DRIVER ALERT AND BEHAVIOR MONITORING SYSTEM USING VIDEO ANALYSIS

KEVIN CHEW

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

DRIVER ALERT AND BEHAVIOR MONITORING SYSTEM USING VIDEO ANALYSIS

KEVIN CHEW

This report is submitted in partial fulfilment of the requirements for the degree of Bachelor of Electronic Engineering with Honours

Faculty of Electronic and Computer Engineering
Universiti Teknikal Malaysia Melaka

2019/2020

DECLARATION

I declare that this report entitled “Driver Alert And Behaviour Monitoring System Using Video Analysis” is the result of my own work except for quotes as cited in the references.

Signature :

Author : KEVIN CHEW

Date : 23 JUNE 2020

APPROVAL

I hereby declare that I have read this thesis and in my opinion this thesis is sufficient in terms of scope and quality for the award of Bachelor of Electronic Engineering with Honours.

Signature : .....

Supervisor Name : DR ZARINA BINTI MOHD NOH

Date : 1st July 2020

DEDICATION

Special dedication to my loving parents, Iely Wong and Chew Kong Min, my kind hearted supervisor, Dr Zarina Binti Mohd Noh and thanks to my dearest friends.

ABSTRACT

Over the years, the rate of road accidents was seemingly increased. With this, road accidents had become one of the major causes of human death. According to data of research, one of the major cause of road accidents are approximately 1.35million deaths every year which were caused by road accidents and 20 to 50 million non-fatal injuries. NHTSA research shows that one of the main cause of road accidents is human error. To counter this problem, a driver monitoring system is being introduced. A driver monitoring system is a system which can detects the behaviour of the driver and alert the driver corresponding to their behaviour. In this project, a driver monitoring system is developed using image processing technique. This system mainly detects drowsy behaviour and yawning behaviour of the driver. The system will alert the driver differently depending on their behaviour. Besides that, the system is also integrated with IoT technology which the database of the system is connected to a cloud storage so that any inappropriate behaviour will be recorded and sent to the clod storage. The accuracy of the system is tested to be 92% accurate for detecting drowsy behaviour and 100% accurate for detecting yawning behaviour. Even though the system yields high accuracy in detecting the driver's inappropriate behaviour, various improvement can still be implemented onto the system for better performance.

ABSTRAK

Selama bertahun-tahun, kadar kemalangan jalan raya semakin meningkat. Dengan ini, kemalangan jalan raya menjadi salah satu penyebab utama kematian manusia. Menurut data penyelidikan, kadar kematian kemalangan jalan raya adalah sekitar 1.35 juta kematian setiap tahun dan 20 hingga 50 juta kecederaan tidak membawa maut. Penyelidikan NHTSA menunjukkan bahawa salah satu penyebab utama kemalangan jalan raya adalah disebabkan oleh kecuian manusia. Untuk mengatasi masalah ini, sistem pengawasan kelakuan pemandu diperkenalkan. Sistem pengawasan kelakuan pemandu bertujuan untuk memantau kelakuan pemandu dan memberi amaran kepada pemandu yang mamandu secara bahaya. Dalam projek ini, teknik pemprosesan bayangan telah digunakan untuk untuk mengesan kelakuan manusia. Sistem ini boleh mengesan kelakuan mengantuk dan kelakuan yang menguap. Sistem ini akan memberi amaran yang berbeza kepada pemandu bergantung pada kelakuan mereka. Sistem ini telah diuji dan didapati 92% tepat untuk mengesan kelakuan mengantuk dan 100% tepat untuk mengesan kelakuan menguap. Walaupun sistem ini mendapat ketepatan yang tinggi dalam mengesan tingkah laku pemandu yang bahaya, pelbagai peningkatan masih boleh dibuat untuk meningkatkan prestasi sistem ini.

ACKNOWLEDGEMENTS

As the 4th year student in Universiti Teknikal Malaysia Melaka (UTeM), I would like to thank those who were always motivate, support and guide me along my final year project accomplishment. I would first like to thank my final year project supervisor Dr Zarina Binti Mohd Noh. The door to Dr Zarina's office was always open whenever I had doubt and questions about my project. She consistently steered me to the right path and always give effective suggestion to help me achieve better goals of research. Furthermore, I would like to give gratitude to Dr. Radi Husin bin Ramlee who as my panel that evaluate my thesis paper. I appreciate his suggestion and comment after the seminar that helps me improve my research. In addition, I also want to thank to my friends who always be by my side to help me whenever I faced any problem and troubles. Finally, I would like to give my deepest appreciation to my family that always give me strength, confidence and support throughout my university's life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

For examples:

DMS	:	Driver Monitoring System
CCD	:	Charge-coupled Device
LED	:	Light-emitting Diode
NHTSA	:	National Highway Traffic Safety Administration
ADAS	:	Advanced Driver Assistance System
IoT	:	Internet of Things
IR	:	Infrared
ASM	:	Active Shape Model
PCA	:	Principle Component Analysis
SVM	:	Support Vector Machine
PERCLOS	:	Percentage of Eye Closure
EAR	:	Eye Aspect Ratio
PVT	:	Psychomotor Vigilance Task
IDE	:	Integrated Development Environment
OS	:	Operating System
SSH	:	Secure Shell
FPS	:	Frame per Second

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

Ever since car was developed in the early 1800, it had become the main transportation tool for mankind. At this era, almost every family owns at least a car. Aside from transporting people, vehicle such as van and trucks are used to transport items. Even though cars had brought a lot of convenience for us, they also put ourselves in danger if we handle it carelessly. Accidents often happens as the amount of vehicle on the road is continuously increasing. The cause of accidents might be varied depends on the situation. We can't control other people action but we can do our part in reducing the accident rate by driving safely. No matter how careful we are, there are times when we are careless. One of the most common of careless behavior of a driver is drowsy driving. We tend to lose focus on the road when we are sleepy.

This project focuses on developing a system which able to detects human's behavior which might potentially cause road accidents while driving. The system works by recording the driver's behavior using a camera and then process the video to determine their behaviors. When the driver poses dangerous behavior such as closing their eye due to fatigue or turning their head away from the road, the system will alert the driver. Road accidents often occurs due to the reckless behavior of driver. Some behavior was intentionally committed such as speeding, drug and drunk driving, but some behaviors were not intended. For example, some drivers start closing their eye slowly due to fatigue and some were subconsciously distracted. While some cars like the Toyota and Lexus had already have this system embedded onto their vehicle [1], most of the cars still doesn't have this safety feature. So, in this project, the system uses a Raspberry Pi as its main central processing unit as it is capable of perform video analysis and can connects to a camera module. The prototype of this project is meant to be small and portable so it can be mounted on any cars.

1.1 Background of Project

The Driver Monitoring System (DMS) or Driver Attention Monitor is a safety system for vehicles introduced by Toyota in year 2006 [1] . In the early year, the DMS is equipped with a CCD camera which is placed on a steering and able to tracks driver's eye using infrared LED detectors. The system will alert the driver when the driver is not paying attention on the road ahead. In year 2008, Toyota's system had developed a DMS with eye-monitoring feature [2]. Beside this, there are also a few other types of DMS technology. One of them which is the steering pattern monitoring which the system will monitor and recognizes the steering wheel angle as the steering pattern of the wheel is reflecting the driver's action.

Another type of DMS is the type which did physiology measurement on the driver [3]. This type of system requires a lot of sensor embedded onto the car to monitor the driver. The monitored criteria include heart rate, body temperature, brain activity, muscle activity and much more. The latest technology used in DMS is of course the system which monitors the driver's behavior using a camera and an image processing computer. Image processing is a technique or a method to perform certain operation on an image to extract useful information. In this context, the DMS will be attached with a camera which is placed facing the driver's face in order to capture and record their behavior. This type of system mostly detects drowsiness and distraction. The camera will record the video and image into to central processing unit and the facial data will be extracted and processed to determine the driver's current behavior. This system will normally attach with an alert function which comes in either a sounding alarm or LED light blinking for notification.

1.2 Problem Statement

According to World Health Organization [4], there are approximately 1.35 million people die due to road traffic accident and crashes every year. More than half of total road accidents deaths are among vulnerable road users such as cyclist, motorcyclist and pedestrians. This is due to the low safety their vehicle provides. Most of the cause of these cases involves a car or truck. The rate of casualty from car and trucks are lower because their vehicles are equipped with better safety system.

Around 93% of the world's total fatal cases on the road occurs in country that have low and mid-income. Besides, even road accidents didn't cause death injury, there are still 20 to 50 million people suffer from non-fatal injury due to road accidents. Among these victims, many of them had suffer from permanent disability. According to

National Highway Traffic Safety Administration, more than 94% if serious road accidents are caused by human error. Mistakes that had been made to contribute in increasing the total number of road crashes includes drowsy driving, distracted driving, drunk driving and others [5]. Most of these cases can be prevented if the driver is alert of their own behavior. Over fatigue while driving will cause driver to lose focus on the road and not paying attention on their surroundings. This might cause the driver to bump into road obstacles of worst cases, people. Same thing can happen when the driver is distracted while driving. The source of distraction might come from anywhere. The most common type of distraction is when the driver is adjusting the rear mirror, answering the phone and adjusting the radio. To solve these problems, a Driver Alert and Monitoring System is introduced.

A driver monitoring system should be able to distinguish between safe and dangerous behavior the driver poses to implement further action. There are a lot of safety technology which have the same function which include steering wheel angle sensors, outward facing cameras and multiple sensors attached to the vehicle. There are some disadvantages for these type of system in which the data input for these systems are more than a camera based driver monitoring system. They either require a lot of sensors with along with data fusion to determine the safety level or the result obtained might not be accurate due to surrounding factors. Huge amount of input data may introduce more error as each data obtained might contains errors. So, to solve this problem, a system with less input data is introduced to reduce the error and increase the processing speed for analytics and algorithm. The Driver Alert and Monitoring System only detects and extract the human's facial feature for its input [6]. Real time image processing technique is used to analyze the input data to determine the type of behavior.

Accuracy and response speed are importance when it comes to safety issue. There are several techniques which can be used in in image processing for determining the driver's behavior. Different type of technique used may result in different accuracy and response time for the system. So, in order for the system to applicable on the vehicle, the system must be accurate and fast in detecting the driver's behavior. The response time and accuracy of the system must be analyzed in order for it to comply for the situation. An air bag safety system is set as a comparison as both of it serves the same safety purpose.

1.3 Objectives

The objectives of this project are:

- To develop a system which can monitor and determine driver's behavior and alert the driver when they pose dangerous behavior such as yawning and drowsy symptom.
- To analyze the accuracy and decency of the system based on the data collected.
- To implement IoT into the monitoring system in order to enable real-time monitoring.

