International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-9 Issue-2, July 2020

4D – Distracted Driver Detection Device

Ananya Sridhar

Abstract*: This paper describes an inexpensive distracted driver detection device built using a Raspberry PI3, a video camera, and python code. Distracted and drowsy driving are two of the leading causes of automobile accidents in the United States, and this inexpensive standalone can help prevent those deaths. live video feed of the driver's face is read in and through the facial landmark detector within DLIB, the co-ordinates of the eyes and mouth are extracted in real time. The ratio of the distance between the eyelids in the vertical direction to the horizontal direction defined as Eye Open Ratio is used to evaluate if the eye is open or closed and a similar ratio on the mouth(Mouth Open Ratio) is used to identify if it is open or closed. By looking at the eye and mouth open ratio for several consecutive frames, it is determined with >95% accuracy whether the driver is drowsy, distracted, or yawning. If any of these behaviors are noted, the device will prompt an audible warning to encourage the driver to focus on the road. The prototype was tested under a variety of lighting conditions from dark to bright light and on different subjects with and without glasses. This test data was used to determine the threshold for when the eye or mouth is determines open or closed. Additionally, the prototype's settings are customizable for a primary driver to further improve the accuracy. The device connects to a smart phone and sends information with the time stamp of the distracted driver incident. This device can be used to prevent distracted or drowsy driving-related deaths, and is an inexpensive attachment that can easily be fitted into a preexisting vehicle.

Keywords*: Raspberry Pi, Python, Night vision camera. Distracted driving

I. INTRODUCTION

Distracted driving accounts for approximately 25% of all motor vehicle crash fatalities. Driver distraction is reported to be responsible for more than 58% of crashes in young drivers. Additionally, drowsiness as a result of driver fatigue is responsible for a large number of accidents.

This paper uses readily available electronic resources to build a prototype that uses facial recognition to determine if a driver is distracted, drowsy or sleepy. A Raspberry Pi3 minicomputer, a night vision camera module, buzzer, LED lights, and other components were used to build the prototype. Python code was written and used to communicate between the different elements of Raspberry Pi. The finished prototype captures images of the driver's face and uses facial recognition to determine who the driver is. It then feeds driver specific information into the program. Video of the driver is captured and is used to determine if the eyes are open or closed. The prototype was tested extensively

under different light conditions and on a variety of subjects and found to work robustly. The device sounds an alarm, an audible alert, and a red light goes off when it detects that the driver is starting to doze off or is distracted. The device connects to a smart phone and sends information with the time stamp of the distracted driver incident.

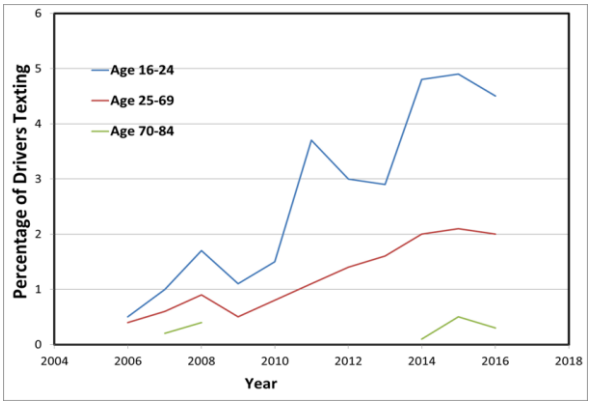

Figure 1. National Highway Traffic Safety Administration statistics on driving while texting

II. HARDWARE DESCRIPTION

Overview of Raspberry Pi

The Raspberry Pi 3 was used as the processor for this prototype. The Raspberry Pi3B was chosen for its compact form factor, its ability to process a large amount of data, and for its built in Wi-Fi and Bluetooth capabilities. Its features include 1.2 GHZ Quad Core ARM V8 Processor, inbuilt Wi-Fi and Bluetooth, 1 HDMI Port , 4 USB Ports, 1GB of RAM, Micro SD card slot, 1 micro USB slot, Video core 4 model GPU, and a 3.5mm audio jack output

Other key Hardware components

Other key components used in the prototype were a Night vision Camera connected via USB port to the Raspberry Pi, a basic buzzer, green and red LED lights, wires, a speaker connected to the 3.5mm audio jack, a lithium ion portable battery and an Android Smart phone.

