Drowsiness detection with OpenCV

Anakha K Babu, Bhagya C and Jithinya K
Department of Computer Science and Engineering
School of Engineering,Cochin University of Science and Technology

***Abstract*—Fatigue is one of the major causes of accidents in the world. Detecting the drowsiness of the driver is one of the surest ways of measuring driver fatigue. In this project we aim to develop a prototype drowsiness detection system. This system works by monitoring the eyes of the driver by a camera and sounding an alarm when he/she is drowsy.The system so designed is a non-intrusive real-time monitoring system. The priority is on improving the safety of the driver without being obtrusive. In this project the eye blink of the driver is detected. If the drivers eyes remain closed for more than a certain period of time, the driver is said to be drowsy and an alarm is sounded. The programming for this is done in OpenCV using the facial landmark localization for the detection of facial features.The pretrained 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.And the 68 point IBUG 300- W dataset which the dlib facial landmark predictor was trained on.B uilding upon this knowledge and develop a computer vision application that is capable of detecting and counting blinks in video streams using facial landmarks and OpenCV.To build our blink detector, we will be computing a metric called the eye aspect ratio (EAR).These EAR value is used to monitor the drivers drowsiness .Implementing these in Raspberry Pi 3 due to the real-world implications of building a driver drowsiness detector using very affordable hardware.**

I. Introduction

According to available statistical data, over 1.3 million people die each year on the road and 20 to 50 million people suffer nonÂŋfatal injuries due to road accidents.There are numerous nonÂŋdriver related causes of car accident including road conditions, the weather and the mechanical performance of a car.However, a significant number of car accidents are caused by driver error. Driver error includes drunkenness, fatigue, and drowsiness. Many factors can affect a driverâĂŹs ability to control a motor vehicle, such as natural reflexes, recognition and perception. The diminishing of these factors can eventually reduce a driver vigilance level.These statistics suggest that driver drowsiness is one of the main causes of road accidents. A driver who falls asleep at the wheel loses control of the vehicle, an action which often results in a crash with either another vehicle or stationary objects. In order to prevent these devastating accidents,the state of drowsiness of the driver should be monitored.The aim of this project is to develop a prototype drowsiness detection system. The focus will be placed on designing a system that will accurately monitor the open or closed state of the driver is eyes in realÂŋtime. By monitoring the eyes, it is believed that the symptoms of driver fatigue can be detected early enough to avoid a car accident by play an alarm if the driver feels drowsy. Detection of fatigue involves the observation of eye movements and blink patterns in a sequence of images of a face.

We are implementing this with openCV(open source computer visoin)is a library of programming functions mainly aimed at realÂŋtime computer vision.OpenCV was designed for computational efficiency and having a high focus on realÂŋtime image detection.One of OpenCVs goals is to provide a simpleÂŋtoÂŋuse computer vision infrastructure which helps people to build highly sophisticated vision applications fast. The OpenCV library, containing over 500 functions, spans many areas in vision. Because computer vision and machine learning often goes handÂŋinÂŋ hand,OpenCV also has a complete, generalÂŋpurpose, Machine Learning Library (MLL).This sub library is focused on statistical pattern recognition and clustering. The MLL is very useful for the vision functions that are the basis of OpenCV is usefulness, but is general enough to be used for any machine learning problem.Primary interface of OpenCV is in C++.There are also C, Python and

JAVA full interfacesPurpose of using openCV as follows:
1)OpenCV was designed for image processing. Every function and data structure has been designed with an Image Processing application in mind.
2)OpenCV however, we get actual real time processing at around 30 frames being processed per second.Sure we pay the price for speed a more cryptic language to deal with, but it is definitely worth it.
3)With OpenCV, we can get away with as littleas 10mb RAM for a real time application. Although with todays computers, the RAM factor is not a big thing to be worried about. However, our drowsiness detection system is to be used inside a car in a way that is nonÂŋintrusive and small so a low processing requirement is vital. 4)It provides hundreds of functions for the capture, analysis, andmanipulation of visual data and can eliminate some of the hassleprogrammers face when developing applications that rely on computer vision. We introduce, a computer vision system that can automatically detect driver drowsiness in a realÂŋ time video stream and then play an alarm if the driver appears to be drowsy.So the procedures of