1.4 Scope of Project

This Driver Alert and Monitoring System in this project is meant to be compatible for all vehicle, so Raspberry Pi 4 model B is used as the main processing unit as not all cars have their own Advanced Driver Assistance System (ADAS) [7]. The prototype should be portable and able to install on all type of vehicles. A camera module is attached onto the Raspberry Pi to record the video of driver. A wired camera module is chosen compared to a USB camera or a wireless webcam because the

camera module can transfer data faster compared to the other two [8]. The camera module is installed directly onto its camera port thus increasing the data transferring speed. The camera should be able to record the driver's face directly on a certain angle to ensure the captured data is accurate. The Raspberry Pi will act as the main processing unit to process the input data from the camera. The programming language used is Python [9]. The framework used in this system is the OpenCV [10]. OpenCV is a library of programming function which used mainly for real-time computer vision, so it is suitable for the real time video analysis task in this project. OpenCV is used to locate and detects the driver's face.

1.5 Report Outline

Chapter 1 is the introduction chapter for this project. This chapter will cover the terms, background and brief explanation of this project. The problem statement based on case studies along with the objectives to achieve would also be highlighted in this chapter along with the scope of work for this project.

In Chapter 2, background study includes a detailed review of researched area. Past research and history on the problem stated are being studied. Current information and solution regarding the issue were also indicated in this part.

In the methodology chapter, research methods that will be implemented in this project is described in detail. Flow chart and graph will be included to portrait a more detailed and understandable concept. Each method used are categorized in different section for the ease of describing different steps involved in succeeding different objectives.

For Chapter 4, the title of this chapter is result and discussion. In this chapter, the illustration of the final prototype will be presented and each detail of the prototype is described and discussed. The outcome of the system would also be included and each data will be discussed.

The final chapter portraits the detailed summary of the whole project. The conclusion section will describe the summary of methods implemented and the analysis method together with results obtained. As for the future works section, possible future improvement and further development of the current project will be discussed.

CHAPTER 2

BACKGROUND STUDY

2.1 Related Projects

Through research and survey on the Internet, few similar research papers which are related to this project were found. After studying those papers, some summary and comparison between those papers were done.

Firstly, in the research paper titled “Driver Monitoring System for Automotive Safety” [11], a sensor base driver monitoring system is introduced. The system consists of three pressure sensors, a light sensor and a microcontroller. The light sensor is used to determine daytime or nighttime during driving session. The three pressure sensors are positioned onto the steering wheel, armrest and gearshift. In this research, pressure sensor FSR400 is used. These sensors operate by sensing forces acting on it. This operation can be compared with the operation of a resistive potentiometer which

changes its value proportion to the force applied. The microcontroller will read the changes in the resistor in the form of analog input and determine the force applied. This sensor is said to hold a precision of 10%. When force is applied to the sensor, the resistance would decrease. These three sensors embedded onto the steering wheel, armrest and gear transmit data to the microcontroller regarding of the position of the driver. This system operates by sensing the position and force applied by the driver hand while driving. In this system, two pressure must be activated (applied force). If less than two pressure sensor is activated, the system will alarm the user through a buzzer. When the driver weakens their grip on the steering wheel due to fatigue, the alarm will also goes off after a short time alerting the driver. Figure 2.1 shows the algorithm of the system.

Figure 2.1: Algorithm of the system [11].

Secondly, in the paper "Intelligent Driver Monitoring and Vehicle control system"[12]. A IoT base driver monitoring system embedded with multiple sensor is made. Details such as drowsiness and alcohol consumption is collected and sent to a

particular webpage. An Arduino is used as the central processing unit for this system as the microcontroller is open source and easy to obtain. The Arduino will collect output from the sensor and determine the action required. Other sensor which are used in this system are ultrasonic sensor, alcohol sensor and infrared sensor. The ultrasonic sensor is used to measure the distance between vehicles. When a vehicle is getting nearer to the driver's vehicle, the buzzer will ring alarming the driver. This detail can be used to determine the fault of driver when accident occurs. For the alcohol sensor, model MQ3 is used. This sensor will detect the alcohol level in the car. When the alcohol level exceeds a certain limit, the engine of the vehicle will turn off preventing the driver from driving the vehicle while drunk. The infrared sensor is used as an eye blink detector. The infrared (IR) sensor is placed on a glass worn by the driver. The IR transmitter and receiver are connected onto the frame of the glass. When the driver closes their eyes for several seconds, the sensor will send a signal to the Arduino and the car will be stopped. All of the data collected by the Arduino will be sent to a particular webpage for further reference. An ESP8266 Wi-Fi module is used as it is compatible for the Arduino to get internet connection. Figure 2.2 shows the block diagram of the system.

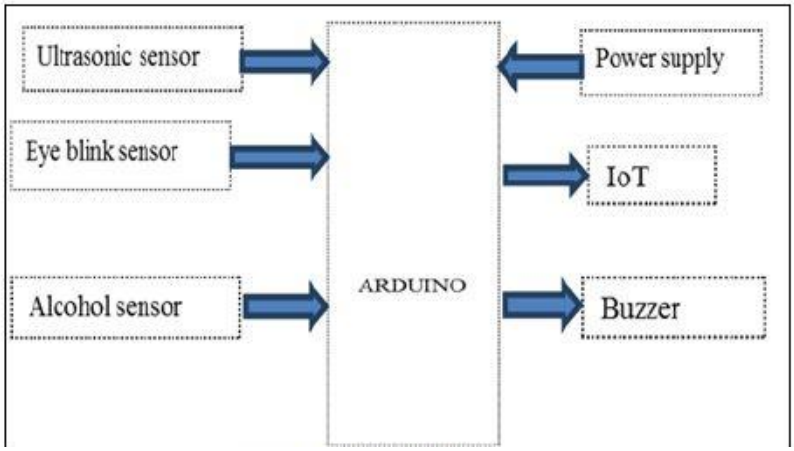

Figure 2.2: Block diagram of the system [12].

In the next paper, in a research titled “Yawning Detection Using Embedded Smart Camera” [13] ， a driver monitoring system using smart cameras was developed. This system focuses on detecting driver’s fatigue by yawning detection. The system first detects the person’s face, then it will proceed to detect the person’s mouth. To detect the human face, the researchers used Viola Jones algorithm. This face detection method has been implemented in the OpenCV software library. The Viola-Jones theory is able to process image very rapidly with high detection rate. There are three technique involved in this method. The first is an integral image for fast feature evaluation and decreasing feature complexity for each frame. The second technique is a process of creating a classifier. The last technique is a method of combining classifiers in Cascade structure. The Viola Jones method is not efficient enough to run real time detection on camera. To overcome this, they introduced a fast and memory efficient face detection algorithm based on Viola-Jones. Firstly, data of the trained features in each node are extracted and stored in five different files in the smart camera. This will help save the computation time as these files will be used in detection algorithm. This way, the monitoring system can utilize the saved values instead of training the classifier and applying the integral image to find the features from the beginning. Each of the five files contains one of the following data:

i. Feature Coordinate

ii. Feature Threshold

iii. Feature value

iv. Stage classifier

v. Feature weight

For face detection, the system the develop is able to detects faces with different sizes. Furthermore, the system can lock-on to the subject's face when the driver's face is detected instead of keep on searching for new faces. This can be useful when there are multiple faces captured by the camera in one frame. For mouth detection, similar method was used but the lower half of the face is chosen for search region. After detecting mouth, the last step is to detect yawning. The first step is to convert the image captured by the smart camera into grayscale image. Then, the histogram of grayscale image is obtained through counting the number of occurrence of each gray level. The histogram of the close mouth position in the first frame will saved as reference frame for further calculations. Back projection theory is used to determine yawning. In order to calculate back projection, the histogram of reference frame is computed and compared with the following frames. In this context, the calculated new mouth histogram is compared with reference mouth histogram. The system will detects the driver as yawning when they satisfy two condition which is when the ratio of the number of black pixels of reference and current frame is greater than a certain threshold and ratio of black pixels in the mouth region and white region is greater than another threshold.

The next paper which is titled "Intelligent Driver Monitoring System Using Camera" [14], Active Shape Model(ASM) [15] was used to detect the driver's behavior. The ASM are statistical models of shape of objects which has been deformed to fit a new example of object in a new image. The shapes are constrained by the point distribution model (PDM). PDM are models representing the mean geometry of a shape and some statistical modes of geometric variation inferred from trained shapes. In ASM, the shape of an object is represented in a form of a set of points. The ASM algorithms aims to match the target model to a new set of image. The ASM works by

generating a suggested shape by surveying around each point on the image for a better position. Then the ASM will conform the suggested shape to the point distributed model. With featured based approach, the system will process the input image and identify plus extract the targeted facial feature such as lips, eyes, nose and mouth. After that, the system will compute the geometric relationships between those facial points to reduce the input facial image to a vector of geometric feature. When the driver is in front of the camera, the ASM will be used to locate the driver's eyes from the front-view camera image. The ASM algorithm will find out the position and scale features of the driver's face. The system will automatically create a rectangle box around the human face when the driver's face is detected. After the facial features has been extracted, the triangle areas of these features are evaluated. When the driver turns away from the camera, the alarm will be activated. Figure 2.3 shows the flow chart of the approach.

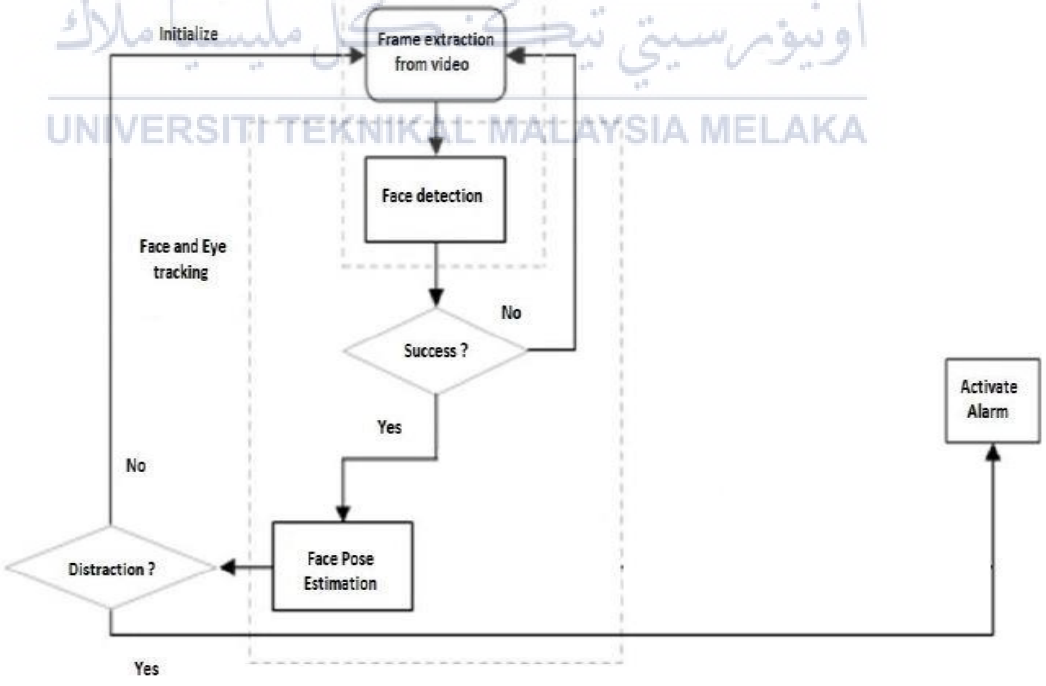

Figure 2.3: Flowchart of the approach [14].