III. HARDWARE IMPLMENTATION

OpenCV (Open Source Computer Vision Library) and ***Dlib*** were downloaded onto the Raspberry Pi. ***OpenCV*** is an open source computer vision and machine learning software library. ***OpenCV*** provides a common open source platform for computer vision applications.

Manuscript received on May 25, 2020.
Revised Manuscript received on June 29, 2020.
Manuscript published on July 30, 2020.
* Correspondence Author
Miss Ananya Sridhar, IB world school, PISD, Plano Texas, USA. E-mail: nasasridhar@gmail.com

Dlib is a general purpose cross-platform software library written in the programming language C++. Within ***Dlib*** there is a facial landmarks data package that scans in an image and is able to identify a face

and describe it as a map of 68 co-ordinates. These co-ordinates will be used to develop the software used in this prototype.

Facial Landmarks

The facial landmark detector implemented inside Dlib produces 68 *(x, y)*-coordinates that map to *specific facial structures*. These 68-point mappings were obtained by training a shape predictor, and it is part of the open source code available.

The facial land-marking aspect of the program is used to label and identify key facial attributes of an image, such as the eyes. There are two steps to facial land marking, detecting the face in the image, and then detecting key facial structures within the face pictured. This is how the face appears after it has been mapped. These annotations are part of the 68 point iBUG 300-W dataset which the ***Dlib*** facial landmark predictor was trained on.

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[2].

This method starts by using:

1. 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.

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

Given this training data, an ensemble of regression trees is 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.

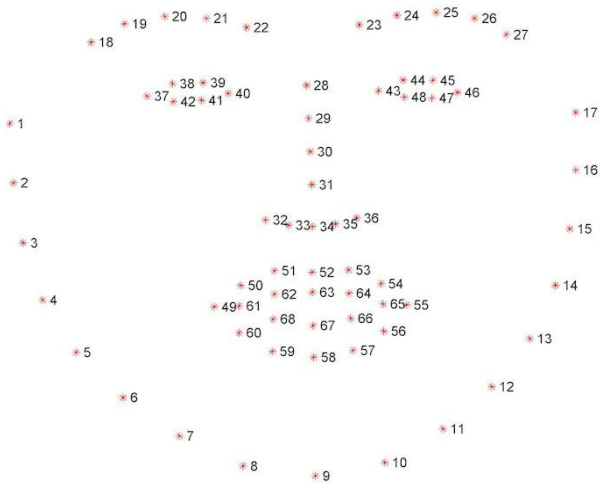

Figure 2: Facial map breaking down features into 68 points

The first step is determining the important regions of the face, such as the eyes, nose, mouth, and jawline.

Examining the image, we can see that facial regions can be accessed via simple Python indexing: For example

1. The mouth can be accessed through points [49, 68].
2. The right eye using [37, 42].
3. The left eye with [43, 48].

The data set was downloaded from the ***Dlib*** site. This set uses the concept described in a paper written by Tereza Soukupová and Jan Čech, a 68 point facial mapping system.

The ratio of the height of the eye to its width is used to determine if the eye is open or closed.

