

Drowsiness Detection and Focus Control System



By

Jawad Hussain Shah
Syed Muhammad Ali Fatmi

Department of Information Technology
University of Haripur
June, 2022

Drowsiness Detection and Focus Control System



By

Jawad Hussain Shah, Syed Muhammad Ali Fatmi
F18-0333 , F18-0338

Supervised by

Habib Akbar

Department of Information Technology
University of Haripur
June, 2022

Drowsiness Detection and Focus Control System



By

Jawad Hussain Shah, Syed Muhammad Ali Fatmi

A Dissertation Submitted in Partial Fulfilment for the Degree of
BACHELORS OF SCIENCE
IN
SOFTWARE ENGINEERING

Department of Information Technology
University of Haripur
June, 2022

Drowsiness Detection and Focus Control System

By

Jawad Hussain Shah, Syed Muhammad Ali Fatmi

CERTIFICATE

A THESIS SUBMITTED IN THE PARTIAL FULFILMENT OF THE
REQUIRMENTS FOR THE DEGREE OF BACHELORS
IN COMPUTER SCIENCE

We accept this dissertation as conforming to the required standards

Habib Akbar
Project Supervisor

Mr. Adeel Ahmad
Final Year Project
Coordinator

Examiner Name
External Examiner

Prof. Mukhtaj Khan
Head of Department

Department of Information Technology
University of Haripur
June, 2022

Declaration

I hereby declare that this dissertation is neither as a whole or part thereof has been copied out from any source. If any part of this thesis is proved to be copied or found to be a report of some other, we shall bear the consequences. No portion of this work presented in the thesis has been submitted in support of any application from any other degree of qualification or any other university or any other institute of learning.

I further declare that this research and all the associated documents with this, reports and all records are submitted as a partial requirement of the degree of Bachelor in Software Engineering. We understand and transfer copyrights for these materials to University of Haripur and we will not sale this research and documents for getting any financial gains. It is further declared that I developed this and also this thesis is entirely on the basis of our personal efforts made under the sincere guidance of the project supervisor Habib Akbar Department of Information Technology, University of Haripur, Haripur.

Date: 28th June, 2022

Jawad Hussain Shah Syed Muhammad Ali Fatmi

Plagiarism Certificate

This thesis has been checked for Plagiarism. Turnitin report endorsed by Supervisor is attached at the end of thesis.

Date: 29th Jun, 2022

Student Name(s)

Student Signature

Supervisor Signature

Dedicated to

*This Piece of Work is Dedicated to Our Caring
Parents, Friends, Teachers and Loved Ones for Their
Endless Efforts, Prayers and Worthy Encouragement.*

Acknowledgements

First, we want to thank Allah The Almighty for blessings us with his grace and taking our endeavor to a successful culmination. We extend our sincere and heartfelt thanks to our esteemed guide, Mr. Habib Akbar, for providing us with the right guidance and advice at the crucial junctures and for showing us the right way. We would also want to thank to all our friends who gave us an inspiration to struggle towards our goal. Last but not the least, we would like to thank our family who made us capable of achieving the accomplishment by providing us with every facility we needed, and supporting us through all the tough times of our life.

Abstract

In past few years it has noted that the massive increase in the transportation due to the growth of population increasing, when traffic increases chances of accident respectively increase, there is one of the most occurring cause of road accident is drowsiness or focus loss of the driver, drowsiness is the state before going to sleep state and focus control means driver stop seen front of car or during conversation with sitting person along the driving seat seeing him as a result accident occurs, the proposed system aim to monitor the driver to detect the drowsiness and focus of the driver to minimize the ratio of accident which occurs due to drowsiness and focus loss, the system detect through distance between eyelashes point and measure the eyes edges point distance for the focus control, we using mediapipe for the eyes point extractions, whenever measured distance between of eyelashes points or edges point reached at mention point, the system will start alerting the driver through the install speaker in the system , whenever alarm start, the driver become alert and chance of accident become reduced.

Table of Content

Declaration		i
Dedication		iii
Acknowledgements		iv
Abstract		v
List of Figures		ix
Chapter 1	Introduction	1
1.1	About the Project	1
1.2	Purpose	2
1.3	Scope	2
1.4	Goals and Objectives	2
1.5	Main functionality	3
1.6	Constraints	3
Chapter 2	Existing System	4
2.1	Steering Wheel Movement (SWM)	4
	2.1.1 Flaws in SWM	5
2.2	Standard Deviation of Lane Position (SDLP)	5

	2.2.1	Flaws in SDLP	5
2.3	IOT based Detection System (IR Sensor)		6
	2.3.1	Flaws in IOT base Detection System	7
Chapter 3		Proposed System	8
3.1	Requirement Analysis		8
	3.1.1	Functional Requirements	9
		3.1.1.1 Video Recording	9
		3.1.1.2 Face Detection	9
		3.1.1.3 Eyes Detection	9
		3.1.1.4 Drowsiness Detection	9
	3.1.2	None Functional Requirements	10
		3.1.2.1 Portable	10
		3.1.2.2 Reliable	10
		3.1.2.3 Accurate	10
		3.1.2.4 Maintainable	10
		3.1.2.5 Efficient	10
		3.1.2.6 Affordable	10
3.2	Technology Used		10
	3.2.1	MediaPipe Face Mesh Technology	10
	3.2.2	Models	11
		3.2.2.1 FACE DETECTION MODEL	11
		3.2.2.2 BlazeFace Model	11
		3.2.2.3 Face Landmark Model	12
	3.2.4	Facial Mapping (landmarks) Dllib	13
3.3	Hardware Requirements		14
	3.3.1	Pi Camera Module	14