Aside from these, in the journal "A Real-Time Monitoring System for the Drivers Using PCA and SVM" [16], the author designed an economical monitoring and detection system for drowsy drivers. The main idea in this journal is to design a system with a simple wireless camera installed directly facing the driver and using MATLAB system as the processing system. The system will be able to monitor the face of driver and tracks the eye level to determine the level of drowsiness. Viola-Jones algorithm will be used to detects the driver's face and eyes [17]. Principal components analysis (PCA) and support vector machine (SVM) is used for eyes close estimation [18].

The first step of the proposed system is face detection from video obtained. This task will be accomplished by Viola-Jones object detection framework using Haar-like features [16]. When capturing video in real life scenario, vary levels of brightness and pose variation together with motion blur might affect the processing speed of the system. These problems will affect the quality of the face video captured and affect the result. To overcome this problem. PCA and SVM is introduced. These methods will aid the searching of feature searched, image frame, Euclidean Distance and facial expression in the video. So when the system starts, the camera will start recording the facial video of the driver. Detector API of Viola-Jones algorithm from MATLAB is used to detect face in the image. When face is detected, another Cascade Object Detector is used for eyes detection. These extracted data will be sent to PCA and SVM. In PCA, the system will first get a database of a set of image and then find the mean of the images.

Secondly, the difference between the mean image and each image in the database is compared and recorded [16]. Then, the covariance matrix of data from previous step is obtained. Next, the system will start obtaining the Eigen value and Eigen

Vectors to find the Eigen faces. After obtaining, the Eigen faces, it will proceed to get the eigen vectors. When unknown image was found, previous step will be echoed to get the weight vector for test image. Lastly, it will find the Euclidian distance between weight vectors of unknown image with database images. If the distance obtained is within a threshold, the test image will consider to be in the database. Otherwise it will be unauthorized. In Figure 2.4, the flowchart of the system is portrayed. Based on the flowchart, drowsiness and fatigue will be detected based on the image's frame mean of trained video that is stored inside the system. The system will continuously check for the match of facial expression stored in the database. Drowsiness will be detected according to the image frame. The alarm will turn on once drowsiness is detected.

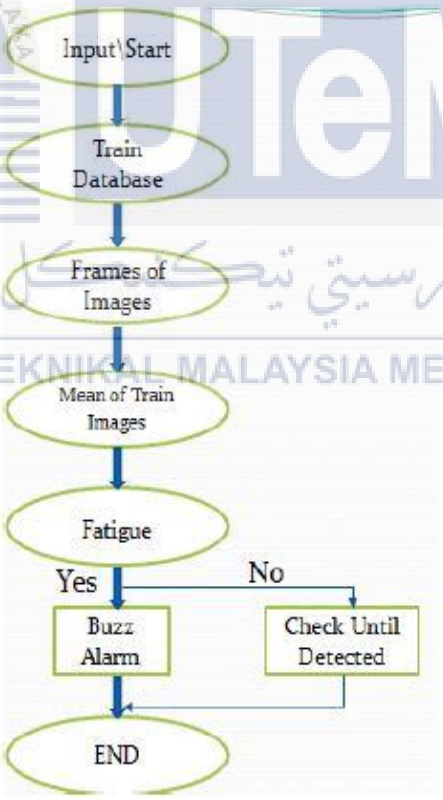

Figure 2.4: Flowchart of the proposed system [16].

In the paper “Driver Drowsiness Detection Based on Face Feature and PERCLOS” written by Suhandi Junaedi and Habibullah Akbar PERCLOS estimation is used along with iris area estimation[19]. The overall process starts with video input, detect face, detect eyes, extract iris region, morphological operation and finally PERCLOS estimation. The input image is the videos of drivers with various condition. Each frame of the video is processed with a few process namely histogram equalization, face and eye detection, iris region extraction, morphological filters, iris area estimation, PERCLOS measurement and finally dataset analysis.

Histogram equalizing is performed because the video is taken from a real life situation which had various illumination condition [19]. The input pixel of each frame is transformed into a new intensity value based on the scale factor of cumulative histogram. Viola-Jones method is used for face and eye detection here. For iris region extraction, the circular Hough transform was used to isolate iris which are darker than other areas od the eyes with a circular shape. When Hough algorithm locate the iris, the edge detection method by Sobel is then implemented to detect the boundaries of iris project.

After that, several morphological filters was used [19]. Firstly, more pixels were added to the boundary of the iris object through dilation operator. Then, flood-fill operation were used to fill holes and reconstruction of iris shape. Moving on, erode operator is used as a inverse dilation process to reduce the size of iris area. After obtaining the iris, Hampel identifier is used to ensure no noise present in the iris region before the iris is estimated.

Lastly, PERCLOS measurement is used. PERCLOS is defined as the percentage of the total frame that eye is closed during a certain time interval. The PERCLOS is

calculated based on the iris's area estimation. In this paper, left iris, right iris and total value of both iris is extracted. Three threshold (60%,70% and 80%) were used and tested in this study. Equation 2.0 shows the calculation of PERCLOS percentage [19]. Figure 2.5 illustrates the flow chart of the proposed drowsiness detection method used

$$PERCLOS(\%) = \frac{sum\ of\ frames\ when\ the\ eyes\ is\ closed}{intrval\ of\ frames}\ x\ 100 \qquad 2.0$$

.

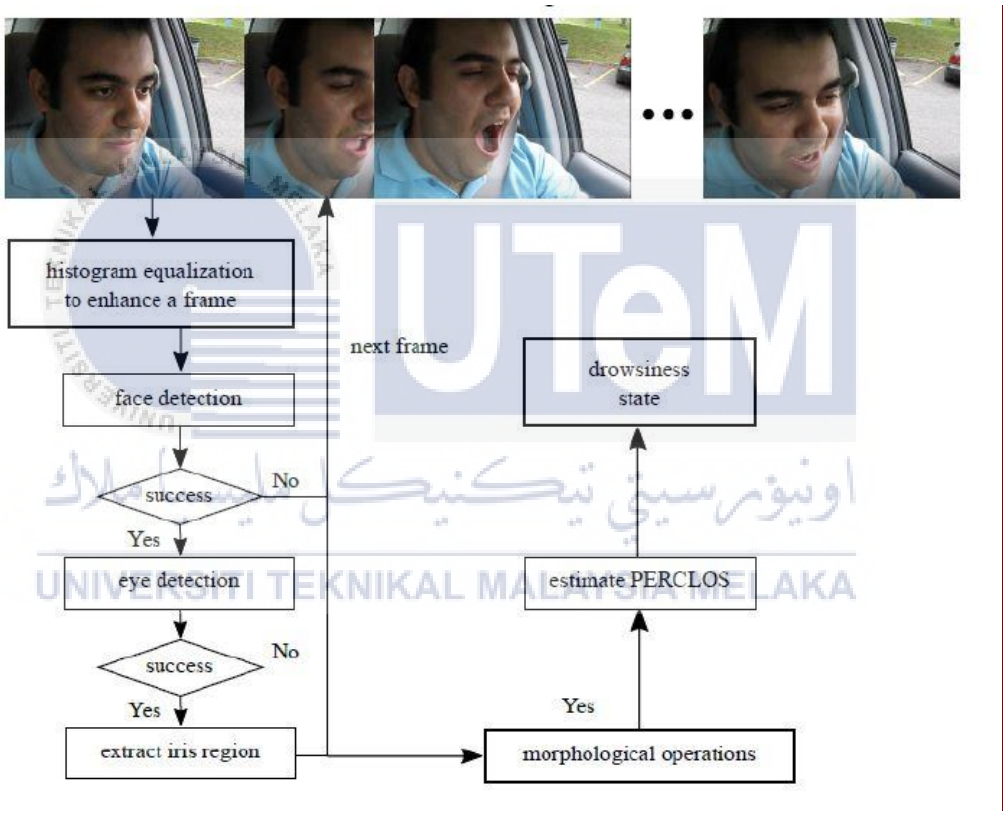

Figure 2.5: Proposed method's flow chart [19].

2.2 Theory

2.2.1 OpenCV Face Detection

OpenCV is a library of programming which is mainly used for image processing. It is available for free on the open source Berkely Software Distribution license [20]. OpenCV was started as a research project by Intel. It contains higher level of algorithms for face detection, feature matching and tracking. It also contains low level image processing function. There are a few main image processing techniques which were used namely image filtering, image transformation, object tracking and feature detection.

Image filtering is a technique used for modifying or to enhance an image [20]. There are two types of image filtering which are linear filtering and non-linear image filtering. In linear image filtering, the value of an output pixel is a linear combination of the value of pixels of the input pixel's neighbor and vice versa for non-linear image filtering.

Image transformation is a technique used to create a new image from multiple source which highlight a few particular properties of interest which is better than the original image [20]. Simple arithmetic operations are applied for basic image transformation. Changes that occurs between images collected from different timestamp are identified using image subtraction.

Object tracking is a process of locating one or multiple object over an array of image. It is the most important component in various application in computer vision [21]. As for feature detection in computer vision, feature is also described as an "interesting" part of an image and is used as a starting point for a lot of computer vision algorithm. Feature detection is a process to find a specific feature of visual

stimulus such as angle, lines or edges. It is very helpful for making local decisions about the information contents is an image.

Face detection is where an input image is searched to locate a face and then the image is processed to extract the facial detail. OpenCV has a face detector named " Haar Cascade classifier". Feeding an input image, either from a camera or video, the face detector will examine and process the image to classify whether it contains faces or not. Figure 2.5 shows the face detection using OpenCV [20].

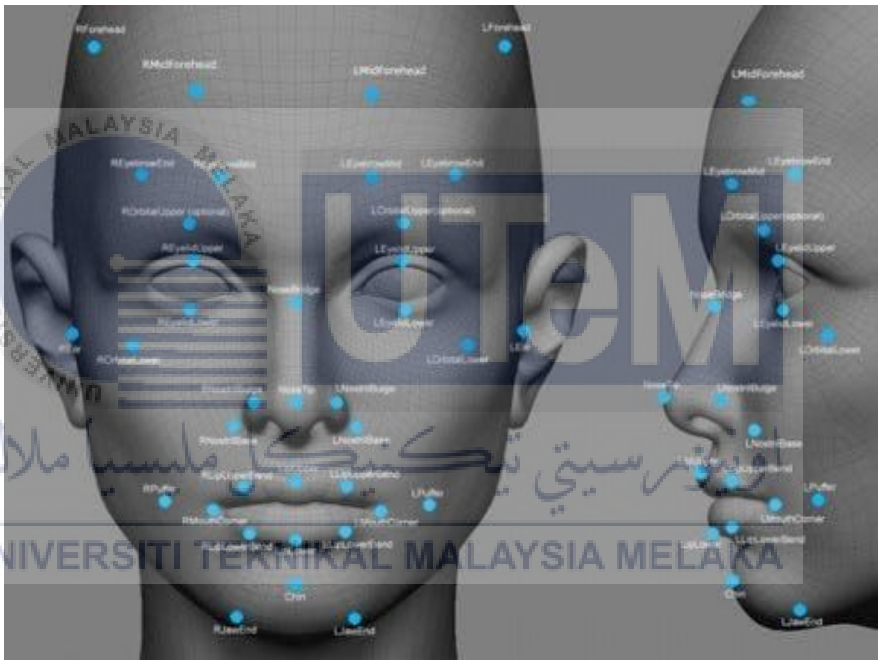

Figure 2.6: Face detection using OpenCV [20].