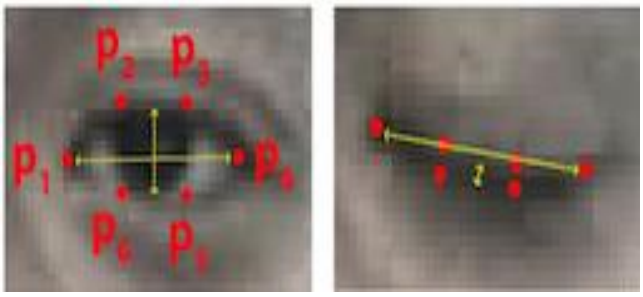

Figure 3- Determining if the eye is open or closed using Facial mapping and Eye open Ratio [1]

$$EOR = \frac{||P2-P6||+||P3-P5||}{2||P1-P4||}$$

Similarly, the MOR (mouth open Ratio) was calculated to determine if the driver is yawning.

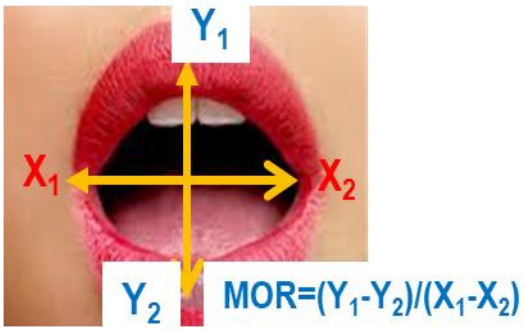

Figure 4- Determination of Mouth Open Ratio (MOR)

Since the goal was to process a live video, it was determined that ***OpenCV's*** Haar cascade face detector handled video feed much better than ***Dlib*** HOG + Linear SVM face detector. Using Haar Cascade results is a slight loss in accuracy compared to HOG, but that is made up for by the increased sampling on a video feed enabled by faster processing.

Below is a description of the setup procedure

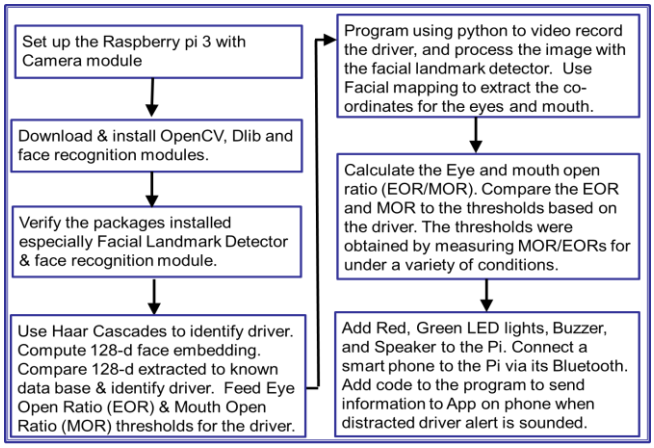

Figure 5- Steps to setup the hardware for the prototype

IV. OPTIMIZING THE PROTOTYPE

The ratio of the vertical distance between the eyelids and the horizontal distance (Called EOR - Eye Open Ratio) was set at 0.2 in order to get the initial set of data collection. The following optimizations were made The time it took for the facial recognition program to map out the face into co-ordinates was found to be a function of the size of the image. If it was too large an image it took a very long time to map and if it was made extremely small there were concerns about the accuracy of the mapping. Several test images were generated with different size and quality levels and fed into the facial mapping tool. It was found that using a 640X480 without reducing the quality was the best compromise. Extensive tests were done using 10 different subjects, and more than 1000 unique data points were collected in a controlled environment with different: ambient lighting conditions (Very Bright, Normal, Low Light and Night), and with and without glasses
It was found that using the minimum of the left eye and right eye EOR resulted in the elimination of the false flags.

A similar procedure was used to collect data for the mouth open ratio.

