

# **FATIGUENESS AND UNCONCIOUS STATE DETECTION FOR DRIVERS IN E-VEHICLES**

**A PROJECT REPORT**

*Submitted by*

**PUVIYAAL V [REGISTER NO:211419104205]**

**RAKSHANA B [REGISTER NO:211419104212]**

**SIVAPRIYA S [REGISTER NO: 211419104256]**

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.**

**ANNA UNIVERSITY : CHENNAI 600 025**

**APRIL 2023**

**PANIMALAR ENGINEERING COLLEGE**  
**(An Autonomous Institution, Affiliated to Anna University,**  
**Chennai)**

**BONAFIDE CERTIFICATE**

Certified that this project report “**FATIGUENESS AND UNCONCIOUS STATE DETECTION FOR DRIVERS IN E-VEHICLES**” is the bonafide work of “**PUVIYAAL V [REGISTER NO: 211419104205], RAKSHANA B [REGISTER NO: 211419104212] and SIVAPRIYA S [REGISTER NO: 211419104256]**” who carried out the project work under my supervision.

**SIGNATURE**

**Dr.L.JABASHEELA,M.E.,Ph.D**  
**HEAD OF THE DEPARTMENT,**  
DEPARTMENT OF CSE,  
PANIMALAR ENGINEERING  
COLLEGE,  
NAZARATHPETTAI,  
POONAMALLEE,  
CHENNAI-600 123

**SIGNATURE**

**MRS.M.MAHESHWARI M.E.,Ph.D**  
**ASSOCIATE PROFESSOR,**  
DEPARTMENT OF CSE,  
PANIMALAR ENGINEERING  
COLLEGE,  
NAZARATHPETTAI,  
POONAMALLEE,  
CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the End Semester Project Viva-Voce Examination held on.....

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## **DECLARATION BY THE STUDENT**

We **PUVIYAAL V [REGISTER NO: 211419104205], RAKSHANA B [REGISTER NO: 211419104212] and SIVAPRIYA S [REGISTER NO: 211419104256]** hereby declare that this project report titled “**FATIGUENESS AND UNCONCIOUS STATE DETECTION FOR DRIVERS IN E-VEHICLES**”, under the guidance of **MRS.M.MAHESHWARI, M.E., Ph.D., ASSOCIATE PROFESSOR** , is the orginial work done by us and we have not plagiarized or submitted to any other degree in any university by us.

**1.PUVIYAAL V**

**2.RAKSHANA B**

**3.SIVAPRIYA S**

## ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our beloved Directors **Tmt.C.VIJAYARAJESWARI, Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.**, for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. L.JABASHEELA , M.E.,Ph.D.**, for the support extended throughout the project.

We would like to thank my **Project Guide Mrs.M.MAHESHWARI, M.E., Ph.D., ASSOCIATE PROFESSOR** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

1.PUVIYAAL

2. RAKSHANA B

3.SIVAPRIYA S

## **ABSTRACT**

The most frequent consequence of falling asleep—amounting to more than 40% of the reported incidents—was crossing of the right edge-line before awaking, whereas crossing of the centre line was reported by 16%. Drivers ‘lack of awareness of important precursors of falling asleep—like highway hypnosis, driving without awareness, and similar phenomena—as well as a reluctance to discontinue driving despite feeling tired, are pointed out as likely contributors to sleep-related accidents. The envisioned vehicle-based driver drowsiness detection system would continuously and unobtrusively monitor driver performance and driver psychophysiological status (in particular eye closure). The system may be programmed to provide an immediate warning signal when drowsiness is detected. If the Drowsiness state is detected this system will automatically alert the driver, if it continues it’ll slow down the vehicle and park the vehicle on left side of the road and lock the vehicle After that the vehicle will be unlocked only by Highway Patrol Department officer as a proof of concept.

**Keywords** — Microcontroller, LCD ,Drowsiness Detection,Biometric Authentication,E-Vehicles.

# TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	<b>ABSTRACT</b>	iii
	<b>LIST OF TABLES</b>	viii
	<b>LIST OF FIGURES</b>	ix
	<b>LIST OF SYMBOLS</b>	x
<b>1.</b>	<b>INTRODUCTION</b>	
	1.1 Problem Definition	1
	1.2 Scope of the project	1
<b>2.</b>	<b>LITERATURE SURVEY</b>	2
<b>3.</b>	<b>SYSTEM ANALYSIS</b>	
	3.1 Existing System	7
	3.2 Proposed system	7
	3.3 Feasibility Study	8
	3.4 Hardware Environment	10
	3.5 Software Environment	10
<b>4.</b>	<b>SYSTEM DESIGN</b>	
	4.1 UML Diagrams	13

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
<b>5.</b>	<b>SYSTEM ARCHITECTURE</b>	
	5.1 Module Design Specification	14
	5.2 Algorithms	15
<b>6.</b>	<b>SYSTEM IMPLEMENTATION</b>	
	6.1 Source code	16
<b>7.</b>	<b>SYSTEM TESTING / PERFORMANCE ANALYSIS</b>	
	7.1 Test Cases	41
<b>8.</b>	<b>CONCLUSION</b>	
	8.1 Conclusion	42
	8.2 Future Enhancements	42
	<b>APPENDICES</b>	
	A.1 Sample Screens	43
	<b>REFERENCES</b>	47

## **LIST OF TABLES**

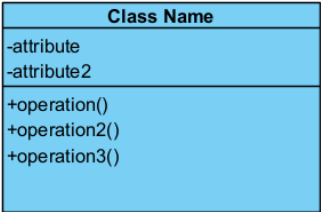


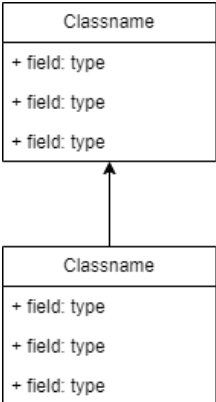
<b>TABLE NO.</b>	<b>TABLE DESCRIPTION</b>	<b>PAGE NO.</b>
7.1	TEST CASE AND REPORTS	41

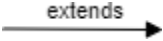






## LIST OF FIGURES

FIG NO.	FIGURE DESCRIPTION	PAGE NO.
4.1	UML DIAGRAM	19
4.2	ACTIVITY DIAGRAM	21
	SYSTEM	30
5.1	ARCHITECTURE DIAGRAM STRUCTURE OF THE DATASET	32
A.1	ENGINE AND LCD DISPLAY CONNECTED TO AUDUINO BOARD	41
A.2	BLUETOOTH CONNECTED WITH USB TO SYSTEM CAMERA	42
A.3	OUTPUT SCREEN	43
A.4	OUTPUT SCREEN WHEN DROWSINESS DETECTED	44
A.5	PARKED STATUS	45

## LIST OF SYMBOLS

S.NO	NOTATION NAME	NOTATION	DESCRIPTION
1.	Class		Represents a collection of similar entities grouped together.
2.	Association		Associations represents a static relationships between the classes.
3.	Actor		It aggregates several classes into a single class.
4.	Aggregation		Interaction between the system and the external environment.

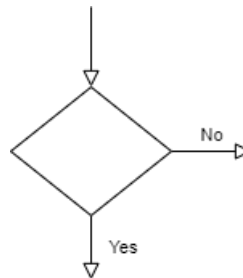
5.	Relation(uses)	<b>uses</b>	Used for additional process communication.
6.	Relation(extends)		Extends relationship is used when one use case is similar to another use case but does a bit more.
7.	Communication		Communication between various use cases.
8.	State		State of the process.
9.	Initial State		Initial state of the object.
10.	Final state		Final state of the object.

11. Control flow



Represents various control flow between the states.

12. Decision box



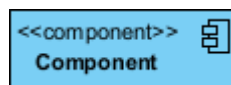
Represents decisions making process from a constraint.

13. Use case



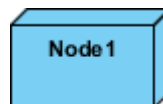
Interact ion between the system and external environment.

14. Component



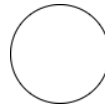
Represents physical modules which is a collection of components.

15. Node



Represents physical modules which are a collection of components.

16. Data  
Process/State



A circle in DFD represents a state or process which has been triggered due to some event or action.

17. External Entity



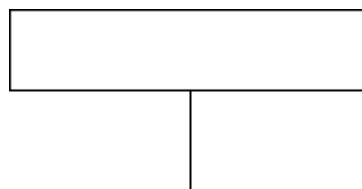
Represents external entities such as keyboard, sensors, etc.,

18. Transition



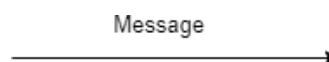
Represents communication that occurs between processes.

19. Object Lifeline



Representation of the vertical dimensions of the objects communication.

