CHAPTER I

INTRODUCTION

DRIVER DROWSINESS DETECTION SYSTEM

Drowsiness of the drivers is one of the key issues for majority of road accidents. Drowsiness threatens the road safety and causes severe injuries sometimes, resulting in fatality of the victim and economical losses. Drowsiness implies feeling lethargic, lack of concentration, tired eyes of the drivers while driving vehicles. Most of the accidents happen in India due to the lack of concentration of the driver. Performance of the driver gradually deteriorates owing to drowsiness. To avoid this anomaly, we developed a system that is able to detect the drowsiness nature of the driver and alert him immediately. This system captures images as a video stream through a camera, detects the face and localizes the eyes. The eyes are then analysed for drowsiness detection using Perclos algorithm. Based on the result, the driver is alerted for drowsiness through an alarm system.

1.1 SYSTEM OVERVIEW

1.1 DIFFERENT APPROACHES TO DETECTING DROWSINESS:

There are different approaches to identify drowsiness state of the driver. They can be categorised into the following three main categories:

1.1.1 BEHAVIOURAL PARAMETERS-BASED TECHNIQUES:

Measuring the driver's fatigue without using non-invasive instruments comes in this category. Analysing the behaviour of the driver based on his/her eye closure ratio, blink frequency, yawning, position of the facial expressions. The parameter used in this system is the eye-closure ratio of the driver.

1.1.2 VEHICULAR PARAMETERS-BASED TECHNIQUES:

Measuring the fatigue nature of the driver through vehicle driving patterns comes under this category. These parameters include lane changing patterns, steering wheel angle, steering wheel grip force, vehicle speed variability and many more.

1.1.3 PHYSIOLOGICAL PARAMETERS-BASED TECHNIQUES:

Measuring the drowsiness of the driver based on the physical conditions of the driver fall under this category. Such parameters may be respiration rate, heart-beat rate, body temperature and many more.

Among other various approaches, these physiological parameters provide the most accurate results since they are based on the biological nature of the driver.

All the above approaches have their own advantages and disadvantages. Based on the desired result accuracy, any approach can be used. Physiological approach includes wearing of the equipment on the driver's body. This equipment includes electrodes to detect the pulse rate of the driver which might make the driver uncomfortable while driving. This also can't be assured that the driver always wears such equipment while driving which may result in inefficient results. Hence there is a hindrance using the physiological approach. Vehicular- based approach is always based on the efficiency of the driver and his condition. There are also constraints like the road condition and the type of vehicle which may change regularly. Hence it is best to follow the behavioural based approach through visual assessment of the driver from a camera. There shall be no equipment attached to the driver. Hence this technique is always the best approach and can be implemented in any vehicle without any modifications.

1.1.4 DIGITAL IMAGE PROCESSING

The term digital image processing generally refers to processing of a two-dimensional picture by a digital computer. In a broader context, it implies digital processing of any two- dimensional data. A digital image is an array of real numbers represented by a finite number of bits. The principle advantage of Digital Image Processing methods is its versatility, repeatability and the preservation of original data precision.

Pixel:

Pixel is the smallest element of an image. Each pixel corresponds to any one value. In an 8-bit gray scale image, the value of the pixel between 0 and 255. The values of a pixel at any point correspond to the intensity of the light photons striking at that point. Each pixel stores a value proportional to the light intensity at that particular location.

Digital image:

A digital image is nothing more than data numbers indicating variations of red, green, and blue at a particular location on a grid of pixels.

Gray level:

The value of the pixel at any point denotes the intensity of image at that location, and that is also known as gray level. Generally to convert an image to gray scale, the equation that was used previously is : Grayscale = (Red + Green + Blue / 3).But as red has more wavelength we use the equation:

$$Grayscale = ((0.3 * R) + (0.59 * G) + (0.11 * B)).$$

1.2 MOTIVATION FOR THE WORK

Now-a-days, there is huge increase in private transportation day by day in this modernize world. It will be tedious and bored for driving when it is for long time distance. One of the main causes behind the driver’s lack of alertness is due to long time travelling without sleep and rest. Tired driver can get drowsy while driving. Every fraction of seconds drowsiness can turn into dangerous and life-threatening accidents may lead to death also. To prevent this type of incidents, it is required to monitor driver’s alertness continuously and when it detects drowsiness, the driver should be alerted. Through this we can reduce significant number of accidents and can save lives of people.

1.3 PROBLEM STATEMENT

Many of the road accidents will occur due to drowsiness of the driver. Drowsiness can be detected by monitoring the driver through continuous video stream with a mobile or camera. The general objective is to create a model that will indicate whether a person is feeling drowsy or not. The model takes image for every second and check for eye blinking and calculate the time for eye closed by perclos algorithm. If the blinking is high and eye is closed for certain amount of time then it will indicate driver through a sound.

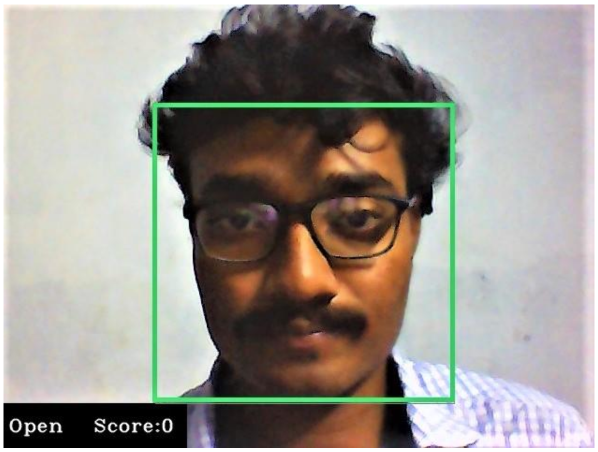
CHAPTER II

LITERATURE SURVEY

In computer science, image processing is the use of computer algorithms to perform image processing on images. As a subcategory or field of digital signal processing, image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the buildup of noise and signal distortion during processing. Since images are defined over two dimensions digital image processing may be modelled in the form of multidimensional system.

2.1 DROWSINESS DETECTION THROUGH REGION OF INTEREST:

Region of interest (ROI) can detect a driver’s face. As can be seen in the Green rectangle is the region of interest. The way to create an ROI area is to first obtain the green rectangle area from the Haar Cascade Classifier in the first frame, which includes height, width. Then, the rectangle is scaled up to create region of interest. There are several steps to calculate the ROI area and we have to calculate ROI for each and every region of interest



Region of Interest

DISADVANTAGES OF REGION OF INTEREST

1. It is uses extra frames or squares to detect face detection.
2. It can’t find in low light.
3. Why to use again region of interest while Haar cascade classifier can do the same process?
4. It can’t detect while using glasses in driving.

2.2 DETECTION OF DROWSINESS THROUGH LBPH:

In this algorithm the faces are detected by using local binary patterns histograms (LBPH). The first computational step in lbph is to create an intermediate images that describes the original image in a binary format. The image is converted into matrix form and we need to take a central value of the matrix to be used as and threshold value. This value is used to define neighbouring values which can be set to to either 0 or 1. The values which are 1 in the matrix form are to be considered and the remaining values are discarded. The values represent each pixel. Through this the region of face can be detected.

DISADVANTAGES OF LBHP

1. It produces less urate results
2. The computational time is high.
3. This will work only if the data samples are less.

2.3 BEHAVIOURAL BASED TECHNIQUES

The different techniques used in behavioural based parameters are:

2.3.1. EYE TRACKING AND DYNAMIC TEMPLATE MATCHING

To avoid road accidents, real time driver fatigue detection system based on vision is proposed. Firstly, system detects the face of driver from the input images using HSI color model. Secondly, Sobel edge operator is used to locate the eyes positions and gets the images of eye as the dynamic template for the tracking of eye. Then the obtained images are converted to HSI colour model to decide that whether the eyes are close or open to judge the drowsiness of driver. The experiments use four test videos for the tracking of eyes and face detection. The proposed system is compared with the labelled data which is annotated by the experts. The average correct rate of proposed system reaches up to 99.01 % and the precision to 88.90 %.

2.3.2. MOUTH AND YAWNING ANALYSIS

Fatigue is the major reason for road accidents. To avoid the issue, Sarada Devi and Bajaj proposed the driver fatigue detection system based on mouth and yawning analysis. Firstly, the system locates and tracks the mouth of a driver using cascade of classifier training and mouth detection from the input images. Then, the images of mouth and yawning are trained using SVM. In the end, SVM is used to classify the regions of mouth to detects the yawning and alerts the fatigue. For experiment, authors collect some videos and select 20 yawning images and more than 100 normal videos as dataset. The results show that the proposed system gives better results as compared to the system using geometric features. The proposed system detects yawning, alerts the fatigue earlier and facilitates to make the driver safe.

2.3.3. FACIAL EXPRESSIONS METHOD

Laboratory condition using Finite Element Analysis is used by the researchers which is a complex system that contains the database of facial expression as a template and detect the drowsiness on the basis of results from database. Similarly, Assari and Rahmati present the hardware-based Driver Drowsiness Detection system based on facial expressions. The hardware system uses infrared light as it has giving many benefits like ease of use, independent of lightning conditions of environment. The system firstly uses the technique of background subtraction to determines the face region from the input images. Then using horizontal projection and template matching, facial expressions are obtained. After that in the tracking phase, elements found earlier are followed up using template matching and then investigates the incidence of sleepiness using the determination of facial states from the changes of the facial components. Changing in the three main elements such as eye brow rising, yawning and eye closure for the certain period are taken as the initial indications for drowsiness and the system generates the alert. The experiment is performed in the real driving scenario. For testing, images are acquired by the webcam under different conditions of lighting and from different people. The results investigate that the system produces appropriate response in the presence of beard or glasses and mustache on the face of driver.

