Real-Time Eye Blink Detection Using Facial Landmarks & ASSIGNING PARTICULAR OUTPUT BASED ON BLINKING

A Mini-project Report submitted in partial fulfillment of the requirements for the award of the degree of

Master of Science in Computer Science

by

Arghyadeep Mondal

19419CMP001

Department of Computer Science

Institute of Science

Banaras Hindu University, Varanasi – 221005

May 2018

CANDIDATE'S DECLARATION

I **Arghyadeep Mondal** hereby certify that the work, which is being presented in the Mini-project report, entitled **Real-Time Eye Blink Detection Using Facial Landmarks** in partial fulfillment of the requirement for the award of the Degree of **Master of Science in Computer Science** and submitted to the institution is an authentic record of my/our own work carried out during the period November-2020 to February-2021 under the supervision of **Dr. Manoj Kumar Singh**. I also cited the reference about the text(s) /figure(s) /table(s) /equation(s) from where they have been taken.

The matter presented in this Mini-project as not been submitted elsewhere for the award of any other degree or diploma from any Institutions.

Date: Signature of the Candidate

The Viva-Voce examination of ___*Candidate Name*___, M.Sc. (Computer Science) Student has been held on _____.

Signature of the Supervisor

ABSTRACT

The main goal of this project is to detect the eye of a patient and count the blinking frequency of the eye to satisfy their need if any. At the elementary process, we detect the face of the patient and predict the shape of the face using the Dlib module after having video frames using the imutils module through camera, which was assigned to monitor the patient all the time. Other than this we will need a facial key points detector that can detect eyes in real-time. For this we will use a pre-trained network in the dlib library which can detect ’68 key points’ that is presented in this paper. The required pre-trained model can be downloaded from the internet. The dlib is used because it can give predictions in real-time.

The eye aspect ratio which is calculated after eye detection, will inspect the eye's present state i.e., open or close. In case of the value of EAR is greater than the predefined threshold, then we assume patient's that winking. This process is followed by the process of counting the frequency of the eye blinking. If the count value matches with the given threshold value, we would end with a prompting message.

Keywords: OpenCV, Dlib, Eye Tracking, EAR, Blink Detection, Blink Frequency Count.

TABLE OF CONTENTS

LIST OF FIGURES

Figure No. **Title** **Page No.**

LIST OF ABBREVIATIONS

ROI	Region of Interest
HOG	Histogram of Oriented Gradients
SVM	Support Vector Machine
EAR	Eye Aspect Ratio
CNN	Convolutional Neural Network
GPU	Graphics Processing Unit
IOT	Internet of Things
IEEE	Institute of Electrical and Electronics Engineers

1. INTRODUCTION:

Let us assume, a situation that a patient who has throat problem, is admitted in a hospital. In this scenario, it is compulsory to recruit a nurse to observe him/her all the time where the patient is unable to talk but can see, hear etc. The particular nurse sitting next to him won't be available even all the time, the nurse will be replaced by another nurse. In this era of technology, we can think a better solution for it excluding the traditional approach.

We can detect the face using Image Processing, the so-called technique is Face Detection. We can tell the patient to wink specified times if any service needed. Then we can use, the above-mentioned face detection process to satisfy patient's criteria, which will be more convenient way of dealing such problem like this.

2. PROPOSED APPROACH:

Our aim is to make out the exact time when the patient actually needs any help without any type of delay or interruption using the following methods:

- After getting the desired video frames, we detect the patient’s face in real time.
- Then the next module is designed to detect the eye from the image frame.
- Through which we can calculate the Eye Aspect Ratio using Euclidean Distance.
- Smaller value of EAR than the given threshold defines the drowsiness of the eye.
- Each time the patient winks, we count the frequency of the blinking.
- In case of equal value of both the threshold and blinking frequency, we prompt a message.

3. IMPLEMENTATION:

At the very first of the process, we get the image frames of the patient face from the installed camera. With respect to the criteria of Dlib (The module we are going to use in our implementation) a front facing camera is installed, which is connected to the executing device.

For sake of our project, it's better to mention that, CNN is much more robust and reliable compared to Dlib. The CNN based detector is capable of detecting faces almost in all angles. Unfortunately, it is not suitable for real time video. It is meant to be executed on a GPU. To get the same speed as the Dlib detector, we might need to run on a powerful GPU.