2.2.2 Haar Cascade

Haar cascade [22] is a machine learning object detection algorithm which is proposed in 2001 by Paul Viola and Michael Jones. They adapted the idea of Haar Wavelets and developed the Haar-like features. Initially to detect object or faces, we were directly computing the image's pixels. In face detection [22], the human's face possess face that have quite similar properties which are nose, eyes, lips and ears. The

relative size and contrast of them are similar for every human being. This uniformity of information and features can be replicated using Haar-like feature. The Haar-like feature consists of adjacent rectangular windows at specific location. It will amplify the pixel intensities in every region and calculates the differences between the regions. The output value will help categorize the specific location on the human face. Figure 2.7 shows a sample of Haar-like feature in face detection.

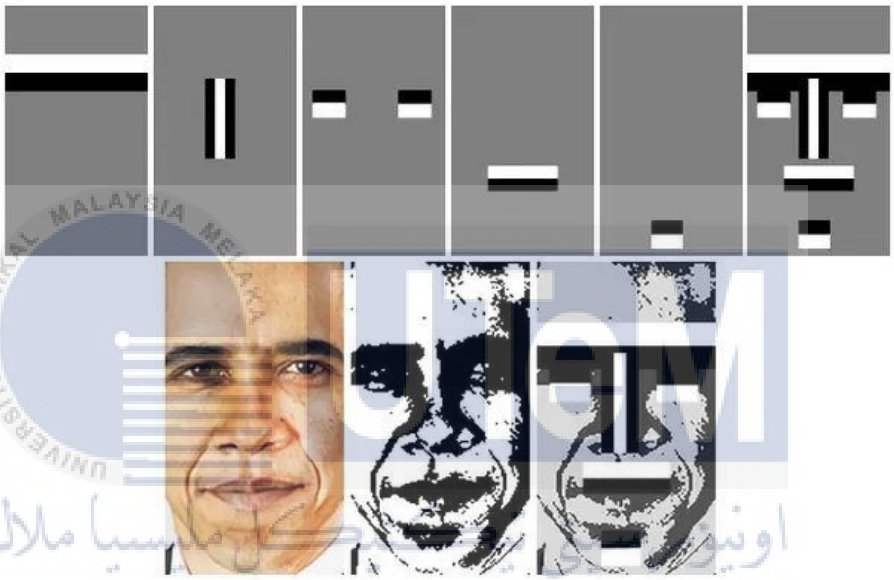

Figure 2.7: Haar-like feature in face detection [21].

2.2.3 Viola-Jones Algorithm

Concept of Haar-like feature is used in Viola-Jones algorithm [17]. For face detection, Viola-Jones algorithm goes through three stages. These three stages are:

1. ***Computation of Integral Images:***

   This process uses the concept of Haar-like feature but it computes the features through the concept of integral image. Integral image is a

concept of Summed Area Table where both data structure and an algorithm are used effectively to calculate the sum of values in a rectangular grid. Thus integral image calculates the Haar-like features in constant time. The integral image at certain location (x,y) given is the sum of pixels above and to the left of that coordinate. The formula for Summed Area Table is given in equation 2.1.

$$I(x,y) = \sum_{\substack{x' \le x \\ y' \le y}} i(x',y') \qquad 2.1$$

Where *i(x,y)* is the value of pixel at coordinate x and y.

2. ***Usage of Adaboost Algorithm***

As the number of features computed in first step is huge, Adaboost algorithm is used to select features which we are interested in. This will help us to detect the object more accurately. Adaboost Algorithm will select principle features to train the classifier that would be used. The main objective of this is to create a stronger classifier from a linear combination of weaker classifier.

3. ***Creating Cascade Structure***

The cascade structure is made up of classifiers. This structure works by using the initial classifier is used to reject majority of sub-windows at the beginning of the process. At the end, a more complex classifier was used to achieve a low false positive rates. These classifiers are trained with the Adaboost Algorithm.

Even though Viola-Jones algorithms are vastly used in object detection, there are still some disadvantages exist despite of the advantages it provides. Table 2.1 show the advantages and disadvantages of Viola-Jones Algorithm [17].

Table 2.1: Advantages and disadvantages of Viola-Jones algorithm

Advantages	Disadvantages
Extremely fast and efficient in feature selection	Less effective in detecting non-frontal object in images.
Capable of scaling the features in an image	Varies results when detecting image under different light condition
Can be used to detect various type of object	Multiple calculation and computation of same parts in an image may occurs.

2.2.4 Eye Blink Detection

The eye blink is a fast closing and reopening of human's eye. Each individual eye blink's speed is vary. The patterns differs in the speed of them closing and reopening their eyes. The eye blink of human last approximately 100ms to 400ms. To detect eye blink, a state-of-the-art facial landmark detector is proposed to localize the eyes and eyelid contours, From the landmark of in the image, an algorithm is derived to estimate the closing and reopening of the eye. The proposed algorithm is named Eye Aspect Ratio (EAR)[23]. The equation of EAR is given as equation 2.2.

$$\mathrm{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|} \quad 2.2$$

Where p1 to p6 are the 2D landmarks on the human's eye. Figure 2.8 shows the image of facial landmark on human's eye. The left side is the landmark detected when human eyes are open and the image on the right is when the human's eyes are closed.

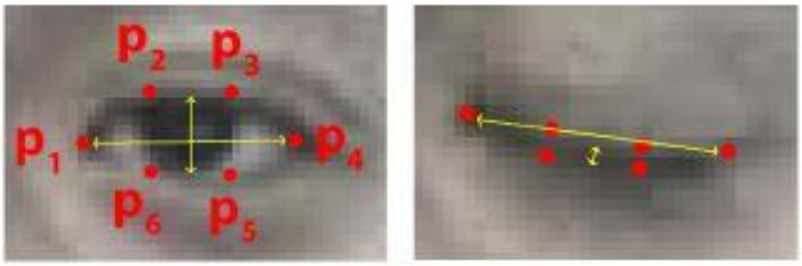

Figure 2.8: Facial landmark on human's eye [22].

From equation 2.2, the EAR is mostly constant when the eyes is open and gradually reduce to zero when the eyes is closing, The aspect ratio of the open eyes has a small variance on different individuals and is fully invariant to a uniform scaling of the image and in-plane rotation of the face.

2.2.5 Percentage of Eyelid Closure (PERCLOS)

In year 1994, Percentage of Eyelid Closure (PERCLOS) was introduced as alertness measure by Wierwille [24]. PERCLOS become the only one measure to measure alertness compared to other methods that were emerged later. After the publication of the Dinges report namely "Cumulative Sleepiness, Mood Disturbance, and Psychomotor Vigilance Performance Decrements During a Week of Sleep Restricted to 4-5 Hours per Night" few years after PERCLOS was suggested, PERCLOS had become the accepted standard for measuring alertness with Psychomotor Vigilance Task (PVT) as a label. PVT is defined as a sustained-attention

or reaction-timed task which measures the speed of a subject or a person respond to visual stimulus.

In a publication from the US Department of Transportation, it was stated that almost all technologies shows the potential for detecting drowsiness b predicting lapses in one or multiple subjects but only PERCLOS was able to correlated highly with PVT lapses within and between the subjects [24]. The study shows that PERCLOS had better performance compared to eye blink measure and head movement measure.

CHAPTER 3

METHODOLOGY

In this section, the procedures and technique which are used in this project is explained discussed. After doing some research, it is stated that Raspberry Pi is capable of performing computer vision image processing using Python. So, for ease of troubleshooting, all programming was done in Raspberry Pi's built in Integrated Development Environment (IDE).

3.1 Overall Project Methodology

Figure 3.1 illustrate the block diagram of the project. The camera used in this project is Pi Camera Model 1.3. The Pi Camera is responsible of recording and streaming live video into the Raspberry Pi for analysis. System used for driver monitoring will be written and compiled within Raspberry Pi. The Raspberry Pi will

process the video streamed in real time for monitoring purpose. If the driver is detected to be drowsy, the Raspberry Pi will capture the image at current time and upload to Dropbox cloud storage. At the same time, the speaker connected to the Raspberry Pi will send alert according to the driver's behavior.

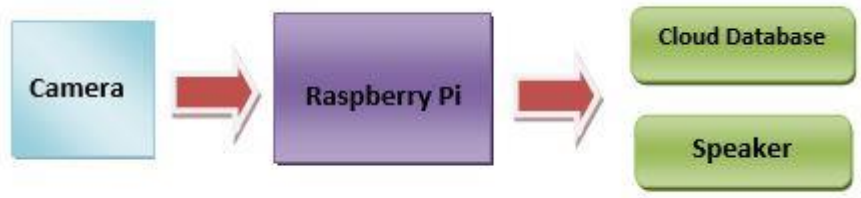

Figure 3.1: Block diagram of the system

Figure 3.2 show the procedure flow of the whole project. Various research were conducted before starting the project to get a better understanding about the objectives and to identify the problem. After doing sufficient research, the next step is to do the literature review of similar project. The aim for this is to understand and compare the similar research to understand which and what method used is the best for the project. When all required research is done, the next step is to plan the prototype. At first, the hardware for this project is determine. For this project, Raspberry Pi 4 Model B is chosen as it is cheap and capable of running computer vision image processing. Then, the suitable framework for machine learning is determined. Prototype development is proceeded when things are done here. For the development part, the first thing to do is to write the program for the system. The system is wrote using Python Programming in the Raspberry Pi's IDE. After finishing writing the program for the system, the prototype is tested and run through the troubleshooting process. When everything is done, the result obtained is analyzed for its accuracy.