	3.3.2	Raspberry Pi Module	15
	3.3.3	Speaker	16
3.4	System Design		17
	3.4.1	Activity Diagram	17
	3.4.2	Sequence Diagram	18
Chapter 4		Implementation	19
4.1	Core Libraries		19
	4.1.1	OpenCV (cv2)	19
	4.1.2	MediaPipe	19
	4.1.3	Time	20
	4.1.4	Math	20
	4.1.5	Util	20
	4.1.6	PlaySound	20
4.2	Core Functions		21
	4.2.	Constructor	21
	4.2.	Alert	24
	4.2.	LandmarkDetection	24
	4.2.	Lm_Distance	25
Chapter 5		Conclusion and Future Work	26
5.1	Conclusion		26
5.2	Future Work		26
	4.2.	Integration	27
	4.2.	Autopilot Mode	27
	4.2.	Night Vision Camera	27
	4.2.	Auto ON OFF	27
References			28

List of Figures

Figure	Description	Page
1	Figure 1 IR Sensor	7
2	Figure 2 Drowsiness Detection	9
3	Figure 3 Face Landmark	12
4	Figure 4 Facial mapping with dilib	13
5	Figure 5(A) Pi Cam	14
6	Figure 5(B) Raspberry Pi	15
7	Figure 5(C) Speaker	16

Chapter 1

Introduction

1. Introduction

In past few years the demand of transportation increased, when transportation increased the chances of accident also increased. One of the main cause of cause is driver's drowsiness or focus lost. Identifying the levels of drivers' drowsiness has an important role in minimizing the number of fatal injuries in traffic accident. Recent statistics and reports show that 20 to 50 million people are killed or injured in car crashes all over the world. Assessments conducted by US NHTSA (National Highway Traffic Safety Administration) showed that 100000 car accidents occur every year for which the drivers' drowsiness is one of the principal contributors. [1]

1.1. About the project

Due to increasing in the number of transportation the occurrence of accident also increased. There are a lot of reasons; one of them is Driver's Drowsiness. The aim of our project is to detect the Driver's Drowsiness and focus control by tracking eye blinking and head

movement. Our system will detect the abnormal behavior of the eyes, if any occurs the system will alert the drivers by using alarming system.

1.2. Purpose

As we describe in the upper part of the document The main and most vital role played in the road accident is driver drowsiness, so the main purpose of our system is to detect the drowsiness state of the driver and once the state is occurring then the system responds to that state and that respond will help the driver in order to prevent the driver from going into sleep state, that result the driver keep awake and hopefully the road accident occurrence will have reduced.

1.3. Scope

Generally, this system can be implement an any vehicle (LTV/HTV) but the main scope of this system is for the LTV/HTV (heavy traffic vehicle), because they travel for long route to ship material from one place to another and the driver become tired as well as they are also alone. Due to considering these situation this system can be helpful for them and save many lives.

1.4. Goal and Objectives

- a.** Save life
- b.** Reduce rate of accident
- c.** Keep alert
- d.** Prevent from sleep state
- e.** Cost Effective

1.5. Main functionality

The main functionality of the system is to detect the blinking of eyes and head movement through camera recording from real time video frames and if system found the eyes closed or lost focus for more than specified time the system will warn through alarm.

1.6. Constraint

- a.** The system will not work in the following situation:
- b. If driver wear the glasses.
- c. If there is darkness or night.
- d. If Batteries are dies.

Chapter 2

Existing System

2.1. Steering Wheel Movement (SWM)

It measured using steering angle sensor and it is a widely used vehicle-based measure for detecting the level of driver drowsiness [2,3,4]. Using an angle sensor mounted on the steering column, the driver's steering behavior is measured. When drowsy, the number of micro-corrections on the steering wheel reduces compared to normal driving [5]. Fairclough and Graham found that sleep deprived drivers made fewer steering wheel reversals than normal drivers [4]. To eliminate the effect of lane changes, the researchers considered only small steering wheel movements (between 0.5° and 5°), which are needed to adjust the lateral position within the lane [2]. Hence, based on small SWMs, it is possible to determine the drowsiness state of the driver and thus provide an alert if needed. In a simulated environment, light side winds that pushed the car to the right side of the road were added along a curved road in order to create variations in the lateral position and force the drivers to make corrective SWMs [3]. Car companies, such as Nissan and Renault, have adopted SWMs but it works in very limited situations [6]. This is because they can function reliably only at particular

environments and are too dependent on the geometric characteristics of the road and to a lesser extent on the kinetic characteristics of the vehicle [6].