3.1. FACE DETECTION:

Face detection is the most popular area of research in the vision of computer science. It is a computer technology which is being used in a variety of applications that identifies human faces in digital images. The research under this field is expanding in many areas of science such as psychology. Localization of human faces is considered as the primary and the initial stage in study of face detection. For example, in home video surveillance etc. MATLAB and OpenCV are the popular tools for creating such prototypes and systems. But in this paper, we are going to use mostly Dlib and OpenCV to detect facial landmarks.

Facial landmarks are used to localize and represent salient regions of the face, such as:

- Eyes
- Eyebrows
- Nose
- Mouth
- Jawline

Figure 1: Facial Landmarks

Facial landmarks have been successfully applied to face alignment, head pose estimation, face swapping, blink detection and much more.

3.1.1 WHAT ARE FACIAL LANDMARKS?

Detecting facial landmarks is a subset of the shape prediction problem. Given an input image (and normally an ROI that specifies the object of interest), a shape predictor attempts to localize key points of interest along the shape.

In the context of facial landmarks, our goal is to detect important facial structures on the face using shape prediction methods.

Detecting facial landmarks is therefore a two-step process:

- Localize the face in the image.
- Detect the key facial structures on the face ROI.

3.1.2. VARIETY OF FACE DETECTION:

Face detection can be achieved in a number of ways.

We could use OpenCV's built-in Haar cascades.

We might apply a pre-trained HOG + Linear SVM object detector specifically for the task of face detection.

Or we might even use deep learning-based algorithms for face localization.

In either case, the actual algorithm used to detect the face in the image doesn't matter. Instead, what's important is that through some method we obtain the face bounding box (i.e., the (x, y)-coordinates of the face in the image).

The facial landmark detector included in the Dlib library starts by using:

- A training set of labeled facial landmarks on an image. These images are manually labeled,
- Priors, of more specifically, the probability on distance between pairs of input pixels.

The end result is a facial landmark detector that can be used to detect facial landmarks in real-time with high quality predictions.

3.1.3. DLIB'S FACIAL LANDMARK DETECTOR:

The pre-trained facial landmark detector inside the dlib library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on the face.

The indexes of the 68 coordinates can be visualized on the image below:

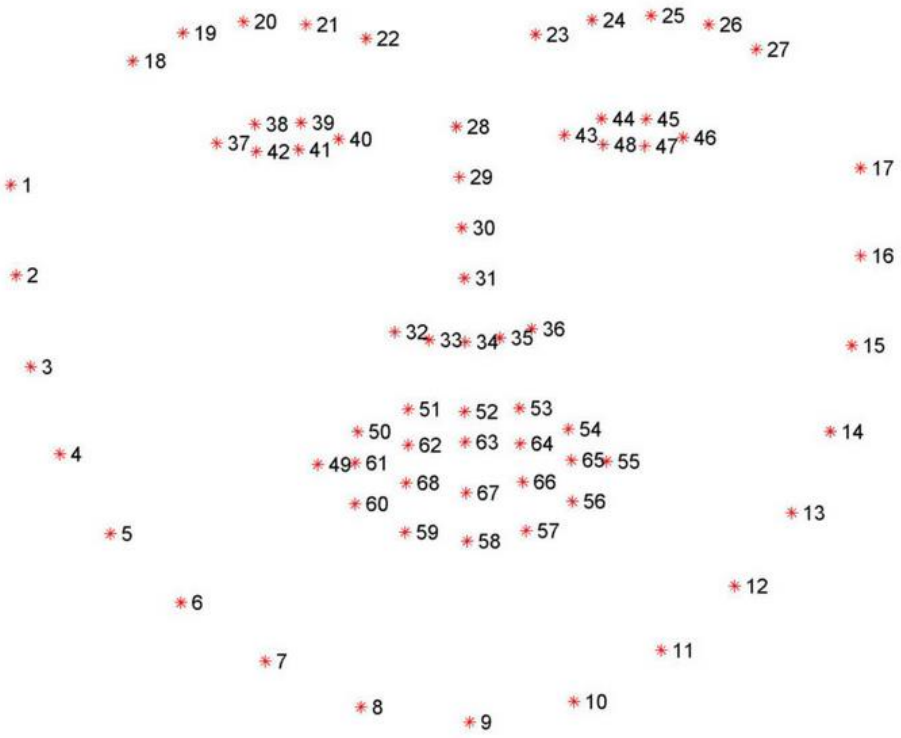

Figure 2: 68 Facial Landmarks

These annotations are part of the 68-point iBUG 300-W dataset which the Dlib facial landmark predictor was trained on. Regardless of which dataset is used, the same Dlib framework can be leveraged to train a shape predictor on the input training data.