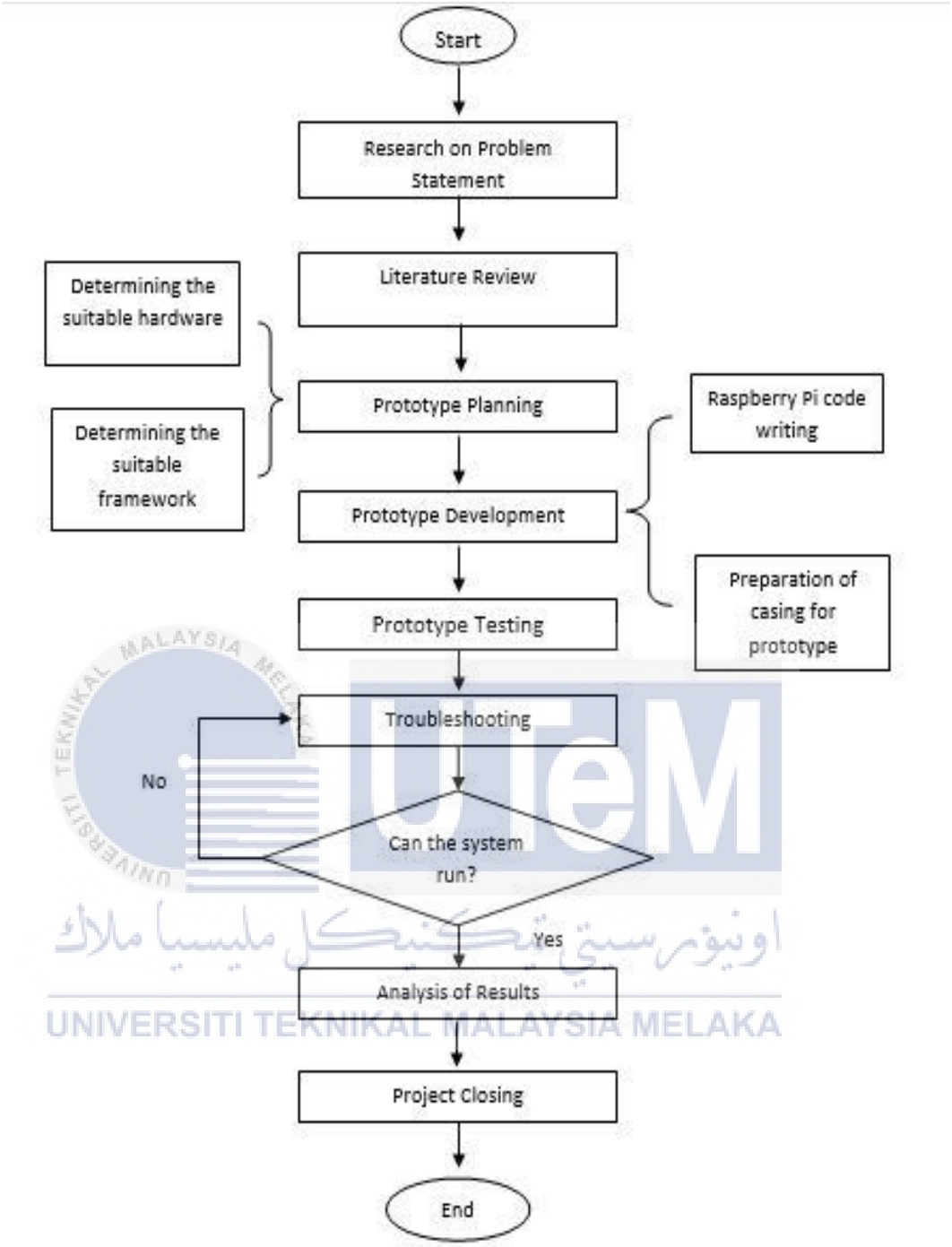

Figure 3.2: Flowchart of the project

3.2 Software Implementation

The first objective focuses in developing a program which can be run in Raspberry Pi for the driver monitoring system.

3.2.1 Raspberry Pi Setup

Raspberry Pi is used as the main processing unit in this project. In order to access to Raspberry Pi, a mouse, keyboard and a monitor is required. In this case, the Raspberry Pi is connected to laptop to use the laptop's screen, mouse and keyboard. Before connecting to the laptop, the SD card of the Raspberry Pi must be configured. An operating system (OS) must be installed in order for the Pi to work. In this project, Raspbian OS is chosen as it can work in headless mode which mean no monitor is required. To install the OS into the SD card, Win32DiskImager is used to write the Raspbian image file into the SD card creating a virtual disk drive.

After finish configuring the SD card, the next step is to connect the Pi to the laptop. This can be done through Virtual Network Computing. Virtual Network Computing is a graphical desktop sharing system which uses the Remote Frame Buffer protocol to control other computer remotely by transmitting the mouse and keyboard events from computer to one other relaying the graphical screen updates back to the other direction over a certain network. For this project, VNC Viewer is installed onto the laptop to gain access to the Raspberry Pi. In order for Raspberry Pi to connect to the laptop over a network, the laptop will share its internet through a LAN cable to provide internet for the Raspberry Pi. After that, Secure Shell (SSH) on Raspberry Pi must be enabled for first time connection between the laptop and the Raspberry Pi. When all are set, login as Raspberry Pi user is required in the SSH terminal, and a server is created for the connection. Figure 3.3 shows the interface of SSH terminal.

Figure 3.3: SSH interface

After accessing into the Raspberry Pi through the laptop, the next step is to download and update the Raspberry Pi's package files. Python3 programming is installed into the Pi as the default python version is 2.7. Python 3 is required in this project for better performance. When software of the Pi is done updating, the next step is to install the camera module in order for the Pi Camera to work. Then, the framework, OpenCV is installed. OpenCV 3.4.2 is used instead of the latest version 4.1.0 because it is more stable and less bug. Various library such as iMutils, DLib and NumPy was installed into the Pi for the system to run.

3.2.2 Python programming for Driver Monitoring System

This system mostly focuses on face detection together with eye level detection. For face detection, the Haar Cascade classifier is used. OpenCV is equipped with its own detector and classifier. So, all that need to do is to load the Cascade classifier from the project directory at the beginning of the program. As for eye level and yawning detection, the first step is to get the facial landmark through the Dlib's facial landmark predictor file. The predictor file will place 68 different (x,y) coordinate landmark onto

the human's face and from here, the eye's landmark is extracted and the eye level is calculated. Figure 3.4 shows the 68 facial landmarks on the face.

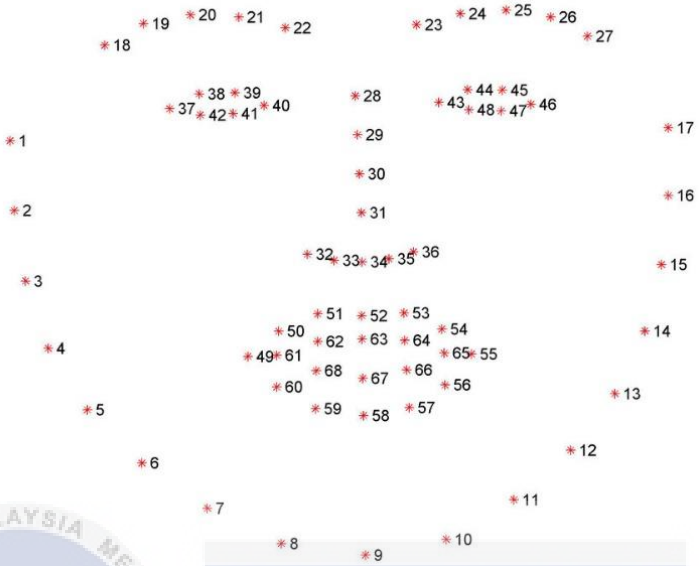

Figure 3.4: 68 Facial landmarks [23]

The eye level is calculated through the Eye Aspect Ratio function. How this algorithm work in this system is that the Euclidean distance between the 6 points on the eye calculated and the final ratio is obtained through the function shown in Figure 3.5. Figure 3.5 shows the function written in the python programming to compute the EAR.

```
A = euclidean_dist(eye[1], eye[5])
B = euclidean_dist(eye[2], eye[4])
C = euclidean_dist(eye[0], eye[3])
ear = (A + B) / (2.0 * C)
```

Figure 3.5: EAR function

As for yawning detection, yawning is characterized by the opening of the mouth. So from the 68 facial landmarks, a few coordinate to indicate the bottom and top lip were extracted. From the facial land marks, point 50,53 and 61,64 is extracted to be

the upper and lower part of top lip and point 56,59 and 65,68 is extracted as the upper and lower level of the bottom lip. Figure 3.6 shows the facial landmark of the lip and Figure 3.7 below shows the coding used to extract the coordinates of the lips.

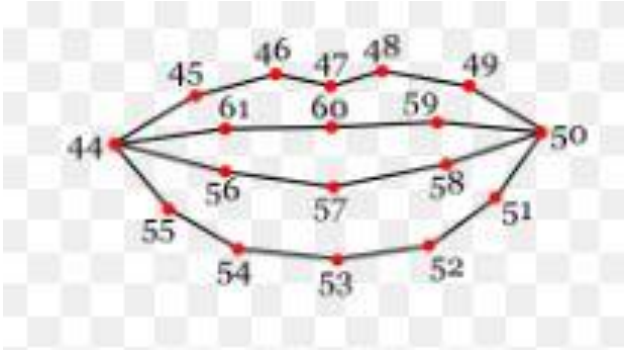

Figure 3.6: Facial landmarks of the lip [23]

```
def lip_distance(shape):
    top_lip = shape[50:53]
    top_lip = np.concatenate((top_lip, shape[61:64]))

    low_lip = shape[56:59]
    low_lip = np.concatenate((low_lip, shape[65:68]))

    top_mean = np.mean(top_lip, axis=0)
    low_mean = np.mean(low_lip, axis=0)
```


Figure 3.7: Extraction of lips coordinates

Then, the mean value of the top and bottom lip will be extracted using the algorithm from numpy. Lastly, the yawning level will be calculated based on the opening of the mouth by subtracting the top lip. mean value by the bottom lip's mean value. Figure 3.8 shows the line of coding used to determine the lip's distance.

```
distance = abs(top_mean[1] - low_mean[1])
return distance
```

Figure 3.8: Coding for lip's distance

To detect drowsiness, the eye aspect ratio's threshold is set to a value which the eye is near to complete close at this value. For each frame when the EAR is lower than the threshold value, the counter will start by adding 1 count. So, when the driver had close their eyes, the counter will start counting. If the driver closes their eye for more 2 seconds, an alert will goes off nudging the driver and the camera will capture the driver's picture at current time and send it to Dropbox storage. 2 seconds had been set as the threshold of drowsiness is based on the research done by B. Mohana and C. M. Sheela Rani in their research paper titled "Drowsiness Detection Based on Eye Closure and Yawning Detection"[25]. The counter will reset once the driver had opened their eyes or when the EAR computed by the system is more than the threshold value.

For yawning detection, the threshold is set after doing some analysis based on the data gathered on the following analysis part. So when the yawning threshold is exceeded by the driver, the alert will be triggered.

For the output result, a window will pop up showing the real time footage being recorded by the camera. The EAR value, yawning value and the fps of the video will be indicated on the output window. The EAR value and fps value will only start indicates when face is detected. Else when there is no face indicate, both the value will not be indicated and the Pi will be in a resting condition. Figure 3.9 shows the output window of the system. Figure 3.10 shows the overall flow chart of the system program's operation.