2.3.4. YAWNING EXTRACTION METHOD

Fatigue or drowsiness is the major reason for road accidents. To prevent the issue, Alioua proposed the efficient system for monitoring the driver fatigue using Yawning extraction. Firstly, face region is obtained from the images using Support Vector Machine (SVM) technique to reduce the edge required cost. The proposed method is used to localize the mouth, detection technique is used to detects facial edges, then compute vertical projection on the lower half face to detect the right and left region boundaries and then compute the horizontal projection on the resulting region to detect the upper and lower limit of mouth and mouth localized region is obtained. Finally, to detect the yawning, Circular Hough Transform (CHT) is executed on the images of mouth region to identify the wide-open mouth. If the system finds notable number of continuous frames where the mouth is widely open, system generates the alert. The results are compared with the other edge detectors like Sobel, Prewitt, Roberts, Canny. The experiment uses 6 videos representing real driving conditions and results are presented in the form of confusion matrix. The proposed method achieves 98% accuracy and outperforms all other edge detection techniques.

2.3.5. EYE CLOSURE AND HEAD POSTURES METHOD

Teyeb proposed the Drowsy Driver Detection using Eye Closure and Head postures. Firstly, video is captured using webcam and for each frame of video, following operations are performed. To detect the ROI (face and eyes), viola-jones method is used. The face is partitioned in to three areas and the top one presenting the aye area is browsed by the Haar classifier. Then to detect the eye state, Wavelet Network based on neural network is used to train the images then the coefficients learning images is compared with the coefficients of the testing images and tells which class it belongs. When the closed eye is identified in the frames then the eye closure duration is calculated, if the value exceeds the pre-defined time then the drowsiness state is detected. Then the developed system estimates the head movements which are: left, right, forward, backward inclination and left or right rotation. The captured video is segmented into frames and extract the images of head and determines the coordinates of image.

Then the images are compared to determine the inclined state of head and same case with other head postures. Finally, the system combines the eye closure duration and head posture estimation to measure the drowsiness. To evaluate the system, experiment is performed on 10 volunteers in various situations. And results show that the systems achieve the accuracy of 80%.

2.3.6. REAL TIME ANALYSIS USING EYE AND YAWNING

Kumar proposed the real time analysis of Driver Fatigue Detection using behavioral measures and gestures like eye blink, head movement and yawning to identify the drivers' state.

The basic purpose of the proposed method is to detect the close eye and open mouth simultaneously and generates an alarm on positive detection. The system firstly captures the real time video using the camera mounted in front of the driver. Then the frames of captured video are used to detect the face and eyes by applying the viola-jones method, with the training set of face and eyes provided in OpenCV. Small rectangle is drawn around the center of eye and matrix is created that shows that the Region of Interest (ROI) that is eyes used in the next step. Since the both eyes blink at the same time that's why only the right eye is examined to detect the close eye state. If the eye is closed for certain amount of time it will be considered as closed eye. To determine the eye state, firstly the eye ball color is acquired by sampling the RGB components on the center of eye pixel. Then the absolute thresholding is done on the eye ROI based on eye ball color and intensity map is obtained on Y-axis that show the distribution of pixels on y-axis which gives the height of eye ball and compared that value with threshold value which is 4 to distinguish the open and close eye. After that, if the eye blink is detected in each frame it will be considered as 1 and stored in the buffer and after the 100 frames, eye blinking rate is calculated. Then to detect the yawning motion of the mouth, contour finding algorithm is used to measure the size of mouth. If the height is greater than the certain threshold. It means person is taking yawning. To evaluate the performance of the proposed system, system has been measured under different conditions like persons with glasses, without glasses, with moustache and without moustache for 20 days in different timings. The system performs best when the drivers are without glasses.

2.3.7. EYE BLINK DETECTION METHOD

Ahmad and Borolie proposed the Driver Drowsiness System based on non-intrusive machine-based concepts. The system consists of a web camera which is placed in front of the driver. Online videos as well as saved videos for simulation purposed are considered. Firstly, camera records the facial expressions and head movements of the driver. Then the video is converted into frames and each frame is processed one by one. Face is detected from frames using Viola-jones algorithm. Then the required features like eyes, mouth and head from face are extracted using cascade classifier. Region of interest on face is indicated by rectangles. Here the main attribute of detecting drowsiness is eyes blinking, varies from 12 to19 per minute normally and indicates the drowsiness if the frequency is less than the normal range. Instead of calculating eye blinking, average drowsiness is calculated. The detected eye is equivalent to zero (closed eye) and non-zero values are indicated as partially or fully open eyes. The equation is used to calculate the average.

$$\%d = \frac{\text{No. of closed evesfound}}{\text{no. of frames}}$$

If the value is more than the set threshold value, then system generates the alarm to alert the driver. Moreover, yawning is also considered to generate the alert. Online and offline are videos are used for experiment which are performed on two different systems. The results show that the system achieves the efficiency up to 90%.

2.3.8. EYE CLOSENESS DETECTION METHOD

Khunpisuth creates an experiment the calculates the drowsiness level of driver using Raspberry Pi camera and Raspberry Pi 3 model B. Firstly Pi camera captures video and to detect face regions in the images, Haar cascade classifier from Viola-Jones method is used. Several images are trained in different lights. The percentage of 83.09 % is achieved based on the case study with 10 volunteers. Blue rectangle shows the Region of Interest (ROI) that is face. Again, Haar cascade classifier is applied on the last obtained frame which reduces the size of ROI. After the face detection, drowsiness level is calculated using eye blink rate. Eye region is detected using template matching on the face and authors uses three templates to check the eye blink and aye area. CV_TM_CCOEFF_NORMED from OpenCV is considered as it gives improved results than other methods of template matching. The integration of eyes and face detection permits the checking of an eye blinking and closeness rate. If the eyes are closed, then the value of closed eye is higher than the open the eyes and opposite case if eyes are open. Authors assumed that Haar cascade classifier will work if the face is front facing position. That why authors proposed the method to rotate the tilted face back in to the front-facing position. Firstly, determines whether the head is tilt or not then calculates the degrees of rotation (angle). After the accurate detection of face and eyes, drowsiness level of driver is determined. If the drivers blink eyes too frequently, he system indicates he drowsiness. When the level reaches to one hundred, a loud sound will be generated to alert the driver.

The proposed method is compared with Haar cascade and results shows that the proposed method achieves the accuracy of 99.59 %. It works in all lighting conditions and able to detect the face wearing glasses.

2.4 VEHICULAR PARAMETER-BASED TECHNIQUES

2.4.1. REAL TIME LANE DETECTION SYSTEM

Road accidents have become common in the present era, causing the severe damage to the property and also to the lives of people travelling. There are many reasons of road accidents like: rash driving, inexperience, ignoring signboards, jumping signal etc. To address the issues, Katyal *et al.* proposed the Drivers' Drowsiness Detection system. The system works in two phases: firstly, detects lane based on Hough transform. Secondly, detects the drivers' eyes to detect the drowsiness. For eye detection, firstly use viola jones method to detect face, then do the image segmentation, after that otsu thresholding is done and canny edge detection is applied. The obtained results is integrated with the circle detection hough transform method to detect eyes to detect the fatigue level. It will also work in low lightning conditions. Result shows that the proposed system is useful for the drivers travelling on lengthy routes, driving late night, drivers who drink and drive.

2.4.2. TIME SERIES ANALYSIS OF STEERING WHEEL ANGULAR VELOCITY

To avoid the road accidents, Zhenhai proposed the Driver Drowsiness Detection method using time series analysis of steering wheel angular velocity. The method firstly analyses the behavior of steering below the fatigue, then temporal detection window is used as the detection feature to determine the angular velocity of steering wheel during time-series. In the temporal window, if the detection feature satisfies the variability constraints and extent constraints, then the state of drowsiness is detected accordingly. The experiment based on real testers is performed, and results shows that the proposed method outperforms the previous methods and useful in the real world.

2.4.3. STEERING WHEEL ANGLE FOR REAL DRIVING CONDITIONS FOR DDT

To avoid road accidents, Li proposed the online detection of Drowsiness Detection System to monitor the fatigue level of drivers under real conditions using Steering Wheel Angles (SWA). The data of SWA is collected from the sensors attached on the steering lever. The system firstly extracts the features of Approximate Entropy (ApEn) from fixed sliding windows on time series of real time steering wheel angles, then the system linearizes the features of ApEn using the deviation of adaptive linear piecewise fitting method. After that the system calculates the warping distance between the series of linear features of sample data. Finally, system determine the drowsiness state of drivers using warping distance according to the designed decision classifier. The empirical analysis uses the data collected in 14.68 hrs. driving under real road conditions and evaluated on two fatigue levels: drowsy and awake. Results show that the proposed sys- tem is capable for working online with an accuracy of 78.01 % and useful for the prevention of road accidents caused by drivers' fatigue.