3.2. EYE BLINK DETECTION:

Detecting eye blinks is important for instance in systems that monitor a human operator vigilance. Unlike traditional image processing methods for computing blinks which typically involve some combination of:

- Eye localization.
- Thresholding to find the whites of the eyes.
- Determining if the “white” region of the eyes disappears for a period of time (indicating a blink).

The eye aspect ratio is instead a much more elegant solution that involves a very simple calculation based on the ratio of distances between facial landmarks of the eyes.
This method for eye blink detection is fast, efficient, and easy to implement.

3.2.1. The “Eye Aspect Ratio” (EAR):

We can apply facial landmark detection to localize important regions of the face, including eyes, eyebrows, nose, ears, and mouth. This also implies that we can extract specific facial structures by knowing the indexes of the particular face parts.

In terms of blink detection, we are only interested in two sets of facial structures — the eyes.

Each eye is represented by 6 (x, y)-coordinates, starting at the left-corner of the eye (as if you were looking at the person), and then working clockwise around the remainder of the region.

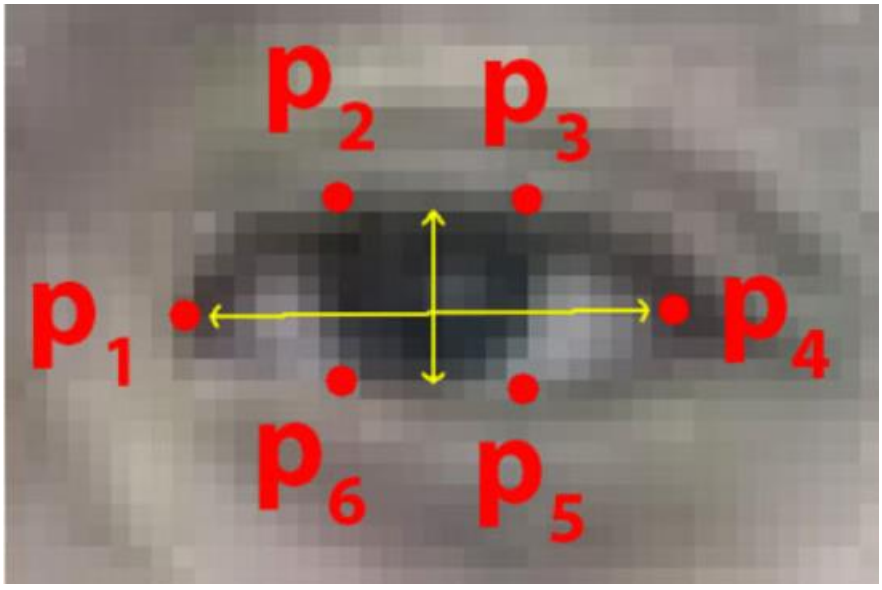

Figure 3: The 6 facial landmarks associated with the eye.

There is a relation between the width and the height of these coordinates. Based on the work by Soukupová and Čech in their 2016 paper, Real-Time Eye Blink Detection using Facial Landmarks, we can then derive an equation that reflects this relation called the eye aspect ratio (EAR):

$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Figure 4: The eye aspect ratio equation

Where p1, …, p6 are 2D facial landmark locations.

The numerator of this equation computes the distance between the vertical eye landmarks while the denominator computes the distance between horizontal eye landmarks, weighting the denominator appropriately since there is only one set of horizontal points but two sets of vertical points.

3.2.2. IMPORTANCE OF EYE ASPECT RATIO:

The eye aspect ratio is approximately constant while the eye is open, but will rapidly fall to zero when a blink is taking place. Using this simple equation, we can avoid image processing techniques and simply rely on the ratio of eye landmark distances to determine if a person is blinking.