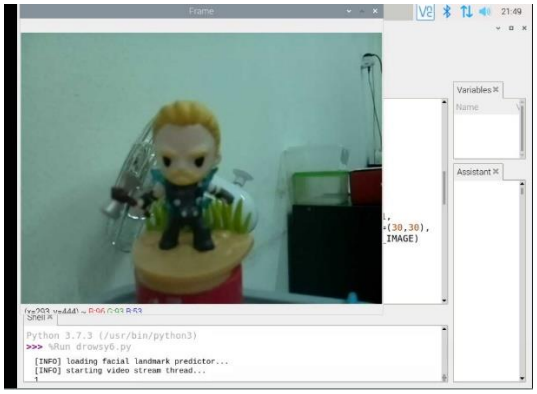

Figure 3.9: Output window

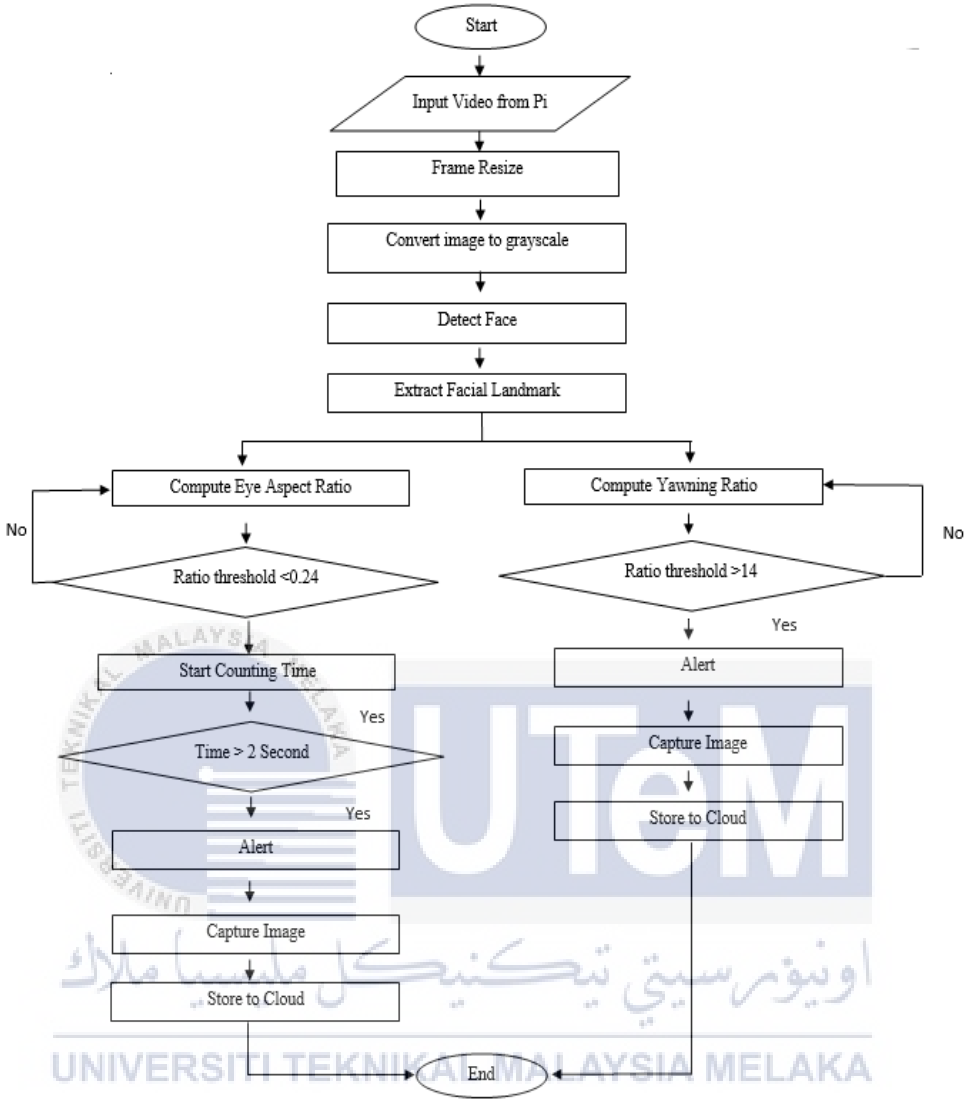

Figure 3.10: Flowchart of the program

3.3 Hardware Implementation

For the hardware of this project, Raspberry Pi 4 Model B will be the main processing unit. In this section, the process of the Raspberry Pi is divided into input, processing and output.

For the Raspberry Pi to work in full performance, 5V3A of power supply is required and USB Type-C Data cable is required for power transfer. The main input for this

system is the footage of the driver. For this case, Pi camera 1.3 was chosen to capture the footage of the driver. Pi Camera was used instead of webcam because Pi Camera is directly interfaced to the Raspberry Pi while webcam is attached to the Raspberry Pi through USB port. So, Pi Camera would provide a better performance compare to webcam. For the Pi Camera to work properly, it must be enabled using the raspi-config command along with navigation to enable the hardware part. Then, the camera is tested by using a simple code to capture an image for the purpose of ensuring that the camera is working properly. Figure 3.11 shows the image of Pi camera attached to the Raspberry Pi.

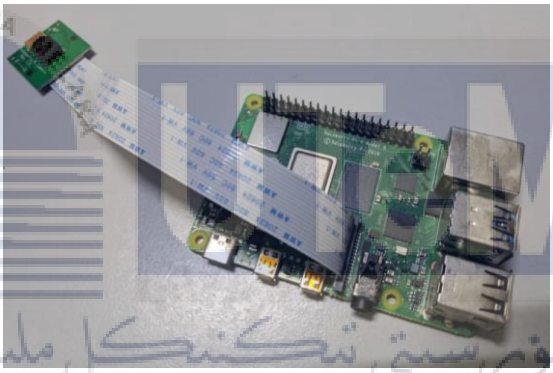

Figure 3.11: Pi camera attached to the Raspberry Pi 4 board

For the processing part, the Raspberry Pi is responsible to processes the input data in real time. The Raspberry Pi 4 is equipped with a 1.5 GHz 64-bit quad-core ARM Cortex-A72 processor. This processor is powerful enough to process the input data from the camera and execute the image processing program. The performance of the raspberry pi is further improved through multithreading. One of the issue for Raspberry Pi 4 is that the processor might overheat through time. To solve this issue, cooling fan and heatsink were installed.

In the output part, any result from the executed program will be shown in the Raspberry Pi's terminal and can be accessible through connecting to a monitor. The results from the raspberry pi includes frame per second of the video thread, eye aspect ratio value and yawn value. The system will also alert the driver through a speaker connected to the Raspberry Pi via 3.5mm audio jack. Speaker is chosen over a buzzer because the output sound of the speaker can be modified compared to a buzzer which only can release a limited type of sound. With speaker, various sound alert can be used based on the type of behavior detected by the system. The Raspberry Pi also have their build in WiFi module. This enable the Raspberry Pi to connect to the internet without extra hardware extension. As the output of Raspberry Pi includes sending data to the cloud storage, the WiFi connection must be intact.

3.4 Analysis of Result

3.4.1 Analysis on the decency of the system

In this section, the decency of the system is analyzed. What is focused here is that the threshold value used in this system, namely the yawning threshold and the EAR threshold. The purpose of this analysis is to determine the suitable threshold that can detects the drowsy behavior and yawning behavior. From previous research, it is shown that every threshold on every paper is differ. This is due to different algorithm and different system use to process each video input. So to determine the suitable threshold value for both yawning and EAR, a set of images which portraits people who are yawning and people who are closing their eyes are chosen. The Raspberry Pi will extract the EAR value and yawning value of each image obtained through the Pi Camera. Then, the highest and lowest value will be extracted from the results. These value will be further analyzed to determine the best value to be set as a threshold.

Image is chosen as a standard for obtaining the threshold because image is stationary and will give an absolute value for each activity. Compare to a real-life video, which consist a lot of movement and ambient disturbance, the value obtained might not be accurate as the value will deviate a lot on every frame of recording. As a conclusion, images tends to gives a better result compared to video.

3.4.2 Analysis on the accuracy

The second part of the result analysis is the analysis on the accuracy of the system. The accuracy of the system is defined as how well does the system works in detecting both drowsiness and yawning. So, in this section, a set of motion videos which shows people yawning and closing their eyes are selected. The camera is pointed at the motion videos and the results are collected. The total number of successful detection will be divided by the total number of videos selected to be tested by the Raspberry Pi. Equation 3.1 shows the equation used to calculate accuracy of the system.

$$\%accuracy = \frac{Number\ of\ succesfull\ detection}{Total\ motion\ videos\ tested} * 100\% \quad (3.1)$$

After obtaining the results, the data are tabulated for the ease of analyzing. The accuracy of the system will be determined after all test were run for both type of detection. If the accuracy of the system falls below a certain value, the system will be categorized as not accurate.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Results

After compiling the program in the Raspberry Pi, a few test run is done to preview the results. When the program is executed, a window will pop up displaying the video captured by the Pi camera. The respective windows will display the frame per second (fps), yawning ratio and the EAR value. The program can be run through the IDE window of Raspberry Pi. Figure 4.1 shows the pop-up window in IDE.

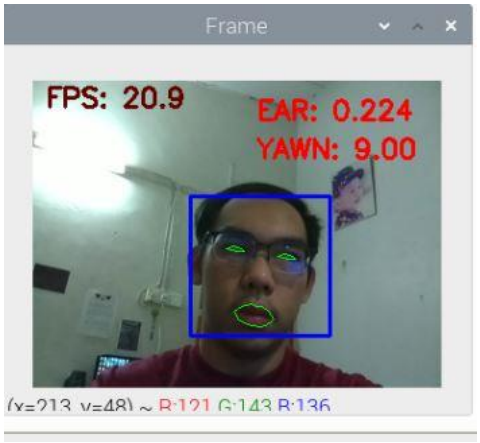

Figure 4.1: Pop-up window.

The image inside the blue frame is the face detected by the system whereas the green contour covers the eye and mouth level. The fps value is displayed on the top left corner whereas yawn ratio and EAR value is displayed on top right of the window.

When the target face is detected to exhibit drowsy or yawning behavior, a warning message will be displayed and an alert will goes off. Figure 4.2 and figure 4.3 shows the output image when the warning is displayed for both drowsy and yawning alert respectively.

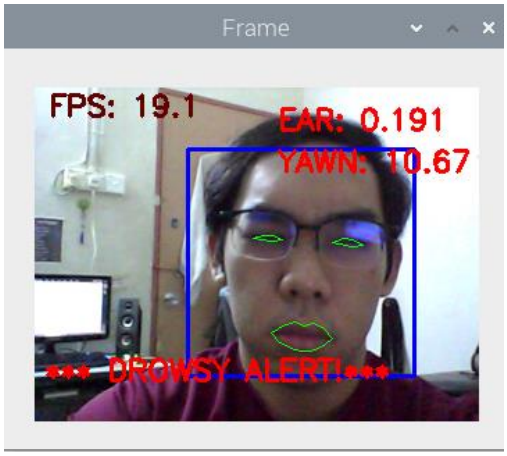

Figure 4.2: Drowsy alert.

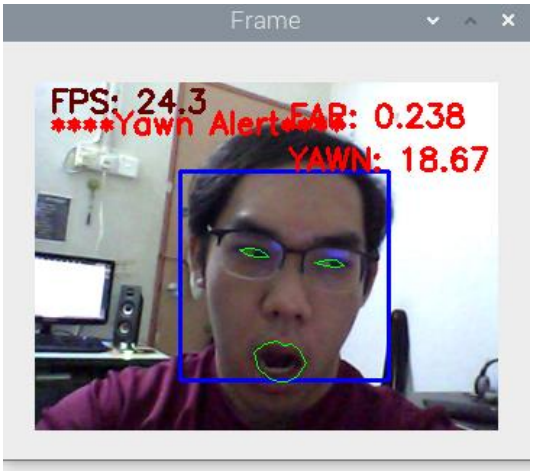

Figure 4.3: Yawning alert.