20. Message



Represents the message exchanged.

# **1. INTRODUCTION**

## **1.1 PROBLEM DEFINITION**

The goal of the project is to detect drowsiness while driving and inform the driver by a alarm at the appropriate time to avoid any mishaps. The project employs a CNN model to determine whether or not a person is drowsy based on whether or not their eyes are closed or open. The idea has a direct application in the vehicle sector, making driving safer and lowering the number of people killed in car accidents caused by drowsy driving. If the eyes closed for a long time the engine of the vehicle stops automatically and displays that the car is parked . After that only the patrol officer can authenticate and operate .

## **1.2 SCOPE OF THE PROJECT**

The purpose of the drowsiness detection system is to aid in the prevention of accidents passenger and commercial vehicles. The system will detect the early symptoms of drowsiness before the driver has fully lost all attentiveness and warn the driver that they are no longer capable of operating the vehicle safely.

## 2.LITERATURE SURVEY

**1. Title:** Drowsiness Detection System Utilizing Physiological Signals.

**Author:** Trupti K. Dange, T. S. Yengatiwar.

**Publication:** 2013.

**Methodology used:**

The Physiological parameters-based techniques detect drowsiness based on drivers' physical conditions such as heart rate, pulse rate, breathing rate, respiratory rate and body temperature, etc. These biological parameters are more reliable and accurate in drowsiness detection as they are concerned with what is happening with driver physically. Fatigue or drowsiness, change the physiological parameters such as a decrease in blood pressure, heart rate and body temperature, etc. Physiological parameters-based drowsiness detection systems detect these changes and alert the driver when he is in the state, near to sleep. A list of physiological condition-based drowsiness detection system. These measures are invasive, so require electrodes to be directly placed on the driver's body.

**Drawbacks:**

- In order to analyze the drowsiness state of the driver 200 images of a driver during a regular driving process were acquisition.
- In this study a Real Time Driver Fatigue Detection SVM Based on SVM Algorithm is presented whose test results showed that the accuracy rate of fatigue detection is up to 97.93%
- It was assumed that the drowsiness is linked to the images in which the driver has closed eyes, and the alert state is linked to the images in which the driver has opened eyes.

## **2. Title:** Drowsiness Detection with OpenCV (Using Eye Aspect Ratio)

**Author:** Adrian Rosebrock.

**Publication:** 2017.

### **Methodology used:**

A real-time algorithm to detect eye blinks in a video sequence from a standard camera is proposed. Recent landmark detectors, trained on in-the wild datasets exhibit excellent robustness against a head orientation with respect to a camera, varying illumination and facial expressions. We show that the landmarks are detected precisely enough to reliably estimate the level of the eye opening. The proposed algorithm therefore estimates the landmark positions, extracts a single scalar quantity – eye aspect ratio (EAR) – characterizing the eye opening in each frame. 6 Many methods have been proposed to automatically detect eye blinks in a video sequence. Several methods are based on motion estimation in the eye region. Typically, the face and eyes are detected by a Viola-Jones type detector. Next, motion in the eye area is estimated from optical flow, by sparse tracking, or by frame-to-frame intensity differencing and adaptive thresholding. Finally, a decision is made whether the eyes are or are not covered by eyelids.

### **Drawbacks:**

- These modern landmark detectors are trained on “in-the-wild datasets” and they are thus robust to varying illumination, various facial expressions, and moderate non-frontal head rotations.
- EAR may not necessarily recognize the eye blinks correctly, a classifier that takes a larger temporal window of a frame into account is trained. The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye.



**3. Title:** Real Time Driver Fatigue Detection Based on SVM Algorithm.

**Authors:** Burcu Kir Savas, Yasar Becerkli.

**Publication:** 2018.

**Methodology used:**

In this study, SVM based driver fatigue prediction system is proposed to increase driver safety. The proposed system has five stages: PERCLOS, count of yawn, internal zone of the mouth opening, count of eye blinking and head detection to extract attributes from video recordings. The classification stage is done with Support Vector Machine (SVM). While the YawDD dataset is used during the training phase of the classification, real-time video recordings are used during the test phase. SVM is a classification algorithm separating data items. This algorithm proposed by Vladimir N. Vapnik based on statistical learning theory. SVM, one of the machine learning methods, is widely used in the field of pattern recognition Whereas label 0 means that the driver is tired, label 1 means the driver is non-tired. Thus, it is aimed to distinguish tired drivers (driver fatigue) from non-tired drivers. In training driving simulation experiments, we had 10 volunteers (5 men and 5 women) whose ages were between 18-30. All the volunteers drove training simulation during the experiments. Attributions obtained in real time during the driving when fatigue detection system was working are indicated.

**Drawbacks:**

- Nowadays, robust real-time facial landmark detectors that capture most of the characteristic points on a human face image, including eye corners and eyelids, are available.
- Recent landmark detectors, trained on in-the wild datasets exhibit excellent robustness against a head orientations,

**4. Title:** Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State.

**Authors:** Venkata Rami Reddy Chirra, Srinivasulu Reddy Uyyala and Venkata Krishna Kishore Kolli.

**Publication:** 2019.

**Methodology used:**

Whole face region may not be required to detect the drowsiness but only eyes region is enough for detecting drowsiness. At first step by using the Viola-jones face detection algorithm face is detected from the images. Once the face is detected, Viola-jones eye detection algorithm is used to extract the eye region from the facial images. it is the first algorithm used for face detection. For the face detection the Viola-Jones algorithm having three techniques those are Haar-like features, Ada boost and Cascade classifier. In this work, Viola-Jones object detection algorithm with Haar cascade classifier was used and implemented using OPEN CV with Face and eye region are detected using ViolaJones detection algorithm. Stacked deep convolution neural network is developed to extract features and used for learning phase. A SoftMax layer in CNN classifier is used to classify the driver as sleep or non-sleep. Proposed system achieved 96.42% accuracy. Proposed system 465 effectively identifies the state of driver and alert with an alarm when the model predicts drowsy output state continuously. In future we will use transfer learning to improve the performance of the system.

**Drawbacks:**

- The mechanisms detecting fatigue and sleepiness while driving has been categorized into broad approaches, including vehicle-based.
- The project is developed to keep the vehicle secure and protect it by occupation of intruders. It uses eye blink sensor, 8051 microcontroller.

## **5. Title:** Accident Detection and Alerting Systems

**Author:** Abdulkadir Shehu Bari

**Publication:** 2022

### **Methodology used:**

With vehicles becoming increasingly affordable, there has been a surge in the number of vehicles on the roads on an average all over the world. The swift development of technology and infrastructure has made our lives easier nowadays. The initiation of technology has likewise increased the traffic hazards and road accidents take place regularly, which causes massive loss of life and property because of the poor emergency facilities. Accidents bring devastation upon victims, causing the loss of precious time and money. It has been established that a majority of accidents become fatalities because of a lack of communication to the concerned medical authorities and the consequent lack of immediate medical support. By monitoring the information from the accelerometer and the vibration sensor, a severe accident can be recognized. It then sends the alert message through the GSM Module, including the latitude and longitude data provided by the GPS module room, any rescue team, or to the car owners. This system can prove to be a lifesaver in isolated areas where an accident has occurred and no one is around in order to report the accident. Through this system, an accident can be detected and a life can be saved by the quick response from the emergency services. In this research, I have reviewed many papers related to the accident detection system and its impact of application.

### **Disadvantages:**

- In some places where there is no provision of GSM networks, it is difficult for communication.
- The biggest problem is the blockage of signal transmission by mountains, high buildings, and tunnels.

## **3. SYSTEM ANALYSIS**

### **3.1 EXISTING SYSTEM**

The existing systems require a kit attached to drivers body. It makes them inconvenient. Detection accuracy is very low and time taken to detect is higher than proposed system . The existing system the vehicle is implemented using obstacle detection and alerting user in any vehicle using ultrasonic sensor.

#### **Dis-advantages:**

- There is a more chance of sensor failure which is unpredictable during driving periods.
- The sensor setup is more expensive.
- The existing system is less effective and takes more time for detection when compared to the proposed system

### **3.2 PROPOSED SYSTEM**

In our method eye is the decision parameter for finding the state of driver. Though detection of eye may be easier to locate, but it's really quite complicated. At this point it performs the detection of eye in the required particular region with the use of detection of several features. It is a time taking process. When eye detection is done then the result is matched with the reference or threshold value for deciding the state of the driver. If the Drowsiness state is detected this system will automatically alert the driver, if it continues it'll slow down the vehicle and park the vehicle on left side of the road and lock the vehicle After that the vehicle will be unlocked only by Highway Patrol Department officer as a proof of concept.

