

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAUM-590010



A Report on Project Work

DRIVER DROWSINESS DETECTION AND ALARM SYSTEM

*Submitted in partial fulfillment of the
Requirements for the award of the degree of*

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

By

**Prajval P
Skandesh Suresh
Yekshith Bushan N**

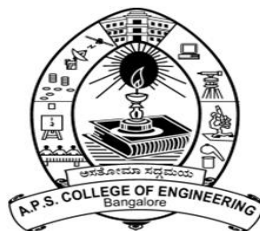
**1AP17EC033
1AP17EC040
1AP17EC047**

Under the Guidance of

Ms. Usha H N

Assistant Professor

Dept. of Electronics and Communication Engineering,
A.P.S. College of Engineering,
Bangalore -560082



Carried out at

Dept. of Electronics and Communication Engineering
A.P.S. College of Engineering,
Bangalore-560082

**A.P.S. COLLEGE OF ENGINEERING
BANGALORE**

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING



CERTIFICATE

Certified that the project work entitled **“DRIVER DROWSINESS DETECTION AND ALARM SYSTEM”** is a bona fide work carried out in partial fulfillment for the degree of Bachelor of Engineering in Electronics and Communication Engineering of the Visvesvaraya Technological University, Belgaum during the year 2020-21. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the Bachelor of Engineering Degree.

Prajval P

Skandesh Suresh

Yekshith Bushan N

1AP17EC033

1AP17EC040

1AP17EC047

Ms. Usha H N
Assistant Professor
Dept. of ECE,
A.P.S.C.E,
Bangalore-560082

Dr. Jagadeesh H S
H.O.D,
Dept. of ECE
A.P.S.C.E,
Bangalore- 560082

Dr. Jagadeesh H S
Principal,
A.P.S.C.E,
Bangalore-560082

Name of examiners

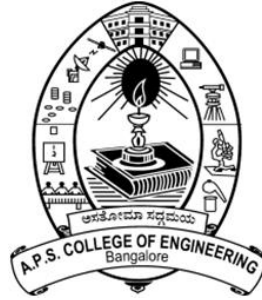
Signature with date

1.

2.

**A.P.S. COLLEGE OF ENGINEERING
BANGALORE**

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING



DECLARATION

We **SKANDESH SURESH, YEKSHITH BUSHAN N, PRAJVAL P**, bearing USNs **1AP17EC040, 1AP17EC047, 1AP17EC033**, students of the final semester B.E., **Department of Electronics & Communication Engineering, APS College of Engineering, Bangalore**, hereby declare that the project work entitled **“DRIVER DROWSINESS DETECTION AND ALARM SYSTEM”** has been independently carried out by us at APS College of Engineering, Bangalore and submitted in partial fulfillment of the requirements for the award of the degree in Bachelor of Engineering in Electronics & Communication Engineering of Visvesvaraya Technological University, Belgaum, during the academic year of 2020-21.

Place: Bangalore

Date: /07/2021

Prajval P (1AP17EC033)

Skandesh Suresh (1AP17EC040)

Yekshith Bushan N (1AP17EC047)

ACKNOWLEDGEMENT

We would like to express our heart-felt thanks to all those who have made this project possible.

We are grateful to our HOD and Principal, Dr. Jagadeesh H S, Dept. of ECE, A.P.S.C.E, for permitting us to take up this project and for his continuous support and encouragement.

We would like to express our gratitude towards our project guide, Ms. Usha H N, Assistant Professor, Dept. of ECE, A.P.S.C.E. for her constant support, encouragement, motivation, criticism, ideas which were valuable to the project implementation.

We would also like to thank all the faculties of our department for their support.

Last, but not the least, we would like to thank our parents and friends whose constant encouragement and support was crucial in execution and completion of this work

Prajval P	(1AP17EC033)
Skandesh Suresh	(1AP17EC040)
Yekshith Bushan N	(1AP17EC047)

ABSTRACT

The modern technology has developed well and has brought about advancements in driving systems. Yet the number of road accidents in India and worldwide have not dropped significantly. The major reasons for the same are drowsiness and fatigue. Hence the driver drowsiness and fatigue detection are a major area which could be improved to prevent a large number of sleep induced road accidents. We hope to develop a Real Time Drowsiness Detection System to install in motor vehicles with the help of conventional Computer Vision applications. The system would employ various parameters such as blink-rate, eye closure, yawning into consideration to effectively identify the drowsiness of the driver in real time, and also the physical condition of the vehicle taking many factors into consideration such as temperature, weather and speed of the vehicle. The proposed methods with the help of the additional sensors installed, we hope to contribute in reducing the number of road accidents using simple methodologies and approaches.

List of Figures

Figure 3.1: System Architecture.....	9
Figure 4.1: Algorithm Flowchart.....	13
Figure 4.2: Visualization of the 68 facial landmark coordinates.....	15
Figure 4.3: Coordinates to calculate EAR.....	15
Figure 4.4: Coordinates to calculate MAR.....	16
Figure 4.5: Ideal Screen output.....	18
Figure 4.6: Drowsiness Detection.....	19
Figure 4.7: Yawning Detection.....	19
Figure 4.8: Flowchart for sensor functioning.....	21
Figure 4.9: MQ-9 Sensor.....	23
Figure 4.10: MQ-9 connection with Arduino UNO.....	23
Figure 4.11: MQ-9 Sensor Dimensions.....	24
Figure 4.12: MQ-9 Sensor sensitivity adjustment.....	24
Figure 4.13: MQ-9 Sensor Serial output.....	24
Figure 4.14: DTH-11 Specifications.....	26
Figure 4.15: DTH-11 Temperature Sensor.....	26
Figure 4.16: DTH-11 Components.....	26
Figure 4.17: LM-35 Temperature Sensor.....	26
Figure 4.18: LM-35 Sensor components.....	26
Figure 4.19: DTH-11 connection with Arduino UNO.....	27
Figure 4.20: DTH-11 Arduino output.....	27
Figure 4.21: Ultrasonic Sensor.....	29
Figure 4.22: Pin Configurations of Ultrasonic Sensor.....	29
Figure 4.23: HC SR04 connection with Arduino UNO.....	29
Figure 4.24: HC SR04 output.....	29
Figure 4.25: Flowchart of Ultrasonic Sensor functioning.....	30
Figure 4.26: Sensor activity in case of accident.....	31
Figure 5.1: Raspberry Pi 3 Model B.....	32
Figure 5.2: Raspberry Pi OS Logo.....	33
Figure 5.3: Python Programming Language Logo.....	34
Figure 5.4: Visual Studio Code Logo.....	35
Figure 5.5: C Language Logo.....	36
Figure 5.6: Arduino IDE Logo.....	37

Figure 5.7: Dlib Library Logo.....	38
Figure 5.8: NumPy Library Logo.....	39
Figure 5.9: OpenCV Library Logo.....	40
Figure 5.10: TensorFlow Library Logo.....	41
Figure 5.11: Keras Library Logo.....	42
Figure 5.12: Pygame Library Logo.....	43
Figure 5.13: Argparse module Logo.....	44
Figure 5.14: SciPy Library Logo.....	45

Table of Contents

1.	Chapter 1: Introduction.....	1
1.1	Project Introduction	
1.2	Brief history into Facial recognition algorithms	
2.	Chapter 2: Literature Survey.....	6
2.1	Driver Drowsiness Detection System	
2.2	Driver Fatigue Detection Based on Eye Tracking	
2.3	Drowsiness Detection of a Driver using Conventional Computer Vision Articles	
2.4	Vehicle speed sensing and smoke detecting System	
2.5	Development of Sensor and Optimal Placement for Smoke detection in an Electric Vehicle Battery Pack	
2.6	Obstacle Avoidance with Ultrasonic Sensors	
3.	Chapter 3: System Architecture.....	9
3.1	Raspberry Pi	
3.2	Camera	
3.3	Alarm System	
3.4	Sensors	
4.	Chapter 4: Proposed Method.....	11
4.1	Technical Terms and Features	
4.2	Algorithm	
4.3	Sensors	
5.	Chapter 5: Hardware and Software Requirements.....	32
5.1	Hardware Requirements	
5.2	Software Requirements	
5.3	Libraries	

6.	Chapter 6: Advantages and Disadvantages.....	48
6.1	Advantages	
6.2	Disadvantages	
7.	Chapter 7: Applications.....	49
8.	Chapter 8: Conclusion.....	50
9.	Chapter 9: Future Scope.....	51

References

Chapter 1

Introduction

1.1 Project Introduction

Automotive population is increasing exponentially in our country. The Biggest problem regarding the increased use of vehicles is the rising number of road accidents. Road accidents are undoubtedly a global menace in our country. The frequency of road accidents in India is among the highest in the world. According to the reports of the National Crime Records Bureau (NCRB) about 4,37,396 road accidents were recorded across India in 2019, resulting in the death of 154732 people. The Global Status Report on Road Safety published by the World Health Organization (WHO) identified the major causes of road accidents are due to errors and carelessness of the driver.

The major driver errors are caused by drowsiness, drunken and reckless behavior of the driver. The resulted errors and mistakes contribute much loss to humanity. In order to minimize the effects of driver abnormalities, a system for abnormality monitoring must be inbuilt with the vehicle.

Drowsiness Detection and Alarm System will be developed using non-intrusive machine vision-based concepts. We hope to test the proposed method to detect fatigue after carefully observing and analyzing the driver's face and eyes. In such a case when fatigue is detected, a warning signal is issued to alert the driver. The system deals with using information obtained for the binary version of the image to find the edges of the face, which narrows down the area of where the eyes may exist. Considering the knowledge that the eye regions in the face present great intensity changes, the eyes are located by finding the significant intensity changes in the face.

The sensors will be an additional feature to this concept where it will help determine the external factors of the vehicle like the temperature, weather outside and if any harmful gases emitted by the engine is causing the drivers drowsiness.

1.2 Brief History into Facial Recognition Algorithms

1.2.1 The dawn of Facial Recognition – 1960s

The earliest pioneers of facial recognition were Woody Bledsoe, Helen Chan Wolf and Charles Bisson. In 1964 and 1965, Bledsoe, along with Wolf and Bisson began work using computers to recognize the human face.

Due to the funding of the project originating from an unnamed intelligence agency, much of their work was never published. However, it was later revealed that their initial work involved the manual marking of various “landmarks” on the face such as eye centres, mouth etc. These were then mathematically rotated by a computer to compensate for pose variation. The distances between landmarks were also automatically computed and compared between images to determine identity.

These earliest steps into Facial Recognition by Bledsoe, Wolf and Bisson were severely hampered by the technology of the era, but it remains an important first step in proving that Facial Recognition was a viable biometric.

1.2.2 Advancing the accuracy of Facial Recognition – 1970s

Carrying on from the initial work of Bledsoe, the baton was picked up in the 1970s by Goldstein, Harmon and Lesk who extended the work to include 21 specific subjective markers including hair colour and lip thickness in order to automate the recognition.

While the accuracy advanced, the measurements and locations still needed to be manually computed which proved to be extremely labour intensive yet still represents an advancement on Bledsoe’s RAND Tablet technology

1.2.3 Linear Algebra used for Facial Recognition – 1980s/90s

It wasn’t until the late 1980s that we saw further progress with the development of Facial Recognition software as a viable biometric for businesses. In 1988, Sirovich and Kirby began applying linear algebra to the problem of facial recognition.

A system that came to be known as Eigenface showed that feature analysis on a collection of facial images could form a set of basic features. They were also able to show that less than one hundred values were required in order to accurately code a normalized facial image.

In 1991, Turk and Pentland carried on the work of Sirovich and Kirby by discovering how to detect faces within an image which led to the earliest instances of automatic facial recognition. This significant breakthrough was hindered by technological and environmental factors; however, it paved the way for future developments in Facial Recognition technology.

Most naturally, we think of an image as a matrix of pixel values. For simplicity, we restrict our attention to grayscale images.

1.2.4 FERET Programme – 1990s/2000s

The National Institute of Standards and Technology (NIST) began Face Recognition Vendor Tests (FRVT) in the early 2000s. Building on FERET, FRVTs were designed to provide independent government evaluations of facial recognition systems that were commercially available, as well as prototype technologies. These evaluations were designed to provide law enforcement agencies and the U.S. government with information necessary to determine the best ways to deploy facial recognition technology.

1.2.5 Face Recognition Grand Challenge – 2006

Launched in 2006, the primary goal of the Face Recognition Grand Challenge (FRGC) was to promote and advance face recognition technology designed to support existing face recognition efforts in the U.S. Government.

The FRGC evaluated the latest face recognition algorithms available. High-resolution face images, 3D face scans, and iris images were used in the tests. The results indicated that the new algorithms were 10 times more accurate than the face recognition algorithms of 2002 and 100 times more accurate than those of 1995, showing the advancements of facial recognition technology over the past decade.

1.2.6 Social Media – 2010 – Current

Back in 2010, Facebook began implementing facial recognition functionality that helped identify people whose faces may feature in the photos that Facebook users update daily. The feature was instantly controversial with the news media, sparking a slew of privacy-related articles. However, Facebook users by and large did not seem to mind. Having no apparent negative impact on the website's usage or popularity, more than 350 million photos are uploaded and tagged using face recognition each day.

1.2.7 iPhone X – 2017

Facial Recognition technology advanced rapidly from 2010 onwards and September 12, 2017 was another significant breakthrough for the integration of facial recognition into our day to day lives. This was the date that Apple launched the iPhone X – the first iPhone users could unlock with FaceID – Apple's marketing term for facial recognition. The technology is based on PrimeSense's previous work with low-cost infrared depth perception that was the basis of the Kinetic motion sensor.