2.4.4. AUTOMATIC DETECTION OF DRIVER FATIGUE

To address the issue of drivers' fatigue, an online detection of drivers' fatigue using the Steering Wheel Angles (SWA) and Yaw Angles (YA) information in the real driving conditions is proposed. The system firstly investigates the operation features of SWA and YA in the different states of fatigue, after that calculates the ApEn features on time series of shot sliding window, then using the dynamic time series of non-linear feature construction theory and taking features of fatigue as input, designs a 2-6-6-3 multi-level Back Propagation (BP) neural network classifier to deter- mine the fatigue detection. For empirical analysis, 15 hours long experiment is performed in real road conditions. The experts evaluated the retrieved data and categorized in three levels of fatigue: drowsy, very drowsy, and awake. And the experiment achieves the average accuracy of 88.02% in fatigue detection and valuable for the engineering applications.

2.5 DROWSINESS DETECTION THROUGH PHYSIOLOGICAL APPROACH

Physiological measures have much of the time been utilized for drowsiness discovery as they can give an immediate and objective measure. Conceivable measures are EEG, eyelid closure, movements of eye, heart rate, size of pupil, skin conductance and creation of the cortical. Among these procedures, the systems that are best, in light of precision are the ones in view of physiological experience of human. There are two ways for implementing this procedure. Measurement of changes in physiological signs for example, waves of human brain, blinking of eyes and heart rate; and physical changes measurement for example, drooping stance, leaning of the head of driver and the open/close conditions of the eyes.

2.5.1. EEG-BASED DRIVER FATIGUE DETECTION

EEG is a technique for measuring the electrical action created by the nerve cells of the human brain, basically the cortical movement. The EEG- action is available all the time and recording show both arbitrary and periodic behaviour. The fundamental inception of the EEG is the neuronal action in the cerebral cortex; however some action like wise starts from the thalamus and from subcritical parts of the human brain. The EEG speaks to the summation of excitatory and inhibitory postsynaptic possibilities in the nerve cells. The musical movement is because of the synchronous actuation of the nerve cells. The drivers' fatigue detection system using Electroencephalogram (EEG) signals is proposed to avoid the road accidents usually caused due to drivers' fatigue. The proposed method firstly finds the index related to different drowsiness levels. The system takes EEG signal as input which is calculated by a cheap single electrode neuro signal acquisition device. To evaluate the proposed method, data set for simulated car driver under the different levels of drowsiness is collected locally. And result shows that the proposed system can detect all subjects of tiredness.

Disadvantages:

1. High cost
2. Sensors Required
3. Can't see behind objects
4. Takes longer time

2.5.2. WAVELET ANALYSIS OF HEART RATE VARIABILITY & SVM CLASSIFIER

Li and Chung proposed the driver drowsiness detection that uses wavelet analysis of Heart Rate Variability (HRV) and Support Vector Machine (SVM) classifier. The basic purpose is to categorize the alert and drowsy drivers using the wavelet transform of HRV signals over short durations. The system firstly takes Photo PlethysmoGraphy (PPG) signal as input and divide it into 1-minute intervals and then verify two driving events using average percentage of eyelid closure over pupil over time (PERCLOS) measurement over the interval. Secondly, the system performs the feature extraction of HRV time series based on Fast Fourier Trans- form (FFT) and wavelet. A Receiver Operation Curve (ROC) and SVM classifier is used for feature extraction and classification respectively. The analysis of ROC shows that the wavelet-based method gives improved results than the FFT-based method. Finally, the real time requirements for drowsiness detection, FFT and wavelet features are used to train the SVM classifier extracted from the HRV signals. The performance of classification using the wavelet-based features achieve the accuracy of 95%, sensitivity to 95% and specificity to 95%. The FFT-based results achieve the accuracy of 68.85. The results show that wavelet-based methods perform better than the FFT-based methods.

2.5.3. PULSE SENSOR METHOD

Mostly, previous studies focus on the physical conditions of drivers to detect drowsiness. That's why Rahim detects the drowsy drivers using infrared heart-rate sensors or pulse sensors. The pulse sensor measures the heart pulse rate from drivers' finger or hand. The sensor is attached with the finger or hand, detects the amount of blood flowing through the finger. Then amount of the blood’s oxygen is shown in the finger, which causes the infrared light to reflect off and to the transmitter. The sensor picks up the fluctuation of oxygen that are connected to the Arduino as microcontroller. Then, the heart pulse rate is visualizing by the software processing of HRV frequency domain. Experimental results show that LF/HF (Low to high frequency) ratio decreases as drivers go from the state of being awake to the drowsy and many road accidents can be avoided if an alert is sent ontime.

2.5.4. WEARABLE DRIVER DROWSINESS DETECTION SYSTEM

Mobile based applications have been developed to detect the drowsiness of drivers. But mobile phones distract the drivers’ attention and may cause accident. To address the issue, Leng proposed the wearable-type drowsiness detection system. The system uses self-designed wrist band consists of PPG signal and galvanic skin response sensor. The data collected from the sensors are delivered to the mobile device which acts as the main evaluating unit. The collected data are examined with the motion sensors that are built-in in the mobiles. Then five features are extracted from the data: heart rate, respiratory rate, stress level, pulse rate variability, and adjustment counter. The features are moreover used as the computation parameters to the SVM classifier to determine the drowsiness state. The experimental results show that the accuracy of the proposed system reaches up to 98.02 %. Mobile phone generates graphical and vibrational alarm to alert the driver.

2.5.5. WIRELESS WEARABLESMETHOD

To avoid the disastrous road accidents, Warwick proposed the idea for drowsiness detection system using wearable Bio sensor called Bio- harness.

The system has two phases. In the first phase, the physiological data of driver is collected using bio-harness and then analyzes the data to find the key parameters like ECG, heart rate, posture and others related to the drowsiness. In the second phase, drowsiness detection algorithm will be designed and develop a mobile app to alert the drowsy drivers.

2.5.6. DRIVER FATIGUE DETECTION SYSTEM

The basic of the system is to detect the drowsiness when the vehicle is in the motion. The system has three components: external hardware (sensors and camera), data processing module and alert unit. Hardware unit communicates over the USB port with the rest of the system. Physio- logical and physical factors like pulse rate, yawning, closed eyes, blink duration and others are continuously monitored using somatic sensor. The processing module uses the combi- nation of the factors to detect drowsiness. In the end, alert unit alerts the driver at multiple stages according to the severity of the symptoms.

2.5.7. HYBRID APPROACH UTILIZING PHYSIOLOGICAL FEATURES

To improve the performance of detection, Awais proposed the hybrid method which integrates the features of ECG and EEG. The method firstly extracts the time and frequency domain features like time domain statistical descriptors, complexity measures and power spectral measures from EEG. Then using ECG, features like heart rate, HRV, low frequency, high frequency and LF/HF ratio. After that, subjective sleepiness is measured to study its relationship with drowsiness. To select only statistically significant features, t-tests is used that can differentiate between the drowsy and alert. The features extracted from ECG and EEG are integrated to study the improvements in the performance using SVM. The other main contribution is to study the channel reduction and its impact on the performance of detection. The method measures the differences between the drowsy and alert state from physiological data collected from the driving simulated-based study. Monotonous driving environment is used to induce the drowsiness in the participants. The proposed method demonstrated that combining ECG and EEG improves the performance of system in differentiating the drowsy and alert states, instead of using them alone. The analysis of channel reduction confirms that the accuracy level reaches to 80% by just combining the ECG and EEG. The performance of the system indicates that the proposed system is feasible for practical drowsiness detection system.

CHAPTER III

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The current drowsiness detection systems include the usage of the devices that detect the respiration rate, heart rate, blood pressure, etc. These devices can cause the driver to be uncomfortable for driving. Cannot be assured that the drivers wear these devices all the time while driving. May get lost or improper functioning which may lead to low accuracy in the result. The existing system does not produce good results in low light conditions. If the light conditions are dark or too low it is unable to detect the face and eyes of the driver which results in lower accuracy.

3.2 PROPOSED SYSTEM

The drowsiness detection based on eye state has been done accurately based on the varying features and factors, and also with the help of expert's knowledge. Predicting the facial landmarks and detecting the eye-state and displaying the driver status on the screen and in the App is the most necessity ingredient for drowsiness detection.

Generally, the driving person feels drowsy due to continues driving for long hours or Physical illness or might be drunken and this leads to major road accidents. Our aim is to detect the drowsiness, make them alert to prevent accidents and generate a notification in the app and an alarm sound.

3.2.1 Advantages of Proposed System

- The proposed system achieved more than 95% accurate result.
- Convenient approach to detect the drowsiness.

- Technically possible for practical implementation.
- Whoever, Installed the App will get a notification about the driver’s active status.

3.3 REQUIREMENT SPECIFICATION

3.3.1 Software Requirements

A software requirement specification is a description of a software system to be developed. It lays out functional and non-functional requirements and may include a set of usecases that describe user interaction that the software must provide.

1. Technology : Anaconda Navigator and Jupyter Notebook
2. Programming Language : Python3
3. Packages : Numpy, Keras, Tensorflow, OpenCV, Pygame
4. Operating Systems : Windows 10 -64 bit and above

3.3.2 Hardware Requirements

These pre-requisites are known as (computer) System requirements and are often used as a guidelines as opposed to an absolute rule . Most software defines two set of system requirements: Minimum and recommended.