### **3.3 FEASIBILITY:**

Feasibility Study All systems are feasible when provided with unlimited resources and infinite time. But unfortunately, this condition does not prevail in the practical world. So it is both necessary and prudent to evaluate the feasibility of the system at the earliest possible time. Months or years of effort, thousands of rupees and untold professional embarrassment can be averted if an ill- conceived system is recognized early in the definition phase. Feasibility & risk analysis are related in many ways. If project risk is great, the feasibility of producing quality software is reduced. In this case three key considerations involved in the feasibility analysis are

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

#### **ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased. Bitcoin Price Prediction using Machine Learning Dept. of ISE, CMRIT 2019-20 Page 18

#### **TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high

demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement as only minimal or null changes are required for implementing this system .

## **SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

### **3.4 Hardware Requirements :**

1. Atmega328 Microcontroller
2. LCD Base
3. Liquid Crystal Display 16x02
4. 12v/1A battery
5. L298N
6. DC Motor
7. Wheels
8. Buzzer
9. Bluetooth Module
10. Cable

### **3.5 Software Requirements :**

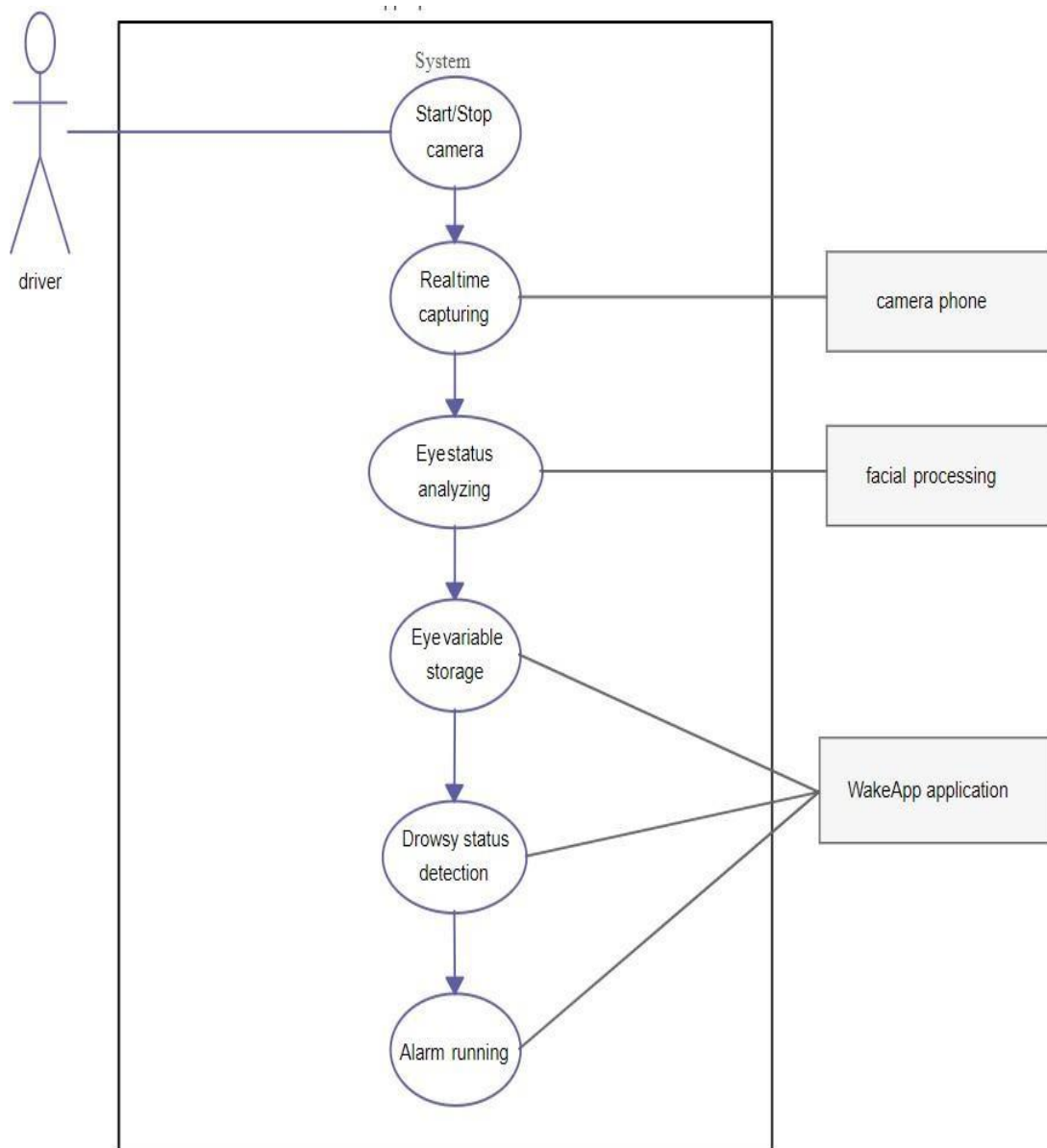
1. Arduino IDE
2. Python IDLE
3. OpenCV

## 4.

## 4.SYSTEM DESIGN

### 4.1 UML DIAGRAMS

#### 4.1.1 USE CASE DIAGRAM

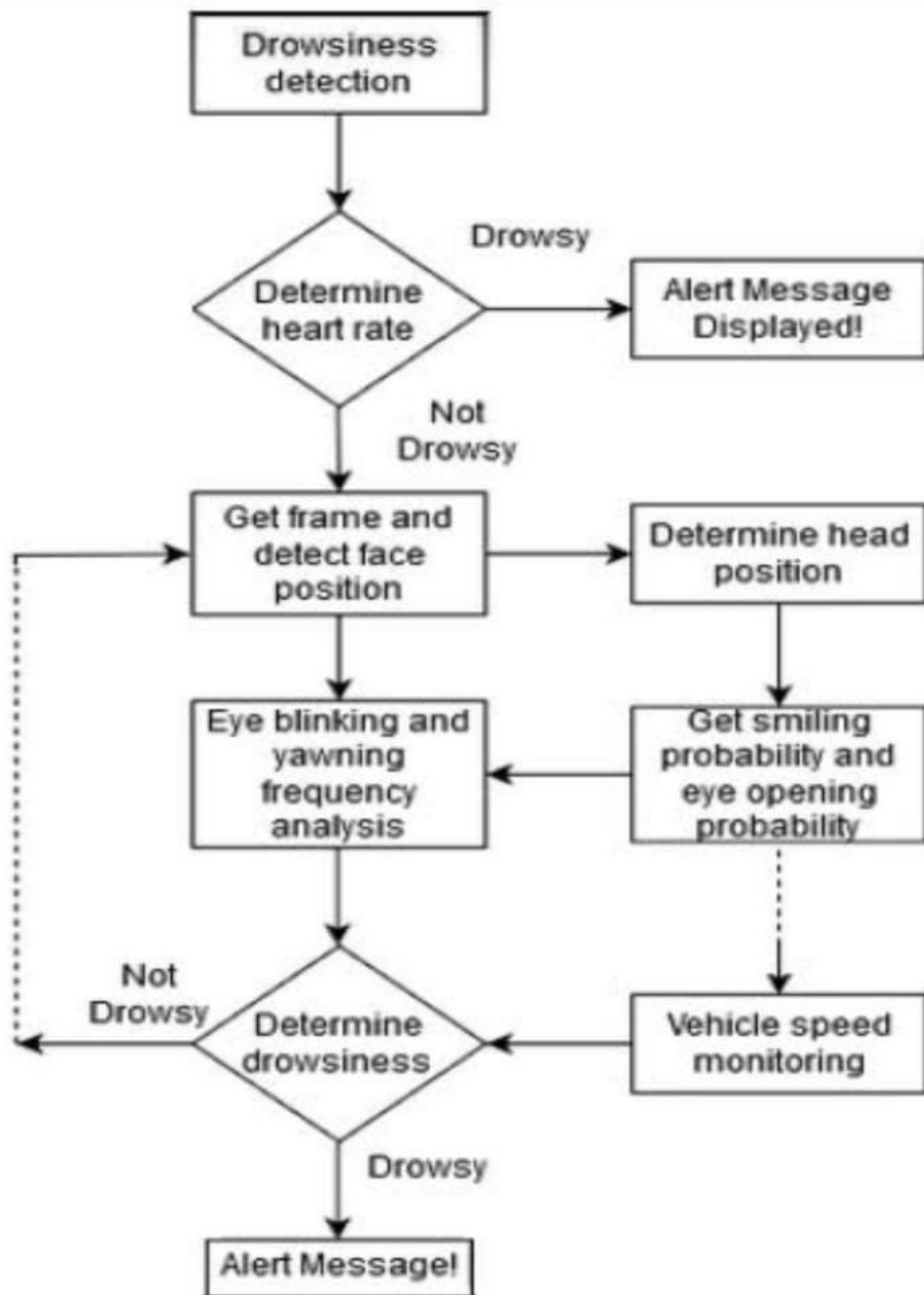


**FIG 4.1 USE CASE**

Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally.