drowsiness detection with opencv as follows.First, we will setup a camera that monitors a stream for faces.Look for faces in the input video stream.If a face is found, we apply facial landmark detection and extract the eye regions.Apply facial landmark localization to extract the eye regions from the face.Compute the eye aspect ratio to determine if the eyes are closed.If the eye aspect ratio indicates that the eyes have been closed for a sufficiently long enough amount of time, we will sound an alarm to wake up the driver

II. Problem Statement

In the 21st century, driver drowsiness has continued to be a major challenge contributing to a large number of accidents on our roads. In India, driver drowsiness especially among long distance truck drivers, public service vehicles drivers and private vehicle drivers is a major concern. This continues despite the government putting in place several measures to address the problem; measures including regulation of the public vehicle travel time, increasing the number of drivers for buses that travel at night, use of alcohol blows to detect drunk drivers among many others.Providing drowsiness detection system among drivers has not been achieved making it difficult to enforce relevant legislations. A few systems are available in the market however; they are expensive making them a reserve for a few who can afford the cost of the current vehicles fitted with search technologies. There is hence great need to provide drowsiness detection system that are affordable to the many who are low income earners and also public service vehicles to help address the many accidents associated with drowsiness.The Image processing used here achieves highly accurate and reliable detection of drowsiness than others.

III. Literature Review

Tereza SoukupovÃą and Jan ÄŇech intoduced Real-Time Eye Blink Detection using Facial Landmarks.A real-time algorithm to detect eye blinks in a video sequence from a standard camera is proposed. Recent landmark detectors, trained on in the wild datasets exhibit excellent robustness against a head orientation with respect to a camera, varying illumination and facial expressions. We show that the landmarks are detected precisely enough to reliably estimate the level of the eye opening. The proposed algorithm therefore estimates the landmark positions, extracts a single scalar quantity eye aspect ratio (EAR) characterizing the eye opening in each frame. Finally, an SVM classifier detects eye blinks as a pattern of EAR values in a short temporal window. The simple algorithm outperforms the state of the art results on two standard datasets.

A major drawback of the previous approaches is that they usually implicitly impose too strong requirements on the setup, in the sense of a relative face camera pose (head orientation), image resolution, illumination, motion

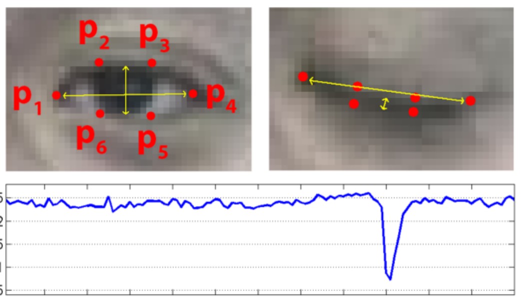

Figure 1: Open and closed eyes with landmarks p_i automatically detected by [1]. The eye aspect ratio EAR in Eq. (1) plotted for several frames of a video sequence. A single blink is present.

dynamics, etc. Especially the heuristic methods that use raw image intensity are likely to be very sensitive despite their real time performance.The eye blink is a fast closing and reopening of a human eye. Each individual has a little bit different pattern of blinks. The pattern differs in the speed of closing and opening, a degree of squeezing the eye and in a blink duration. The eye blink lasts approximately 100 to 400 ms. We propose to exploit state of the art facial landmark detectors to localize the eyes and eyelid contours. From the landmarks detected in the image,we derive the eye aspect ratio (EAR) that is used as an estimate of the eye opening state. Since the perframe EAR may not necessarily recognize the eye blinks correctly, a classifier that takes a larger temporal window of a frame into account is trained.