2.1.1. Flaws in SWM

The flaw in the steering wheel movement whenever state road is ahead and no need of steering movement at that point system generate errors

2.2. Standard Deviation of Lane Position (SDLP)

It is an another measure through which the level of driver drowsiness can be evaluated [7]. In a simulated environment, the software itself gives the SDLP and in case of field experiments the position of lane is tracked using an external camera. Ingre *et al.* conducted an experiment to derive numerical statistics based on SDLP and found that, as KSS ratings increased, SDLP (meters) also increased [7]. For example, KSS ratings of 1, 5, 8, and 9 corresponded to SDLP measurements of 0.19, 0.26, 0.36 and 0.47, respectively. The SDLP was calculated based on the average of 20 participants; however, with some drivers, the SDLP did not exceeded 0.25 m even for a KSS rating of 9. In the above experiment by performing correlation analysis on a subject to subject basis significant difference is noted. Another limitation of SDLP is that it is purely dependent on external factors like road marking, climatic and lighting conditions. In summary, many studies have determined that vehicle-based measures are a poor predictor of performance error risk due to drowsiness. Moreover, vehicular-based metrics are not specific to drowsiness. SDLP can also be caused by any type of impaired driving, including driving under the influence of alcohol or other drugs, especially depressants. [8–10][1]

2.2.1. Flaw in Standard Deviation of Lane Position

Whenever, driver intentionally move the vehicle against the line like parking or if any area where lane not exist or lane may be unclear, so the existing system alert for wrong information.

2.3. IOT base detection system (IR SENSOR)

To identify the drowsiness. IR sensor consists of infrared transmitter and receiver. Infrared transmitter emits infrared rays. The transmitted IR rays are received by IR receiver [14]. The IR transmitter and IR receiver are arranged in parallel. When the signal is given, the IR sensor starts functioning and IR transmitter emits the infrared rays to the receiver. The comparator is coupled with IR receiver. The operational amplifier is attached to comparator. To the inverting input terminal of the comparator the reference voltage is given, the comparator is linked to receiver. When there is a disruption is present in the IR rays between sender (transmitter) and recipient (receiver), the IR receiver will not conduct. Hence the voltage at the inverting input terminal is lower than the voltage at the non-inverting input. Therefore, the output of comparator is high. The output voltage of comparator is given to microcontroller. When IR receiver receives the rays from transmitter, the IR receiver becomes conducting since the voltage at the non-inverting terminal is lower than voltage at the inverting terminal. Therefore, output of comparator is low. Hence the output of comparator is set to controller. This circuit is used for counting eyelid movement.

Fig (1)

2.3.1. Flaw in IOT base detection system

The problem is in IOT that if a person closes his or her eyes for the sometime there is high chance that IR sensor receive low intense of light due to distraction of light and system considered the awoken state and cannot alarm although the person is in sleep mode and chance of accident occurrence.



Fig .1

Chapter 3

Proposed System

3.1. Requirements Analysis

Requirement analysis is the process of determining user assumptions of a new or reformed project and a well-defined stage in the Software Development Life Cycle. Requirements are an explanation of how the system should behave or an illustration of the system properties or characteristics. It can also be a 'what' plan to be expected. The Software Requirements Analysis process involves the task of promoting and documenting the needs of all these users, modeling and analyzing these requirements and documenting them as the basis for program development. The requirements are divided into two parts.

- I. Functional
- II. Non-Functional

3.1.1. Functional requirements

The proposed system contains following primary components

3.1.1.1. Video recording

Camera will record the continuous video streams.

3.1.1.2. Face detection

After recorded video streams, the video stream is converted to number frames in order to detect the face.

3.1.1.3. Eyes detection

Detected face will be analyzed by face landmark algorithm, which can detect eyes from face.

3.1.1.4. Drowsiness detection

The above mentioned requirements (detection techniques) works base on python library (open-cv), Open-CV work on real time image processing, based on computer vision algorithm when eyes are detected through land marks algorithm, on the bases of eyes landmark points, distance between horizontal and vertical eye landmark can measure, on the basis of distance, drowsiness can be detected shown in fig (2) below.

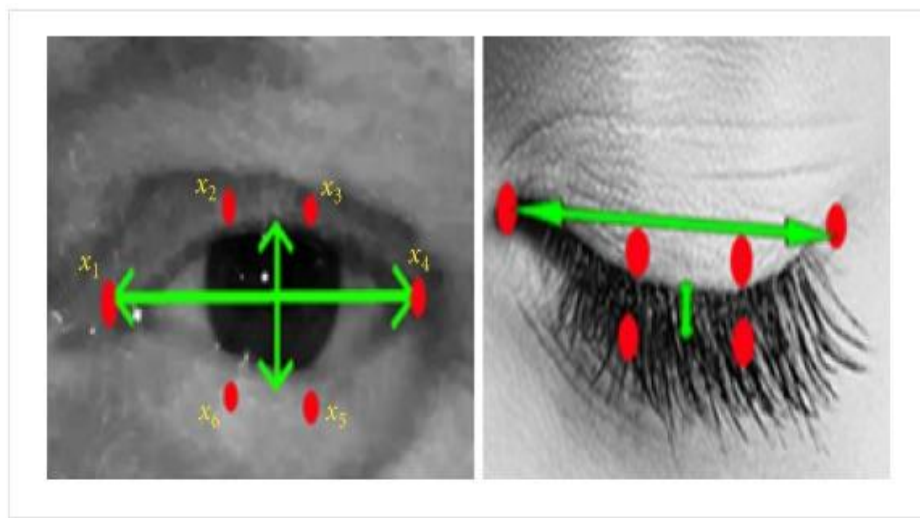


Fig .2

3.1.2. Nonfunctional requirements

3.1.2.1. Portable

Proposed system is can easily install in any kind of vehicle, and size of system small that have no effect on wide angle view