### 4.1.2 ACTIVITY DIAGRAM



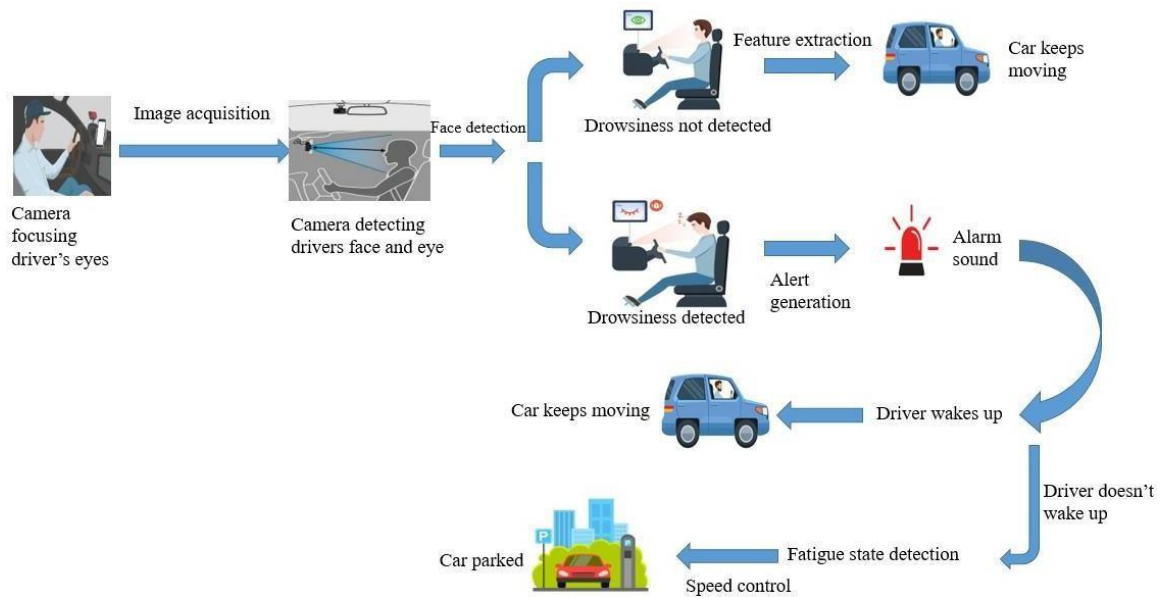
**FIG 4.2 ACTIVITY DIAGRAM**

Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. Activity diagram is some time considered as the flow chart. Although the diagrams look like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.

## 5.SYSTEM ARCHITECTURE

### 5.1 SYSTEM MODULE SPECIFICATION

#### SYSTEM ARCHITECTURE



**FIG 5.1 SYSTEM ARCHITECTURE**

To design and implement a driver fatigue detection-based accident prevention system to reduce the number of accidents caused by driver fatigue. The system will use computer vision and machine learning techniques to detect signs of driver fatigue and provide hardware based accident prevention system to save the driver from mishaps.

## MODULE DESCRIPTION

There are some modules in this methodology. They are,

- MODULE 1:Image Acquisition
- MODULE 2:Face Detection
- MODULE 3:Feature Extraction
- MODULE 4:Machine Learning
- MODULE 5:Alert Generation
- MODULE 6:Speed Control

- **MODULE 1:IMAGE ACQUISITION**

The camera mounted on the dashboard of the vehicle captures real-time images of the driver's face.

- **MODULE 2:FACE DETECTION**

The system uses computer vision algorithms to detect and isolate the driver's face.

- **MODULE 3:FEATURE EXTRACTION**

The system extracts eye features such as eye closure. After extracting the eye closure feature the system processes the eye movement.

- **MODULE 4:Machine Learning**

The system uses machine learning algorithms to analyze the extracted features and detect signs of fatigue.

- **MODULE 5:Alert Generation**

If signs of fatigue are detected, the system generates an alert to the driver through visual and alert cues.

- **MODULE 6:SPEED CONTROL**

The system may also automatically adjust the vehicle's speed or control to prevent accidents caused by driver fatigue.

## 5.2 ALGORITHM

The algorithm used in Fatigueness and unconscious state detection for drivers in E-vehicles is,

❖ Haar Cascades

### Haar Cascades

Haar cascade is an algorithm that can detect objects in images, irrespective of their scale in image and location. This algorithm is not so complex and can run in real-time. We can train a haar-cascade detector to detect various objects like cars, bikes, buildings, fruits, etc. Haar cascade uses the cascading window, and it tries to compute features in every window and classify whether it could be an object. This is the first example of object detection using a haar cascade, where we will detect human faces from a picture using a pre-trained haar cascade. This piece of work was done long before the Deep Learning Era had even started. But it's an excellent work in comparison to the powerful models that can be built with the modern day Deep Learning Techniques. The algorithm is still found to be used almost everywhere.

## 6. SYSTEM IMPLEMENTATION

### 6.1 Source code:

#### **Drowsiness.py:**

```
# import the necessary packages
from imutils.video import VideoStream
from imutils import face_utils
import numpy as np
import argparse
import imutils
import time
import dlib
import cv2
import serial

# define constants, one for the eye aspect ratio to indicate
# frames the eye must be below the threshold for to set off the
EYE_AR_THRESH = 0.26
ser = serial.Serial('COM5', 115200)

def euclidean_dist(ptA, ptB):
    # compute and return the euclidean distance between the two
    # points
    return np.linalg.norm(ptA - ptB)

def eye_aspect_ratio(eye):
    # compute the euclidean distances between the two sets of
    # vertical eye landmarks (x, y)-coordinates
    A = euclidean_dist(eye[1], eye[5])
    B = euclidean_dist(eye[2], eye[4])
    # compute the euclidean distance between the horizontal
    # eye landmark (x, y)-coordinates
```

```

C = euclidean_dist(eye[0], eye[3])
# compute the eye aspect ratio
ear = (A + B) / (2.0 * C)
# return the eye aspect ratio
return ear

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-c", "--cascade",
required=False,default="haarcascade_frontalface_default.xml",
    help = "path to where the face cascade resides")
ap.add_argument("-p", "--shape-predictor",
required=False,default="shape_predictor_68_face_landmarks.dat",
    help="path to facial landmark predictor")
ap.add_argument("-a", "--alarm", type=int, default=1,
    help="boolean used to indicate if TraffHat should be used")
args = vars(ap.parse_args())
# initialize the frame counter
COUNTER = 0
# load OpenCV's Haar cascade for face detection (which is faster than
# dlib's built-in HOG detector, but less accurate), then create the
# facial landmark predictor
print("[INFO] loading facial landmark predictor...")
detector = cv2.CascadeClassifier(args["cascade"])
predictor = dlib.shape_predictor(args["shape_predictor"])
# grab the indexes of the facial landmarks for the left and
# right eye, respectively
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
# start the video stream thread
print("[INFO] starting video stream thread...")
vs = VideoStream(src=0).start()

```

```

# vs = VideoStream(usePiCamera=True).start()
print("[INFO] Please Wait for hardware booting...")
time.sleep(10.0)
found=set()
# loop over frames from the video stream
while True:
    # grab the frame from the threaded video file stream, resize
    # it, and convert it to grayscale
    # channels)
    frame = vs.read()
    frame = imutils.resize(frame, width=450)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # detect faces in the grayscale frame
    rects = detector.detectMultiScale(gray, scaleFactor=1.1,
        minNeighbors=5, minSize=(30, 30),
        flags=cv2.CASCADE_SCALE_IMAGE)
    # loop over the face detections
    for (x, y, w, h) in rects:
        # construct a dlib rectangle object from the Haar cascade
        # bounding box
        rect = dlib.rectangle(int(x), int(y), int(x + w),
            int(y + h) # determine the facial landmarks for the face region, then
        # convert the facial landmark (x, y)-coordinates to a NumPy
        # array
        shape = predictor(gray, rect)
        shape = face_utils.shape_to_np(shape)
        # extract the left and right eye coordinates, then use the
        # coordinates to compute the eye aspect ratio for both eyes
        leftEye = shape[lStart:lEnd]
        rightEye = shape[rStart:rEnd]
        leftEAR = eye_aspect_ratio(leftEye)
        rightEAR = eye_aspect_ratio(rightEye)