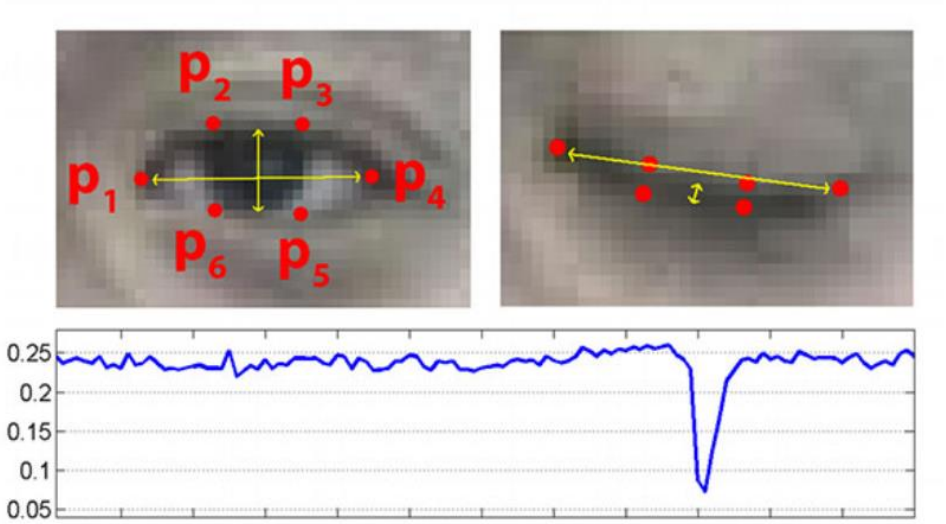

Figure 5: Eye Aspect Ratio of open and close eyes

To make this clearer, consider the following figure. On the top-left we have an eye that is fully open — the eye aspect ratio here would be large(r) and relatively constant over time.

However, once the person blinks (top-right) the eye aspect ratio decreases dramatically, approaching zero.

The bottom figure plots a graph of the eye aspect ratio over time for a video clip. As we can see, the eye aspect ratio is constant, then rapidly drops close to zero, then increases again, indicating a single blink has taken place.

3.3. ALGORITHM:

Priority needs to be given to the algorithms and functions mentioned below to obtain our goal.

3.3.1 CALCULATE EAR:

INPUT:

Eye indices, or landmarks of an eye.

OUTPUT:

EAR value of the eye.

EAR (eye) begin

1. Compute the Euclidean distances between the two sets of vertical eye landmarks (x, y)-coordinates.
    a) A = Euclidean distance (eye [1], eye [5])
    b) B = Euclidean distance (eye [2], eye [4])
2. Compute the Euclidean distance between the horizontal eye landmark (x, y)-coordinates
    a) C = Euclidean distance (eye [0], eye [3])
3. Compute the eye aspect ratio
    a) ear = (A + B) / (2.0 * C)
4. return ear

3.3.2 DETECTING DROWSINESS AND COUNT BLINKING FREQUENCY:

INPUT:

Video stream from web camera.

OUTPUT:

Frames containing the prompting message in help needed or not.

Main () begin

1. Start the video stream thread named vs
2. Loop over frames from the video stream while True
3. If this is a file video stream, then we need to check if there are any more frames left in the buffer to process
4. Grab the frame from the threaded video file stream by vs.read (), resize it using imutils.resize and convert it to grayscale channels by cv2.cvtColor
5. Detect faces in the grayscale frame using dlib.get_frontal_face_detector
6. Loop over the face detections
7. Determine the facial landmarks for the face region using dlib.shape_predictor method and 68 facial landmarks file as it's argument, then convert the facial landmark (x, y)-coordinates to a NumPy array
8. Extract the left and right eye coordinates, then use the coordinates to compute the eye aspect ratio for both eyes by using EAR function
9. Average the eye aspect ratio together for both eyes
10. Compute the convex hull for the left and right eye, then visualize each of the eyes (as human eyes are convex shaped)
11. Check to see if the eye aspect ratio is below the blink threshold (0.3), and if so, increment the blink frame counter
12. Otherwise, the eye aspect ratio is not below the blink threshold and if the eyes were closed for a sufficient number of times then increment the total number of blinks
13. Reset the eye frame counter

14. Draw the total number of blinks on the frame along with the computed eye aspect ratio for the frame by using cv2.putText
15. Show the frame
16. If the `q` key was pressed, break from the step 2 loop
17. Lastly, do a bit of cleanup