3.1.2.2. Reliable

In every environment system support and work 24/7

3.1.2.3. Accurate

System is highly accurate detect the target point accurately without any error

3.1.2.4. Maintainable

System is easy to maintain, having proper documentation

3.1.2.5. Efficient

Efficient in term of processing speed of frame is high and provide output quickly

3.1.2.6. Affordable

System is cheap and easily available and spare parts also available in market at the low price

3.2. Technology used

3.2.1. MediaPipe Face Mesh Technology

MediaPipe Face Mesh is a solution that estimates 468 3D face landmarks in real-time even on mobile devices. It employs machine learning (ML) to infer the 3D facial surface, requiring only a single camera input without the need for a dedicated depth sensor.[12]

3.2.2. Models

3.2.2.1. FACE DETECTION MODEL

The face detector is the same BlazeFace model used in MediaPipe Face Detection

3.2.2.2. BlazeFace Model

BlazeFace is a machine learning model developed by Google to rapidly detect the location and keypoints of faces. The position of the face and the keypoints of the face can be obtained simultaneously. There are six key points: eyes, nose, ears, and mouth. It is also possible to detect multiple people at the same time.[11]

3.2.2.3. FACE LANDMARK MODEL

For 3D face landmarks we employed transfer learning and trained a network with several objectives: the network simultaneously predicts 3D landmark coordinates on synthetic rendered data and 2D semantic contours on annotated real-world data. The resulting network provided us with reasonable 3D landmark predictions not just on synthetic but also on real-world data. [12]



Fig .3

3.2.3. Facial mapping (landmarks) with Dlib

Identifying faces in photos or videos is very cool, but this isn't enough information to create powerful applications, we need more information about the person's face, like position, whether the mouth is opened or closed, whether the eyes are opened, closed, looking up and etc. The Dlib, a library capable of giving you 68 points (landmarks) of the face.

It's a landmark's facial detector with pre-trained models, the dlib is used to estimate the location of 68 coordinates (x, y) that map the facial points on a person's face like image below.[13]

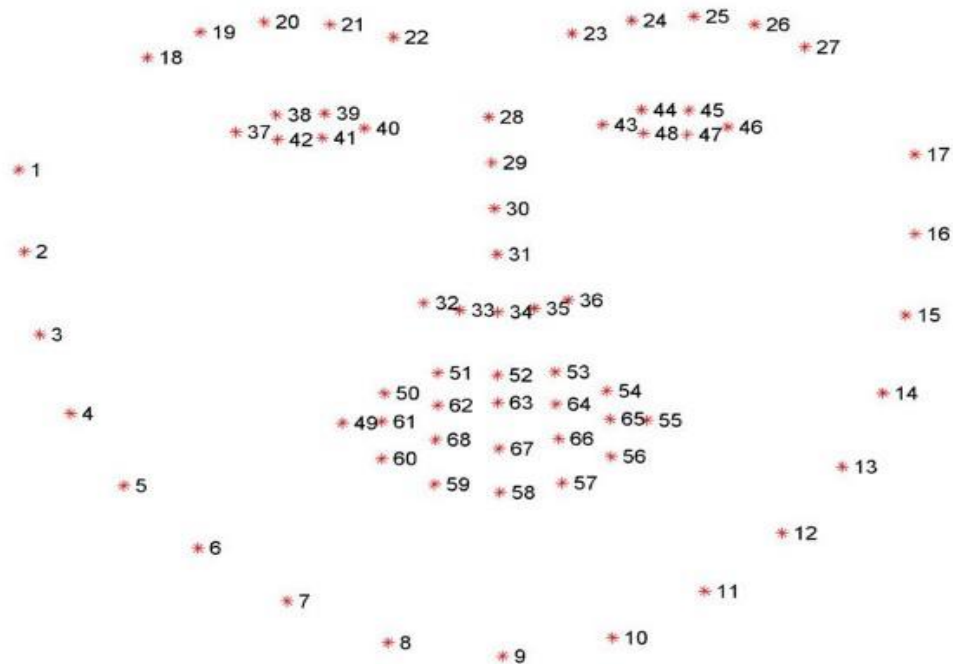


Fig .4

3.3. Hardware requirements

The proposed system will contain following components

3.3.1. Pi Camera Module

Pi camera module use instead of original camera module, to capture high definition(HD) video, pi camera can operate in different video modes which connect to a port where cable on the raspberry pi module that is show in figure 3(A)