When the drowsy alert is triggered, the system will automatically capture the image of the target. The captured image will then be uploaded to a Dropbox cloud drive so that it could be used for further references. The Dropbox is accessible via smartphone's app as well as computer's software or web browser. The image will be named based on the time when the image is captured. Figure 4.4 shows the sample display of Dropbox on smartphone.

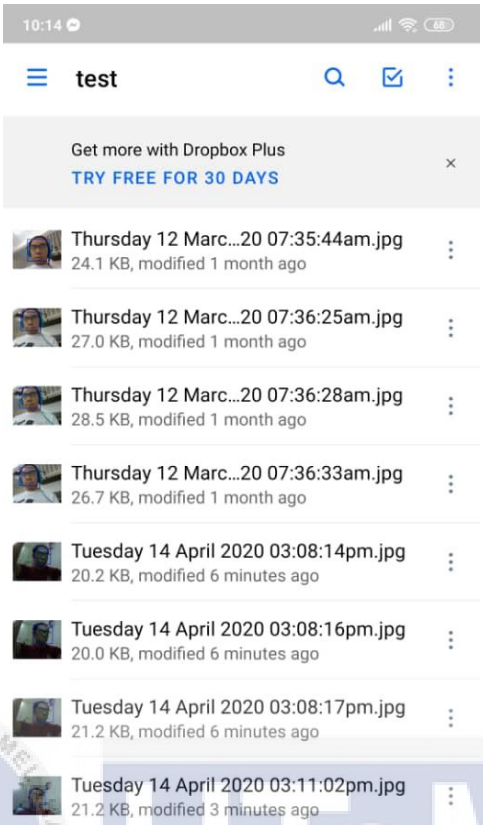

Figure 4.4: Dropbox app on smartphone

4.2 System decency's analysis

In this section, the decency of the system is analyzed. The yawning behavior and drowsy behavior is determined by the yawning threshold value and EAR threshold value. Both these value varies depending on the setting of the system which includes the input's frame size and the type of algorithm used. So in order to determine the suitable value for both thresholds. A set of experiment is conducted to obtain such value.

4.2.1 Analysis on EAR threshold

For EAR threshold, 4 sets of sample image which consist of male and female, wearing and not wearing glasses were chosen. Each sample set consists of 6 different image portraying people from different ethics closing their eyes. Figure 4.5 shows the portion image used for this analysis.

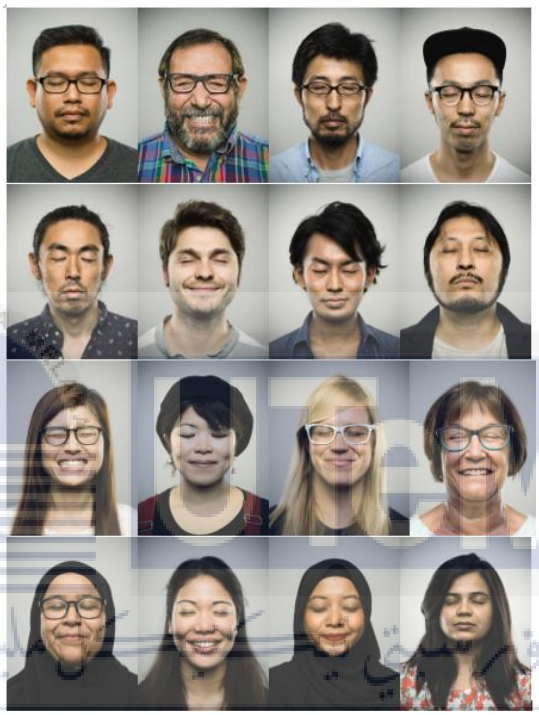

Figure 4.5: Sample image used for determining the drowsiness EAR threshold

A sample EAR value of 20 frames were taken and the results were tabulated. The highest and lowest value is obtained and shown in the table. Tables 4.1, 4.2, 4.3, and 4.4 show the result obtained from the test according to their group.

Table 4.1: Drowsiness EAR values of male subject with spectacles on

Image / Frame	1	2	3	4	5	6
1	0.166	0.202	0.191	0.206	0.101	0.22
2	0.214	0.196	0.121	0.181	0.118	0.229
3	0.205	0.152	0.204	0.19	0.087	0.14
4	0.208	0.189	0.149	0.166	0.087	0.219
5	0.171	0.157	0.162	0.178	0.114	0.201
6	0.159	0.191	0.148	0.175	0.104	0.218
7	0.185	0.2	0.177	0.17	0.142	0.214
8	0.2	0.105	0.184	0.174	0.136	0.222
9	0.199	0.118	0.182	0.187	0.121	0.217
10	0.169	0.134	0.141	0.172	0.104	0.185
11	0.184	0.153	0.166	0.168	0.104	0.162
12	0.19	0.159	0.175	0.173	0.023	0.216
13	0.178	0.152	0.146	0.192	0.092	0.215
14	0.187	0.189	0.165	0.184	0.087	0.196
15	0.162	0.203	0.161	0.19	0.042	0.189
16	0.167	0.156	0.156	0.189	0.122	0.183
17	0.195	0.103	0.152	0.183	0.151	0.18
18	0.206	0.099	0.207	0.164	0.08	0.222
19	0.178	0.174	0.167	0.179	0.021	0.179
20	0.213	0.172	0.13	0.205	0.092	0.154
Highest Value						0.229
Lowest Value						0.021

Table 4.2:Drowsiness EAR values of male subject without spectacles

Image / Frame	1	2	3	4	5	6
1	0.192	0.189	0.173	0.144	0.168	0.186
2	0.133	0.192	0.136	0.15	0.154	0.044
3	0.23	0.135	0.182	0.137	0.199	0.159
4	0.199	0.178	0.173	0.148	0.178	0.111
5	0.153	0.222	0.133	0.137	0.199	0.1
6	0.112	0.155	0.124	0.109	0.176	0.1
7	0.205	0.176	0.205	0.217	0.169	0.228
8	0.218	0.164	0.162	0.125	0.168	0.111
9	0.173	0.176	0.142	0.241	0.181	0.227
10	0.179	0.176	0.128	0.178	0.177	0.143
11	0.191	0.192	0.159	0.125	0.183	0.211
12	0.187	0.178	0.124	0.122	0.142	0.166
13	0.188	0.146	0.142	0.102	0.192	0.028
14	0.192	0.161	0.135	0.158	0.125	0.083
15	0.179	0.172	0.137	0.156	0.167	0.031
16	0.187	0.143	0.106	0.21	0.136	0.031
17	0.178	0.181	0.138	0.15	0.18	0.059
18	0.179	0.166	0.124	0.118	0.179	0.028
19	0.187	0.156	0.127	0.146	0.181	0.034
20	0.199	0.219	0.112	0.148	0.17	0.121
Highest Value						0.241
Lowest Value						0.028

Table 4.3: Drowsiness EAR values of female subject with spectacles on

Image / Frame	1	2	3	4	5	6
1	0.202	0.158	0.237	0.137	0.155	0.084
2	0.161	0.162	0.214	0.182	0.209	0.084
3	0.197	0.139	0.179	0.192	0.163	0.101
4	0.228	0.118	0.231	0.175	0.194	0.078
5	0.223	0.127	0.192	0.186	0.193	0.082
6	0.204	0.122	0.21	0.228	0.163	0.065
7	0.226	0.072	0.213	0.199	0.217	0.087
8	0.242	0.083	0.193	0.193	0.156	0.112
9	0.229	0.099	0.214	0.196	0.194	0.094
10	0.162	0.106	0.169	0.176	0.172	0.103
11	0.153	0.137	0.161	0.128	0.211	0.116
12	0.168	0.137	0.232	0.128	0.164	0.123
13	0.22	0.157	0.161	0.212	0.22	0.079
14	0.209	0.184	0.214	0.232	0.178	0.119
15	0.173	0.146	0.22	0.208	0.159	0.097
16	0.233	0.125	0.183	0.184	0.196	0.05
17	0.201	0.094	0.201	0.21	0.166	0.065
18	0.163	0.135	0.196	0.205	0.213	0.081
19	0.11	0.071	0.16	0.229	0.219	0.108
20	0.196	0.152	0.212	0.213	0.174	0.087
Highest Value						0.242
Lowest Value						0.05

Table 4.4: Drowsiness EAR value of female subject without spectacles

Image / Frame	1	2	3	4	5	6
1	0.214	0.202	0.191	0.206	0.101	0.22
2	0.239	0.196	0.121	0.181	0.118	0.229
3	0.183	0.152	0.204	0.19	0.087	0.14
4	0.186	0.189	0.149	0.166	0.087	0.219
5	0.17	0.157	0.162	0.178	0.114	0.201
6	0.218	0.191	0.148	0.175	0.104	0.218
7	0.219	0.2	0.177	0.17	0.142	0.214
8	0.242	0.105	0.184	0.174	0.136	0.222
9	0.149	0.118	0.182	0.187	0.121	0.217
10	0.216	0.134	0.141	0.172	0.104	0.185
11	0.145	0.153	0.166	0.168	0.104	0.162
12	0.155	0.159	0.175	0.173	0.023	0.216
13	0.222	0.152	0.146	0.192	0.092	0.215
14	0.183	0.189	0.165	0.184	0.087	0.196
15	0.183	0.203	0.161	0.19	0.042	0.189
16	0.225	0.156	0.156	0.189	0.122	0.183
17	0.222	0.103	0.152	0.183	0.151	0.18
18	0.23	0.099	0.207	0.164	0.08	0.222
19	0.199	0.174	0.167	0.179	0.021	0.179
20	0.159	0.172	0.13	0.205	0.092	0.154
Highest Value						0.244
Lowest Value						0.066

From the gathered result, it is observed that the lowest value detected by the system is 0.021 while the highest value detected by the system is 0.244. To ensure that the

system can detect drowsiness more accurately, the highest value obtained is chosen as the threshold. The reason for this is to ensure the system is more sensitive while detecting the closing of the eyes.

There is a concern which that the value obtained preciously might have a conflict with normal eye level value as there are people who poses smaller eye sizes. So, to ensure that there is no conflict between the drowsy EAR threshold and normal eye level EAR value, another analysis is done. Another set of sample images portraying people with normal eye level is chosen. Figure 4.6 shows portion of the sample image for the analysis.