1.2.8 NEC and Facial Recognition

Border controls, airlines, airports, transport hubs, stadiums, mega events, concerts, conferences. Biometrics are playing a growing role not only in the real-time policing and securing of increasingly crowded and varied venues worldwide, but also in ensuring a smooth, enjoyable experience for the citizens who visit them.

As a long-time committed pioneer of biometric research and solutions, NEC has developed multi-modal technologies including face, iris and voice recognition, finger and palmprint identification, and ear acoustic authentication, and supplemented them with AI and data analytics to enhance situational awareness and facilitate effective real-time or post-event action in both law-enforcement and consumer-oriented spheres.

Face recognition can often prove one of the best biometrics because images can be taken without touching or interacting with the individual being identified, and those images recorded and instantly checked against existing databases.

NEC's face recognition offers high-performance, scalable solutions for the most demanding real-time or post-event requirements. With face surveillance, search, identification and verification functions all on a single platform, it can be easily integrated into existing surveillance systems to extract faces in real time, match against an existing database or watchlist and produce real-time alerts to help reduce public safety risks.

With the ability to process and analyse multiple camera feeds and thousands of faces per minute, NEC's powerful face recognition is able to address the largest and most difficult security challenges with unparalleled efficiency, sensitivity, and perception.

In response to booming demand for varied biometrics, NEC has expanded its traditional range of face recognition applications from law-enforcement and security provision to new areas, and now boasts more than 1,000 active systems in over 70 countries and regions spanning police, immigration control agencies, national ID, banking, entertainment, stadium, conference venue systems, and many more.

Right now, many of our face recognition technology solutions are blazing a trail for first-time use in new areas and venues worldwide, such as end-to-end airport travel experiences in the U.S., mega event surveillance in Japan, and EU Summit security in Europe. In 2019, NEC was named Frost & Sullivan Asia Pacific Biometrics Company of the Year in recognition of its leading position in the

industry and foresight in innovating and developing future face recognition biometric solutions that maximize customer value and experience.

1.2.9 Balancing Security and Privacy

As the leading global pioneer of face recognition and other biometric technologies, and champion of our NEC Safer Cities Vision, NEC is fully vested in developing biometric recognition solutions and services that contribute to the creation of safe, secure, equal and efficient communities around the world.

Having positioned safety business as one of our key pillars of growth, we are keen to encourage businesses, consumers, and governments to work together to help balance the need for privacy with the benefits of protecting our society, securing our borders and providing consumer convenience without the fear of negative consequences.

At NEC, we strongly believe that face recognition can add significant value to our lives, and we seek to advance these technologies in ways that respect the worldwide principals of freedom, justice, rights to privacy, transparency and continuous improvement.

Chapter 2

Literature Survey

2.1 Driver Drowsiness Detection System

Authors: Athira Gopal, V Vineeth

Publication: Independent Publication

Summary:

The System is developed using non-intrusive machine vision-based concepts. It uses a small monochrome camera that points directly towards the driver's face and monitors the driver's eyes in order to detect fatigue. In case fatigue is detected, a warning signal is issued to alert the driver. Fatigue is detected using various facial recognition and detection algorithms such as Haar Cascade Classifiers and other similar algorithms.

Drawbacks:

The system relies solely on the state of the eyes, i.e., whether the eyes are opened or closed, and sets a threshold to the duration which the eyes can be closed. It doesn't make use of other governing factors like facial features, etc.

2.2 Driver Fatigue Detection Based on Eye Tracking

Authors: Mandalapu Sarada Devi, Dr. Preeti R Bajaj

Publisher: IEEE Computer Society

Summary:

A system is designed that uses video camera that points directly towards the driver's face in order to detect fatigue. If fatigue is detected, a warning signal is issued to alert the driver. Video files recorded by the camera are extracted. They are converted into frames. Once the eyes are located from each frame, by measuring the distances between the intensity changes in the eye area one can determine whether the eyes are open or closed. The system draws the conclusion that the driver is falling asleep and issues a warning signal, based on the number of frames the eyes are found closed.

Drawbacks:

This system is highly dependent on analyzing the intensity changes in the eye region in order to detect fatigue of the driver. Therefore, a fault can easily occur if the driver is not facing the camera or, if the driver closes his/her eyes for prolonged time which may not be due to drowsiness.

2.3 Drowsiness Detection of a Driver using Conventional Computer Vision Articles

Author: Hitendra Garg

Publisher: IEEE Xplore

Summary:

A system which uses pre-existing facial landmark detection to identify the state of drowsiness and fatigue. 68-facial landmarks help in shape prediction to clearly identify the various regions of the face. Each extracted frame is analyzed and to study facial features to determine the EAR and MAR for each frame. EAR and MAR values when exceed their respective threshold values for a certain number of consecutive frames an alert will be sent to the driver.

Drawbacks:

The work is based on behavioral analysis, high end camera installation and conventional algorithm to detect the possible coordinates.

2.4 Vehicle Speed Sensing and Smoke Detecting System

Authors: Prashanna Rangan (Kongu Engineering College)

Publisher: Research-gate publications

Summary:

For the project the drowsiness of the driver can cause due to external factors also. Such factors include Carbon Monoxide gas leakage from the engine or CNG gas from some particular vehicles. Using the MQ9 sensor we can detect such gages and alert the driver to prevent any causalities.

Drawbacks:

The sensor may occasionally fail and hence faulty signals could be sent to the microcontroller board.

2.4 Vehicle speed sensing and smoke detecting system

Authors: Prashanna Rangan (Kongu Engineering College)

Publisher: Research-gate publications

Summary:

For the project the drowsiness of the driver can cause due to external factors also. Such factors include Carbon Monoxide gas leakage from the engine or CNG gas from some particular vehicles. Using the MQ9 sensor we can detect such gases and alert the driver to prevent any casualties.

Drawbacks:

The sensor may occasionally fail and hence faulty signals could be sent to the microcontroller board.

2.5 Development of Sensor and Optimal Placement for Smoke Detection in an Electric Vehicle Battery Pack

Authors: Mukund Kulkarni, Saravanan Meenatchi Sundaram, Vinten Diwakar

Publisher: IEEE Computer Society

Summary:

In this project we plan to implement a sensor inside the dashboard to log the temperature and alert if the temperatures inside the car cross a certain temperature limit. This signal can be sent using the GSM module hence alerting the driver.

Drawback:

The sensor may fail due to weather conditions, and may send false data signals which could be faulty causing a misread.

2.6 Obstacle Avoidance with Ultrasonic Sensors

Authors: Johann Borenstein, Yoram Koren

Publisher: IEEE Computer Society

Summary:

Using the ultrasonic sensors, we can calculate the distance between the car and any obstacle. If the obstacle is very close to the car crossing the set limit this will indicate the driver of the obstacle and hence alerting to stop the movement of the car.

Drawbacks:

The sensor may malfunction, it may show inaccurate data due to dust particles in the sensor.

Chapter 3

System Architecture

The proposed method would make the system in a compact way so it can be installed inside the car in suitable way to monitor the driver's facial landmarks. The below figure represents the system architecture.

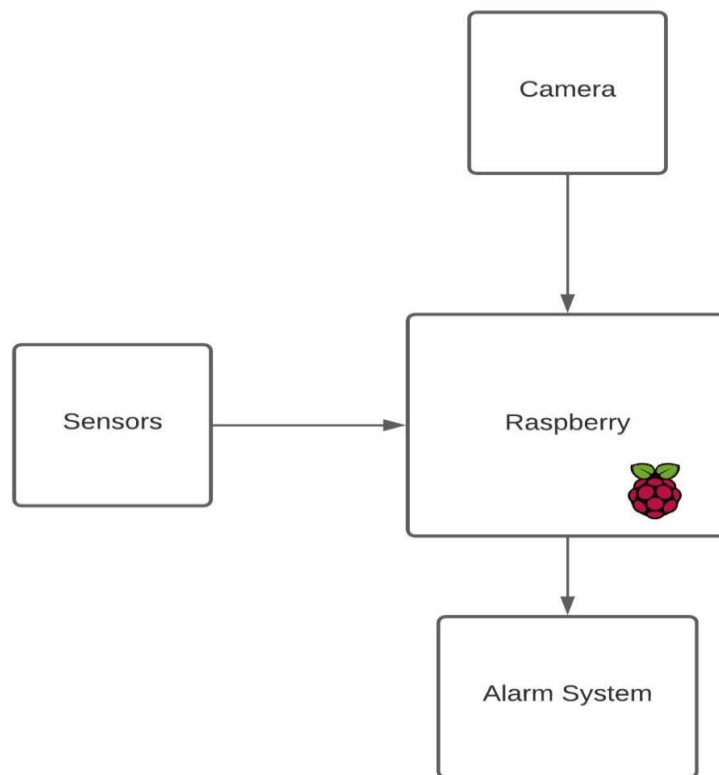


Figure 3.1: System Architecture

3.1 Raspberry Pi

We will be using the Raspberry Pi 3 Model B as our central processing unit. The microcomputer with its feature of 40 GPIO pins will help us integrate multiple sensors and actuators, which would serve the purpose. The compactness of the microcomputer is an added advantage to minimize occupancy issues.

3.2 Camera

The camera would act as the primary sensor which would solely focus on the live recording inside the vehicle and is responsible for the frame acquisition task which plays a major role in the proposed methodology.

3.3 Alarm System

The alarm system would consist of the audio signaling device which would be a buzzer. This component would be used to alert the driver in necessary situations. In most vehicles there is a pre-installed speaker already present which can also be used to send out alerts.

3.4 Sensors

A sensor is a device that detects the change in the environment and responds to some output on the other system. A sensor converts a physical phenomenon into a measurable analog voltage (or sometimes a digital signal) converted into a human-readable display or transmitted for reading or further processing.

The sensors used for this project are for measuring the values of the conditions that can affect the driver of the car or the physical condition of the car. These are used to prevent any accidents or incidents from occurring, by which it can save many lives.

The sensors are additional add-ons which will enable the system to evaluate and determine the external conditions which may affect the driver.

Chapter 4

Proposed Method

4.1 Technical Terms and Features

4.1.1 Machine Vision

Machine vision is the ability of a computer to see; it employs one or more video cameras, analog-to-digital conversion (ADC) and digital signal processing (DSP). The resulting data goes to a computer or robot controller. Machine vision is similar in complexity to voice recognition.

4.1.2 Facial Expressions

A facial expression is one or more motions or positions of the muscles beneath the skin of the face. Facial expressions are a form of nonverbal communication. They are a primary means of conveying social information between humans, but they also occur in most other mammals and some other animal species.

4.1.3 Algorithm

In the most general sense, an algorithm is a series of instructions telling a computer how to transform a set of facts about the world into useful information. The facts are data, and the useful information is knowledge for people, instructions for machines or input for yet another algorithm.

4.1.4 Frame Acquisition

A protocol for creating images or image-derived measurements. In this way, data is captured
Acquisition Duration: The amount of time elapsed during the capture of the data.

4.1.4 Threshold

The magnitude or intensity that must be exceeded for a certain reaction, phenomenon, result, or condition to occur or be manifested. A value above that threshold indicates a condition; a value below indicates the other condition. For example, below a threshold the eye is said to be closed. Above it, the eye is open.

4.1.5 Coordinates

Coordinates are distances or angles, represented by numbers, that uniquely identify points on surfaces of two dimensions (2D) or in space of three dimensions (3D). There are several coordinate schemes commonly used by mathematicians, scientists, and engineers.

4.1.6 Facial-Landmarking

Facial landmarking, defined as the detection and localization of certain key points on the face, plays arguably the important role as an intermediary step for many subsequent face processing operations that ranges from biometric recognition to the understanding of mental states. Though its conceptual simplicity, the computer vision problem has proven extremely challenging due to multitude of compound factors such as pose, expression, occlusion and illumination.

4.1.7 Haar Cascades Classifiers

Haar Cascade classifiers are an effective way for object detection. This method was proposed by Paul Viola and Michael Jones in their paper Rapid Object Detection using a Boosted Cascade of Simple Features .Haar Cascade is a machine learning-based approach where a lot of positive and negative images are used to train the classifier.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features, i.e., edge, line, and four-rectangle features are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting the sum of pixels under the white rectangle from the sum of pixels under the black rectangle.

In an image, most of the image region is non-face region. So it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot.

For this the concept of Cascade of Classifiers was introduced. Instead of applying all the features on a window, group the features into different stages of classifiers and apply one-by-one. If a window fails the first stage, discard it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region.

4.2 Algorithm

4.2.1 Flowchart

The algorithm uses simple machine vision techniques to monitor and capture the driver's facial expressions which would help it to analyze the live feed and determine if the driver is drowsy or not. The below diagram represents the flowchart of the proposed algorithm.