For every video frame, the eye landmarks are detected. The eye aspect ratio (EAR) between height and width of the eye is computed.where p 1 , . . . , p 6 are the 2D landmark locations, depicted in Fig. 1.The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye. It is partially person and head pose insensitive. Aspect ratio of the open eye has a small variance among individuals and it is fully invariant to a uniform scaling of the image and in-plane rotation of the face. Since eye blinking is performed by both eyes synchronously, the EAR of both eyes is averaged. An example of an EAR signal over the video sequence is shown in Fig. 1

However, due to noise in a video stream, subpar facial landmark detections, or fast changes in viewing angle, a simple threshold on the eye aspect ratio could produce a false-positive detection, reporting that a blink had taken place when in reality the person had not blinked.

To make our blink detector more robust to these challenges, SoukupovÃą and ÄŇech recommend:

Computing the eye aspect ratio for the N-th frame, along with the eye aspect ratios for N -6 and N + 6 frames, then concatenating these eye aspect ratios to form

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

a 13 dimensional feature vector. Training a Support Vector Machine (SVM) on these feature vectors.

Soukupova and Cech report that the combination of the temporal-based feature vector and SVM classifier helps reduce false-positive blink detections and improves the overall accuracy of the blink detector.

Vahid Kazemi and Josephine Sullivan introduced One Millisecond Face Alignment with an Ensemble of Regression Trees.This paper addresses the problem of Face Alignment for a single image. We show how an ensemble of regression trees can be used to estimate the faces landmark positions directly from a sparse subset of pixel intensities, achieving super-realtime performance with high quality predictions. We present a general framework based on gradient boosting for learning an ensemble of regression trees that optimizes the sum of square error loss and naturally handles missing or partially labelled data. We show how using appropriate priors exploiting the structure of image data helps with efficient feature selection. Different regularization strategies and its importance to combat overfitting are also investigated. In addition, we analyse the effect of the quantity of training data on the accuracy of the predictions and explore the effect of data augmentation using synthesized data.In this paper we present a new algorithm that performs face alignment in milliseconds and achieves accuracy superior or comparable to state-of-the-art methods on standard datasets. The speed gains over previous methods is a consequence of identifying the essential components of prior face alignment algorithms and then incorporating them in a streamlined formulation into a cascade of high capacity regression functions learnt via gradient boosting. The major contributions of this paper are

1. A novel method for alignment based on ensemble of regression trees that performs shape invariant feature selection while minimizing the same loss function during training time as we want to minimize at test time.

2. We present a natural extension of our method that handles missing or uncertain labels.

3. Quantitative and qualitative results are presented that confirm that our method produces high quality predictions while being much more efficient than the best previous method

The presented framework is faster in reducing the error compared to previous work and can also handle partial or uncertain labels. While major components of our algorithm treat different target dimensions as independent variables, a natural extension of this work would be to take advantage of the correlation of shape parameters for more efficient training and a better use of partial labels.

IV. Modular Breakup and Design

A. Rigging the car with a drowsiness detector

The camera used for this project was a Logitech C920 having features like:

i) Is relatively affordable.

ii)Can shoot in full 1080p.

iii)Is plug-and-play compatible with nearly every device tried it with.

Here we are planning to use Raspberry Pi 3 due to

(1) form factor and

(2) the real-world implications of building a driver drowsiness detector

using very affordable hardware,the Raspberry Pi isnt quite fast enough for real-time facial landmark detection, for the time being, well simply use a standard laptop computer. With all hardware setup, ready to move on to building the actual drowsiness detector using computer vision techniques.To optimize the Raspberry Pi along with the dlib compile to enable real-time facial landmark detection.

B. The drowsiness detector algorithm

step 1. Look for faces in the input video stream

step2.Apply facial landmark localization to extract the eye regions from the face.

Step3.Compute the eye aspect ratio to determine if the eyes are closed.

step4.Sound an alarm if the eyes have been closed for a sufficiently long enough time.

C. Building the drowsiness detector with OpenCV

Import some required Python packages.

1.dlib library To detect and localize facial landmarks we will need the dlib library ,to use facial landmarks for: Face part (i.e., eyes, nose, mouth, etc.) extraction ,Facial alignment , Blink detection

2.imutils package Used for image processing functions to make working with OpenCV easier.A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and both Python 2.7 and Python 3.