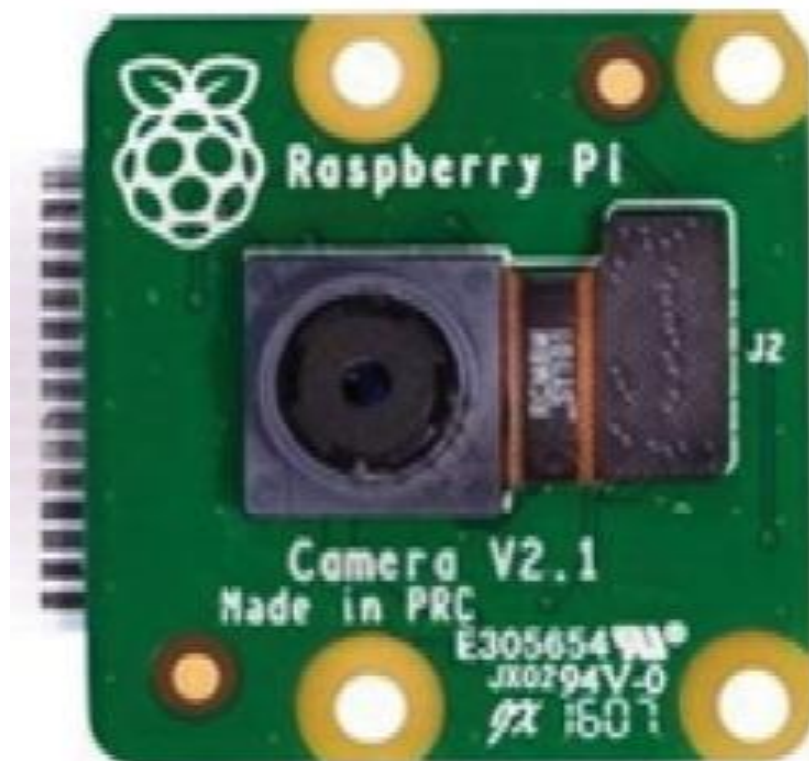


Fig 5(A) Pi Cam

3.3.2. Raspberry Pi Module

A credit card size computer which is embedded with CPU and other parts, and contain Raspbain operating system. It includes microprocessor which designed for window operating system. Shown in fig (b)



Fig 5(B) Raspberry Pi

3.3.3. Speaker

Voice generating device which convert electromagnetic waves into the sound, when the drowsiness or focus lost detected then voltage is provided as an alert to generate regular programed vice sound