3.4 SIMPLE FLOWCHART OF THE PROCESS:

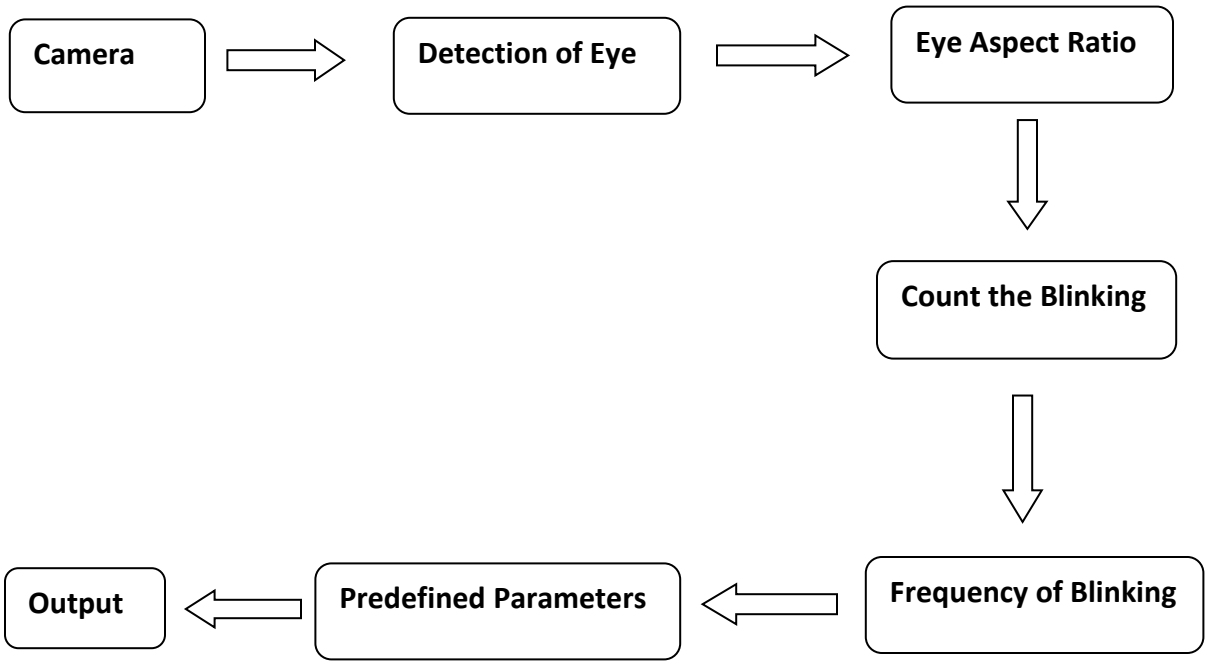

Figure 6: Flowchart of the Process

4. RESULTS AND DISCUSSION:

Finally, we reached our goal, now it's time to discuss our result. Two scenarios can be observed here:

Eventually, everyone winks after some interval of seconds. Hence, the convenient way is not just to count the blinks, but also put some threshold to determine the exact time of need. We defined the threshold here as 3 times or a multiple of 3 times. The value other than threshold will not entertain our approach. (Shown in the figure below.)

Figure 7: Twice Blink, No Response

When the patient actually needs any help, acts as exactly as the instruction given to him/her i.e., to wink 3 times or a multiple of 3 times. Here, our motive of this project goes to the conquest after prompting the message "Need Help". (Shown in the figure below)