3. SciPy SciPy is a Python based ecosystem of open source software for mathematics,science,and engineering.The SciPy package so we can compute the Euclidean distance between facial landmarks points in the eye aspect ratio calculation

4. playsound library In order to actually play WAV/MP3 alarm, we need the playsound library, a pure Python, cross-platform implementation for playing simple sounds.

D. *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 detect important facial structures on the face using shape prediction methods.

Detecting facial landmarks is therefore a two step process:

Step 1: Localize the face in the image.

Step 2: Detect the key facial structures on the face ROI.

Face detection (Step 1) can be achieved in a number of ways.

We could use OpenCVs 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 doesnot matter. Instead, whats important is that through some method we obtain the face bounding box (i.e., the (x, y)-coordinates of the face in the image).

Given the face region we can then apply Step 2: detecting key facial structures in the face region.

There are a variety of facial landmark detectors, but all methods essentially try to localize and label the following facial regions:

Mouth

Right eyebrow

Left eyebrow

Right eye

Left eye

Nose

Jaw

The facial landmark detector included in the dlib library is an implementation of the One Millisecond Face Alignment with an Ensemble of Regression Trees paper by Kazemi and Sullivan (2014).

This method starts by using:

A training set of labeled facial landmarks on an image. These images are manually labeled, specifying specific (x, y)-coordinates of regions surrounding each facial structure.

Priors, of more specifically, the probability on distance between pairs of input pixels.

Given this training data, an ensemble of regression trees are trained to estimate the facial landmark positions directly from the pixel intensities themselves (i.e., no feature extraction is taking place).

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

For more information and details on this specific technique, be sure to read the paper by Kazemi and Sullivan linked to above, along with the official dlib announcement.

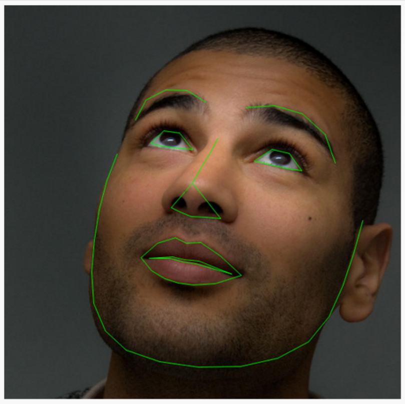

Figure 1: Facial landmarks are used to label and identify key facial attributes in an image (source).

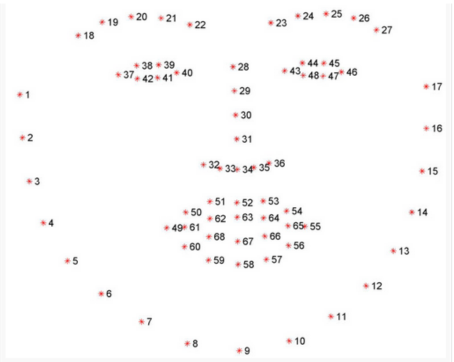

Figure 6: The full set of facial landmarks that can be detected via dlib (higher resolution).

E. *dlib- 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.

These annotations are part of the 68 point iBUG 300-W dataset which the dlib facial landmark predictor was trained on.

Its important to note that other flavors of facial landmark detectors exist, including the 194 point model that can be trained on the HELEN dataset.

Regardless of which dataset is used, the same dlib framework can be leveraged to train a shape predictor on the input training data this is useful if you would like to train facial landmark detectors or custom shape predictors of your own.

F. *EAR CALCULATION*

The return value of the eye aspect ratio will be approximately constant when the eye is open. The value will then rapid decrease towards zero during a blink. If the eye is closed, the eye aspect ratio will again remain approximately constant, but will be much smaller than the ratio when the eye is open. To visualize this, consider the following figure from Soukupova and Cechs 2016 paper, Real-Time Eye Blink Detection using Facial Landmarks.