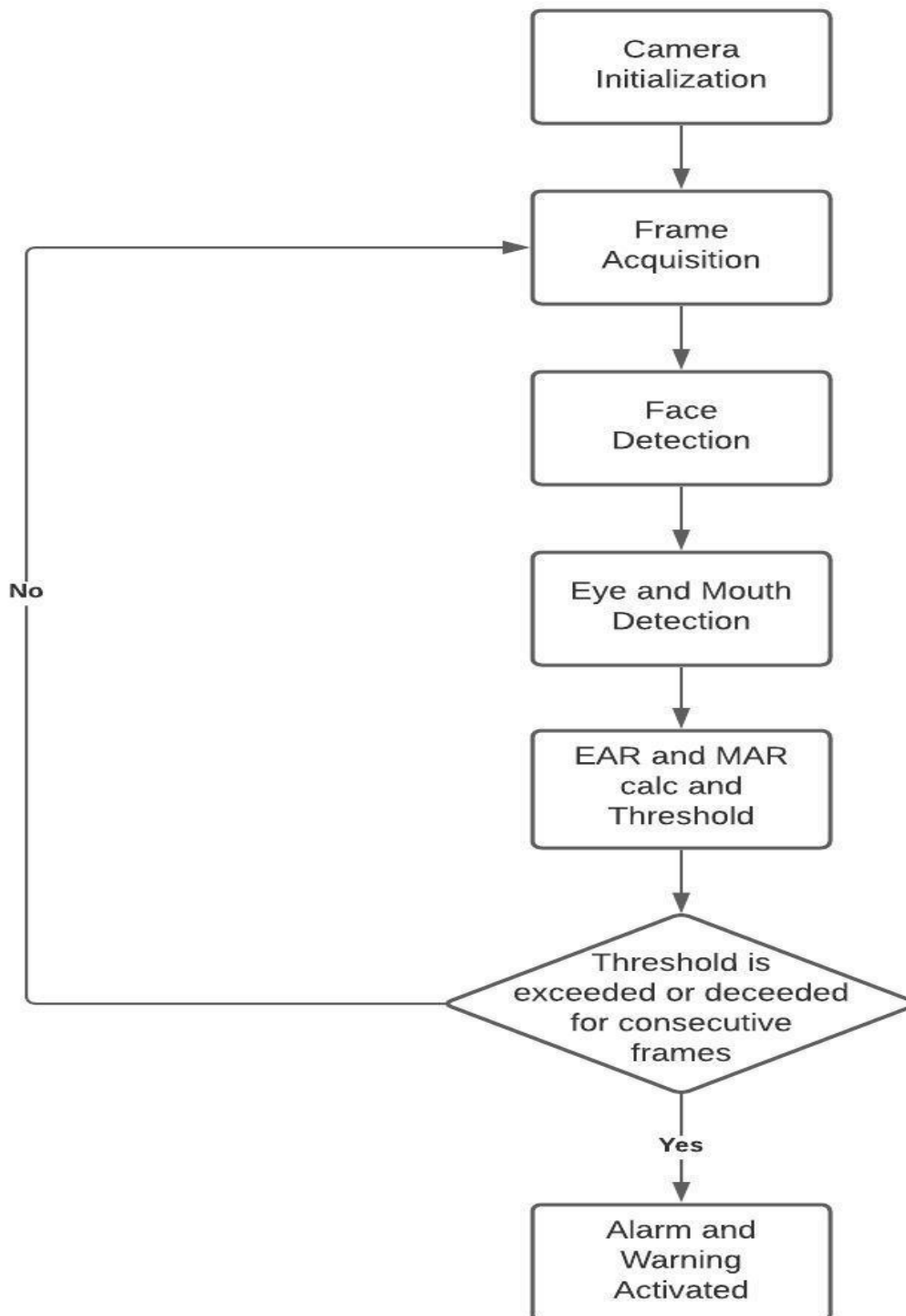


Figure 4.1: Algorithm Flowchart

1. As the device is turned on, the camera has to be initialized first in order get it ready to capture the live feed to process.
2. The second step would be acquiring each frame of the live feed in order to analyse them individually, as the processing takes place in real time.
3. Each frame acquired is assessed and Haar cascades are used for the facial detection process.
4. The 68 – predefined facial landmarks and facial features retrieved from the probe file will help in the detection of the eyes and the mouth.
5. The Eye Aspect Ratio and the Mouth Aspect Ratio is calculated using the given formulae and then compared with the respective thresholds set.
6. The next step is checking if the ideal conditions are not satisfied. If yes, then the alarm or alert system is triggered, if not then the whole process repeats itself and checks for the conditions repetitively.
7. The alarm or alert system when triggered sends out the alert so the individual or the driver notices the abnormality and then switches off automatically when the situation comes back to the ideal state.

4.2.2 Description

1. Pre-existing features for facial landmark detection are implemented to identify the state of drowsiness and fatigue.
2. 68 – facial predefined landmarks help in shape prediction to clearly identify the various regions of the face.
3. The camera is embedded to monitor and capture to extract frames one by one and generate alerts accordingly. Each frame will be analyzed to study the patterns of facial features, using Haar Cascades Classifiers and determine the Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR) for each frame.
4. The program uses a facial training data set to understand where certain points exist on facial structures. The program then plots the same points on the region of interests in other images, if they exist. The Probe document(.dat file) used is pre-trained using a data set containing numerous facial images and the extracted features will be used for the detection of the eye and mouth.
5. The program would use priors to estimate the probable distance between the required key points.
6. For eye blinks we need to pay attention to points 37-46, the points that describe the eyes.
The below figure shows the visualization of the 68 facial landmark coordinates.
7. For yawns we need to pay attention to the points 49 – 68, the points that describe the mouth.
8. EAR (Eye Aspect Ratio) is used to detect a blink which the ratio of vertical distance of the lower and upper eyelids to the horizontal length of an eye. During the eye blinks, the vertical distance



Figure 4.2: Visualization of the 68 facial landmark coordinates

of lower and upper eyelids decreases similarly in opened eye after a blink distance between the eyelids tend to increase. Therefore, the EAR decreases and increases simultaneously, the blink count is incremented. The EAR count less than the threshold value reports suspicion in driver's behaviour.

- A program can determine if a person's eyes are closed if the EAR falls below certain threshold, using (1).
- Frame differencing is another blink detection technique. Essentially the program compares subsequent video frames to determine if there was any movement in the selected eye region.

$$EAR = \frac{||p_2 - p_6|| + ||p_3 - p_5||}{2 * ||p_1 - p_4||} \dots\dots\dots(1)$$



Figure 4.3: Coordinates to calculate EAR

9. The MAR (Mouth Aspect Ratio) is represented by 8- coordinates in landmarks predictor function and the yawn count is incremented as soon as the MAR value increases and exceeds the threshold value.
- The MAR is a constant value when the mouth is closed and gradually rises to 1 when the mouth is opened
 - A program can determine if the person is yawning if the MAR rises above a certain threshold, using (2).
 - Frame differencing is again used to determine the state of yawning.

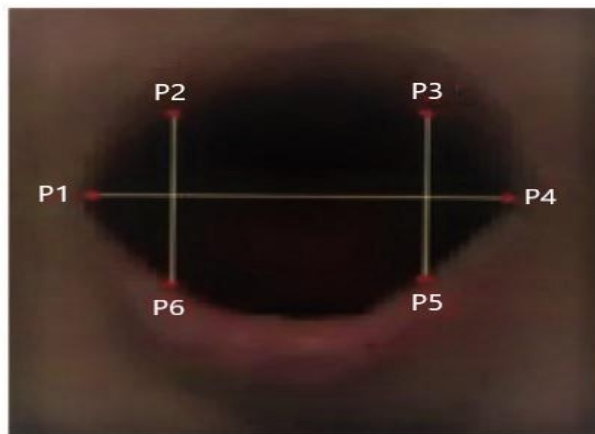


Figure 4.4: Coordinates to calculate MAR

$$\text{MAR} = \frac{|p_2 - p_6| + |p_3 - p_5|}{2 * |p_1 - p_4|} \dots\dots(2)$$

10. Blink rate and yawn counts exceed specified threshold values respectively for a certain number of consecutive frames, the system will assume that the driver is dozing off and will activate the alarm to alert the driver until the driver wakes up. This technique is called frame differencing.

There have been multiple constants and features set up the program. These constants and features in some cases can also be considered the thresholds set up in order trigger the alert system.

The important thresholds which were set are:

- Eye Aspect Ratio Threshold

In the program the EYE_ASPECT_RATIO_THRESHOLD is set to 0.3. This is supposed to be the minimum threshold meaning when the Eye Aspect Ratio calculated falls

below the specified threshold for a said number of consecutive frames then the alert system is supposed to be activated.

- Eye Aspect Ratio Consecutive Frames

In the program the EYE_ASPECT_RATIO_CONSEC_FRAMES is set to 40. It means that if the Eye Aspect Ratio falls below the set threshold for 40 consecutive frames, then the alert system is triggered.

- Mouth Aspect Ratio Threshold

In the program the MOUTH_ASPECT_RATIO_THRESHOLD is set to 0.7. This is supposed to be the maximum threshold meaning when the Mouth Aspect Ratio calculated rises above the specified threshold for a said number of consecutive frames then the alert is supposed to be activated.

- Mouth Aspect Ratio Consecutive Frames

In the program the MOUTH_ASPECT_RATIO_CONSEC_FRAMES is set to 50. It means that if the Mouth Aspect Ratio rises above the set threshold for 50 consecutive frames, then the alert system is triggered.

- The camera is given some waiting time to get initialized.
- The Euclidean function is used to calculate the distance between the facial landmarks indicated for use in the calculation of Eye Aspect Ratio and Mouth Aspect Ratio.
- The face detection is carried out by the frontal face detector from the dlib library.
- The shape predictor function from the Dlib library is used to read the probe file. The probe file would help in recognizing the 68 – facial landmarks.
- Once the 68-facial landmarks are recognised, the frame is further processed.
- We finally use the put Text function from the cv2 library to display the state of drowsiness or yawning after detection on the output screen window.

4.2.3 Output Screen Displays

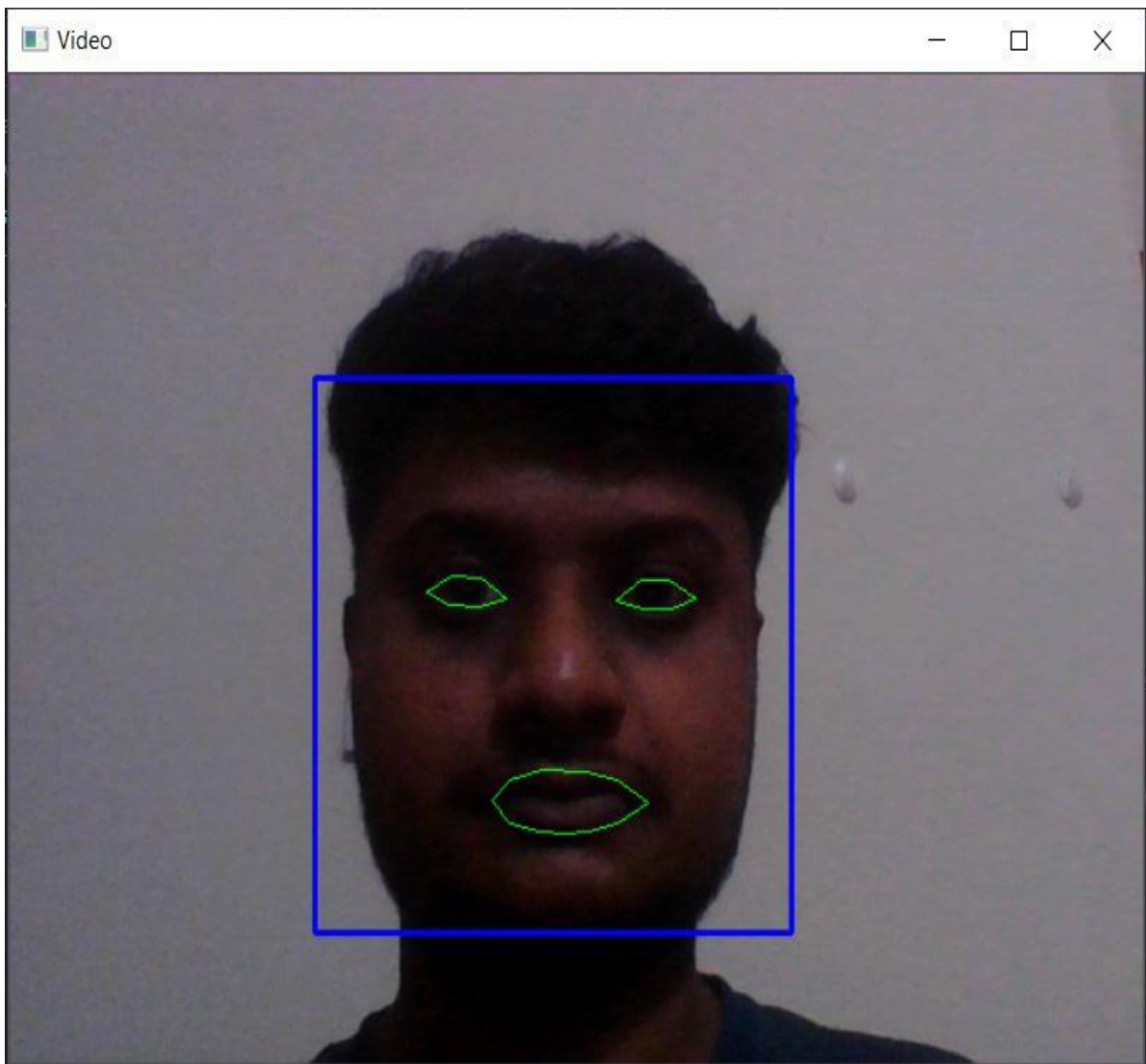


Figure 4.5: Ideal Screen Output

The figure 4.5 shows the ideal screen output where, the eyes and the mouth are in their ideal state. There is no detection of any drowsiness and yawning. The Eye Aspect Ratio (EAR) is above the minimum threshold, and the Mouth Aspect Ratio (MAR) is below the maximum threshold. The eye blinks would be ignored and considered redundant if the individual does not have the EAR below the minimum threshold for more than 50 consecutive frames.

The figure 4.6 shows the detection of the drowsiness state of the driver. Here it is clear that the individual has the Eye Aspect Ratio below the minimum threshold for more than 50 consecutive frames and due to this condition, the program triggers the alarm system and the device would alert the driver so the individual becomes aware of the condition and then take the necessary action to further improve the driving experience.