- Processor: 64 bit, quad-core, 2.5 GHz minimum per core
- RAM: 4 GB or more.
- HDD: 20 GB of available space or more.
- Display: Dual XGA (1024 x 768) or higher resolution monitors.
- Camera: A detachable webcam.
- Keyboard: A standard keyboard.

3.4 LANGUAGE SPECIFICATION

3.4.1 Python Programming Language

Python is a high-level, general-purpose and a very popular programming language. Python programming language (latest Python 3) is being used in web development, Machine Learning applications, along with all cutting edge technology in Software Industry. Python Programming Language is very well suited for Beginners, also for experienced programmers with other programming languages like C++ and Java..

Python offers code that is readable and easy to understand. Instead of having punctuations and signs, it offers simplicity that enables the developers to write reliable programs. Using this language, the developers can put all their efforts into developing a solution for an ML project. This programming language allows them to focus more on the project requirements rather than the technical nuances.

Most of the programmers find Python more intuitive than other programming languages. According to the developer community, Python is more intuitive than other programming languages. Python is quite suitable for collaboration when multiple developers are involved. As it is a general-purpose language, it is capable of performing a set of complex machine tasks that enable the developers to build prototypes quickly.

Platform independence is one of the keys to the popularity of this programming language. Python is supported by several platforms including Linux, macOS, and Windows. The Python code can be used to create executable programs for different operating systems.

Apart from that, the developers can also use powerful graphics processing units to train their ML models and projects. Python is platform-independent and this makes it quite inexpensive and easier to implement.

Benefits

- Implementing the ML algorithms is tricky and requires a lot of time. It is quite important to have a well-structured and well-tested environment to allow the developers to come up with the best coding solutions. Many libraries can be used to reduce the coding time and save tons of development hours. Python has a rich set of tech-stack that includes extensive libraries like,

✓ Scikit-learn TensorFlow, and Keras for machine learning

✓ Pandas for general computing

✓ NumPy for computing data analysis

✓ SciPy for advanced computing

✓ Seaborn for advanced data visualization

With these libraries and solutions, developing and launching a product is faster in the market place.

- One of the key aspects of Python development is the object-oriented approach. Python recognizes the concept of object encapsulation and polymorphism. This object-oriented approach allows the programs to be efficient in the long run.

- IDLE (Integrated Development & Learning Environment)is a powerful IDE that is dedicated to software development. This environment offers a bundle of tools and utilities to work with. You get access to a powerful editor to manage all the code. Apart from that, you can also perform other functions such as auto-code completion and syntax highlighting.

Running Python on different platforms simultaneously is quite a seamless experience. Python offers you complete portability in switching from one platform to another without changing the code. This feature makes it one of the highly preferred programming languages in the world.

3.5 ALGORITHM SPECIFICATION

CONVOLUTIONAL NEURAL NETWORK (CNN, or ConvNet)- CNN is a particular type of feed-forward neural network in AI. It is widely used for image recognition . CNN represents the input data in the form of multidimensional arrays . It works well for a large number of labeled data.

CNN VGG16- VGG-16 is a convolutional neural network that is 16 layers deep. You can load a pretrained version of the network trained on more than a million images from the ImageNet database.

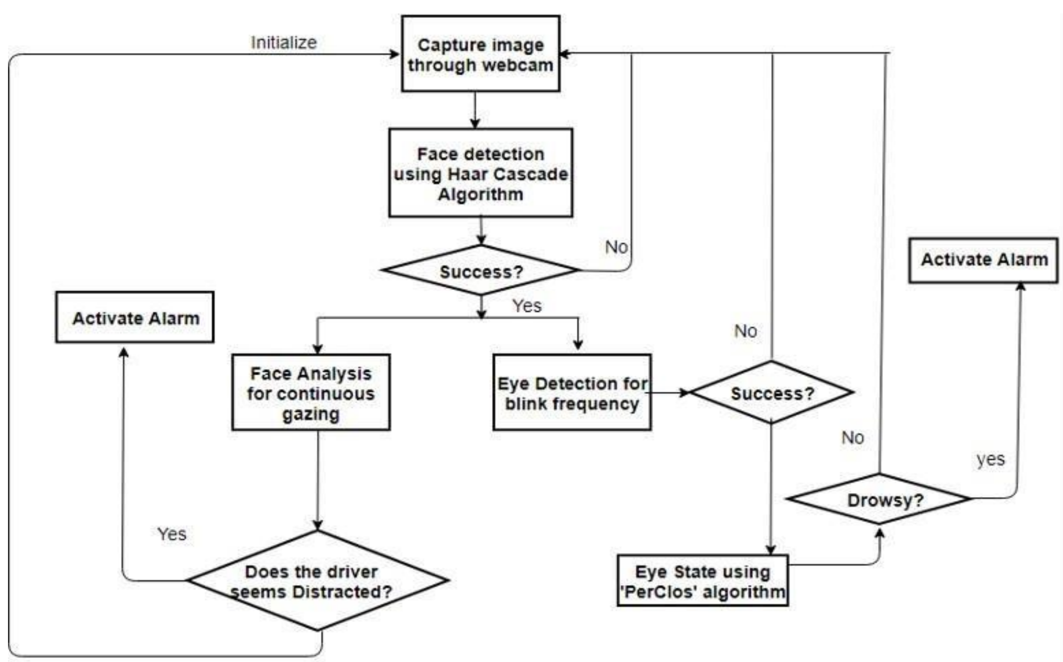
MOBILENET- MobileNet uses depthwise separable convolutions. It significantly reduces the number of parameters when compared to the network with regular convolutions with the same depth in the nets. This results in lightweight deep neural networks. MobileNet is a class of CNN that was open-sourced by Google, and therefore, this gives us an excellent starting point for training our classifiers that are insanely small and insanely fast.

CHAPTER IV

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

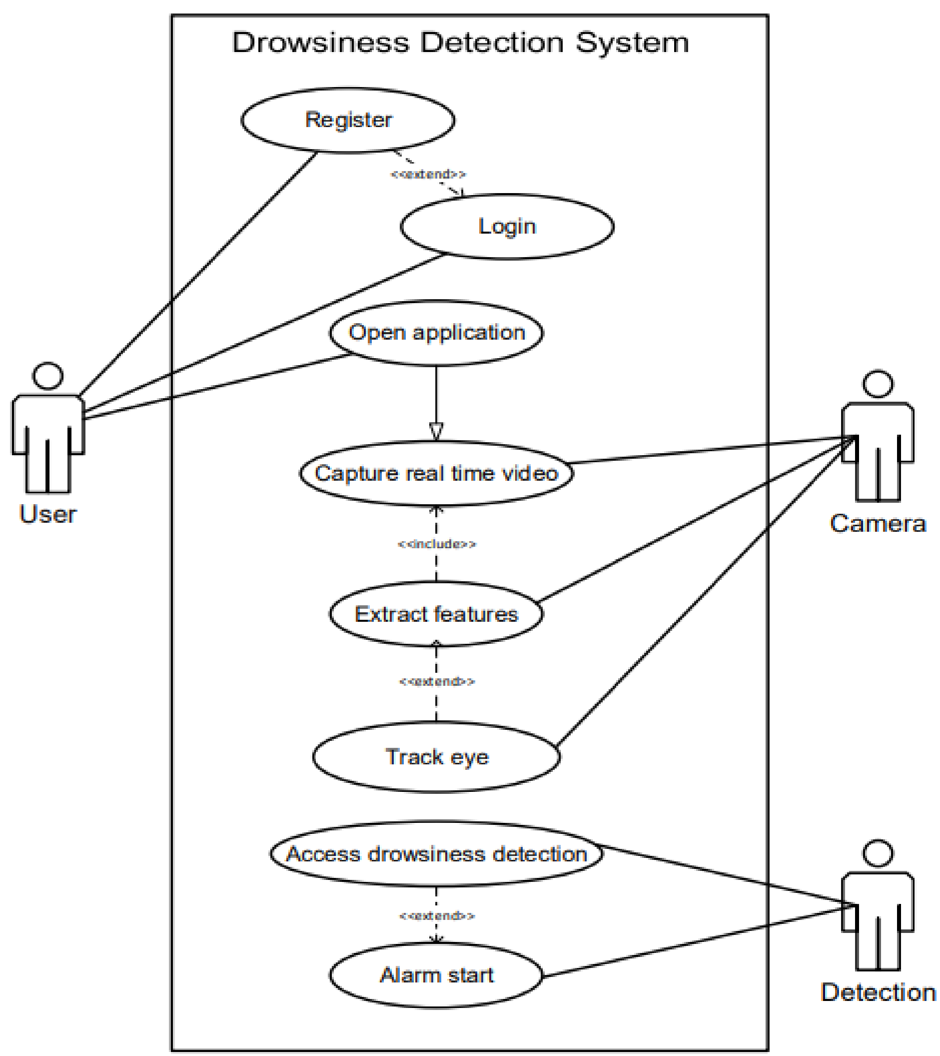
This is the architecture for detecting the drowsiness of the driver. First of all the system captures images through the webcam and after capturing it detects the face through haar cascade algorithm. It uses haar features which can detect the face. If the system founds it as face the it will proceed for next phase i.e eye detection. The eye is also detected using haar cascade features and it is used for blink frequency. The state of eye will be detected using perclos algorithm. Through this algorithm we can find the percentage of time the eye lids remains closed. If it found eyes in closed state then it detects driver in drowsy state and alerts him by an alarm. In some cases distraction can be measured by continuous gazing. The drivers face is analysed continuously to detect any distraction. If found then alarm is activated by the system.