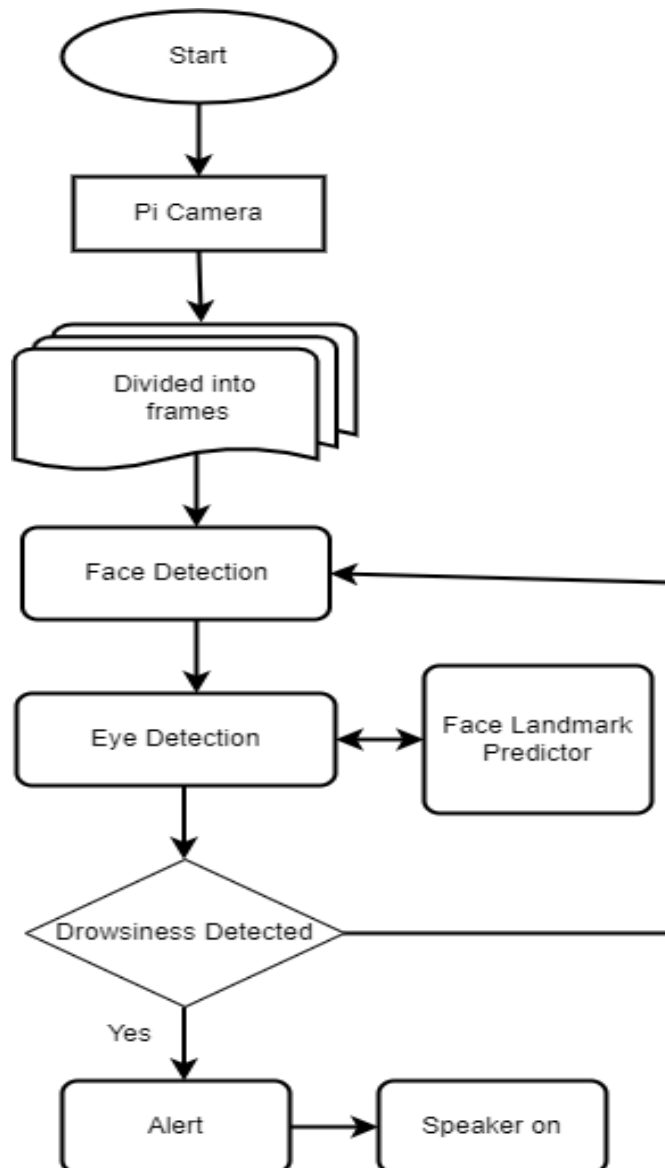


Fig 5(C) Speaker

3.4. System design

3.4.1. Activity Diagram

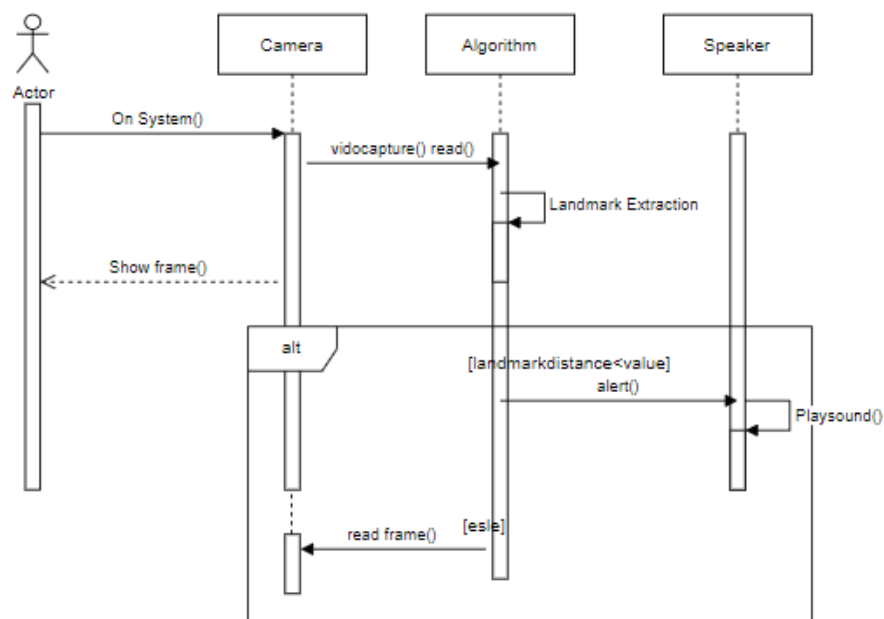
Activity diagram is a UML diagram to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.



3.4.2. Sequence Diagram

A sequence diagram consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction.

In the below diagram first user start video camera for the recording, in the next step camera return to user showing frames and forward captured video to algorithm in the captured video algorithm extract the facial landmarks then algorithm check that if the extracted features distance is less than the value it sends message to the speaker here speaker call play sound function which start mp3 audio file , if the value of distance is greater than the value then algorithm request for the new frames



Chapter 4

Implementation

4.1. Core Libraries

A library is a collection pre-defined classes or module which includes bundles of codes which can be used repeatedly in a program. It make programmer work simpler and easier. In the project we have used the following libraries:

4.1.1. OpenCV(cv2)

OpenCV is open source and cross platform computer vision library. It is use for image processing, computer vision and machine learning. It is use in real-time computer vision application. It mainly focuses on image processing and video capturing in real-time.

4.1.2. Mediapipe

MediaPipe Face Mesh is a solution that estimates 468 3D face landmarks in real-time even on mobile devices. It employs machine learning (ML) to infer the 3D facial surface, requiring only a single camera input without the need for a dedicated depth sensor.[12]

4.1.3. Time

Time library is use for extracting date and time. We used time library for calculating fps(Frame per Second).

4.1.4. Math

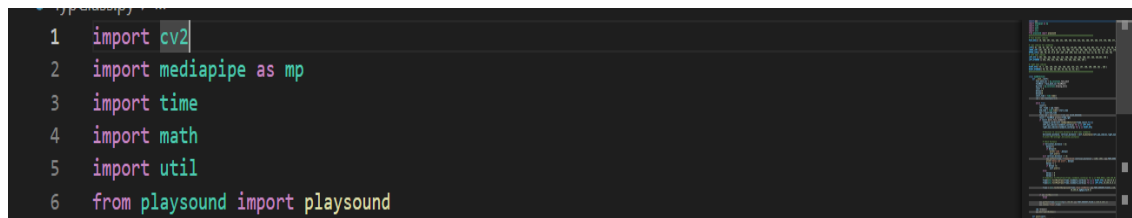
It is mathematical library of python which have built in mathematical operations, like calculating distance, square root, average and percentage etc.

4.1.5. Util

It is self-made library, which is used for coloring, text etc. It contain a functions fillPolyTranse and textWithBackground which is used for given functionality.

4.1.6. PlaySound

It contain single function named playsound(). This function required one argument which is path of file which we want to play. It may be local file path or url of the file on the internet. Playsound library work for both wave and mp3format.



```
1 import cv2
2 import mediapipe as mp
3 import time
4 import math
5 import util
6 from playsound import playsound
```

4.2. Core Functions

Our system contains the following functions.

4.2.1. Constructor

Constructor is as function which automatically called when the object of the class created. Mostly constructor use for initializing variables. Here first we create an object of solutions which in mediapipe contain face_mesh library, then we create an object of face_mesh.FaceMesh() which is responsible for extracting face landmarks point. For recording video frames we use VideoCapture() which is function of OpenCV(cv2).

```
25     def __init__(self):
26         face_mesh_sol = mp.solutions.face_mesh
27         faceMesh = face_mesh_sol.FaceMesh()
28         mp_draw = mp.solutions.drawing_utils
29         count = 0
30         delay1=0
31         delay2=0
32         start_time = time.time()
33         cam = cv2.VideoCapture(0)
```