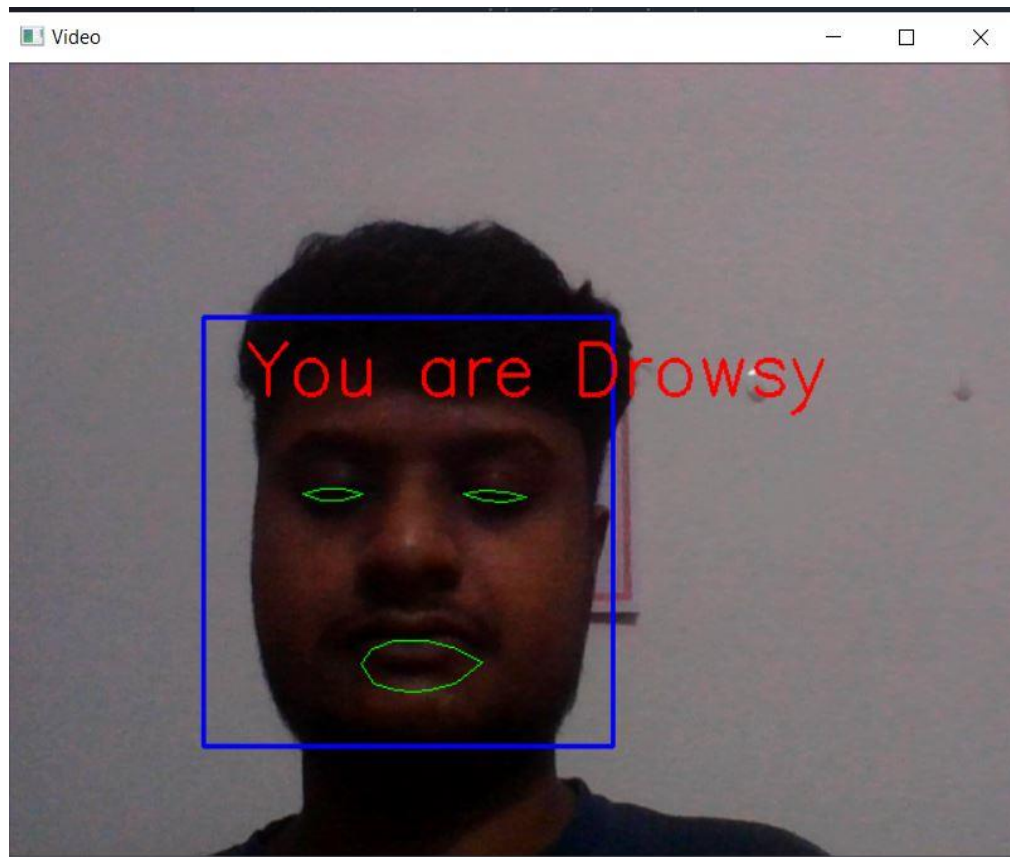


Figure 4.6: Drowsiness Detection

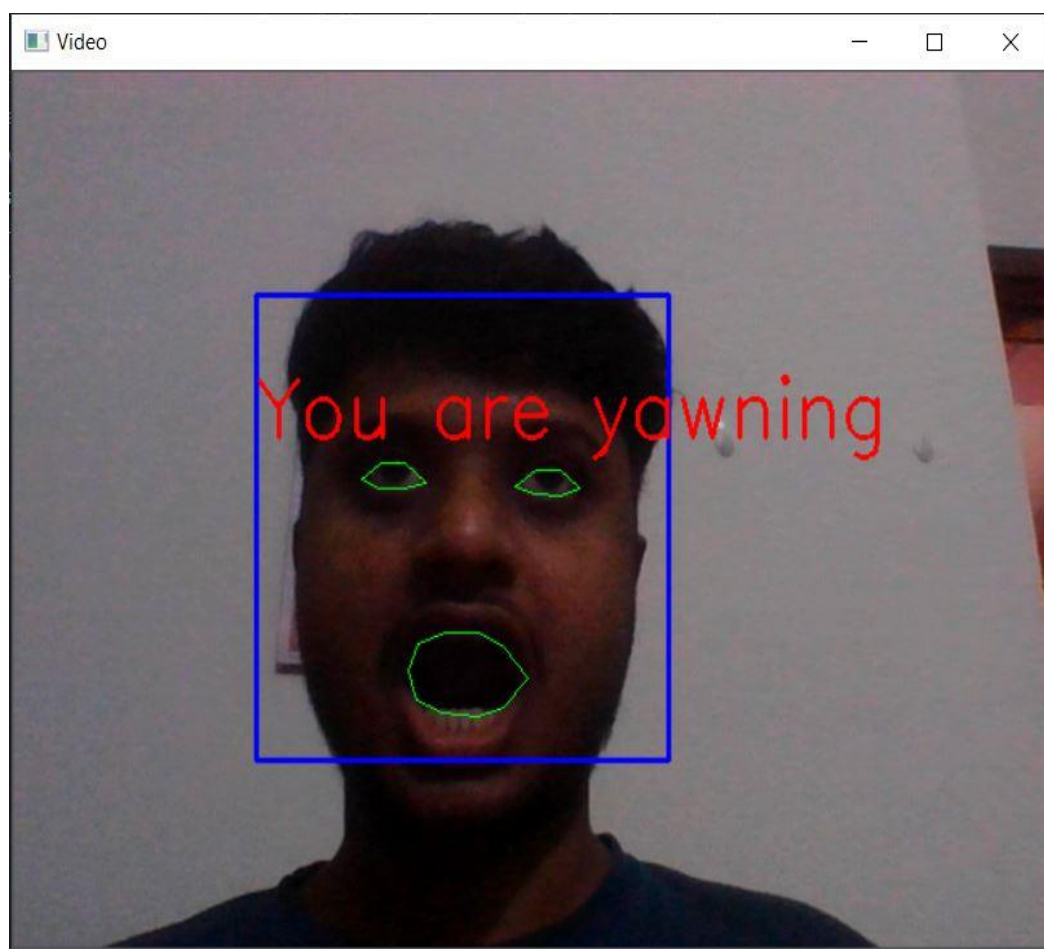


Figure 4.7: Yawning Detection

The figure 4.7 depicts the detection of the yawning state of driver. Here it clearly indicates the yawning state because the Mouth Aspect Ratio is more than the maximum threshold which is set to be 0.6 for 60 consecutive frames and due to this condition, the program would trigger the alarm system again and the device would send the alert in order to make the driver aware of it, so the individual could take the necessary steps to improve the driving experience.

4.3 Sensors

A sensor is a device that detects the change in the environment and responds to some output on the other system. A sensor converts a physical phenomenon into a measurable analog voltage (or sometimes a digital signal) converted into a human-readable display or transmitted for reading or further processing.

One of the best-known sensors is the microphone, which converts sound energy to an electrical signal that can be amplified, transmitted, recorded, and reproduced.

Sensors are used in everyday objects such as touch-sensitive elevator buttons (tactile sensor) and lamps which dim or brighten by touching the base, besides innumerable applications of which most people are never aware. With advances in micromachinery and easy-to-use microcontroller platforms, the uses of sensors have expanded beyond the traditional fields of temperature, pressure or flow measurement,

Sensors are used in our everyday lives. For example, the common mercury thermometer is a very old type of sensor used for measuring temperature. Using colored mercury in a closed tube, it relies on the fact that this chemical has a consistent and linear reaction to changes in temperature.

The Sensors used in this project are: -

1. Smoke and Gas sensor
2. Temperature sensors
3. Ultrasonic sensor

4.3.1 General flowchart for sensor working

The sensor works in the following methodology.

The sensors are equipped and programmed using a micro-processor, they are programmed in a unique way to respond to certain conditions. The sensors are set at a certain specified threshold, the

violation of which can lead to some uncertain conditions. To prevent these conditions an alert is given to the user.

The methodology first step includes, the checking if the driver is getting drowsy, this is done by the algorithms used in this project. The drowsiness of the driver may be a cause for an accident which is a dangerous scenario endangering human lives.

The second step of the methodology, is the action which is taken by the sensors. These actions are the activation of the sensors. The sensors are being monitored by the micro-processor, in this case an Arduino Uno. The program coded into the micro-processor will activate all the sensors and continuously start measuring the values of all the sensors and record them. This will check on the results of all the sensors and evaluate if any set threshold is being violated.

The third step of the methodology, this is the step where it requires human interaction, specifically the driver. The driver will receive an alert if any of the conditions is met. The alert will inform if the driver is drowsy and it will prompt the driver to get alert if drowsy. If any of the sensors are triggered the same alert is informed to the driver and brought to attention. The driver will be equipped with an override switch to counter the alert if a false positive is triggered due to uncontrolled natural conditions. By this the issue is resolved and the driver can now drive more alert with caution in mind, hence a safe drive.

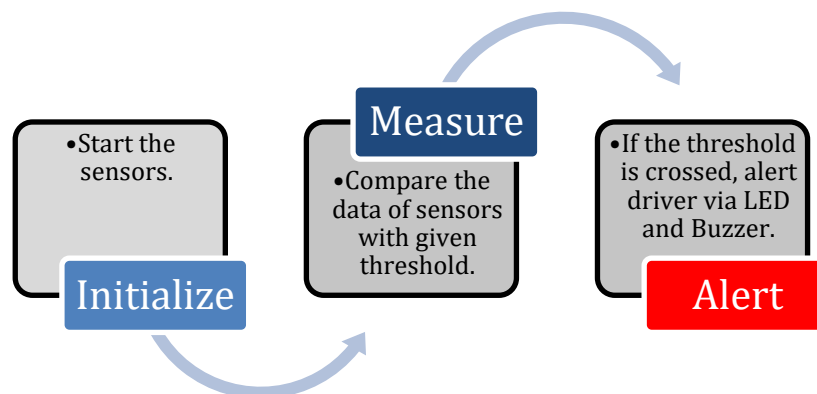


Figure 4.8: Flowchart for sensor functioning

4.3.2 Smoke and Gas Sensor

1. We use the MQ9 gas sensor here.
2. MQ-9 gas sensor module has high sensitivity to Carbon Monoxide, Methane and LPG.
3. Sensitive material of MQ-9 gas sensor is SnO₂, which with lower conductivity in clean air.
4. It makes detection by cycling high and low temperature, and detects CO when low temperature (heated by 1.5V).

5. The sensor's conductivity is higher along with the gas concentration rising.
6. When temperature rises (heated by 5.0V), it detects Methane, Propane, combustible gas, etc and cleans the other gases as it adsorbed under low temperature. MQ-9 usually applied in domestic gas leakage detector, industrial gas detector, portable gas detector, etc.
7. This will detect Carbon Monoxide, a highly harmful and deadly gas and alert its concentrations ranging from 10 to 10,000 ppm.
8. This sensor will also detect smoke, because the sensors conductivity is higher with more concentration of gas and smoke.
9. MQ-9 gas sensor has high sensitivity to Carbon Monoxide, Methane and LPG.

4.3.2.1 Sensitivity Adjustment: -

Resistance value of MQ-9 is different to various kinds and various concentration of gases. So, when using this component, sensitivity adjustment is very necessary. It is recommended that we calibrate the detector for 200ppm and 5000ppm CH₄ or 1000ppm LPG concentration in air and use value of Load

resistance that (RL) about 20 K Ω (10K Ω to 47 K Ω).

When accurately measuring, the proper alarm point for the gas detector should be determined after considering the temperature and humidity influence.

The sensitivity adjusting program:

- a. Connect the sensor to the application circuit.
- b. Turn on the power, keep time of preheating through electricity is over 48 hours.
- c. Adjust the load resistance RL until you get a signal value which is respond to a certain carbon monoxide concentration at the end point of 90 seconds.
- d. Adjust another load resistance RL until you get a signal value which is respond to a CH₄ or LPG concentration at the end point of 60 seconds.