4.2 USE CASE DIAGRAM

The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. These internal and external agents are known as actors. So use case diagram are consists of actors, usecases and their relationships.

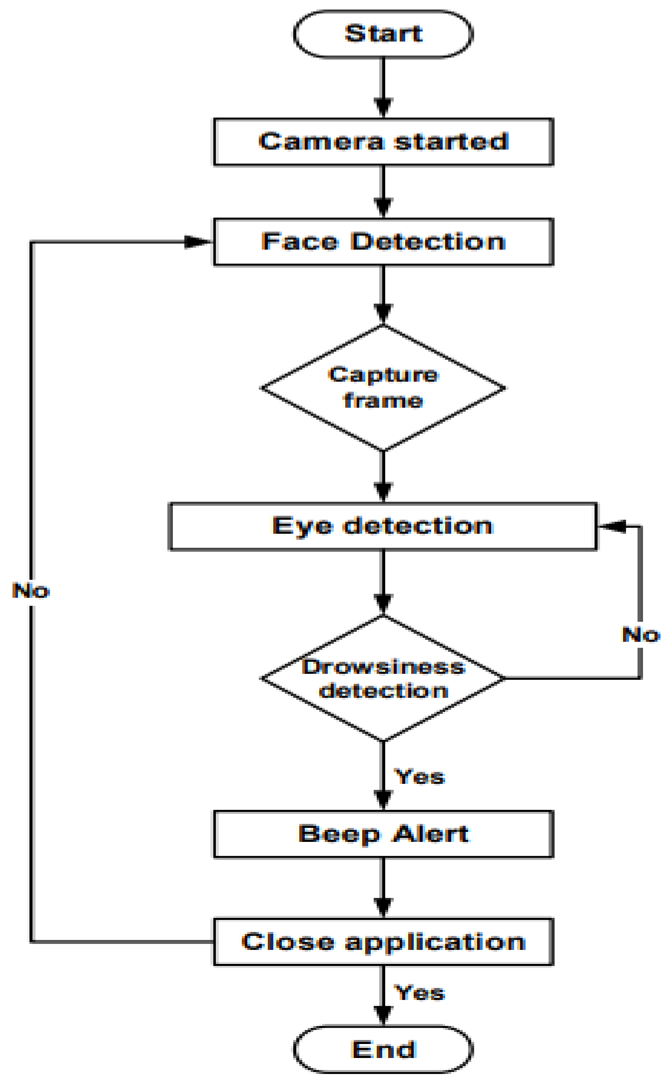
The use case diagram depicts the project flow sequence. It ends at the Alarm Sound which is considered to be the Alert. It travels through all the modules respectively as per Driver Drowsiness.



Use case diagram

4.3 ACTIVITY DIAGRAM

Activity diagram is another important diagram in UML to describe dynamic aspects of the system.Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another.

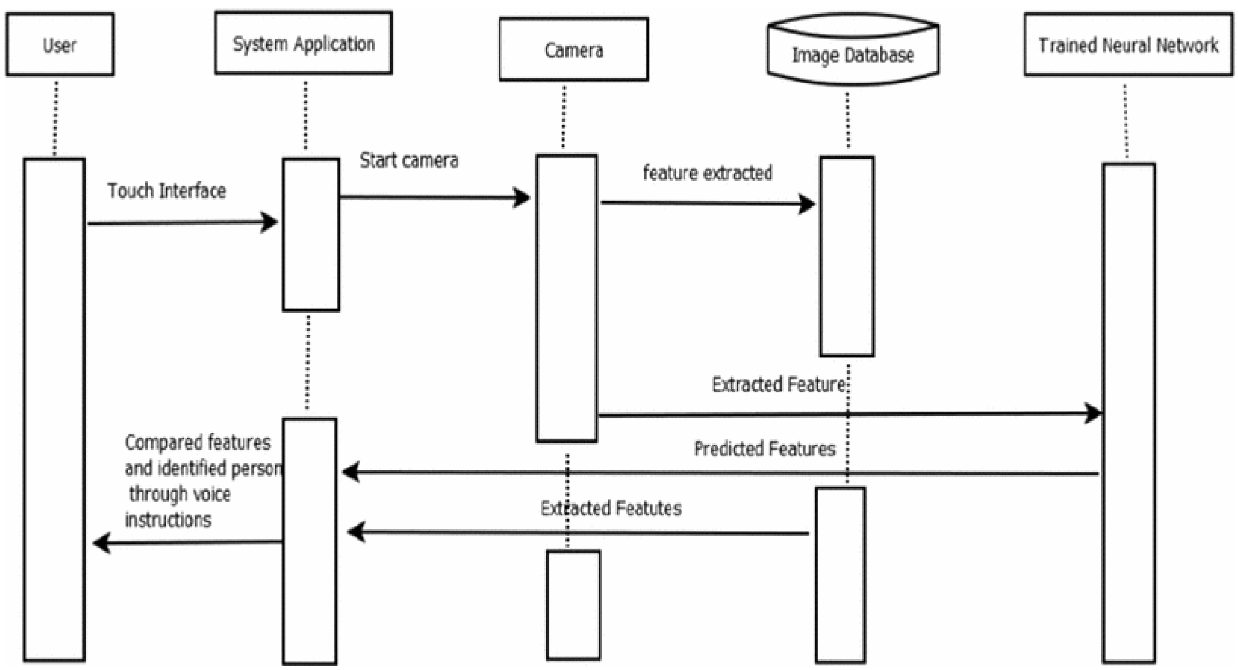


Activity diagram

Activity diagram contains a start button and proceed with the flowchart tools. It contains decision making loops such as the redetection, Face Detection, Eye Detection, etc. It ends with a finish button or Alarm Alert.

4.4 SEQUENCE DIAGRAM

Sequence diagram is the most common kind of interaction diagram, which focuses on the message interchange between a number of lifelines. Sequence diagram describes an interaction by focusing on the sequence of messages that are exchanged, along with their corresponding occurrence specifications on the lifelines.

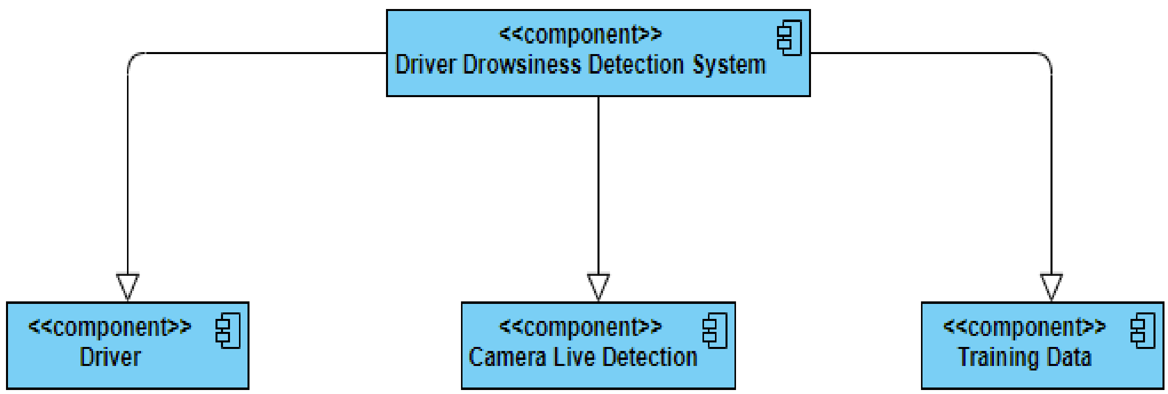


Sequence diagram

Sequence diagram depicts or describes the relationship and communication between the Driver as User, Dashboard as Interface, Camera and database. It shows the sequence flow of the project according to the drivers drowsiness.

4.5 COMPONENT DIAGRAM

Sequence diagram Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. Component diagrams are different in terms of nature and behaviour. Component diagrams are used to model physical aspects of a system.



Component diagram

Component diagram depicts the major components being used in the project. It deals with user, database and admin. The nodes appear as boxes , and the artifacts allocated to each node appear as rectangles within the boxes. A single node in a componenet diagram may conceptually represent multiple physical nodes, such as a cluster of databse servers. There are two type of nodes : Device Node, Execution Environment Node.

CHAPTER V

SYSTEM IMPLEMENTATION

5.1 MODULAR DIVISION:

The entire architecture is divided into 6 modules.

1. Face Detection
2. Eye Detection
3. Face Tracking
4. Eye Tracking
5. Drowsiness Detection
6. Distraction Detection

Face Detection:

This module takes input from the camera and tries to detect a face in the video input. The detection of the face is achieved through the Haar classifiers mainly, the **Frontal face cascade classifier**. The face is detected in a rectangle format and converted to grayscale image and stored in the memory which can be used for training the model.

Eye Detection:

Since the model works on building a detection system for drowsiness we need to focus on the eyes to detect drowsiness. The eyes are detected through the video input by implementing a haar classifier namely **Haar Cascade Eye Classifier**. The eyes are detected in rectangular formats

Face Tracking:

Due to the real-time nature of the project, we need to track the faces continuously for any form of distraction. Hence the faces are continuously detected during the entire time.

Eye Tracking:

The input to this module is taken from the previous module. The eyes state is determined through Perclos algorithm.