```

```

# average the eye aspect ratio together for both eyes
ear = (leftEAR + rightEAR) / 2.0
# compute the convex hull for the left and right eye, then
# visualize each of the eyes
leftEyeHull = cv2.convexHull(leftEye)
rightEyeHull = cv2.convexHull(rightEye)
cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
# check to see if the eye aspect ratio is below the blink
# threshold, and if so, increment the blink frame counter
if ear < EYE_AR_THRESH:
    print('detected')
    if "alert" not in found:
        print("alert")
        ser.write(('alert').encode())
        time.sleep(2)
        found.clear()
        found.add("alert")
        cv2.putText(frame, "DROWSINESS ALERT!", (10, 30),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
# otherwise, the eye aspect ratio is not below the blink
# threshold, so reset the counter and alarm
elif ear > EYE_AR_THRESH+0.05:
    if "run" not in found:
        print("run")
        ser.write(('run').encode())
        time.sleep(2)
        found.clear()
        found.add("run")
    COUNTER = 0
    print("running")

```



```

# draw the computed eye aspect ratio on the frame to help
# with debugging and setting the correct eye aspect ratio
# thresholds and frame counters
cv2.putText(frame, "VAL: {:.3f}".format(ear), (300, 30),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
# show the frame
cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF
# if the ESC key was pressed, break from the loop
if key == 27:
    break
# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()

```

### **Arduino program**

```

#include <LiquidCrystal.h>

#include <SoftwareSerial.h>

//SoftwareSerial ble_soft(2, 3);

#define ble Serial3

LiquidCrystal lcd(8, 9, 10, 11, 12, 13);

// LiquidCrystal lcd(13, 12, 11, 10, 9, 8);

#define splash splash1

#define sp1 4

#define m1 3

#define m2 2

```

```
#define m3 5

#define m4 6

#define sp2 7

#define t1 A0

#define e1 A1

#define key 51

int tt1 = 1;

int tt2 = 2;

int tt3 = 3;

int tt4 = 4;

int tt5 = 5;

int dis_l, dis_f;

int dis_max = 150;

int dis_alt = 30;

int dis_fr;

int dis;

// int dis_max = 10;

int dis_maxx = 50;

int dis_alert = 20;

int dis_min = 4;

int v1, v2, auto_park;

float t = 1;

String IncomingData = "";
```

```

int run_stat = 0;

void (*resetFunc)(void) = 0;

void setup() {

    // put your setup code here, to run once:

    Serial.begin(9600);

    ble.begin(9600);

    LcDSet();

    pinMode(m1, OUTPUT);

    pinMode(m1, OUTPUT);

    pinMode(m2, OUTPUT);

    pinMode(m3, OUTPUT);

    pinMode(m4, OUTPUT);

    pinMode(t1, OUTPUT); // Sets the trigPin as an Output

    pinMode(e1, INPUT);

    pinMode(key, INPUT_PULLUP);

    stopbot();

    delay(2000);

    lcd.clear();

    analogWrite(sp1, 255);

    analogWrite(sp2, 255);

}

void LcDSet() {

    lcd.begin(16, 2);

    splash(0, "Drowsiness");

```

```

splash(1, "RX");

delay(2000);

lcd.clear();

}

void loop() {

    // put your main code here, to run repeatedly:

    dis_l = ultracm(t1, e1);

    dis_fr = dis_l;

    if (run_stat) {

        if (dis_fr < dis_alert) {

            stopbot();

            splash(1, "Object Detected");

            analogWrite(sp1, 0);

        } else if (dis_fr < dis_maxx) {

            analogWrite(sp1, 150);

            delay(50);

            splash(1, "Object Detected");

        } else if (dis_fr > dis_maxx) {

            analogWrite(sp1, 255);

            forward();

            splash(1, "Run");

        }

    }

    lcd.setCursor(0, 0);

```

```

lcd.print("Ultra:  ");

lcd.setCursor(7, 0);

lcd.print(dis_1);

ble_data();

delay(10);

ble.flush();

}

void ble_data() {

    while (ble.available()) {

        IncomingData = ble.readString();

        delayMicroseconds(10);

    }

    // IncomingData = "run";

    if (IncomingData.length() > 0) {

        if (IncomingData == "run") {

            analogWrite(sp1, 255);

            analogWrite(sp2, 255);

            forward();

run_stat = 1;

            //    ble.println("Running");

        }

        if (IncomingData == "alert") {

            auto_park = 1;

            park();

```

```

    //    ble.println("Running");

}

if (IncomingData == "rst") {

    resetFunc();

    //    ble.println("Running");

}

IncomingData = "";

}

}

void park() {

while (auto_park) {

dis_l = ultracm(t1, e1);

dis_fr = dis_l;

if (dis_fr < dis_alert) {

    stopbot();

    splash(1, "Object Detected");

    analogWrite(sp1, 0);

} else if (dis_fr < dis_maxx) {

analogWrite(sp1, 150);

    delay(50);

splash(1, "Object Detected");

} else if (dis_fr > dis_maxx) {

    analogWrite(sp1, 255);

```

```

    splash(1, "Run");

}

t += 0.9;

splash(1, String(t, 1));

dis_l = ultracm(t1, e1);

dis_f = ultracm(t1, e1);


lcd.setCursor(0, 0);

lcd.print("Ultra:  ");

lcd.setCursor(7, 0);

lcd.print(dis_l);

if (t > 0 and t < tt1) {

    if (dis_f > dis_alt) {

        forward();

    } else {

        t = tt1;

    }

} else if (t >= tt1 and t < tt2) {

    if (dis_l > dis_alt) {

        //    forward();

        left();

    } else {

        t = tt2;

    }

}

```

```

} else if (t >= tt2 and t < tt3) {

    if (dis_f > dis_alt) {

        forward();

    } else {

        t = tt3;

    }

}

else if (t >= tt3 and t < tt4) {

    right();

} else if (t >= tt4 and t < tt5) {

    if (dis_f > dis_alt) {

        forward();

    } else {

        t = tt5;

    }

} else if (t >= tt5) {

    stopbot();

    splash(1, "Parked");

    while (1) {

        int key_r = digitalRead(key);

    if (!key_r) {

        resetFunc();

    }

}

```



```
}
```

} This piece of work was done long before the Deep Learning Era had even started. But it's an excellent work in comparison to the powerful models that can be built with the modern day Deep Learning Techniques. The algorithm is still found to be used almost everywhere.

```
while (ble.available()) {  
  
  IncomingData = ble.readString();  
  
  delayMicroseconds(10);  
  
}  
  
if (IncomingData.length() > 0) {  
  
  if (IncomingData == "rst") {  
  
    resetFunc();  
  
    //   ble.println("Running");  
  
  }  
  
  IncomingData = "";  
  
}  
  
}
```

### **Splash.arduino program**

```
void forward()  
  
{  
  
splash(1,"Forward");  
  
Serial.println("FORWARD" );  
  
digitalWrite(m1, HIGH); digitalWrite(m2, LOW);  
  
digitalWrite(m3, LOW); digitalWrite(m4, HIGH);  
  
}
```

```
}
```

```
void backward()
```

```
{
```

```
  splash(1,"Backward");
```

```
  Serial.println("BACKWARD" );
```

```
  digitalWrite(m1, LOW); digitalWrite(m2, HIGH);
```

```
  digitalWrite(m3, HIGH); digitalWrite(m4, LOW);
```

```
}
```

```
void right()
```

```
{
```

```
  splash(1,"Right");
```

```
  Serial.println("RIGHT" );
```

```
  digitalWrite(m1, HIGH); digitalWrite(m2, LOW);
```

```
  digitalWrite(m3, HIGH); digitalWrite(m4, LOW);
```

```
}
```

```
void left()
```

```
{
```

```
  splash(1,"Left");
```

```
  Serial.println("LEFT" );
```

```
  digitalWrite(m1, LOW); digitalWrite(m2, HIGH);
```

```
  digitalWrite(m3, LOW); digitalWrite(m4, HIGH);
```

```
}
```

```

void stopbot()

{

    splash(1,"Stop");

    Serial.println("STOP" );

    digitalWrite(m1, LOW); digitalWrite(m2, LOW);

    digitalWrite(m3, LOW); digitalWrite(m4, LOW);

}

```

[11:46 am, 09/04/2023] Rakshana: void splash1( int row, String txt)

```

{

    int curs = (17 - txt.length()) / 2;

    lcd.setCursor(0, row);

    lcd.print("_____");


    lcd.setCursor(curs, row);

    lcd.print(txt);

    delay(300);

}

```

[11:46 am, 09/04/2023] Rakshana: int ultracm(int trigPin, int echoPin) {

```

    long duration;

    int distance;

    digitalWrite(trigPin, LOW);


    delayMicroseconds(2);