Model no.	MQ-9
Sensor type	Semiconductor
Standard Encapsulation	Bakelite
Detection Gas	Carbon Monoxide (CO), Alcohol, & Methane (CH₄)

Table 4.1: MQ-9 specifications



Figure 4.9: MQ-9 Sensor

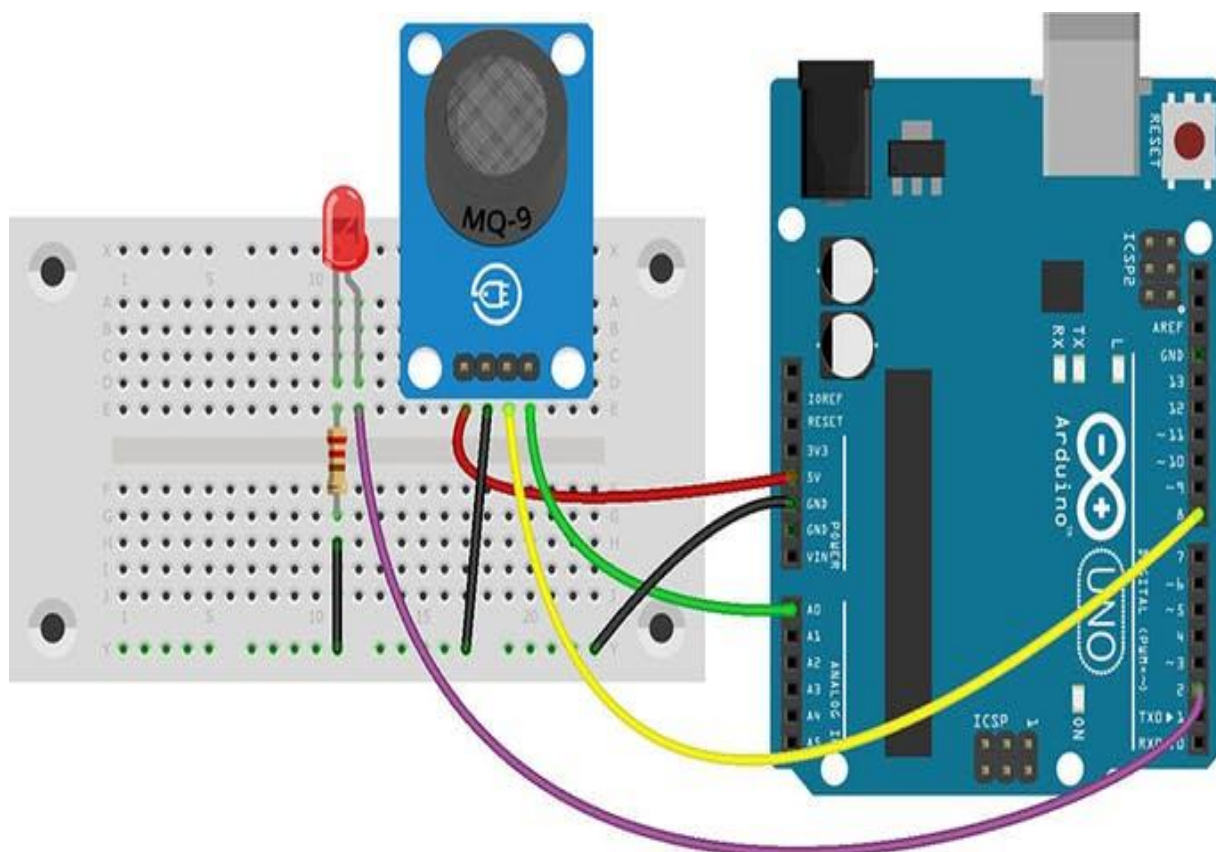


Figure 4.10: MQ-9 Sensor Connection with Arduino UNO

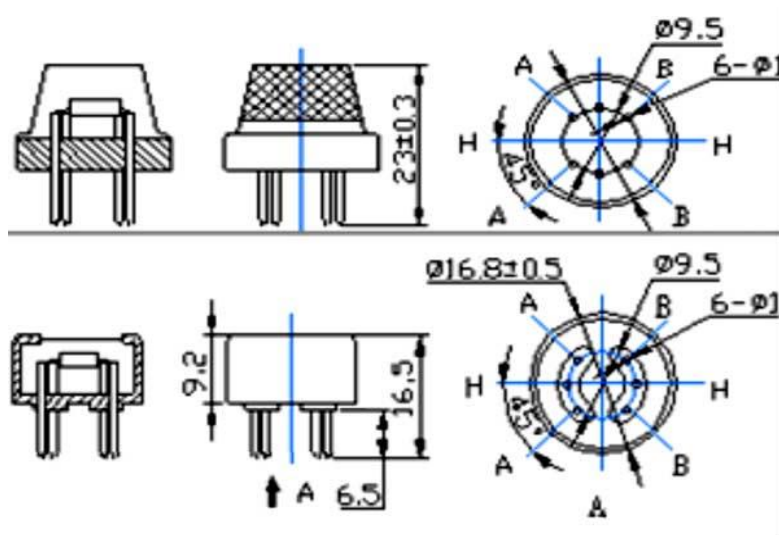


Figure 4.11: MQ-9 Sensor dimensions



Figure 4.12: MQ-9 Sensor sensitivity adjustment

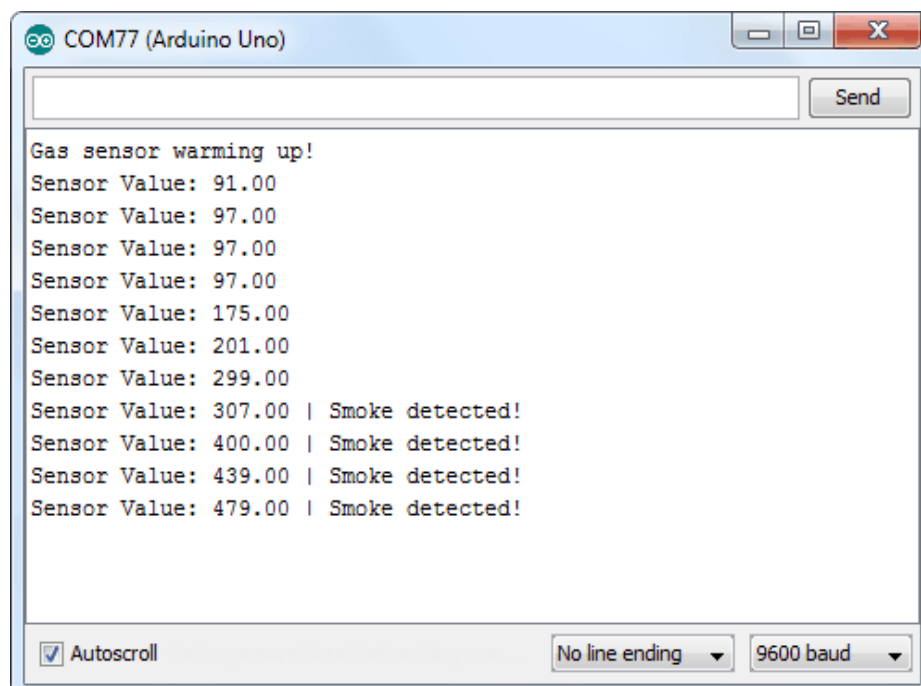


Figure 4.13: MQ-9 Sensor Serial Output

4.3.3 Temperature Sensor

1. DHT11 sensor consists of a capacitive humidity sensing element and a thermistor for sensing temperature.
2. The humidity sensing capacitor has two electrodes with a moisture holding substrate as a dielectric between them.
3. Change in the capacitance value occurs with the change in humidity levels.
4. The IC measure, process this changed resistance values and change them into digital form.
5. It is an electronic device that measures the temperatures of its environment.
6. It converts the input data into electronic data to record and monitor.
7. It can even signal the temperature changes.
8. This can be used in automobiles, where it can detect very high temperatures and alert the driver, it can also detect the battery temperatures (In Electric Vehicles), It can also Alert when the temperature is exceeding the basic regulations.

The applications of the DHT11 Sensor are:

- Measurement of temperature and humidity.
- In local weather stations.
- In Automatic Climate Control systems.
- In Environmental Monitoring systems.

The following points emphasize the technical features and specifications of the DHT11 sensor:

- The temperature range of DHT11 is from 0 to 50 degree Celsius with a 2-degree accuracy. Humidity range of this sensor is from 20 to 80% with 5% accuracy. The sampling rate of this sensor is 1Hz.i.e., it gives one reading for every second. DHT11 is small in size with operating voltage from 3 to 5 volts. The maximum current used while measuring is 2.5mA.
- The DHT11 sensors usually require external pull-up resistor of 10K Ω between VCC and Out pin for proper communication between sensor and the Arduino. However, the module has a built-in pull-up resistor, so you need not add it.
- The module also has a decoupling capacitor for filtering noise on the power supply.
- It has 3 pins: -
 - + (VCC) pin supplies power for the sensor. 5V supply is recommended, although the supply voltage ranges from 3.3V to 5.5V. However, with 3.3V supply voltage, cable length shall not be greater than 1 meter. Otherwise, the line voltage drop will lead to errors in measurement.
 - Out pin is used to communication between the sensor and the Arduino.
 - – (GND) should be connected to the ground of Arduino.

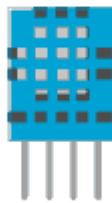
	
DHT11	
Operating Voltage	3 to 5V
Max Operating Current	2.5mA max
Humidity Range	20-80% / 5%
Temperature Range	0-50°C / $\pm 2^\circ\text{C}$
Sampling Rate	1 Hz (reading every second)
Body size	15.5mm x 12mm x 5.5mm
Advantage	Ultra low cost

Figure 4.14: DTH-11 Specifications

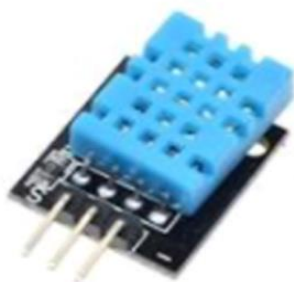


Figure 4.15: DHT-11 Temperature Sensor

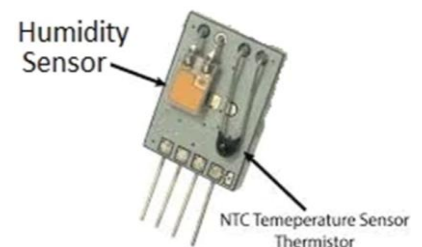
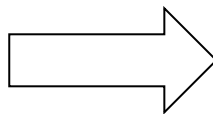
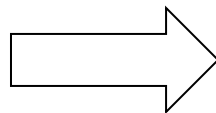


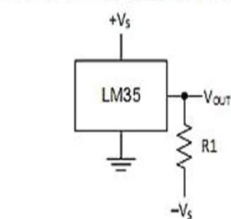
Figure 4.16: DHT-11 Components



Figure 4.17: LM-35 Temperature Sensor



Full-Range Centigrade Temperature Sensor



Choose $R_1 = -V_S / 50 \mu\text{A}$
 $V_{OUT} = 1500 \text{ mV at } 150^\circ\text{C}$
 $V_{OUT} = 250 \text{ mV at } 25^\circ\text{C}$
 $V_{OUT} = -650 \text{ mV at } -55^\circ\text{C}$