VideoCapture() record video as frame, for capturing every frame we use *while condition*. We convert every frame to RGB using *cvtColor()* function. After converting to RGB we pass every frame to *FaceMesh.process()* function which extract the landmarks point. In *if condition* we check that whether the landmark points extracted or not. If true then we pass them to *landmarkDetection()* function which return all landmarks coordinates. Then we extract specific points which we need. We extracted here left eye and right eye indices. After that we pass that indices to the *lm_Distance()* which return average of both horizontal distance and vertical distance between eyebrows of both eyes separately.

```
34
35     while True:
36         count+=1
37         ret, frame = cam.read()
38         end_time = time.time()-start_time
39         fps = count/end_time
40         frame_rgb = cv2.cvtColor(frame,cv2.COLOR_BGR2RGB)
41         result = faceMesh.process(frame_rgb)
42         if result.multi_face_landmarks:
43             landmark_coords=self.landmarkDetection(frame,result,False)
44             left_eye_indices=[landmark_coords[p] for p in LEFT_EYE]
45             right_eye_indices=[landmark_coords[p] for p in RIGHT_EYE]
46
47             # Function to calculate distance of both eyes landmarks
48             horizontal_distance, vertical_distance = self.lm_Distance(left_eye_indices,right
49             # print('HOR-average',horizontal_distance)
50
```

Then we check the horizontal average distance and average vertical distance for some specific values.

If conditions true, then increment the delay variable and check if the delay become true, then call alert function and function perform functionality. If condition false the value of delay variables become zero. By coloring the specific area on the face or whole face we used *fillPolyTrans()* which takes list of landmarks coordinates as parameter. We used *textWithBackground()* for showing the text with color on the background. We use *waitKey()* which use for key commands and we specify which key will be pressed. If that key pressed *waitkey()* perform the function. We also use *putText()* function for showing text on the background. We use *release()* function which is responsible for stop capturing, if we do not stop the capturing, the system will not respond. *destroyAllWindows()* function use in OpenCv to stop the whole program mean to destroy all window.

```
50
51     # Check Distance
52     if horizontal_distance < 22:
53         delay1+=1
54         if delay1>5:
55             print('side ',delay1)
56             self.alert()
57     elif vertical_distance < 7.0:
58         cv2.putText(frame, f'Sleep/blink {vertical_distance}', (100, 100), cv2.FONT_HERSHEY_PLAIN,2, (0, 255, 0))
59         print('delay and alert', delay2)
60         delay2 += 1
61         if delay2 >2:
62             self.alert()
63     else:
64         delay1 = 0
65         delay2 = 0
66     # frame=util.fillPolyTrans(frame,[landmark_coors[p] for p in FACE_OVAL],(255,255,0),0.3)
67     frame=util.fillPolyTrans(frame,[landmark_coors[p] for p in RIGHT_EYE],(0,255,0),0.1)
68     frame=util.fillPolyTrans(frame,[landmark_coors[p] for p in LEFT_EYE],(0,255,0),0.1)
69
70     frame = util.textWithBackground(frame,'Face LandMarks',cv2.FONT_HERSHEY_PLAIN,2,(20,70),2,(0,0,255),
71                                     (0,255,0),bgOpacity=0.3)
72
73     if cv2.waitKey(1)==13:
74         break
75
76     cv2.putText(frame,str(int(fps)),(50,50),cv2.FONT_HERSHEY_PLAIN,1,(255,0,255),2)
77     cv2.imshow('Frame',frame)
78
79     cam.release()
80     cv2.destroyAllWindows()
81
```

4.2.2. Alert

alert() made for play sound when call. In *alert()* function we use builtin function *playsound()* function which is responsible for playing audio file. Audio file may be in the form of wave or in mp3 format.

```
81
82 def alert(self):
83     playsound('alert.mp3')
84
```

4.2.3. LandmarkDetection

The *landmarkdetection()* function took 3 parameters as an argument named as *img*, *result*, *draw*

In *img.shape()* return two values height and width of the image,

In *mesh_coords* we store x and y position of each landmark point value which is returned by *result.multi_face_landmarks.landmarks*, for extracting coordinates of each point multiply each x and y position with *img* height and width respectively.

```
84
85 # Detecting coordinates of landmark
86 def landmarkDetection(self, img, result, draw=False):
87
88     h, w = img.shape[:2]
89     mesh_coords = [(int(point.x * w), int(point.y * h)) for point in result.multi_face_landmarks[0].landmark]
90     if draw:
91         [cv2.circle(img, p, 2, (0, 255, 0), -1) for p in mesh_coords]
92     return mesh_coords
93
```

4.2.4. Lm_Distance

This function takes two arguments of indices of left eye and indices of right eye.

In this function we extracted specific point of eyes. By mean that we extracted position of central point of upper side of the both eyes and lower side of the both eyes and similarly position of the left side of the both eyes and positions of the right side of the both eyes. Then we used built-in function *math.sqrt()* which is responsible for finding square root, to find the distance between points of both eyes like horizontal distance as well as vertical distance. Then we find the average for horizontal distance and vertical distance and return both calculated average for further processing.

```
94 def lm_Distance(self,le_indices,re_indices,img):
95     #Left eye indices point
96     # horizontal line
97     lx1,ly1 = le_indices[0]
98     lx2,ly2 = le_indices[8]
99     #vertical line
100     vlx1,vly1 = le_indices[4]
101     vlx2,vly2 = le_indices[13]
102
103     #Right eye indices point
104     # horizontal line
105     rx1, ry1 = re_indices[0]
106     rx2, ry2 = re_indices[8]
107
108     # vertical line
109     vrx1,vry1 = re_indices[3]
110     vrx2,vry2 = re_indices[12]
111
112     #Calculating distance for verticle line
113     le_distance = math.sqrt((vlx1-vlx2)**2 + (vly1-vly2)**2)
114     re_distance = math.sqrt((vrx1-vrx2)**2 + (vry1-vry2)**2)
115
116     #Calculating distance for horizontal line
117     leh_distance = math.sqrt((lx1-lx2)**2 + (ly1-ly2)**2)
118     reh_distance = math.sqrt((rx1-rx2)**2 + (ry1-ry2)**2)
119
120     # averg_distance fro horizontal line
121     horizontal_average = (leh_distance+reh_distance)//2
122
123     # Average for verticle line
124     verticle_averagDistance = (le_distance+re_distance)//2
125     return horizontal_average, verticle_averagDistance
```