Drowsiness detection:

In the previous module the frequency is calculated and if it remains 0 for a longer period then the driver is alerted for the drowsiness through an alert from the system

Distraction detection:

In the face tracking module the face of the driver is continuously monitored for any frequent movements or the long gaze of the eyes without any blinks which can be treated as lack of concentration of the driver and is alerted by the system for distraction.

5.1.1 Haar Cascade

Haar Cascade is based on the concept of features which are proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It can be used to detect objects from an image or a video.

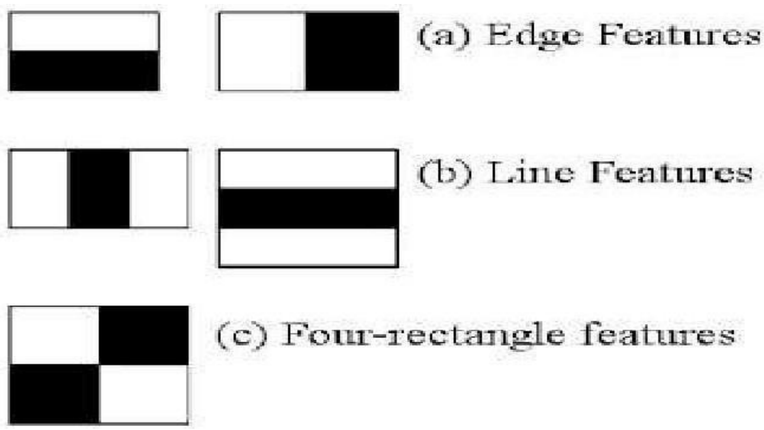
This algorithm comprises of four stages:

- Haar Feature Selection
- Creating Integral Images
- Adaboost Training
- Cascading Classifiers

Though Haar Cascade is used for detecting almost all objects, it is popular for detecting faces in images. Adaboost which both selects the best features and trains the classifiers that use them. This algorithm constructs a “strong” classifier as a linear combination of weighted simple “weak” classifiers.

A Haar feature considers adjacent rectangular regions at a specific location in a detection window, sums up the intensities of the pixels in each region and calculates the difference between these sums. During the detection phase, a window of the target size is moved over the input image, and for each subsection of the image and Haar features are calculated. This difference is then compared to a learned threshold that separates non-objects from objects. Because each Haar feature is only a "weak classifier" i.e. its detection quality is slightly better than random guessing and a large number of Haar features are necessary to describe an object with sufficient accuracy and are therefore they are organized into **cascade classifiers** to form a strong classifier.

Haar-like features are digital image features used in object recognition. They owe their name to their intuitive similarity with Haar wavelets and were used in the first real-time face detector. A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums. This difference is then used to categorize subsections of an image. For example, with a human face, it is a common observation that among all faces the region of the eyes is darker than the region of the cheeks. Therefore, a common Haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region. The position of these rectangles is defined relative to a detection window that acts like a bounding box to the target object.



Different types in feature extraction

Integral image is a data structure and algorithm for generating the sum of values in a rectangular subset of a grid. The goal is reducing the number of computations needed to obtain the summations of pixel intensities within a window.

The idea is transforming an input image into a summed-area table, where the value at any point (x, y) in that table is the sum of all the pixels above and to the left of (x, y), inclusive:

$$I(x,y) = \sum_{\substack{x' \le x \\ y' \le y}} i(x', y')$$

Where *I(x,y)* is the value of the integral image pixel in the position *(x,y)*, while *i(x,y)* is the corresponding intensity in the original image. It is a recursive formula, hence, if we start from one corner of the input image, we will have the same result in the integral image.

1	3
7	9

0	0	0
0	1	4
0	8	20

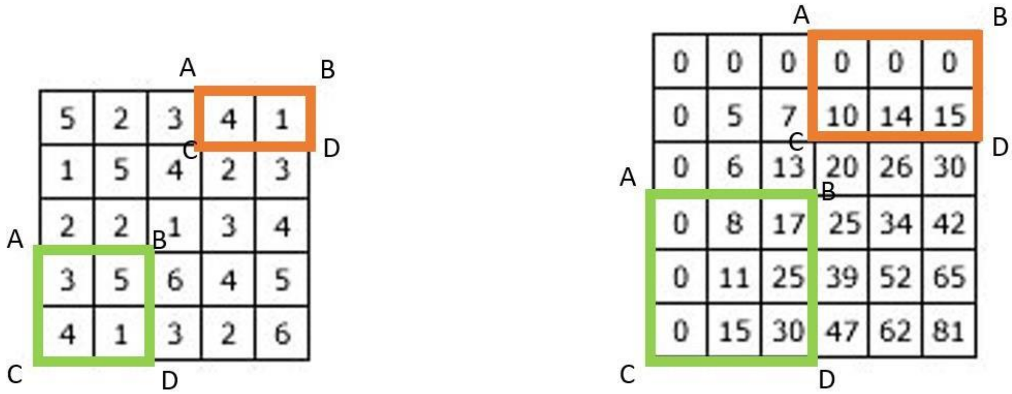
Calculation of integral image from an actual image

We add one row and column of zeros, since we need one step backward in order to start the recursive formula. Hence, if actual image is *w* pixels wide and *h* pixels high, then the integral of this image will be *w+1* pixels wide and *h+1*pixels high.

Moving to the computations, let's start from the first pixel in the original image from fig. 3.3 with intensity 1: the integral image returns exactly the same value, since it is computing (1+0+0). Then, pixel '3' becomes '4', since it is 3+1+0+0. With the same procedure, we obtain an '8' (7+1+0) and a '20' (9+3+1+7).

$$\sum_{\substack{x_0 < x \leq x_1 \\ y_0 < y \leq y_1}} i(x,y) = I(D) + I(A) - I(B) - I(C)$$

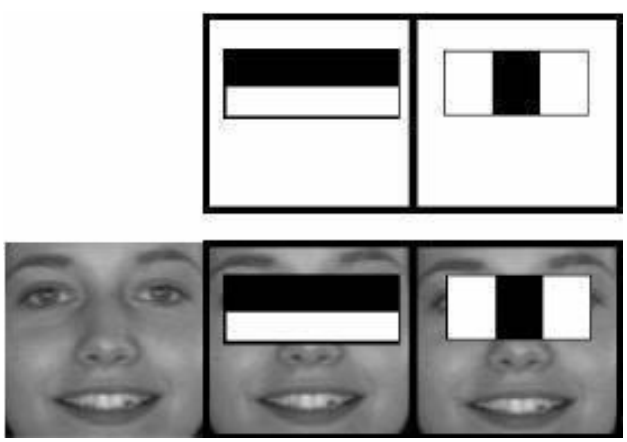
Now, we have a new image. This image is useful in an unique property of the integral image. Indeed, it turned out that if you need to compute the summation within a window in the input image, hence that summation is equal to a linear combination of the corresponding window's corner in the integral image, as follows:



Converting a 5x5 image to a 6x6 integral image

This reduces the number of computations. To give you an idea, consider a 100×100 image with a 9×9 window. We want to compute the sum of the pixel intensities within that window, which requires 8 operations. If we repeat this procedure 100 times, we obtain 800 operations.

Now let’s see the integral image approach. First, we compute the summed-area table, which requires 56 operations. Then, considering the same 9×9 window, to compute the sum of pixel intensity we just need the above formula, which is made of 3 operations. Hence, the total number of operations is 56+3*100=356. As you can see, it is less than a half. This procedure is widely used in computer vision and Haar Cascade algorithm is based exactly on that.



Extracting different features

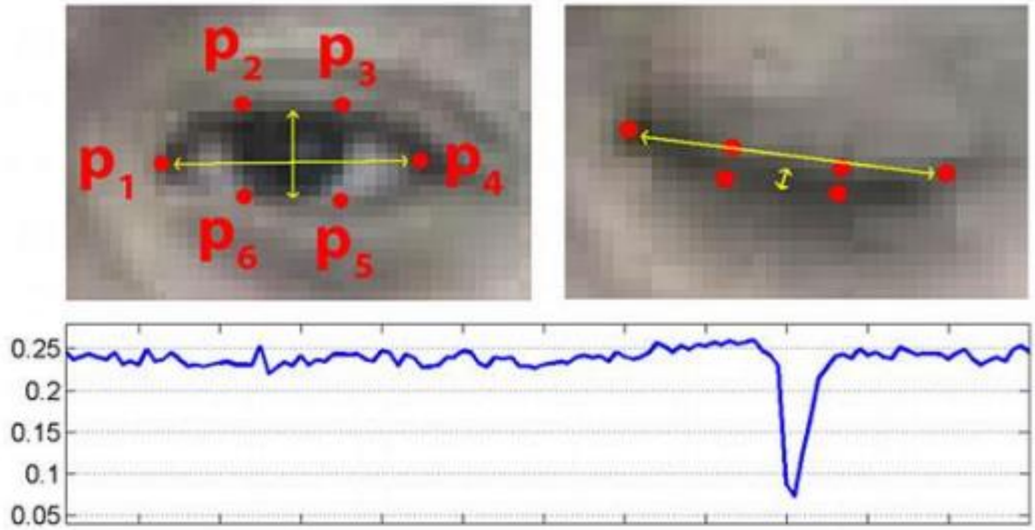
the cascade classifiers for the eyes region. In second picture eyes region is in black colour or both the eyes will have white colours. If the system founds the same pixel intensity at both the places then it detects them as the eyes. The detection is based on the intensity of the pixels around a particular object. The eyes region is white in colour and the region around the eye is black in colour. If there are white pixels found then it detects the region as eye.