Figure 4.18: LM-35 Components

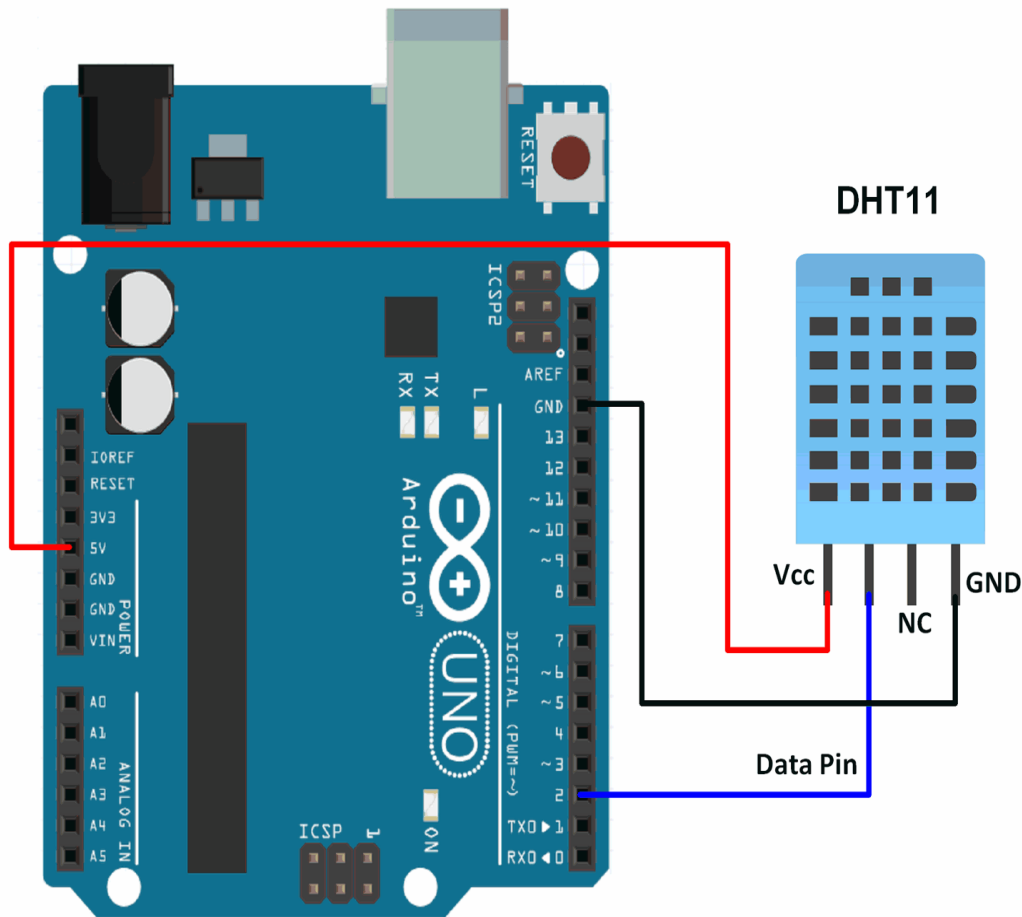


Figure 4.19: DTH-11 Connection with Arduino UNO

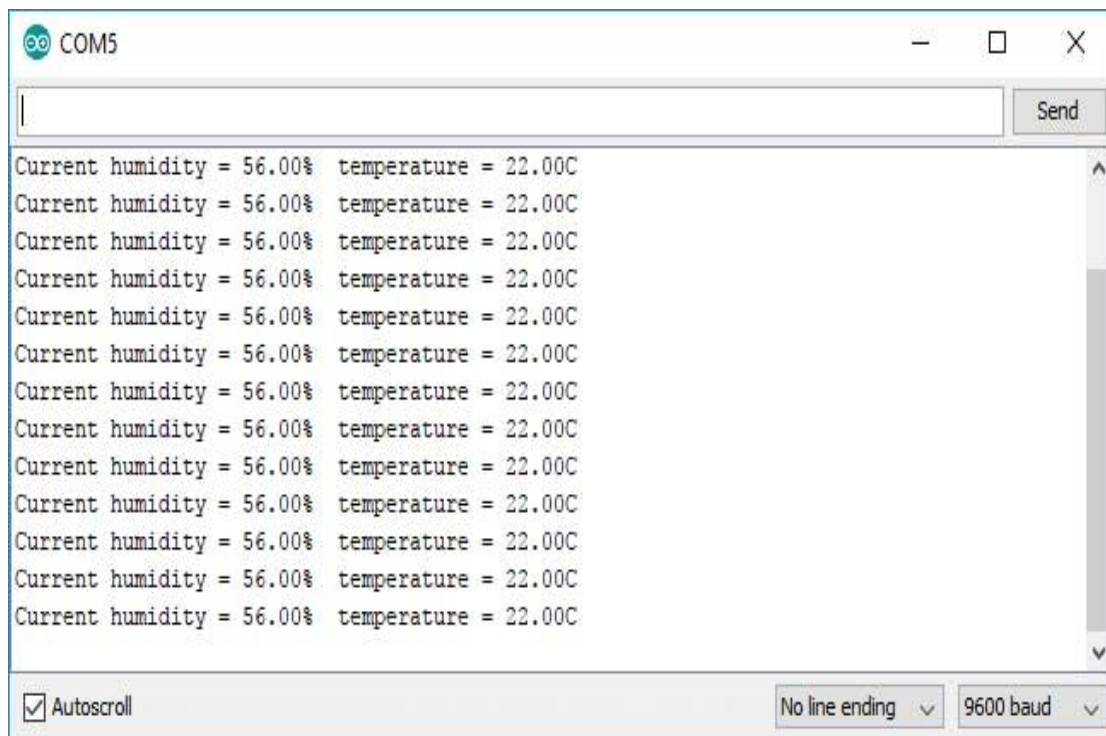


Figure 4.20: DTH-11 Arduino Output

4.3.4 Ultrasonic Sensor (Collision avoidance)

1. We can use two ultrasonic sensors here: HC-SR04 and US-100.
2. At its core, the HC-SR04 Ultrasonic distance sensor consists of two ultrasonic transducers.
3. The one acts as a transmitter which converts electrical signal into 40 KHz ultrasonic sound pulses.
4. The receiver listens for the transmitted pulses.
5. If it receives them, it produces an output pulse whose width can be used to determine the distance the pulse travelled.
6. The sensor is small, easy to use in any robotics project and offers excellent non-contact range detection between 2 cm to 400 cm (that's about an inch to 13 feet) with an accuracy of 3mm.
7. Since it operates on 5 volts, it can be hooked directly to an Arduino or any other 5V logic microcontrollers.
8. Both the sensors combined will alert if there is safe distance between vehicles.
9. As soon as the distance is unsafe, it will indicate the driver to slow down the vehicle and hence preventing an accident.
10. It all starts, when a pulse of at least 10 μ S (10 microseconds) in duration is applied to the Trigger pin
11. . In response to that the sensor transmits a sonic burst of eight pulses at 40 KHz.
12. This 8-pulse pattern makes the “ultrasonic signature” from the device unique, allowing the receiver to differentiate the transmitted pattern from the ambient ultrasonic noise.

The following points emphasize the technical features and specifications of the HC-SR04 sensor:

- Theoretical Measuring Distance is 2cm to 450cm
- Practical Measuring Distance is 2cm to 80cm
- Accuracy is 3mm
- Measuring angle covered is $<15^\circ$
- Operating current is <15 mA
- Operating Frequency is 40Hz
- It has 4 Pins: -
 - VCC is the power supply for HC-SR04 Ultrasonic distance sensor which we connect the 5V pin on the Arduino.
 - Trig (Trigger) pin is used to trigger the ultrasonic sound pulses.
 - Echo pin produces a pulse when the reflected signal is received. The length of the pulse is proportional to the time it took for the transmitted signal to be detected.
 - GND should be connected to the ground of Arduino.

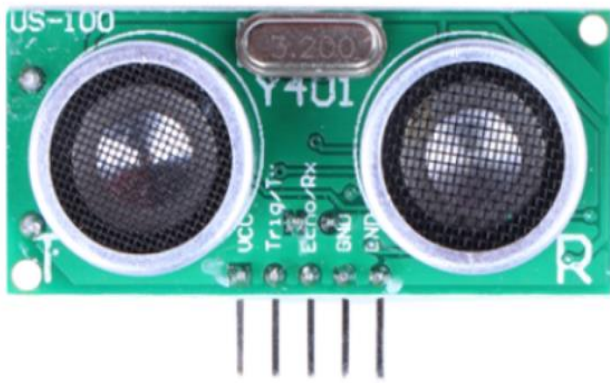


Figure 4.21: Ultra Sonic Sensor

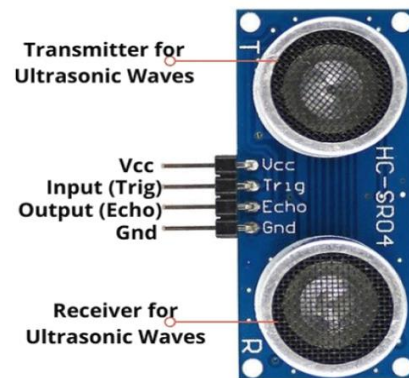


Figure 4.22: Pin Configurations of Ultra-Sonic Sensor

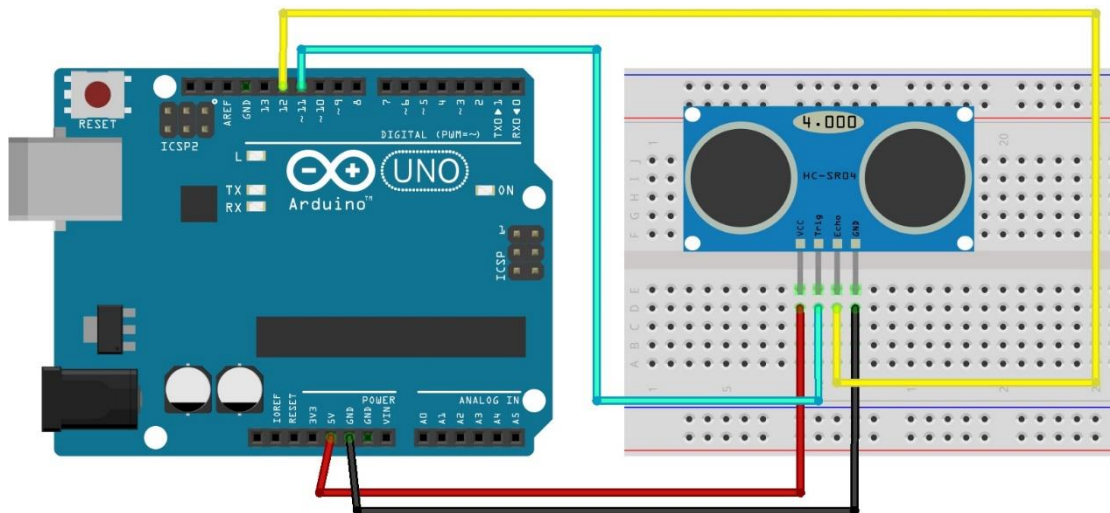


Figure 4.23: HC SR04 Connection with Arduino UNO

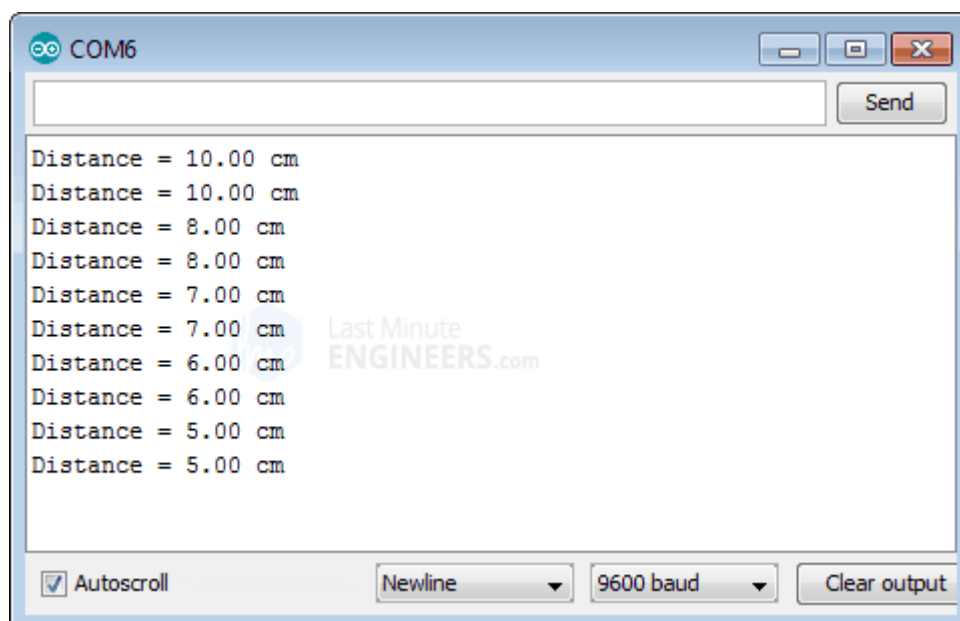


Figure 4.24: HC SR04 Output

4.3.4.1 Ultra-sonic Sensor Functioning

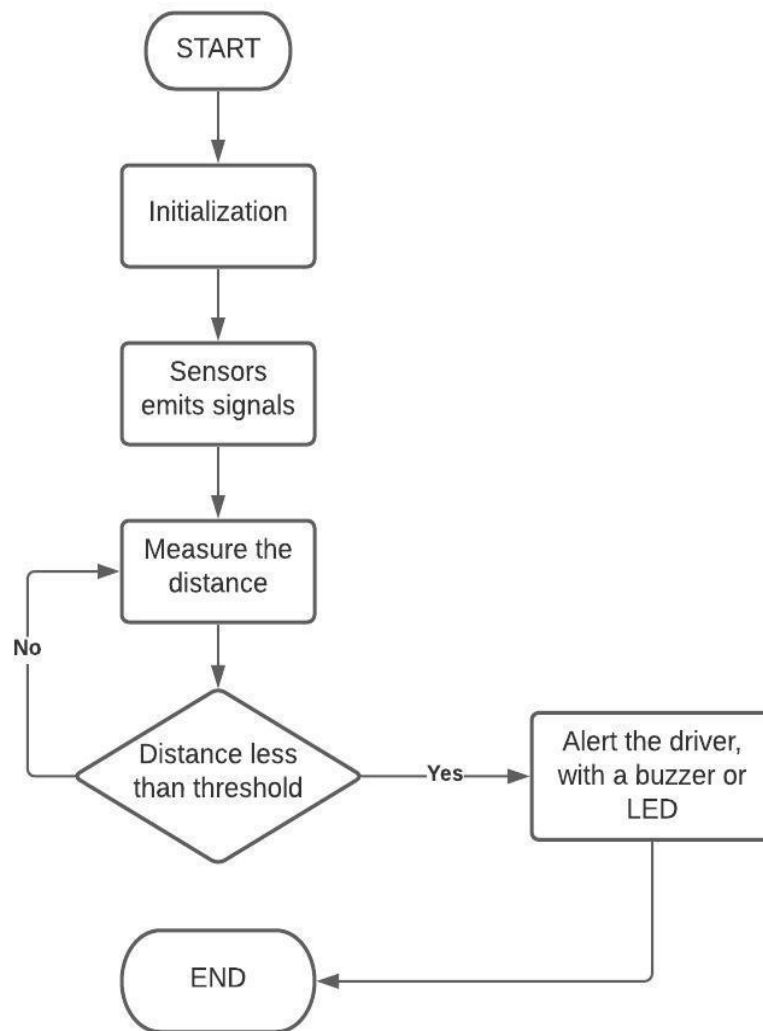


Figure 4.25: Flowchart of Ultrasonic Sensor functioning

- The above flowchart shows the functioning of the Ultrasonic sensor.
- The sensor will initialize first and start emitting ultrasonic signals.
- These Ultrasonic signals will repel if it encounters any obstacles in its way.
- The sensor has two antennas, one for emitting the ultrasonic waves and the other for receiving them.
- Depending on the transmit time and the receive time it will deduce the distance between the sensor and the obstacle.
- There is a threshold set for this sensor, if any obstacle is less than the set value, it gives a high voltage in its output pin.
- This here is connected to the Buzzer and the LED.
- This is how the Ultrasonic sensor functions and alerts if the obstacles are too nearby.