Top-left: A visualization of eye landmarks when then the eye is open. Top-right: Eye landmarks when the eye is closed. Bottom: Plotting the eye aspect ratio over time. The dip in the eye aspect ratio indicates a blink (Figure 1 of SoukupovÃą and ÄŇech).

Consider the figure, Compute the euclidean distances between the two sets of vertical eye landmarks (x,y)-coordinates

let

A = dist.euclidean(eye[1], eye[5])

B = dist.euclidean(eye[2], eye[4])

Compute the euclidean distance between the horizontal eye landmark (x, y)-coordinates

C = dist.euclidean(eye[0], eye[3])

Compute the eye aspect ratio

ear = (A + B) / (2.0 * C)

G. Testing the OpenCV drowsiness detector

1.Testing the script on your local system in the comfort of your home/office before you start to wire up your car for driver drowsiness detection.

2.Testing moved on to some back roads and parking lots were there was very little traffic to continue testing the drowsiness detector.

3.Driving with your eyes closed, even for a second, is dangerous, so taking extra special precautions to ensure that the only person who could be harmed during the experiment .

4.Testing the drowsiness detector is able to detect when a person is at risk of dozing off.

5.Testing the drowsiness detector is able to work in a variety of conditions, including direct sunlight when driving on the road.

6.Testing at low/artificial lighting while in the concrete parking garage.

V. Tools Used

A. Hardware Tools

(i)Camera-Logitech C920- this camera as it:

Is relatively affordable. Can shoot in full 1080p. Is plug-and-play compatible with nearly every device (including the Raspberry Pi).

(ii)Laptop-

(iii)Raspberry Pi-

B. software Tools

(i)Python3 Interpreter.

(ii)OpenCV and Dlib libraries.- OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision

SCHEDULE DESIGN

JAN 1,2 WEEK	**Rigging the car or table with a drowsiness detector**
JAN 3,4 WEEK	**Selecting drowsiness detector algorithm and compare**
FEB 1,2,3 WEEK	**Implementing algorithm in laptop**
FEB 4 ,MAR 1 WEEK	**Implementing algorithm in Raspberry Pi**
MAR 2,3 WEEK	**Testing and Optimization**

VI. Conclusions

The proposed system in this analysis provides accurate detection of driver fatigue. The analysis and design of driver drowsiness detection system is presented. The proposed system is used to avoid various road accidents caused by drowsy driving and it can also help drivers to stay awake when driving by giving a warning when the driver is sleepy. And also this system used for security purpose of a driver. During the monitoring, the system is able to decide if the eyes are opened or closed. When the eyes have been closed for too long, a warning signal is issued. Image processing achieves highly accurate and reliable detection of drowsiness.

References

[1] A. Asthana, S. Zafeoriou, S. Cheng, and M. Pantic. *Incremental face alignment in the wild. In Confer- ence on Computer Vision and Pattern Recognition, 2014.*

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

[3] J. Cech, V. Franc, and J. Matas. *A 3D approach to facial landmarks: Detection, refinement, and track- ing. In Proc. International Conference on Pattern Recognition, 2014..*

[4] M. Chau and M. Betke. *Real time eye tracking and blink detection with USB cameras. Technical Report 2005-12, Boston University Computer Science, May 2005..*

[5] H. Dinh, E. Jovanov, and R. Adhami *Eye blink detection using intensity vertical projection. In In- ternational Multi-Conference on Engineering and Technological Innovation, IMETI 2012..*

[6] D. W. Jacobs, D. J. Kriegman, and N. Kumar. *Localizing parts of faces using a consensus of exem- plars. In CVPR, pages 545âĂŞ552, 2011.*

[7] M. Goffredo, S. Conforto, and M. Schmid. *An adaptive blink detector to initial- ize and update a view-basedremote eye gaze track- ing system in a natural scenario. Pattern Recogn. Lett., 30(12):1144âĂŞ1150, Sept. 2009.*

[8] S.-K. Pavani, C. Butakoff, and A. F. Frangi. *Automatic assessment of eye blinking pat- terns through statistical shape models. In ICVS, 2009.*