```

```

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance = duration * 0.034 / 2;

if (distance > 150) {

    distance = 150;

}

if (distance < 4) {

    distance = 0;

}

return distance;

}

```

## **PY Serial.AI**

```

#include <SoftwareSerial.h>

SoftwareSerial ble_soft(2, 3);

#define ble ble_soft

#define python Serial

#define splash splash1

#define buz A0

String IncomingData = "";

```

```

int trans_dat;

float ti = 0;

void setup() {

  LcDSet();

  ble.begin(9600);

  python.begin(115200);

  python.println("Ready");

  ble.print("rst");

  pinMode(buz, OUTPUT);

  digitalWrite(buz, LOW);

  // digitalWrite(buz, HIGH);

}

```

```

void LcDSet() {

  lcd.begin(16, 2);

  splash(0, "Drowsiness");

  splash(1, "TX");

  delay(2000);

  lcd.clear();

}

void loop() {

  while (python.available()) {

```

```

IncomingData = python.readString();

delayMicroseconds(5);

}

if (IncomingData.length() > 0) {

    if (IncomingData == "run") {

        splash(1, "Run");

        //    ble.print("ru--+;

        ti = 0;

        trans_dat = 1;

        python.println("Running");

    } else if (IncomingData == "alert") {

        //ble.print("alert");

        trans_dat = 2;

        splash(1, "alert");

        python.println("Alert");

    }

    IncomingData = "";

}

if (trans_dat == 1) {

    ble.print("run");

    python.println("running");

    digitalWrite(buz, LOW);

```

```

    delay(1000);

}

else if (trans_dat == 2 and ti > 10) {

    ble.print("alert");

    ti = 0;

    python.println("alerted");

    digitalWrite(buz, LOW);

    delay(2000);

} else if (trans_dat == 2) {

    digitalWrite(buz, HIGH);

    delay(50);

    digitalWrite(buz, LOW);

    ti += 0.1;

}

```

### **Shape predictor . py**

```

# import the necessary packages

from imutils.video import VideoStream

from imutils import face_utils

import numpy as np

import argparse

import imutils

import time

import dlib

import cv2

```

```

import serial

# define constants, one for the eye aspect ratio to indicate
# frames the eye must be below the threshold for to set off the

EYE_AR_THRESH = 0.26

ser = serial.Serial('COM5', 115200)

def euclidean_dist(ptA, ptB):

    # compute and return the euclidean distance between the two
    # points

    return np.linalg.norm(ptA - ptB)

def eye_aspect_ratio(eye):

    # compute the euclidean distances between the two sets of
    # vertical eye landmarks (x, y)-coordinates

    A = euclidean_dist(eye[1], eye[5])

    B = euclidean_dist(eye[2], eye[4])

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

    C = euclidean_dist(eye[0], eye[3])

    # compute the eye aspect ratio

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

    # return the eye aspect ratio

    return ear

# construct the argument parse and parse the arguments

ap = argparse.ArgumentParser()

```



```

ap.add_argument("-c", "--cascade",
required=False,default="haarcascade_frontalface_default.xml",

    help = "path to where the face cascade resides")

ap.add_argument("-p", "--shape-predictor",
required=False,default="shape_predictor_68_face_landmarks.dat",

    help="path to facial landmark predictor")

ap.add_argument("-a", "--alarm", type=int, default=1,

    help="boolean used to indicate if TraffHat should be used")

args = vars(ap.parse_args())

# initialize the frame counter

COUNTER = 0

# load OpenCV's Haar cascade for face detection (which is faster than

# dlib's built-in HOG detector, but less accurate), then create the

# facial landmark predictor

print("[INFO] loading facial landmark predictor...")

detector = cv2.CascadeClassifier(args["cascade"])

predictor = dlib.shape_predictor(args["shape_predictor"])

# grab the indexes of the facial landmarks for the left and

# right eye, respectively

(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]

(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]

# start the video stream thread

```

```

print("[INFO] starting video stream thread...")

vs = VideoStream(src=0).start()

# vs = VideoStream(usePiCamera=True).start()

print("[INFO] Please Wait for hardware booting...")

time.sleep(10.0)

found=set()

# loop over frames from the video stream

while True:

    # grab the frame from the threaded video file stream, resize

    # it, and convert it to grayscale

    # channels)

    frame = vs.read()

    frame = imutils.resize(frame, width=450)

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


    # detect faces in the grayscale frame

    rects = detector.detectMultiScale(gray, scaleFactor=1.1,

        minNeighbors=5, minSize=(30, 30),

        flags=cv2.CASCADE_SCALE_IMAGE)

    # loop over the face detections

    for (x, y, w, h) in rects:

        # construct a dlib rectangle object from the Haar cascade

        # bounding box

        rect = dlib.rectangle(int(x), int(y), int(x + w),

```

```

    int(y + h)

# determine the facial landmarks for the face region, then

# convert the facial landmark (x, y)-coordinates to a NumPy
# array

shape = predictor(gray, rect)

shape = face_utils.shape_to_np(shape)


# extract the left and right eye coordinates, then use the
# coordinates to compute the eye aspect ratio for both eyes

leftEye = shape[lStart:lEnd]

rightEye = shape[rStart:rEnd]

leftEAR = eye_aspect_ratio(leftEye)

rightEAR = eye_aspect_ratio(rightEye)


# average the eye aspect ratio together for both eyes

ear = (leftEAR + rightEAR) / 2.0


# compute the convex hull for the left and right eye, then
# visualize each of the eyes

leftEyeHull = cv2.convexHull(leftEye)

rightEyeHull = cv2.convexHull(rightEye)

cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)

cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)

# check to see if the eye aspect ratio is below the blink

```

```

# threshold, and if so, increment the blink frame counter

if ear < EYE_AR_THRESH:

    print('detected')

    if "alert" not in found:

        print("alert")

        ser.write(('alert').encode())

        time.sleep(2)

        found.clear()

        found.add("alert")

        cv2.putText(frame, "DROWSINESS ALERT!", (10, 30),

                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

# otherwise, the eye aspect ratio is not below the blink

# threshold, so reset the counter and alarm

elif ear > EYE_AR_THRESH+0.05:

    if "run" not in found:

        print("run")

        ser.write(('run').encode())

        time.sleep(2)

        found.clear()

        found.add("run")

    COUNTER = 0

    print("running")

# draw the computed eye aspect ratio on the frame to help

```

```
# with debugging and setting the correct eye aspect ratio

# thresholds and frame counters

cv2.putText(frame, "VAL: {:.3f}".format(ear), (300, 30),

            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

# show the frame

cv2.imshow("Frame", frame)

key = cv2.waitKey(1) & 0xFF

# if the ESC key was pressed, break from the loop

if key == 27:

    break

# do a bit of cleanup

cv2.destroyAllWindows()

vs.stop()
```

## 7. SYSTEM TESTING

### 7.1 TEST CASES AND REPORTS

TEST CASE ID	TEST CASE/ACTION TO BE PERFORMED	EXPECTED RESULT	ACTUAL RESULT	PASS /FAIL
1.	Face detection	Face detection of driver	Detects the face drivers face	Pass
2.	Eye detection	Detection of eye in drivers face	Exactly detected the eye of the drivers face	Pass
3.	Alarm sign when drowsiness detected	Alarming the alarm sound when driver closes the eye	Alarming the alarm sound when driver closes the eye	Pass
4.	Stop the engine when fatigue state detected	Slows down and stops the engine when fatigue state detected	Slows down and stops the engine when fatigue state detected	Pass
5.	Detection of obstacle range and display parked	Detecting the range of obstacles and displaying parked in LCD	Detecting the range of obstacles and displaying parked in LCD.	Pass