4.3.4.2 Sensor Activity in case of accident

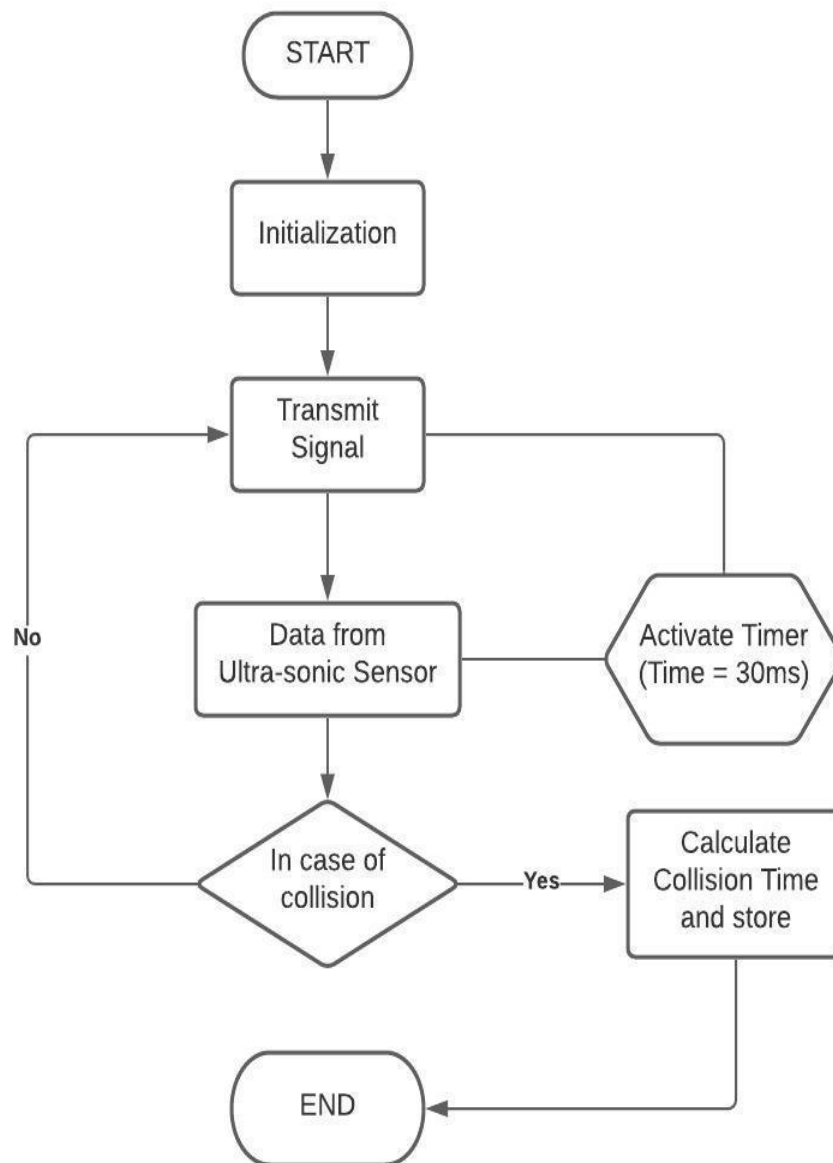


Figure 4.26: Sensor activity in case of accident

- This flowchart represents the Sensor activity in case of an accident.
- The condition of an accident is taken from the readings of the ultrasonic sensor, where if the sensor crosses a reading below the threshold, it is considered to be a possible collision.
- The sensor emits signals with an active timer with a time loop of 30ms.
- If the sensors predict a collision the time is stored.
- If there is no collision predicted, it loops back to transmitting the signal.
- This continues until the sensors are kept ON, they will be continuously recording the value.
- This is how the sensor is used here to predict if there is a collision or an accident.

Chapter 5

Hardware and Software Requirements

5.1 Hardware Requirements

5.1.1 Raspberry Pi 3 Model B

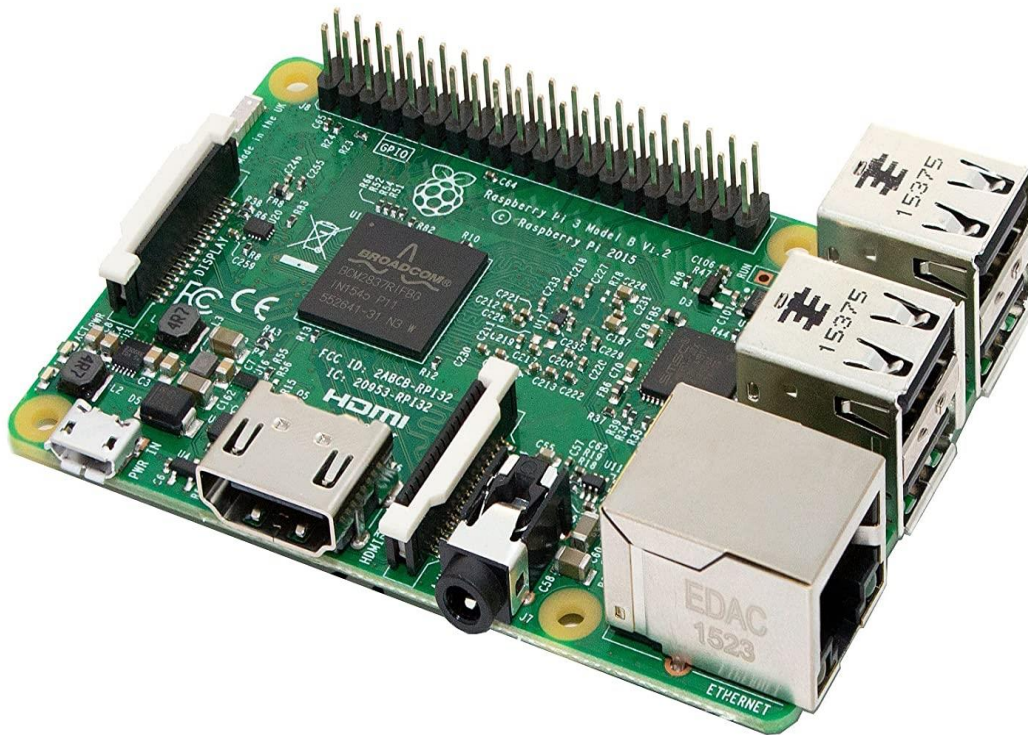


Figure 5.1: Raspberry Pi 3 Model B

The Raspberry Pi 3 Model B is the earliest model of the third-generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B. This powerful credit-card sized single board computer can be used for many applications and supersedes the original Raspberry Pi model B+ and Raspberry Pi 2 model b. Whilst maintaining the popular board format the Raspberry Pi 3 model b brings a more powerful processor, 10x faster than the first-generation Raspberry Pi. Additionally, it adds wireless LAN and Bluetooth connectivity making it an ideal solution for powerful connected designs.

5.1.2 Camera

A camera is an optical instrument used to capture an image. The video camera is a camera used for electronic motion picture acquisition

5.1.3 Buzzer/Alarm Device

A buzzer is an audio signaling device, which may be mechanical, electromechanical or piezoelectric in nature and operates on user input.

5.1.4 Sensors

1. Smoke and Gas sensor
2. Temperature sensors
3. Ultrasonic sensor

The sensors are explained in depth in Chapter 4 under section 4.3, where the roles and the functioning of each of the sensor is briefly depicted.

5.2 Software Requirements

5.2.1 Raspberry Pi OS



Figure 5.2: Raspberry Pi OS Logo

Raspberry Pi OS (formerly **Raspbian**) is a Debian-based operating system for Raspberry Pi. It promotes Python and Scratch as the main programming languages, with support for many other languages. The default firmware is closed source, while unofficial open source is available.

5.2.2 Python Programming Language

Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

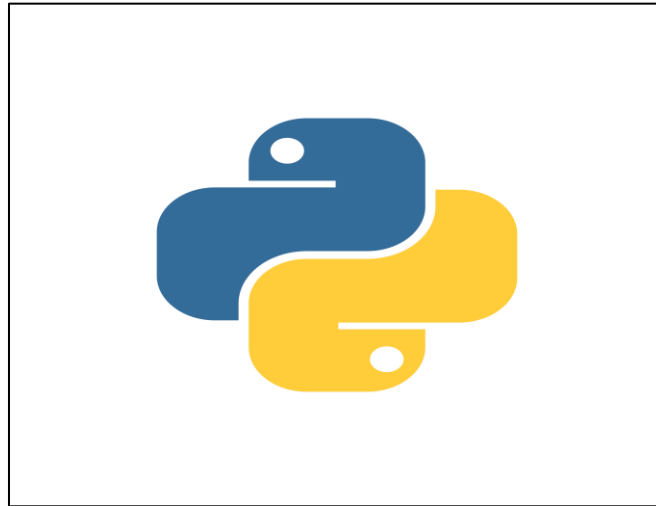


Figure 5.3: Python Programming Language Logo

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a “batteries included” language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting. Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible and much Python 2 code does not run unmodified on Python 3. Python 2 was discontinued with version 2.7.18 in 2020.

The main features of Python Programming Language are:

- Easy to code.
- Free and open-source.
- Object-oriented language.
- GUI Programming Support.
- High-level Language.
- Extensible feature.
- Python is a portable language.
- Python is an integrated language.
- Dynamically typed language.
- Interpreted Language.

5.2.3 VS Code

Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality. Microsoft has released most of Visual Studio Code's source code on the microsoft/vscode repository of GitHub using the "Code – OSS" name, under the permissive MIT License, while the releases by Microsoft are proprietary freeware.

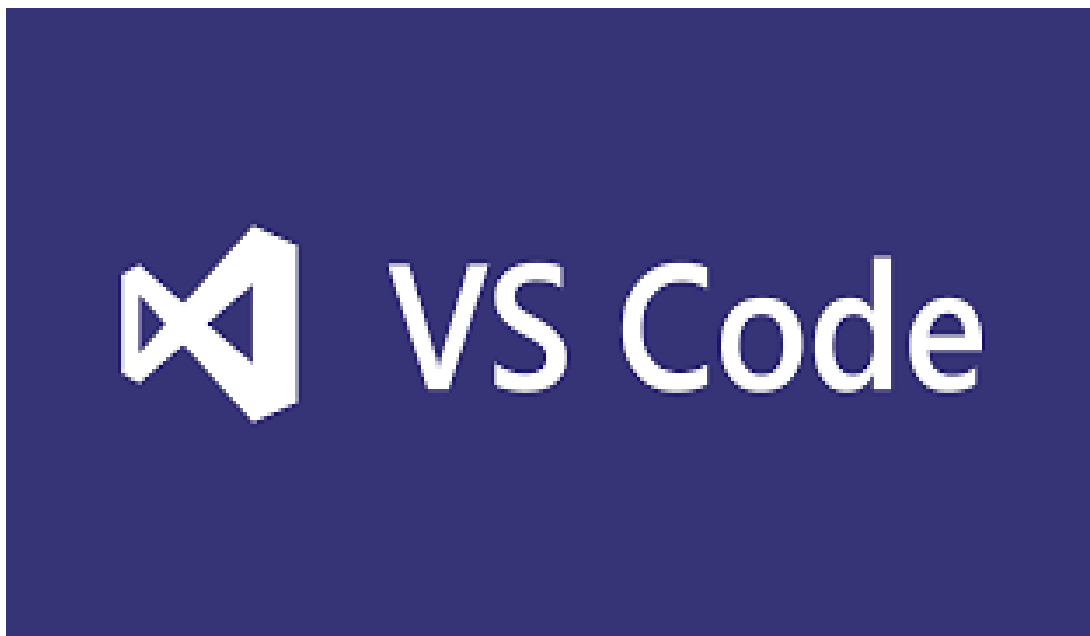


Figure 5.4: Visual Studio Code Logo

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js and C++. It is based on the Electron framework, which is used to develop Node.js Web applications that run on the Blink layout engine. It is basically a code editor redefined and optimized for building and debugging modern web and cloud applications.

The features of the VS Code editor are:

- **Language Support**

Out-of-the-box, Visual Studio Code includes basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for

additional languages can be provided by freely available extensions on the VS Code Marketplace.

- **Data collection**

Visual Studio Code collects usage data and sends it to microsoft, although this can be disabled. In addition, because of the open-source nature of the application, the telemetry code is accessible to the public, who can see exactly what is collected.^[27] According to Microsoft, the data is shared with Microsoft-controlled affiliates and subsidiaries, although enforcement may request it as part of a legal process.

- **Version control**

Source control is a built-in feature of Visual Studio Code. It has a dedicated tab inside of the menu bar where you can access version control settings and view changes made to the current project. To use the feature, you must link Visual Studio Code to any supported version control system (Git, Apache Subversion, Perforce, etc.). This allows you to create repositories as well as make push and pull requests directly from the Visual Studio Code program

5.2.4 C language

C is a procedural programming language. It was initially developed by Dennis Ritchie in the year 1972. It was mainly developed as a system programming language to write an operating system. The main features of the C language include low-level memory access, a simple set of keywords, and a clean style, these features make C language suitable for system programming like an operating system or compiler development.



Figure 5.5: C Language logo

Many later languages have borrowed syntax/features directly or indirectly from the C language. Like syntax of Java, PHP, JavaScript, and many other languages are mainly based on the C language. C++ is nearly a superset of C language (Few programs may compile in C, but not in C++).

The features of C programming language are:

- Procedural language.
- Fast and Efficient.
- Modularity.
- Statically type.
- General-purpose language.
- Rich set of built-in operators.
- Libraries with rich functions.
- Middle-level language.
- Portability.
- Easy to extend.

5.2.5 Arduino IDE



Figure 5.6: Arduino IDE logo

The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub `main()` into an

executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards.

5.3 Libraries

5.3.1 Dlib



Figure 5.7: Dlib Library Logo

Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. It is used in both industry and academia in a wide range of domains including robotics, embedded devices, mobile phones, and large high performance computing environments. Dlib's open-source licensing allows you to use it in any application, free of charge.

Major Features of Dlib are:

- Documentation
- High Quality Portable Code
- Machine Learning Algorithm
- Numerical Algorithm
- Graphical Model Inference Algorithm
- Image Processing

- Threading
- Networking
- Graphical User Interface
- Data Compression and Integrity Algorithm
- Testing
- General Utilities

5.3.2 Numpy



Figure 5.8: NumPy Library Logo

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance.

The features of the numpy library are:

- **Powerful N-Dimensional Arrays:**

Fast and versatile, the NumPy vectorization, indexing, and broadcasting concepts are the de-facto standards of array computing today.

- **Numerical Computing Tools:**

NumPy offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more.

- **Interoperable:**

NumPy supports a wide range of hardware and computing platforms, and plays well with distributed, GPU, and sparse array libraries.

- **Performant:**

The core of NumPy is well-optimized C code. Enjoy the flexibility of Python with the speed of compiled code.

- **Open source:**

Distributed under a liberal BSD License, Numpy is developed and maintained publicly on GitHub by a vibrant, responsive and diverse community.

