ethics approval was not required for this research according to the University of Dhaka's and City University, Bangladesh's guidelines and national laws and regulations. A written and informed consent was obtained from all research participants.

AUTHOR CONTRIBUTIONS

Both authors shared the work and writings equally.

ACKNOWLEDGMENTS

Authors are thankful to the participants for the data collection program for their time and to the Center for Natural Science & Engineering Research (CNSER) for its support for this research. Authors are also greatly thankful to Roman Hošek for supporting this research.

REFERENCES

Ahad, M. A. R. (2011). *Computer Vision and Action Recognition: A Guide for Image Processing and Computer Vision Community for Action Understanding*. Springer. doi:10.2991/978-94-91216-20-6

Ahad, M. A. R. (2012). *Motion History Images for Action Recognition and Understanding*. Springer. doi:10.1007/978-1-4471-4730-5

Ahad, M. A. R. (2013a). Smart approaches for human action recognition. *Pattern Recognit. Lett.* 34, 1769–1770. doi:10.1016/j.patrec.2013.07.006

Ahad, M. A. R. (2013b). Gait analysis: an energy image-based approach. *Int. J. Intell. Comput. Med. Sci. Image Process.* 5, 81–91. doi:10.1080/1931308X.2013.799313

Ahad, M. A. R., Kobashi, S., and Tavares, J. M. R. S. (2018). Advancements of image processing and vision in healthcare. *J. Healthc. Eng.* 2018:8458024. doi:10.1155/2018/8458024

Ahmad, R., and Borole, J. N. (2015). Drowsy driver identification using eye blink detection. *Int. J. Comput. Sci. Inf. Technol.* 6, 270–274.

Amamag, S., Kumaran, R. S., and Gowdy, J. N. (2003). "Real time eye tracking for human computer interfaces," in *Proc. of Int'l. Conf. on Multimedia and Expo (ICME)*, Vol. 3, Baltimore/Washington, DC, 557–560.

Betke, M., Mullally, W., and Magee, J. (2000). "Active detection of eye scleras in real time," in *IEEE Workshop on Human Modeling, Analysis and Synthesis*, Hilton Head, SC.

Calistra, C. (2017). *On Blink Detection and How It Works*. Available at: https://www.kairos.com/blog/130-how-to-use-blink-detection (accessed April, 2017).

Caporarello, L., Di Martino, B., and Martinez, M. (eds). (2014). *Smart Organizations and Smart Artifacts: Fostering Interaction between People, Technologies and Processes*. Springer, 140. doi:10.1007/978-3-319-07040-7

Chau, M., and Betke, M. (2005). *Real Time Eye Tracking and Blink Detection with USB Cameras*. Technical Report No. 2005-12. Boston: Boston University Computer Science.

Chennamma, H. R., and Yuan, X. (2013). A survey on eye-gaze tracking techniques. *Indian J. Comput. Sci. Eng.* 4, 388–393.

Danisman, T., Bilasco, I. M., Djeraba, C., and Ihaddadene, N. (2010). "Drowsy driver detection system using eye blink patterns," in *Intl. Conf. on Machine and Web Intelligence (ICMWI)*, Algiers, Algeria.

Elahi, H. M., Islam, D., Ahmed, I., Kobashi, S., and Ahad, M. A. R. (2013). "Webcam-based accurate eye-central localization," in *Second Intl. Conf. on Robot, Vision and Signal Processing*, Japan, 47–50.

Elleuch, H., Wali, A., and Alimi, A. (2014). "Smart tablet monitoring by a real-time head movement and eye gestures recognition system," in *2014 International Conference on Future Internet of Things and Cloud (FiCloud)*, 393–398.

Freund, Y., and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* 55, 119–139. doi:10.1006/jcss.1997.1504

Galab, M. K., Abdalkader, H. M., and Zayed, H. H. (2014). Adaptive real time eye-blink detection system. *Int. J. Comput. Appl.* 99, 29–36. doi:10.5120/17372-7910

Gaur, R., and Jariwala, K. (2014). A survey on methods and models of eye tracking, head pose and gaze estimation. *JETIR* 1, 265–273.

Grauman, K., Betke, M., Lombardi, J., Gips, J., and Bradski, G. R. (2003). Communication via eye blinks and eyebrow raises: video-based human-computer interfaces. *Univers. Access Inf. Soc.* 2, 359–373. doi:10.1007/s10209-003-0062-x

Hameed, S. (2017). *HaarCascadeEyeTree Eyeglasses Classifier*. Available at: https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_eye_tree_eyeglasses.xml (accessed February, 2017).

Hansen, D. W., and Ji, Q. (2010). In the eye of the beholder: a survey of models for eyes and gaze. *IEEE Trans. PAMI* 32, 478–500. doi:10.1109/TPAMI.2009.30

Haque Syeda, U., Zafar, Z., Zahidul Islam, Z., Mahir Tazwar, S., Jannat Rasna, M., Kise, K., et al. (2017). "Visual face scanning and emotion perception analysis between autistic and typically developing children," in *ACM UbiComp Workshop on Mental Health and Well-being: Sensing and Intervention*, Hawaii.