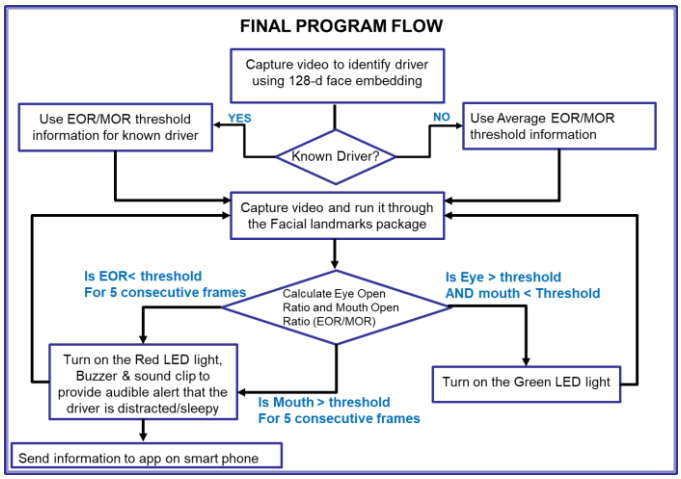

Figure 6- Program flow to detect and warn driver

Driver Identification:

Initial Face detection methods used a technique invented by Paul Viola and Michael Jones in early 2000s. In this program, a method invented in 2005 called Histogram of Oriented Gradients — or just ***HOG*** for short is used.

The goal is to figure out how dark a current pixel is compared to the pixels directly surrounding it. Then, an arrow is drawn showing in which direction the image is getting darker. By repeating that process for **every single pixel** in the image, pixels can be replaced by arrows. These arrows are called *gradients* and they show the flow from light to dark across the entire image.
The image is divided into small squares of 16x16 pixels each. In each square, the number of gradients pointed in each major direction is counted (how many point up, point up-right, point right, etc). That square is then replaced in the image with the arrow directions that were the strongest.
The end result turns the original image into a very simple representation that captures the basic structure of a face.

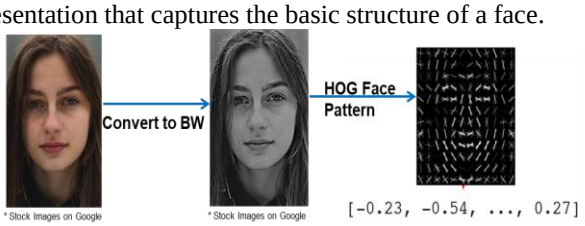

Figure 7- Facial recognition using the face-recognition module method generates a 128-d real-valued number feature vector per face

The recognition works by comparing the unknown face previously found with all the pictures of people that have already been tagged. When a previously tagged face looks very similar to the unknown face, it is identified as that person.

In summary the steps are

1. Create a directory with known images if the drivers. Capture images of each person's face at different angles. Run the program encode_faces.py [4], created by Adrian Rosenbrock of Raspberry PI foundation). This file will find faces in the dataset, encode them into 128-d vectors and save them in a .pickle file.

2. Capture the image of the driver and encode a picture using the HOG algorithm to create a simplified version of the image. Using this simplified image, find the part of the image that most looks like a generic HOG encoding of a face.

3. Figure out the pose of the face by finding the main landmarks in the face. Use those landmarks, to warp the image so that the eyes and mouth are centered.

4. Pass the centered face image and measure features of the face. Save those 128 measurements.

5. Looking at all the faces in the database, to see which person has the closest measurements to the face's measurements. That is the match!

V. DATA/RESULTS

The built prototype works as designed. The prototype was extensively tested and found to work robustly across a variety of light conditions from very bright light to darkness and for drivers with and without glasses. The EOR (Eye open ratio) values were collected and correlated across multiple experiments and the threshold value for optimal performance was determined to be 0.245 (if the value is below this threshold, the driver is marked as sleepy and alerted).