Selecting the most relevant features is performed through Adaboost technique which selects the best features and trains the classifiers that use them. This algorithm uses "Haar Cascade Frontal Face" classifier for detecting the faces since we need to detect only the frontal part of the face.

5.1.2 PerClos

PERcentage of eye CLOSure (PERCLOS) is defined as the proportion of time for which the eyelid remains closed more than 70-80% within a predefined time period. Level of drowsiness can be judged based on the PERCLOS threshold value.

Perclos is a drowsy detection measure used to calculate the percentage of eyelid closure over the pupil over time. It is used by various real-time drowsiness detection systems and is able to yield effective results. Developers use different set of hardware to capture the closure movement of the eyelids for developing the accuracy of the system. This project uses camera mounted on the dashboard of the vehicle and is set up in such a way that the driver is visible on the camera. This helps in better detection of the face and calculating the eyelid closure frequency using Perclos measure. A total of six points are marked for each eye and the Euclidean distance is calculated for each eye. The eye aspect ratio for each eye are then calculated for average eye-aspect ratio.



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

Where

A is the distance between the 2-points (p2 and p6)

B is the distance between the 2-points (p3 and p5)

C is the distance between 2-points (p1 and p4)

The number of frames are 20 for this system with a threshold value of 0.25. Based on this value, the alarm is made to sound. If the eye aspect ratio is less than the threshold value for given number of frames then it will detect the driver as drowsy and alert is given through an alarm.

CHAPTER VI

CONCLUTION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

It completely meets the objectives and requirements of the system. The framework has achieved an unfaltering state where all the bugs have been disposed of. The framework cognizant clients who are familiar with the framework and comprehend it's focal points and the fact that it takes care of the issue of stressing out for individuals having fatigue-related issues to inform them about the drowsiness level while driving.

6.2 FUTURE WORK

The model can be improved incrementally by using other parameters like blink rate, yawning, state of the car, etc. If all these parameters are used it can improve the accuracy by a lot. We plan to further work on the project by adding a sensor to track the heart rate in order to prevent accidents caused due to sudden heart attacks to drivers. Same model and techniques can be used for various other uses like Netflix and other streaming services can detect when the user is asleep and stop the video accordingly. It can also be used in application that prevents the user from sleeping.

APPENDIX I

SAMPLE CODING

<u>Driver Drowsiness Detection System :</u>

```
import cv2
import os
from keras.models import load_model
import numpy as np
from pygame import mixer
import time

mixer.init()
sound = mixer.Sound('alarm.wav')

face = cv2.CascadeClassifier('haar cascade
files\haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier('haar cascade
files\haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier('haar cascade
files\haarcascade_righteye_2splits.xml')

lbl=['Close','Open']

model = load_model('models/cnncat2.h5')
path = os.getcwd()
```

```
cap = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
count=0
score=0
thicc=2
rpred=[99]
lpred=[99]

while(True):
    ret, frame = cap.read()
    height,width = frame.shape[:2]

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces =
face.detectMultiScale(gray,minNeighbors=5,scaleFactor=1.1,minSize=(25,25))
    left_eye = leye.detectMultiScale(gray)
    right_eye =  reye.detectMultiScale(gray)

    cv2.rectangle(frame, (0,height-50) , (200,height) , (0,0,0) ,
thickness=cv2.FILLED )

    for (x,y,w,h) in faces:
        cv2.rectangle(frame, (x,y) , (x+w,y+h) , (100,100,100) , 1 )

    for (x,y,w,h) in right_eye:
        r_eye=frame[y:y+h,x:x+w]
```

```
    count=count+1
    r_eye = cv2.cvtColor(r_eye,cv2.COLOR_BGR2GRAY)
    r_eye = cv2.resize(r_eye,(24,24))
    r_eye= r_eye/255
    r_eye=  r_eye.reshape(24,24,-1)
    r_eye = np.expand_dims(r_eye,axis=0)
    predict_x=model.predict(r_eye)
    rpred=np.argmax(predict_x,axis=1)
    if(rpred[0]==1):
        lbl='Open'
    if(rpred[0]==0):
        lbl='Closed'
    break

for (x,y,w,h) in left_eye:
    l_eye=frame[y:y+h,x:x+w]
    count=count+1
    l_eye = cv2.cvtColor(l_eye,cv2.COLOR_BGR2GRAY)
    l_eye = cv2.resize(l_eye,(24,24))
    l_eye= l_eye/255
    l_eye=l_eye.reshape(24,24,-1)
    l_eye = np.expand_dims(l_eye,axis=0)
    predict_x=model.predict(l_eye)
    lpred=np.argmax(predict_x,axis=1)
    if(lpred[0]==1):
        lbl='Open'
    if(lpred[0]==0):
```

```
            lbl='Closed'
        break

    if(rpred[0]==0 and lpred[0]==0):
        score=score+1
        cv2.putText(frame,"Closed",(10,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
    # if(rpred[0]==1 or lpred[0]==1):
    else:
        score=score-1
        cv2.putText(frame,"Open",(10,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)

    if(score<0):
        score=0
    cv2.putText(frame,'Score:'+str(score),(100,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
    if(score>15):
        #person is feeling sleepy so we beep the alarm
        cv2.imwrite(os.path.join(path,'image.jpg'),frame)
        try:
            sound.play()

        except:  # isplaying = False
            pass
        if(thicc<16):
            thicc= thicc+2
```

```
        else:
            thicc=thicc-2
            if(thicc<2):
                thicc=2
        cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thicc)
    cv2.imshow('frame',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

<u>Training Model :</u>

```
import os
from keras.preprocessing import image
import matplotlib.pyplot as plt
import numpy as np
from keras.utils.np_utils import to_categorical
import random,shutil
from keras.models import Sequential
from keras.layers import Dropout,Conv2D,Flatten,Dense, MaxPooling2D,
BatchNormalization
from keras.models import load_model

def generator(dir, gen=image.ImageDataGenerator(rescale=1./255),
shuffle=True,batch_size=1,target_size=(24,24),class_mode='categorical' ):
```

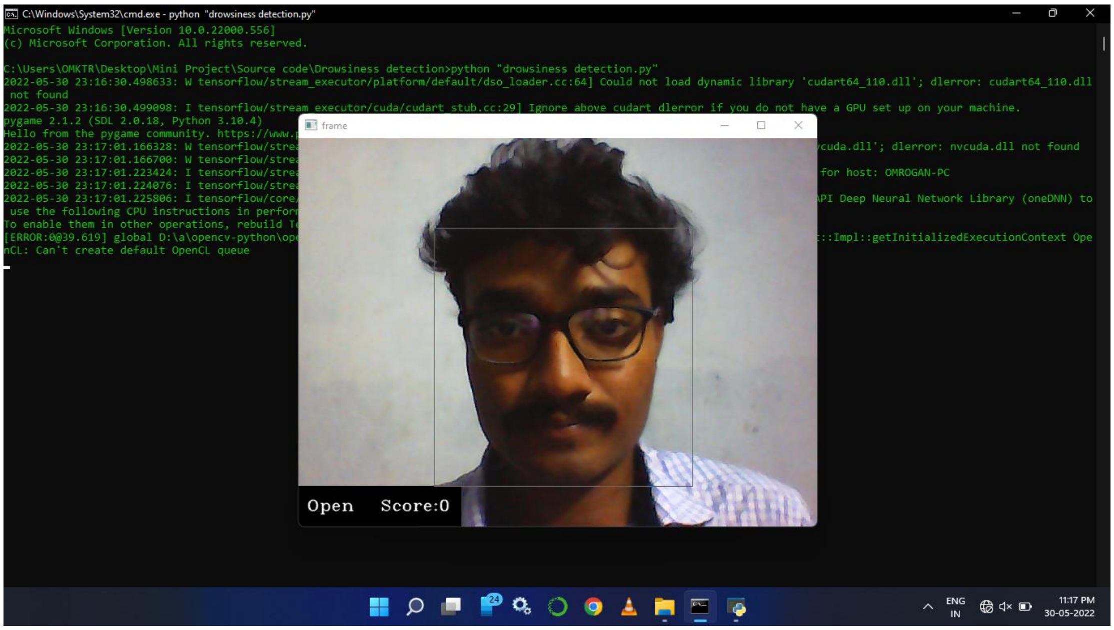
```
    return
gen.flow_from_directory(dir,batch_size=batch_size,shuffle=shuffle,color_mod
e='grayscale',class_mode=class_mode,target_size=target_size)

BS= 32
TS=(24,24)
train_batch= generator('data/train',shuffle=True,
batch_size=BS,target_size=TS)
valid_batch= generator('data/valid',shuffle=True,
batch_size=BS,target_size=TS)
SPE= len(train_batch.classes)//BS
VS = len(valid_batch.classes)//BS
print(SPE,VS)

# img,labels= next(train_batch)
# print(img.shape)

model = Sequential([
    Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(24,24,1)),
    MaxPooling2D(pool_size=(1,1)),
    Conv2D(32,(3,3),activation='relu'),
    MaxPooling2D(pool_size=(1,1)),
#32 convolution filters used each of size 3x3
#again
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(1,1)),
```