## **8.**

# **CONCLUSION**

### **8.1 CONCLUSION**

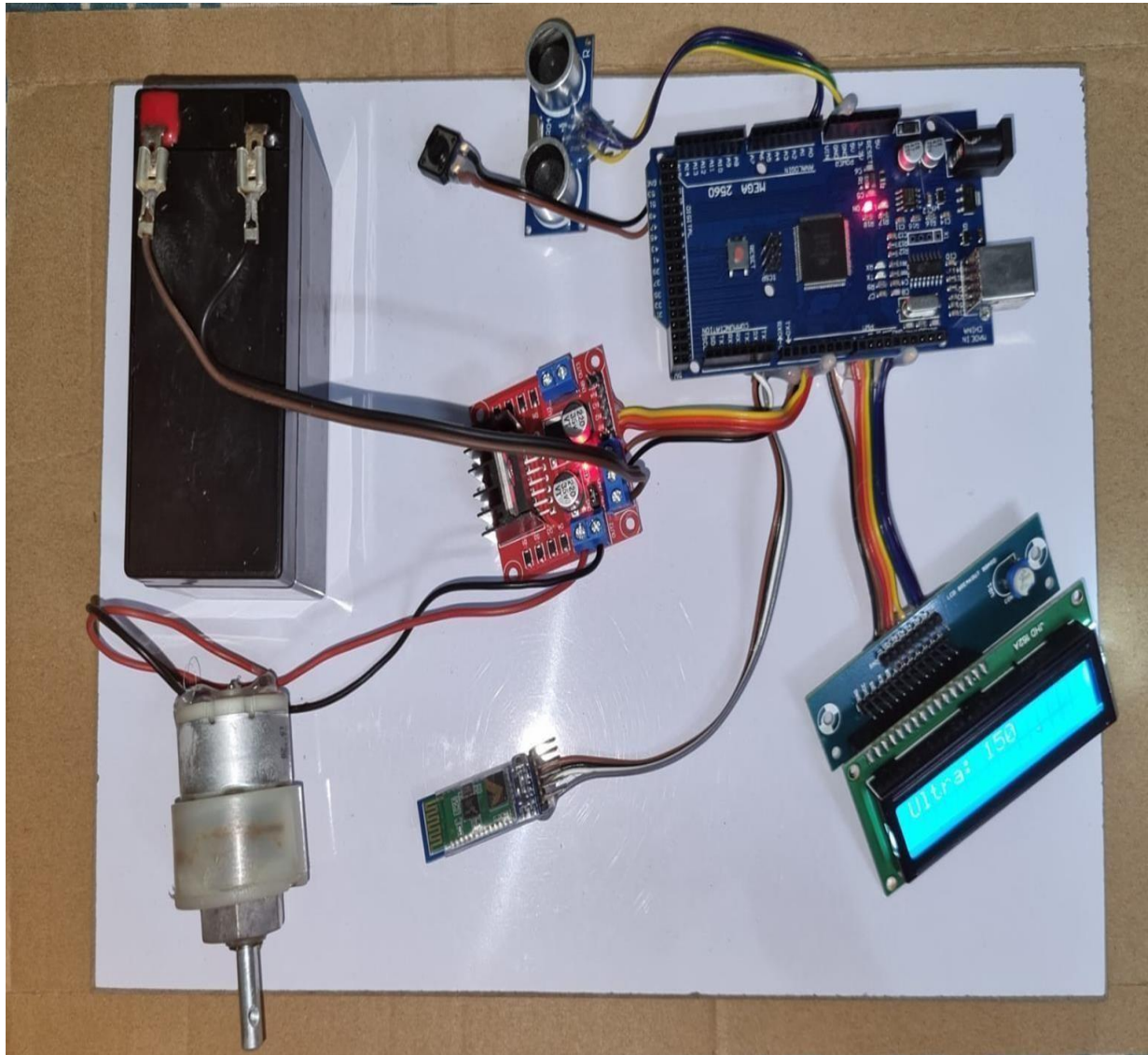
The driver fatigue detection-based accident prevention system is a vital project that can help reduce accidents caused by driver fatigue. The system uses computer vision and machine learning techniques to monitor behavior and identify signs of fatigue. By alerting driver and automatically adjusting the vehicle's speed or control, the system can help prevent accidents caused by driver fatigue and improve road safety.

### **8.2 FUTURE ENHANCEMENT**

The driver fatigue detection-based accident prevention system is an evolving technology that can benefit from further enhancements and improvements. Some of the possible future enhancements for the project are multimodal sensing, Real-time feedback, cloud base data analytics, integration with navigation system and integration with vehicle to vehicle communication.