Haro, A., Flickner, M., and Essa, I. (2000). "Detecting and tracking eyes by using their physiological properties, dynamics, and appearance," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Hilton Head, SC, 163–168.

Hosek, R. (2017). *On OpenCV and Android*. Available at: http://romanhosek.cz/android-eye-detection-and-tracking-with-opencv (accessed January, 2017).

Ince, I. F., and Yang, T.-C. (2009). "A new low-cost eye tracking and blink detection approach: extracting eye features with blob extraction," in *Proc. of the 5th Int'l. Conf. on Emerging Intelligent Computing Technology and Applications*, Ulsan, 526–533.

Jain, A. K., and Zongker, D. (1997). Feature-selection: evaluation, application, and small sample performance. *IEEE Trans. Pattern Anal. Mach. Intell.* 19, 153–158. doi:10.1109/34.574797

Kothari, R., and Mitchell, J. L. (1996). Detection of eye locations in unconstrained visual images. *IEEE Int. Conf. Image Process.* 3, 519–522. doi:10.1109/ICIP.1996.560546

Krolak, A., and Strumillo, P. (2012). Eye-blink detection system for human-computer interaction. *Univers. Access Inf. Soc.* 11, 409–419. doi:10.1007/s10209-011-0256-6

Kroon, B., Hanjalic, A., and Maas, S. (2008). "Eye localization for face matching: is it always useful and under what conditions?" in *Proc. of the 7th ACM International Conference on Image and Video Retrieval (CIVR)*, Niagara Falls, Canada.

Lakhani, K., Chaudhari, A., Kothari, K., and Narula, H. (2015). Image capturing using blink detection. *Int. J. Comput. Sci. Inf. Technol.* 6, 4965–4968.

Liu, P., Guo, J. M., Tseng, S. H., Wong, K., Lee, J. D., Yao, C. C., et al. (2017). Ocular recognition for blinking eyes. *IEEE Trans. Image Process.* doi:10.1109/TIP.2017.2713041

Magee, J. J., Scott, M. R., Waber, B. N., and Betke, M. (2004). "Eyekeys: a real-time vision interface based on gaze detection from a low-grade video camera," in *IEEE Workshop on Real-Time Vision for Human-Computer Interaction (RTV4HCI)*, Washington, DC.

Mohammed, A. A., and Anwer, S. A. (2014). Efficient eye blink detection method for disabled helping domain. *Int. Adv. Comput. Sci. Appl.* 5, 202–206.

Morimoto, C., and Flickner, M. (2000). "Real-time multiple face detection using active illumination," in *4th IEEE Intl. Conf. on Automatic Face and Gesture Recognition (AFGR)*, Grenoble, 8–13.

Morris, T., Blenkhorn, P., and Zaidi, F. (2002). Blink detection for real-time eye tracking. *J. Netw. Comput. Appl.* 25, 129–143. doi:10.1006/jnca.2002.0130

Pauly, L., and Sankar, D. (2016). Non-intrusive eye blink detection from low resolution images using HOG-SVM classifier. *Int. J. Image Graph. Signal Process.* 10, 11–18.

Soukupova, T., and Cech, J. (2016). "Real-time eye blink detection using facial landmarks," in *21st Computer Vision Winter Workshop*, Slovenia.

Timm, F., and Barth, E. (2011). "Accurate eye centre localisation by means of gradients," in *Proc. of Intl. Conf. on Computer Vision Theory and Applications (VISAPP)*, Algarve, 125–130.

Vaitukaitis, V., and Bulling, A. (2012). "Eye gesture recognition on portable devices," in *Proc. of Conf. on Ubiquitous Computing*, Pittsburgh, PA, 711–714.

Varma, A., Arote, S., and Bharti, C. (2012). Accident prevention using eye blinking and head movement. *Int. J. Comput. Appl.* 18–22.

Viola, P., and Jones, M. (2001). "Robust real-time face detection," in *Proc. of 8th Int'l. Conf. on Computer Vision*, Vol. 2, Vancouver, BC, 747.

Viola, P., and Jones, M. (2004a). Robust real-time face detection. *Int. J. Comput. Vis.* 57, 137–154. doi:10.1023/B:VISI.0000013087.49260.fb

Wisniewska, J., Rezaei, M., and Klette, R. (2014). Robust eye gaze estimation. *Comput. Vis. Graph.* LNCS 8671, 636–644.

Worah, G., Khan, A., Kothari, M., and Naik, M. (2017). Monitor eye-care system using blink detection—a convolutional neural net approach. *Int. J. Eng. Res. Technol.* 6, 12–16.

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2018 Noman and Ahad. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.