```
#64 convolution filters used each of size 3x3
#choose the best features via pooling

#randomly turn neurons on and off to improve convergence
    Dropout(0.25),
#flatten since too many dimensions, we only want a classification output
    Flatten(),
#fully connected to get all relevant data
    Dense(128, activation='relu'),
#one more dropout for convergence' sake :)
    Dropout(0.5),
#output a softmax to squash the matrix into output probabilities
    Dense(2, activation='softmax')
])

model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['acc
uracy'])

model.fit_generator(train_batch,
validation_data=valid_batch,epochs=15,steps_per_epoch=SPE
,validation_steps=VS)

model.save('models/cnnCat2.h5', overwrite=True)
```

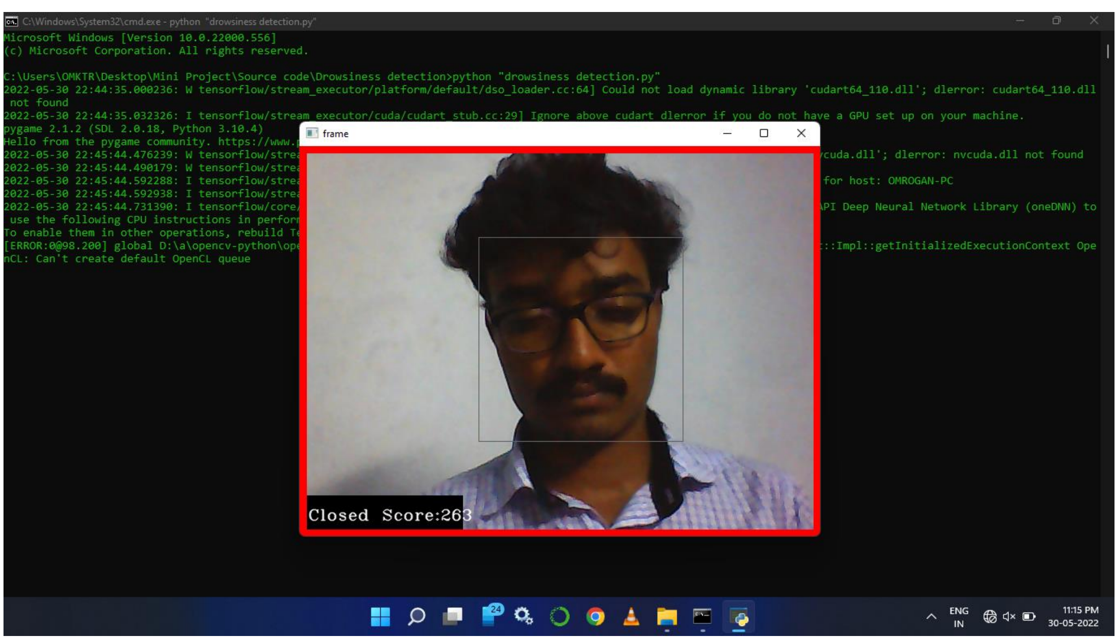
APPENDIX II

SCREENSHOTS & DETECTION

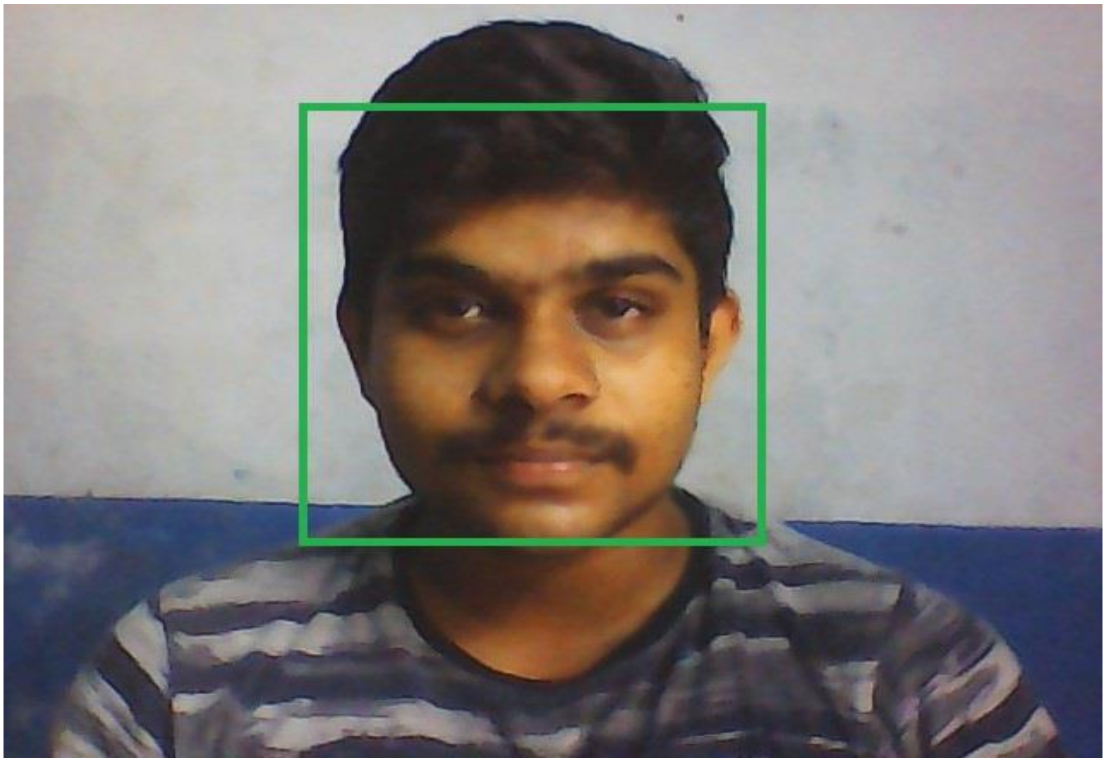
Active :



Drowsy :



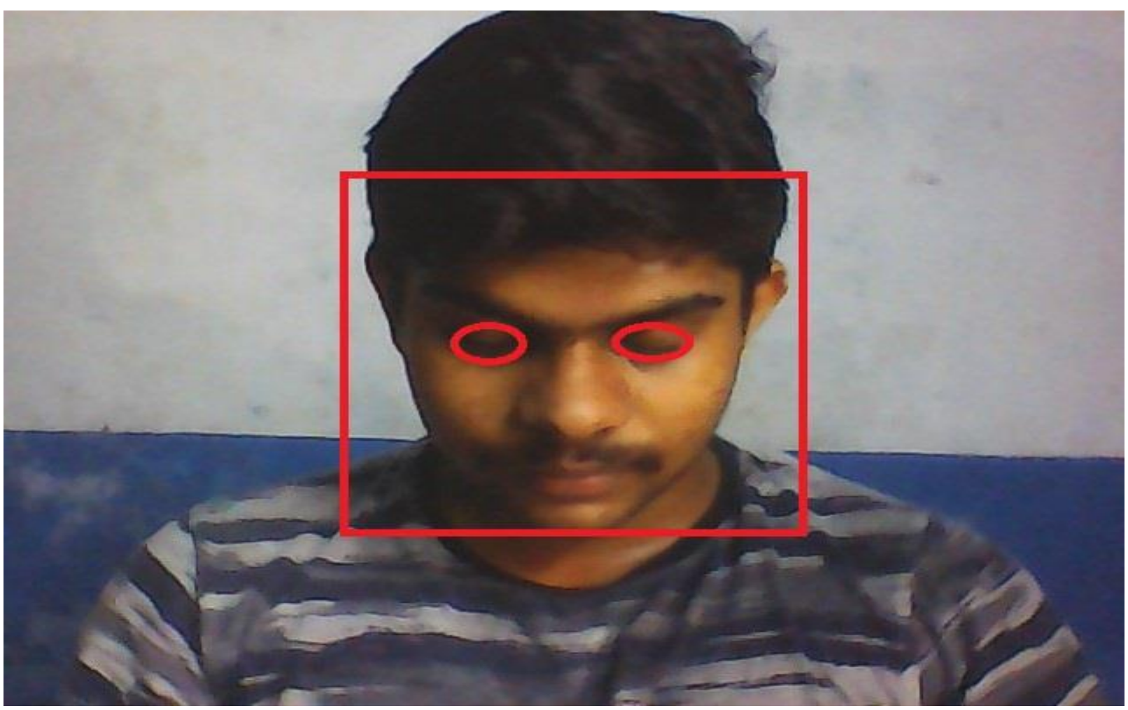
Region Of Interest :



Eye Tracking :

Gray Scale :

Detection :



REFERENCES

1. Computationally efficient face Detection; B. Scholkopf-A.

2. Use of the Hough Transformation to Detect Lines and Curves in Picture; R. Duda and P.E.Hart.

3. Jain, “Face Detection in color Images; R.L. Hsu, M.Abdel-Mottalab, and A.K. Jain

4. Open/Closed eye analysis for Drowsiness Detection; P.R. Tabrizi and R.A. Zoroofi.

5. http://ncrb.gov.in/StatPublications/ADSI/ADSI2015/chapter1A%20traffic %20accidents.pdf.

6. http://www.jotr.in/text.asp?2013/6/1/1/118718

7. http://dlib.net/face_landmark_detection_ex.cpp.html

8. https://data-flair.training/blogs/python-project-driver-drowsiness-detection-system/