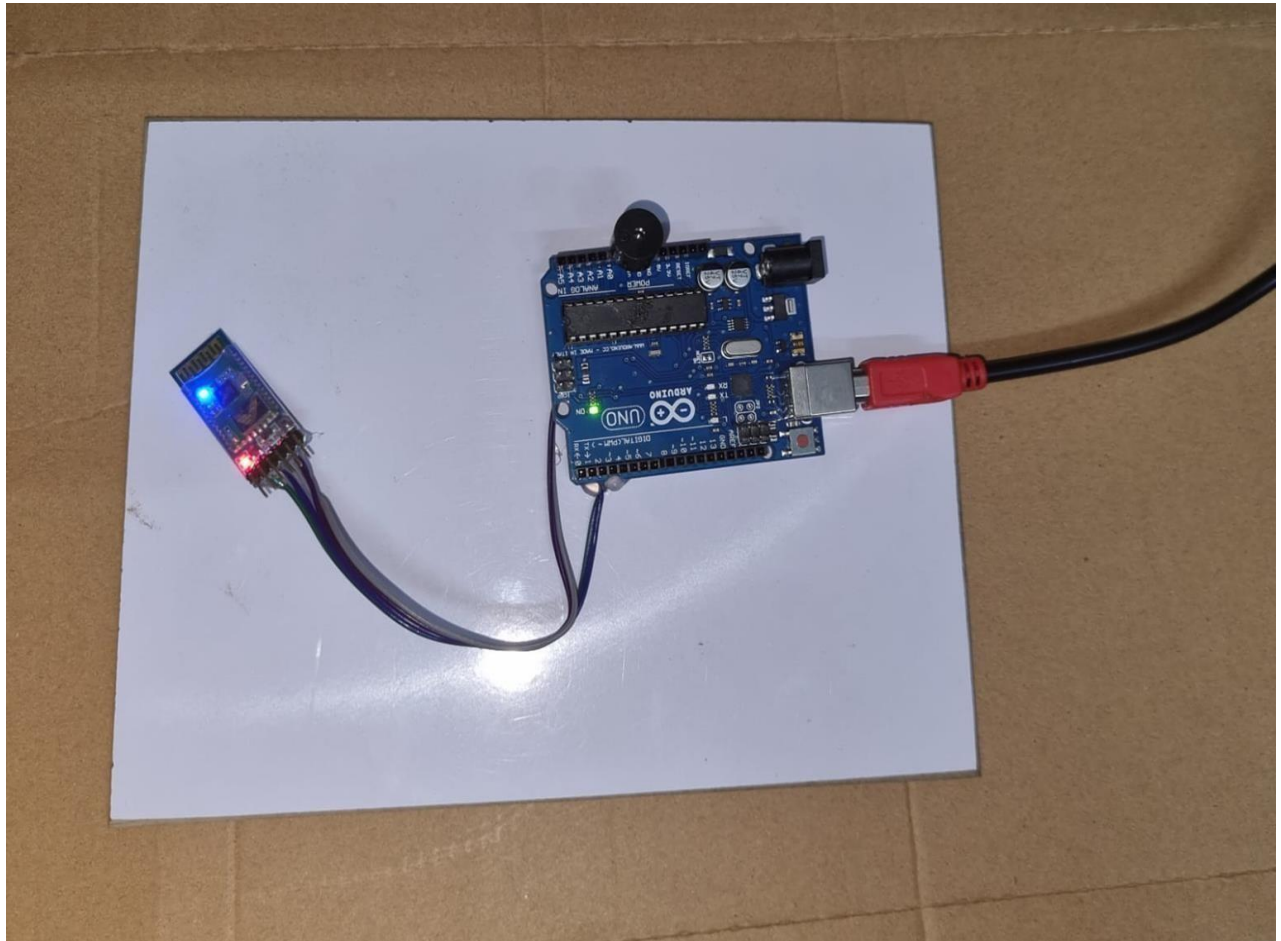
# APPENDICES

## A.1 SAMPLE SCREENS

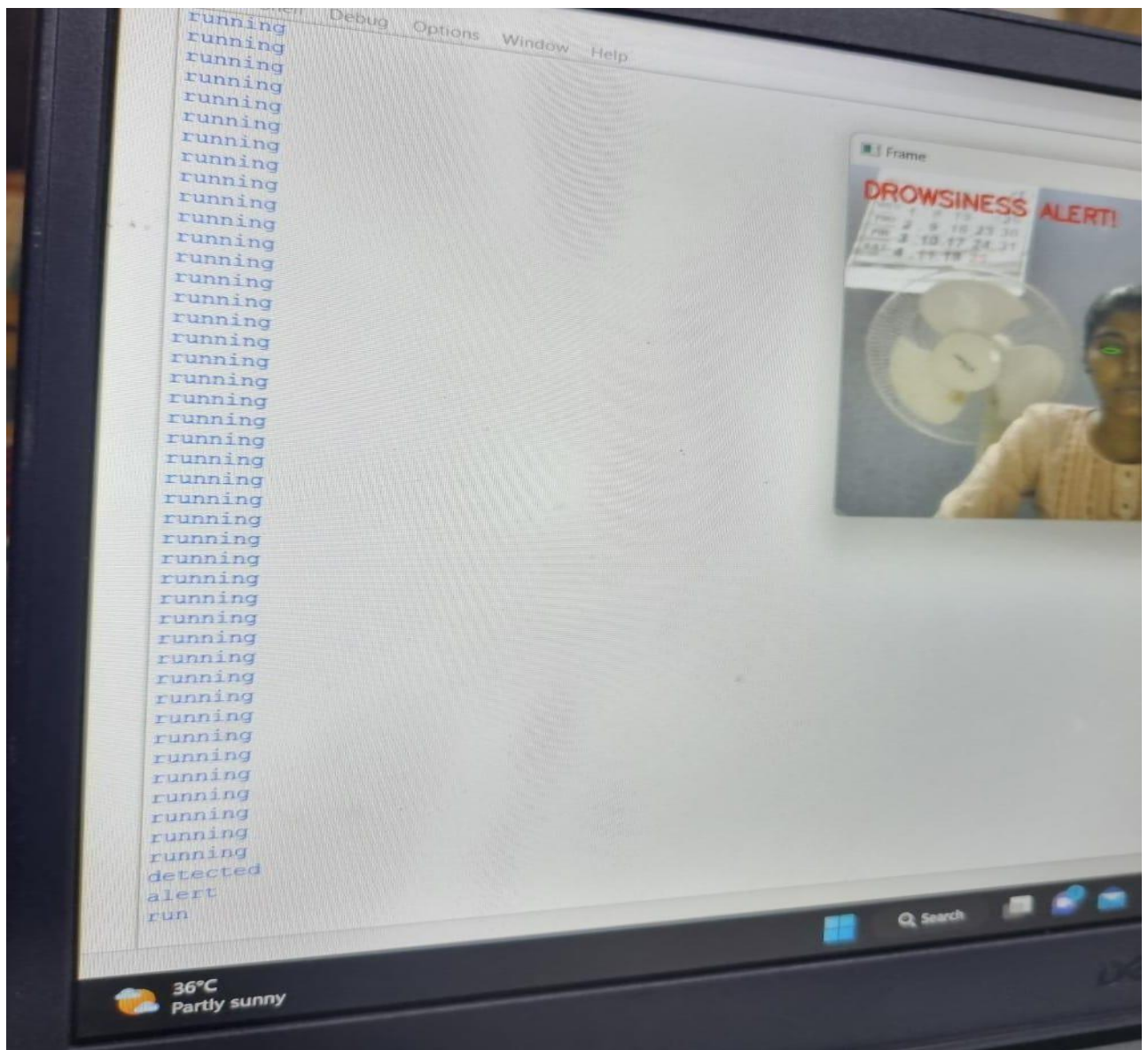


**FIG A.1 Engine and LCD display connected to Arduino board**

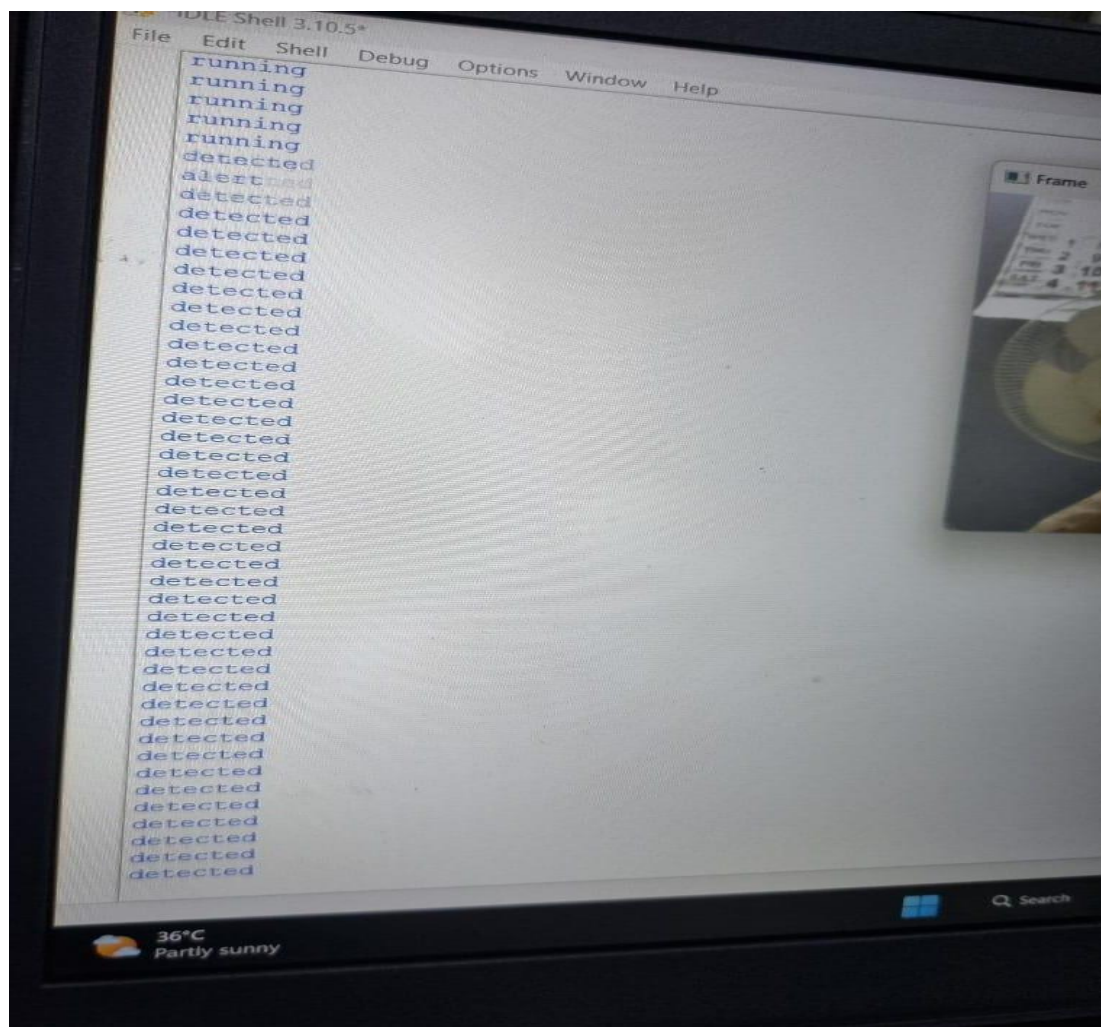




**FIG A.2 Bluetooth connected with USB to system camera**



**FIG A.3 Output screen**



**FIG A.4 Output screen when drowsiness detected**



**FIG A.5 Parked  
status**

## REFERENCE

- [1] Q. Abbas and A. Alsheddy, “Driver fatigue detection systems using multisensors, smartphone, A comparative analysis,” *Sensors, Intell. Sens. Syst. Vehicle*, vol. 21, no. 1, p. 56, Dec. 2020, doi: 10.3390/s21010056.
- [2] Shabinar Binti Abdul Hamid Anis Diyana Rosli Widad Ismail, Aimi Zulliayana Rosli Design and Implementation of RFID-based Anti-Theft System 23 - 25 Nov. 2012 .
- [3] Geeth Jayendra, Sisil Kumarawadu, Lasantha Meegahapola RFID-Based Anti-theft Auto Security System with an Immobilizer 8 – 11 August 2007.
- [4] Vinoth Kumar Sadagopan, Upendran Rajendran, Albert Joe Francis. Anti-Theft Control System Design Using Embedded System 2011.
- [5] Vivek Kumar Sehgal<sup>1\*</sup>, Mudit Singhal<sup>2</sup>, Bhart Mangla<sup>3</sup>, Sudeep Singh<sup>4</sup>, and Shivangi Kulshrestha. An Embedded Interface for GSM Based Car Security System 2012.
- [6] Tanya Ignatenko, Frans M. J. Willems. Biometric Systems: Privacy and Secrecy Aspects 2009.
- [7] M. A. Mohd Nasir, W. Mansor GSM based Motorcycle Security System 2011.
- [8] Kaisheng Zhang, Jiao She and Mingxing Gao and Wenbo Ma Study on the Embedded Fingerprint Image Recognition System 2010.
- [9] Shihab A. Hameed, Shaima Abdulla, Mohd Ershad, Fauzan Zahudi, Aisha Hassan New Automobile Monitoring and Tracking Model: Facilitate Model with Handhelds 17- 19 May 2011.
- [10] Jayanta Kumar Pany & R. N. Das Choudhury Embedded Automobile Engine Locking System, Using GSM Technology 2011.