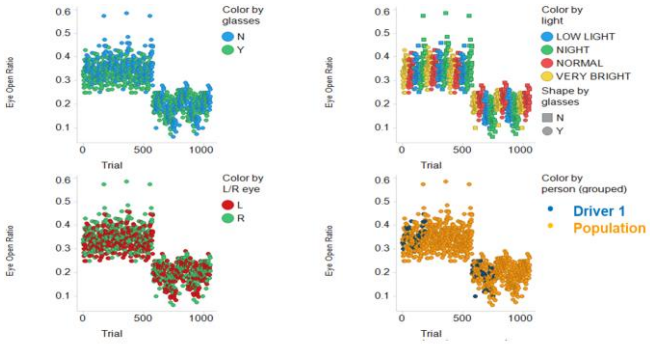

Figure 7- 1052 data points plotted for the eye open ratio clearly shows a transition at 0.22. This can be used to distinguish between an open & closed/distracted eye. If it were to be customized; Driver 1 has a EOR threshold of 0.255

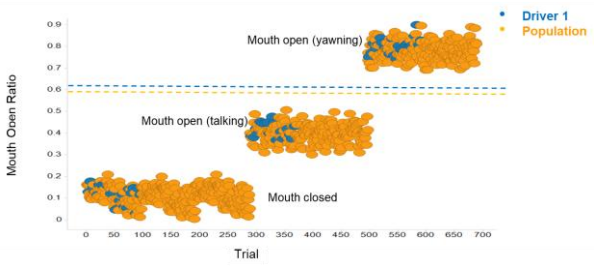

Figure 8 - 678 data points plotted for the mouth open ratio clearly shows a transition at 0.59. This can be used to distinguish between a closed mouth, someone who is talking and someone who is yawning. Driver 1's MOR threshold is 0.62.

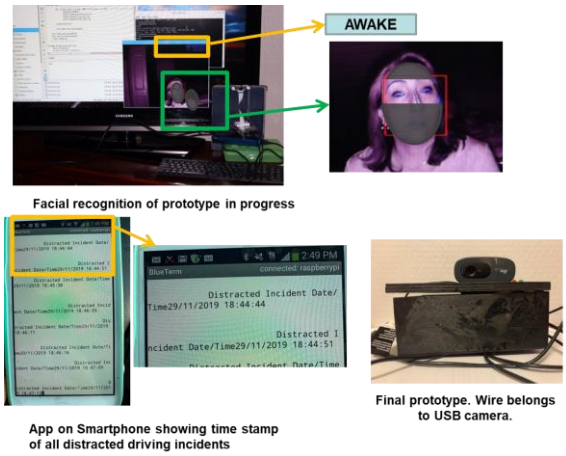

Figure 9- Prototype connected to Smartphone via app recording instances of distracted driving.

Figure 10- Prototype installed in Vehicle.

VI. CONCLUSION

The proposed Distracted Driver Detection Device was built and extensively tested. It functions as designed. When the device is turned on it scans the driver's face and determines if it is a known driver. If that is the case it uses the customized MOR and EOR for that driver and audibly welcomes the driver. If it does not recognize the face it uses the default MOR and EOR of 0.59 and 0.22. Every time a distracted driving incident is detected it logs it and sends the information to a smart phone. This device can be used to prevent distracted driving and any accidents that may stem from that cause. It can also be useful to parents of new drivers who will be alerted if their child is driving unsafely.

REFERENCES

1. Soukupová and Čech, "Real-Time Eye Blink Detection suing Facial Landmarks"; 21st Computer Vision Winter Workshop, 2016
2. V. Kazemi and J. Sullivan, "One Millisecond Face Alignment with an Ensemble of Regression Trees", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1867-1874, 2014.
3. P.Viola and M.Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.
4. Adrian Rosenbrock, tutorials on face recognition, https://www.pyimagesearch.com/

AUTHOR'S PROFILE

Ananya Sridhar is a high school student in the IB program. She has a strong interest in medicine and technology and hopes to one day work in a field that combines both of her passions. In school, she is a member of the National Honor Society, an officer in the TedED club, and a senior staff editor at her school's chapter of Research Crunch. Outside of school, she teaches taekwondo, is an avid participant of Destination Imagination, and is heavily involved at her local zoo. She is a member of the American Junior Academy of Science and has presented her research to scientists and peers across the nation.