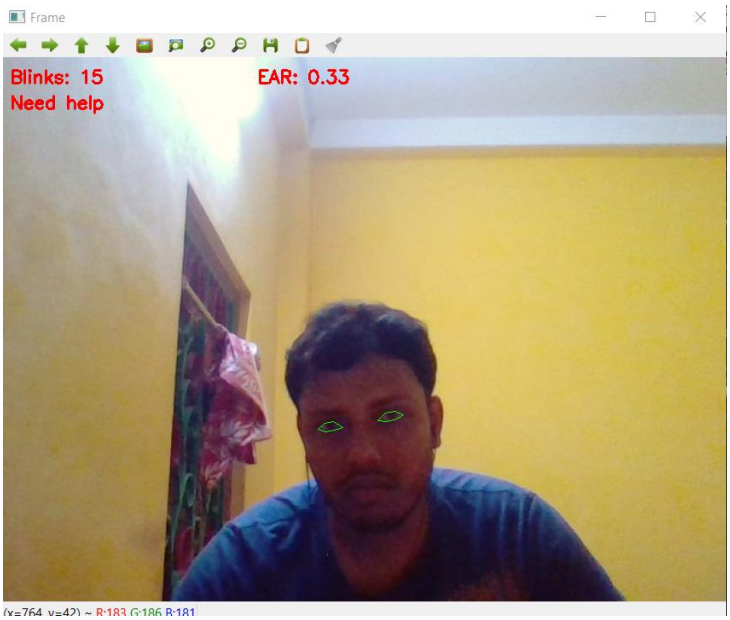

Figure 8: 3x5 times blink, 'Need Help' message

5. CONCLUSION AND FUTURE WORK:

5.1. CONCLUSION:

Here, I presented the algorithm for drowsiness detection and count the number of winks. The proposed method easily detects the blinks and drowsiness as the success rate is high. Our system is designed to detect drowsiness in the real time. The implementation of this approach runs at 25-30 frames per second. The application is implemented in Python using mainly OpenCV and Dlib library in Windows environment with a laptop web camera (should be front facing) view.

The Python Module that are used, presented in the list below:

- Dlib
- OpenCV
- Imutils
- NumPy
- Time
- SciPy

5.2. FUTURE WORKS:

Eye detecting as a tool is now more accessible than ever, and is growing in popularity amongst researchers from a whole host of different disciplines and have the potential to become an even more important component in future perceptual user interfaces.

Although, eye detecting has been massively popular for recent years, but the worthy implementation of this approach has not been so familiar yet. If the subsequent process could be controlled using the prosperous field of web technology or IOT, that the entire system will be accessible to the opted person of authority, will be much more desired.

Greater availability of the hardware configuration could bring us bigger approach of Image Processing or Deep Learning. That the process reaches to more accuracy irrespective of angles of the face or movement.

Beside this, such tool can be applied in various aspects:

- If a driver falls asleep while driving it causes numbers of accident. That's why a constant camera can observe him/her so that installed system can detect his/her eyes if it observes something abnormal then it will prompt a message in the speaker.

- Another application of this tool can be, if a person can't speak or hear, he/she can use his/her eyeblink to express his/her desire and the system will process the information and convert his expression in a written manner or a text output.

REFERENCES:

Journal Papers:

[1] Dhaval Pimplaskar, Dr. M.S. Nagmode, Atul Borkar, Real Time Eye Blinking Detection and Tracking Using Opencv, Int. Journal of Engineering Research and Applications ISSN: 2248-9622, Vol. 3, Issue 5, Sep-Oct 2013, pp.1780-1787

[2] Viola, Paul and Michael J. Jones. "Robust Real Time Face Detection" International Journal of Computer Vision 57.2 (2004): 137 154. Web.

[3] Bradski , Gary R and Adrian Kaehler. Learning Opencv . Sebastopol, CA: O'Reilly, 2008. Print.

[4] QiangJi, Xiaojie Yang, "Real-time eye, gaze, and face pose tracking for monitoring driver vigilance", Journal of Real-Time Imaging, Volume 8 Issue 5, October 2002, ISSN: 10772014, DOI: 10.1006/rtim.2002.0279

[5] Mu Li, Jia-Wei Fu and Bao-Liang Lu, "Estimating Vigilance in Driving Simulation using Probabilistic PCA", 30th Annual International IEEE EMBS Conference Vancouver, British Columbia, Canada, August 20-24, 2008 Electric Engineering and Computer August 19-22, 2011, Jilin, China

[6] Tereza Soukupova´ and Jan C˘ ech, Real-Time Eye Blink Detection using Facial Landmarks, 21st Computer Vision Winter Workshop Luka C˘ehovin, Rok Mandeljc, Vitomir S˘truc (eds.) Rimske Toplice, Slovenia, February 3–5, 2016

[7] A. Asthana, S. Zafeoriou, S. Cheng, and M. Pantic. Incremental face alignment in the wild. In Conference on Computer Vision and Pattern Recognition, 2014.

[8] L. M. Bergasa, J. Nuevo, M. A. Sotelo, and M. Vazquez. Real-time system for monitoring driver vigilance. In IEEE Intelligent Vehicles Symposium, 2004

[9] Kruti Goyal, Kartikey Agarwal, Rishi Kumar, Face Detection and Tracking Using OpenCV, International Conference on Electronics, Communication and Aerospace Technology ICECA 2017

[10] G. Pan, L. Sun, Z. Wu, and S. Lao. Eyeblink-based anti-spoofing in face recognition from a generic webcamera. In ICCV, 2007.