Chapter 5

Conclusion and Future Work

5.1. Conclusion

The proposed system (drowsiness detection and focus control system) will detect the sleepy state of the driver and also check the driver focus to see the straight to the road and don't move head left and right for the long time because whenever driver see alongside the road or start conversation to sitting next to driver seat person, he starts seeing him and in that case focus from driving a car is divert and chances of hitting other vehicle or changing the side of driving to opposite side of driving which cause accident , the propose system detect drowsiness through distance between eyelashes if change occurs then system alert through the alarming, for the focus calculate distance of both edges of eyes whenever person see along the road the distance reduced and alarm start to alert the person in order to keep focus straight rather than seeing surrounding.

5.2. Future Work

We are living in digital era where changes in software system are necessary, we can't claim that our proposed system is complete solution

for rest of life. We will add more features to our project in the future for Drowsiness and focus control system. We will add the following features in our proposed system.

5.2.1. Integration

As we discussed hardware requirements in our document for the integration of our project. We use all the hardware which are mentioned in document. The hardware will be portable, easy to integrate and long lost in each and every vehicle.

5.2.2. Autopilot Mode

If a person not respond when system alert them due to drowsiness, after passing some time the system turn on autopilot mode present in the vehicle.

5.2.3. Night vision camera

As we know that our system is only work form the day time, system not detecting eyes in the dark, we will integrate system to night vision camera which can detect the eyes in the dark and detect the drowsiness and alert the person.

5.2.4. Auto ON OFF

When the vehicle start the system will automatically on and off when the vehicle stop rather than manually pressing on off button.

References

References

1. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3571819/>
2. Otmani S., Pebayle T., Roge J., Muzet A. Effect of driving duration and partial sleep deprivation on subsequent alertness and performance of car drivers. *Physiol. Behav.* 2005;**84**:715–724. [[PubMed](#)] [[Google Scholar](#)]
3. Thiffault P., Bergeron J. Monotony of road environment and driver fatigue: A simulator study. *Accid. Anal. Prevent.* 2003;**35**:381–391. [[PubMed](#)] [[Google Scholar](#)]
4. Fairclough S.H., Graham R. Impairment of driving performance caused by sleep deprivation or alcohol: A comparative study. *J. Hum. Factors Ergon.* 1999;**41**:118–128. [[PubMed](#)] [[Google Scholar](#)]
5. Ruijia F., Guangyuan Z., Bo C. An on-Board System for Detecting Driver Drowsiness Based on Multi-Sensor Data Fusion Using Dempster-Shafer Theory. Proceedings of the International Conference on Networking, Sensing and Control; Okayama, Japan. 26–29 March 2009; pp. 897–902. [[Google Scholar](#)]
6. Vural E. Sabanci University; Istanbul, Turkey: 2009. Video Based Detection of Driver Fatigue. Ph.D. Thesis, [[Google Scholar](#)]
7. Ingre M., Åkerstedt T., Peters B., Anund A., Kecklund G. Subjective sleepiness, simulated driving performance and blink duration: Examining individual differences. *J. Sleep Res.* 2006;**15**:47–53. [[PubMed](#)] [[Google Scholar](#)]

8. Simons R., Martens M., Ramaekers J., Krul A., Klöpping-Ketelaars I., Skopp G. Effects of dexamphetamine with and without alcohol on simulated driving. *Psychopharmacology*. 2012;**222**:391–399. [[PMC free article](#)] [[PubMed](#)] [[Google Scholar](#)]
9. Das D., Zhou S., Lee J.D. Differentiating alcohol-induced driving behavior using steering wheel signals. *IEEE Trans. Intell. Transport. Syst.* 2012;**13**:1355–1368. [[Google Scholar](#)]
10. Mets M.A.J., Kuipers E., de Senerpont Domis L.M., Leenders M., Olivier B., Verster J.C. Effects of alcohol on highway driving in the STISIM driving simulator. *Hum. Psychopharm.* 2011;**26**:434–439. [[PubMed](#)] [[Google Scholar](#)]
11. <https://medium.com/axinc-ai/blazeface-a-machine-learning-model-for-fast-detection-of-face-positions-and-key-points-5dcfb9429d72>
12. https://google.github.io/mediapipe/solutions/face_mesh.html
13. <https://towardsdatascience.com/facial-mapping-landmarks-with-dlib-python-160abcf7d672>
14. Rajasekar.R, Vivek Bharat Pattni, S.Vanangamudi “Drowsy driver sleeping device and driver alert system”, IJSR, Vol.3 Issue4,2014.