5.3.3 Open CV

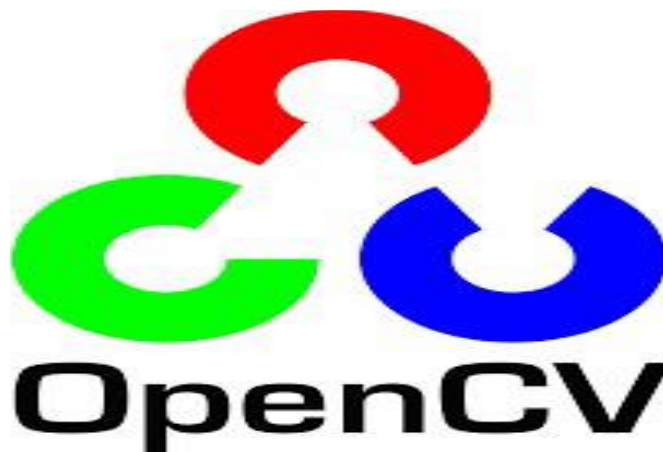


Figure 5.9: Open CV Library Logo

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

Originally developed by Intel, it was later supported by Willow Garage then Itseez. The library is cross-platform and free for use under the open-source Apache 2 License.

OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. All of the new developments and algorithms appear in the C++ interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation.[12] Wrappers in several programming languages have been developed to encourage adoption by a wider audience. In version 3.4, JavaScript bindings for a selected subset of OpenCV functions was released as OpenCV.js, to be used for web platforms.

The features of the Open CV library that make it preferable are:

- Read and write images.
- Capture and save videos.
- Process images.
- Perform feature detection.
- Detect specific objects such as faces, eyes, cars, in the videos or images.
- Analyse the video, i.e., estimate the motion in it, subtract the background, and track objects in it.

5.3.4 TensorFlow

TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. TensorFlow is a symbolic math library based on dataflow and differentiable programming. It is used for both research and production at Google. It was developed by the Google Brain team for internal Google use. It was released under the Apache License 2.0 in 2015.

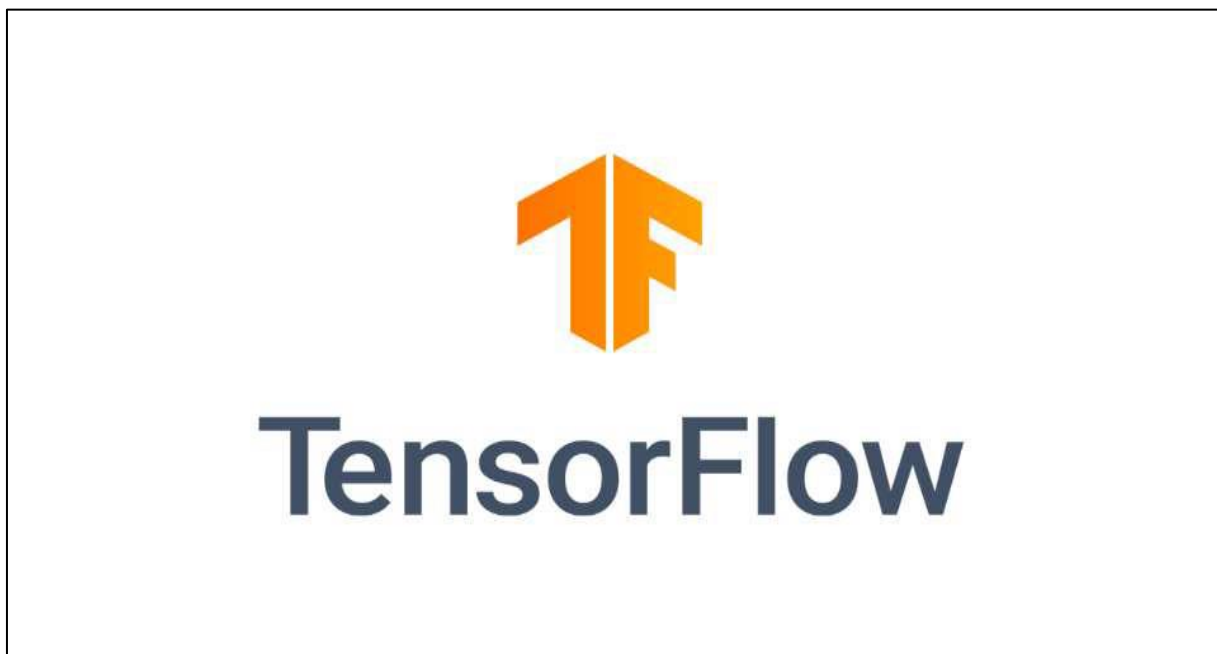


Figure 5.10: TensorFlow Library Logo

TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.

The features of the Tensorflow library that make it attractive are:

- Open-Source library
- Easy to run
- Fast Debugging
- Effective
- Scalable
- Easy experimentation
- Abstraction
- Flexibility

5.3.5 Keras



Figure 5.11: Keras Library Logo

Keras is the high-level API of TensorFlow 2: an approachable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. It provides essential abstractions and building blocks for developing and shipping machine learning solutions with high iteration velocity.

Keras empowers engineers and researchers to take full advantage of the scalability and cross-platform capabilities of TensorFlow 2: you can run Keras on TPU or on large clusters of GPUs, and you can export your Keras models to run in the browser or on a mobile device.

The salient features of the Keras library are:

- Keras is a high-level interface and uses Theano or Tensorflow for its back-end.
- It runs smoothly on both CPU and GPU.
- Keras supports almost all the models of a neural network – fully connected, convolutional, pooling, recurrent, embedding, etc. Furthermore, these models can be combined to build more complex models.

- Keras, being modular in nature, is incredibly expressive, flexible, and apt for innovative research.
- Keras is a completely Python-based framework, which makes it easy to debug and explore.
- Keras contains a large pre-defined dataset. It provides you a variety of datasets. You can use this dataset to be directly importing and loading it.
- Keras contains a number of pre-trained models. These models can be imported from `keras.applications`. These models are useful for feature extraction and fine-tuning. `Keras.application` is a module that contains weights for image classification like VGG16, VGG19, Xception, etc.

5.3.6 Pygame

Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language.



Figure 5.12: Pygame Library Logo

Pygame uses the Simple DirectMedia Layer (SDL) library, with the intention of allowing real-time computer game development without the low-level mechanics of the C programming language and its derivatives. This is based on the assumption that the most expensive functions inside games can be abstracted from the game logic, making it possible to use a high-level programming language, such as Python, to structure the game.

Other features that SDL doesn't have include vector math, collision detection, 2D sprite scene graph management, MIDI support, camera, pixel-array manipulation, transformations, filtering, advanced freetype font support, and drawing.

Applications using Pygame can run on Android phones and tablets with the use of Pygame Subset for Android (pgs4a). Sound, vibration, keyboard, and accelerometer are supported on Android.

The features of the pygame library include:

- Uses optimized C and assembly code for core functions.
- Comes with many operating systems.
- Truly portable.
- Modular.
- You control your main loop.
- Does not require a GUI to use all the functions.
- Fast response to reported bugs.

5.3.7 Time

This module provides various time-related functions. For related functionality, see also the datetime and calendar modules.

Although this module is always available, not all functions are available on all platforms. Most of the functions defined in this module call platform C library functions with the same name. It may sometimes be helpful to consult the platform documentation, because the semantics of these functions varies among platforms.

5.3.8 Argparse



Figure 5.13: Argparse module logo

The `argparse` module makes it easy to write user-friendly command-line interfaces. The program defines what arguments it requires, and `argparse` will figure out how to parse those out of `sys.argv`. The `argparse` module also automatically generates help and usage messages and issues errors when users give the program invalid arguments.

5.3.9 Sci-py

SciPy is a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.



Figure 5.14: SciPy Library Logo

The SciPy library is currently distributed under the BSD license, and its development is sponsored and supported by an open community of developers. It is also supported by NumFOCUS, a community foundation for supporting reproducible and accessible science. The SciPy package of key algorithms and functions are core to Python's scientific computing capabilities.

Available sub-packages include:

- **Cluster:** hierarchical clustering, vector quantization, K-means
- **Constants:** physical constants and conversion factors
- **fft:** Discrete Fourier Transform Algorithms
- **fftpack:** Legacy interface for Discrete Fourier Transforms
- **integrate:** numerical integration routines

- **io**: data input and output
- **linalg**: linear algebra routines
- **misc**: miscellaneous utilities
- **ndimage**: various functions for multi-dimensional image processing
- **ODR**: orthogonal distance regression classes and algorithms
- **optimize**: optimization algorithms including linear programming
- **signal**: signal processing tools
- **sparse**: sparse matrices and related algorithms
- **spatial**: algorithms for spatial structures such as k-d trees, nearest neighbours, Convex hulls.
Etc
- **special**: special functions
- **stats**: statistical functions
- **weave**: tool for writing C/C++ code as Python multiline strings

5.3.10 Simple DHT

This is an Arduino library used for the DHT sensor series. They consist of low-cost temperature and humidity sensors. This library is compatible with all architectures so you should be able to use it on all the Arduino boards.

It is used in the Arduino code in the project for the temperature and humidity sensor, in this case the DHT 11 sensor.

This library makes the use and display of the results very easy with its built-in functionalities. It is simple, Stable and fast Arduino library which was developed by Winlin.

This strictly follows the standard DHT protocol which is embedded within the DHT series sensors. Its quick speed in the display of the results, is because of its high sampling rate which supports 0.5Hz for the DHT-22 sensor and 1Hz for the DHT-11 sensor.

Hence, the simple DHT library is preferred due to the following features:

- **Simple:**
Contains simple C++ code with helpful comments.
- **Stable:**
Strictly follows the standard DHT protocol, hence it is highly stable.
- **Fast:**
Supports 0.5HZ (DHT22) or 1HZ (DHT11) sampling rate.
- **Compatible:**

Simple DHT library is compatible with multiple low-cost temperature and humidity sensors like DHT11 and DHT22.

- **MIT License:**

Simple DHT library is open-source and uses one of the most permissive licenses, enabling it to be used on any project.

Chapter 6

Advantages and Disadvantages

The real time detection system would farewell in most scenarios and the sensors would act as an added advantage and increase the safety measures to help the driver. One of the main challenges is the accuracy of the data and the network issues which would be faced in certain areas. The accuracy is also affected by any unusual behavior of the driver.

6.1 Advantages

- Real-Time monitoring from the system.
- The sensors are cost efficient.
- They don't take up much space.
- Easily replaceable.
- High levels of safety regarding the pollution abatement, due to over usage of the vehicle, is ensured.
- Can prevent accidents and save lives.

6.2 Disadvantages

- The data might not be a 100% accurate.
- False System Failure due to network issues.
- False interpretation due to unusual behaviour by the driver.
- It may malfunction due to external effects.
- To establish a proper energy interface for all the sensors.
- Accuracy in controlling the sensors depends mainly on signal strength.
- The sensor may occasionally fail and hence faulty signals could be sent to the microcontroller board.

Chapter 7

Applications

The following are few of the most prominent applications of the real time system designed using the proposed method:

- **Drowsiness Detection.**

Accurately detect if the driver is drowsy, using computer vision-based algorithms that are integrated into the system.

- **Gas detection and analysis.**

Constantly survey for any harmful gases that may be present in the vehicle, using on-board sensors and warn the driver on a positive detection.

- **Real-time Temperature analysis.**

Continuously monitor the temperature of the driver environment, as it plays a crucial role in the cause of driver drowsiness.

- **Weather Monitoring.**

Prevent accidents by warning the driver of harsh weather conditions, in most cases which may lead to drowsiness of the driver.

Chapter 8

Conclusion

We have developed a Real-Time Driver Drowsiness Detection and Alarm system which is implemented to detect the drowsiness and fatigue of a driver based on the live images captured in order to minimize the rate of road accidents. The proposed algorithm uses simple machine vision techniques to achieve the goal. The objective is also to make the system as compact as possible. The sensors would act as a driver environment monitor and help keep track of certain parameters.

Chapter 9

Future Scope

A much advanced and efficient system can be designed with additional features. The Drowsiness Detection System can be interfaced into a Driver Performance System which would track multiple parameters of the vehicle's movement and the driving style.

References

- [1] Drowsiness Detection of a Driver using Conventional Computer Vision Application, Hitendra Garg, “2020 International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC)”
- [2] Miaou, “Study of Vehicle Scrap page Rates,” Oak Ridge National Laboratory, Oak Ridge, TN, S.P., April 2012.
- [3] Wreggit, S. S., Kim, C. L., and Wierwille, W. W., Fourth Semi-Annual Research Report”, Research on Vehicle-Based Driver Status Performance Monitoring”, Blacksburg, VA: Virginia Polytechnic Institute and State University, ISE Department, January 2013.
- [4] Yi-Qing Wang, An Analysis of the Viola Jones Face Detection Algorithm, Published in IPOL, 2014.
- [5] Mandalapu Sarada Devi, Dr.Preeti R Bajaj, Driver Fatigue Detection Based on Eye Tracking, First Int Conf on Emerging Trends in Engineering and Technology, 2008.
- [6] Prashanna Rangan “Vehicle speed sensing and smoke detecting system”, March 2017
- [7] Rashida Nazir, Ayesha Tariq, “Accident Prevention and Reporting System Using GSM (SIM 900D) and GPS (NMEA 0183)”
- [8] Mukund Kulkarni; Saravanan Meenatchi Sundaram, ” Development of sensor and optimal placement for smoke detection in an electric vehicle battery pack” DOI: 10.1109/ITEC-India.2015.7386868
- [9] Kai-Tai Song; Chih-Hao, ”Design and experimental study of an ultrasonic sensor system for lateral collision avoidance at low speeds” DOI: 10.1109/IVS.2004.1336460