[11] Gary Bradski and Adrian Kaehler, "Learning OpenCV Computer Vision with the Open CV Library", O"Reilly,2008.

[12] Gonzalez, “Digital Image Processing”, Pearson Education India, 2009.

[13] M. Wang, H. P. Chou, C. F. Hsu, S. W. Chen, and C. S. Fuh, “Extracting Driver‟s Facial Features During Driving ”, 2011 14th International IEEE Conference on Intelligent Transportation Systems Washington, DC, USA. October 5-7, 2011

[14] Bergasa, L.M. Nuevo, J.Sotelo, M.A.Barea, R.Lopez, M.E “Real-time system for monitoring driver vigilance”, Intelligent Transportation Systems, IEEE Transactions on March 2006, Volume: 7 Issue: 1, on page(s): 63 – 77, ISSN: 1524-9050

[15] Duan-Sheng Chen , Zheng-Kai Liu, Aug 2011. “Generalized Haar-like features for Fast Face Detection”. Conference on Machine Learning and Cybernetics, 2007. Hong Kong, pp. 2131-2135

[16] Kirby and Sirovich, 1990. Application of KarhunenLoeve procedure for the characterization of human faces. IEEE Trans. pattern analysis and machine intelligence, 12:103-108.

[17] Longin Jan Latecki, Venugopal Rajagopal, Ari Gross, “Image Retrieval and Reversible Illumination Normalization”, SPIEIIS&T Internet Imaging VI, vol. 5670, 2005.

[18] Smita Tripathi,Varsha Sharma.(2011),” Face Detection using Combined Skin Color Detector and Template Matching Method”. International Journal of Computer Applications Volume 26– No.7.

[19] Neetu Saini, Sukhwinder Kaur, Hari Singh(2013). A Review: Face Detection Methods And Algorithms, International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Vol. 2 Issue 6

[20] Jalal, m. (2013). Drowsy Driver Warning System Using Image Processing. INTERNATIONAL JOURNAL OF ENGINEERING DEVELOPMENT AND RESEARCH

[21] WANGHUA DENG1, RUOXUE WU, Real-Time Driver-Drowsiness Detection System Using Facial Features, IEEE Trans. Intell. Transp. Syst., August 21, 2019

[22] Z. Liu, P. Luo, X. Wang, and X. Tang, ``Deep learning face attributes in the wild,'' in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 3730-3738

[23] S. M. Kamel and L. Guan, ``Histogram equalization utilizing spatial correlation for image enhancement,'' Proc. SPIE, vol. 1199, pp. 712-723, Nov. 1989

[24] R. O. Mbouna, S. G. Kong, and M.-G. Chun, ``Visual analysis of eye state and head pose for driver alertness monitoring," IEEE Trans. Intell. Transp. Syst., vol. 14, no. 3, pp. 14621469, Sep. 2013,

[25] A. Dasgupta, D. Rahman, and A. Routray, ``A smartphone-based drowsiness detection and warning system for automotive drivers," IEEE Trans. Intell. Transp. Syst., to be published.

[26] Y. Wu, T. Hassner, K. Kim, G. Medioni, and P. Natarajan, ``Facial landmark detection with tweaked convolutional neural networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 12, pp. 3067-3074, Dec. 2018.

[27] N.Wang and D.-Y.Yeung, ``Learning a deep compact image representation for visual tracking," in Proc. 26th Int. Conf. Neural Inf. Process. Syst., New York, NY, USA: Curran Associates Inc., vol. 1, 2013, pp. 809-817.

[28] D. B. Lucas and T. Kanade, ``An iterative image registration technique with an application to stereo vision," in Proc. 7th Int. Joint Conf. Artif. Intell., San Francisco, CA, USA, vol. 2, 1981, pp. 674-679

[29] M. Kowalski, J. Naruniec, and T. Trzcinski, ``Deep alignment network: A convolutional neural network for robust face alignment," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jul. 2017, pp. 2034-2043

[30] D. E. King, ``Dlib-ml: A machine learning toolkit," J. Mach. Learn. Res., vol. 10, pp. 1755-1758, Jul. 2009.