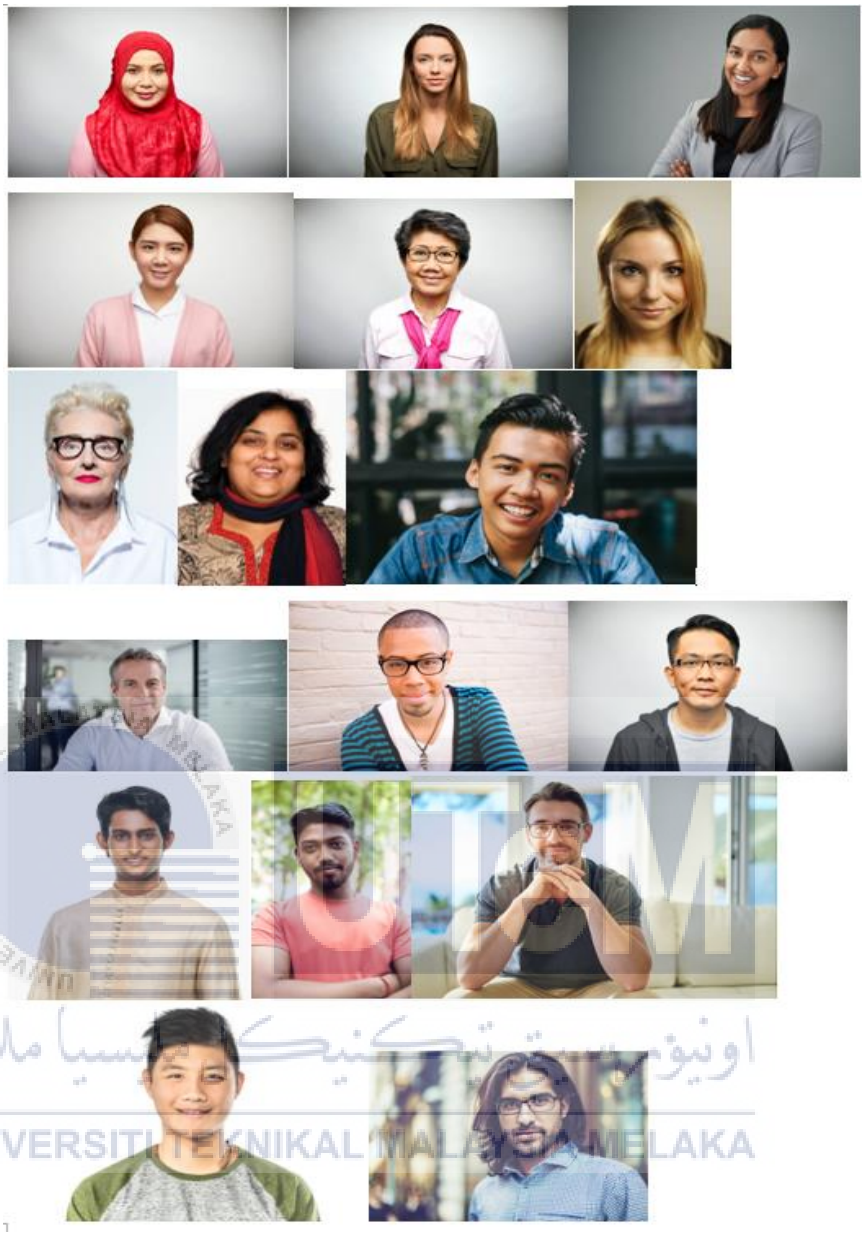

Figure 4.6: Sample image used to determine normal eye level EAR value

For this section, 20 frames of EAR value were to be recorded to determine the normal eye level's EAR value. The highest and lowest value obtained will also be extracted. The obtained results were recorded in Tables 4.5 and 4.6.

Table 4.5: EAR values of male subject's normal eye level

Frame \ Image	1	2	3	4	5	6	7	8	9	10
1	0.258	0.289	0.311	0.252	0.296	0.29	0.279	0.312	0.312	0.268
2	0.265	0.261	0.261	0.355	0.285	0.276	0.266	0.336	0.319	0.256
3	0.321	0.287	0.262	0.268	0.346	0.29	0.269	0.261	0.278	0.254
4	0.26	0.26	0.277	0.261	0.258	0.259	0.253	0.253	0.274	0.262
5	0.333	0.306	0.304	0.289	0.284	0.28	0.258	0.293	0.287	0.259
6	0.281	0.261	0.257	0.266	0.319	0.253	0.255	0.302	0.273	0.259
7	0.277	0.273	0.26	0.267	0.297	0.274	0.283	0.276	0.286	0.259
8	0.297	0.261	0.342	0.279	0.326	0.262	0.263	0.304	0.27	0.257
9	0.311	0.325	0.251	0.273	0.3	0.263	0.257	0.275	0.274	0.294
10	0.263	0.274	0.262	0.278	0.301	0.255	0.289	0.317	0.278	0.254
11	0.265	0.32	0.298	0.276	0.273	0.277	0.264	0.31	0.262	0.289
12	0.278	0.264	0.254	0.255	0.272	0.255	0.28	0.267	0.259	0.259
13	0.338	0.281	0.252	0.258	0.298	0.266	0.271	0.269	0.347	0.256
14	0.317	0.276	0.267	0.251	0.314	0.257	0.256	0.301	0.299	0.266
15	0.266	0.28	0.257	0.277	0.301	0.261	0.251	0.31	0.278	0.259
16	0.293	0.261	0.279	0.252	0.334	0.272	0.251	0.28	0.319	0.258
17	0.253	0.276	0.279	0.288	0.333	0.263	0.269	0.268	0.287	0.256
18	0.277	0.265	0.268	0.291	0.312	0.258	0.252	0.293	0.264	0.254
19	0.316	0.261	0.261	0.278	0.284	0.289	0.279	0.26	0.264	0.282
20	0.338	0.29	0.298	0.285	0.322	0.266	0.263	0.272	0.28	0.257
Highest Value										0.355
Lowest Value										0.251

Table 4.6: EAR values of female subject's normal eye level

Frame \ Image	1	2	3	4	5	6	7	8	9	10
1	0.35	0.342	0.335	0.358	0.309	0.378	0.398	0.343	0.328	0.429
2	0.367	0.31	0.44	0.349	0.294	0.365	0.373	0.338	0.329	0.277
3	0.369	0.33	0.335	0.334	0.263	0.345	0.385	0.253	0.276	0.317
4	0.39	0.308	0.398	0.323	0.27	0.402	0.394	0.346	0.269	0.26
5	0.381	0.327	0.315	0.335	0.292	0.354	0.398	0.274	0.316	0.326
6	0.341	0.345	0.322	0.375	0.273	0.394	0.353	0.316	0.302	0.27
7	0.369	0.359	0.314	0.357	0.284	0.36	0.351	0.312	0.296	0.313
8	0.379	0.326	0.311	0.309	0.281	0.387	0.394	0.317	0.318	0.42
9	0.365	0.34	0.332	0.356	0.273	0.388	0.405	0.349	0.271	0.303
10	0.367	0.336	0.335	0.366	0.311	0.379	0.371	0.322	0.324	0.286
11	0.383	0.347	0.317	0.36	0.281	0.432	0.368	0.329	0.285	0.302
12	0.36	0.333	0.401	0.324	0.267	0.351	0.365	0.313	0.283	0.357
13	0.353	0.364	0.332	0.366	0.301	0.324	0.348	0.264	0.316	0.4
14	0.36	0.339	0.372	0.355	0.289	0.411	0.393	0.251	0.286	0.257
15	0.35	0.349	0.419	0.349	0.295	0.4	0.366	0.278	0.318	0.318
16	0.375	0.337	0.297	0.332	0.273	0.396	0.375	0.284	0.283	0.307
17	0.331	0.367	0.381	0.363	0.311	0.408	0.368	0.333	0.285	0.256
18	0.398	0.327	0.314	0.293	0.305	0.372	0.348	0.277	0.301	0.277
19	0.36	0.368	0.291	0.322	0.284	0.383	0.43	0.352	0.318	0.265
20	0.363	0.365	0.372	0.349	0.273	0.412	0.373	0.306	0.334	0.266
Highest Value										0.44
Lowest Value										0.251

According to the data observed, the lowest EAR value of normal eye level is 0.251 for both the gender and the highest value is 0.455. Comparing this result to drowsiness

EAR, it is observed that the value range for normal eye level is 0.251 to 0.455, while for people closing their eyes is 0.021 to 0.244. The final results show that there is no conflict between these two ranges and hence the value of 0.244 is safe to be used as the threshold for drowsy driving in this project.

4.2.2 Analysis on yawning threshold

For yawning threshold, a sets of sample images were chosen. These images are subject mixed between genders and ethics. Figure 4.7 shows the sample images used for this analysis part.

Figure 4.7: Sample of yawning subject used to determine yawning threshold

A sample of 20 frames data were obtained for yawning from each image and were tabulated for further analysis. Table 4.7 shows the data obtained from the test. The highest and lowest value were extracted from the table.

Table 4.7: Data of yawn level obtained from the Raspberry Pi

Frame \ Image	1	2	3	4	5	6	7	8	9	10
1	17	16.5	18.333	18.667	16.833	20	20.333	17.167	17.333	17.333
2	16.5	18.5	17.5	15.833	17.833	20.167	19.833	18.167	17	16
3	20.833	17.667	17.5	17.167	18.833	20.167	15.167	15	17	15.5
4	16	18.333	17.833	17.333	16.5	27.5	16.667	18.5	17	16.333
5	18.667	17.667	16.333	16.167	17.5	29.167	14	16.333	17.5	15.833
6	16.167	18	18	17	19.167	26.333	17.667	18	16.667	15.667
7	16.833	16.167	17.167	16.833	19.167	29.333	18	18	17.833	16.333
8	20.333	15.833	17.333	17.333	17.667	26.833	16.5	16.667	17.5	15.667
9	19.667	18.667	17.333	18	19	29	20.333	16.333	16.833	16.5
10	17.167	16	16.833	17.833	17	29.833	16.667	17.667	18.333	17.333
11	17	17.333	16.667	16.5	18.5	29	17.167	17.667	18	15.667
12	17.333	16.667	17	18.333	17.5	30.333	16.5	17.5	17.167	16.833
13	16	16.833	17	15.667	18.667	28.667	16.5	17.667	16.333	16.5
14	20.5	15.167	17	18.5	18.5	29.333	19.333	17.5	17.833	15.833
15	18.5	17	18.667	16.333	18.5	31.5	18.167	18.5	16.833	16.167
16	19.667	18.167	17.667	17.667	17.333	29.833	14.5	16.5	16.833	16.667
17	16.833	15.833	18	17.667	19	31.833	14.833	17.5	18.5	15.833
18	20	17.167	17.667	17.5	17.167	30.333	16.167	17.833	17.333	16
19	16	18.333	17.833	17.333	16.5	27.5	16.667	18.5	17	16.333
20	18.667	17.667	16.333	16.167	17.5	29.167	14	16.333	17.5	15.833
Highest Value										31.833
Lowest Value										14

From the data obtained, 14 is the lowest yawn value obtained and 31.833 is the highest value obtained when comparing both male and female subject. Hence, the value 14 is chosen as the threshold value to determine yawning behavior for this project. The reason for this value to be chosen is to increase the sensitivity of the system. In the case of yawning, the higher the value indicates the larger the opening proportion of the mouth, hence person who yawn softer than a certain people might not be indicated as yawning if the threshold is set too high. So by setting the value lower, the sensitivity of the system can be increased and the chance of missing a certain yawning frame can lowered.

4.3 System's Accuracy Analysis

In this section, the analysis of accuracy is divided into two parts. The first part of the analysis is to analyze the accuracy of the system to detects drowsy behavior. The second part is the analysis of the accuracy of the system in detecting yawning behavior. A total sample of 50 subjects were used in this section. The amount of sample used is based on the research paper title "Driver Drowsiness Detection Based on Face Feature and PERCLOS" by Suhandi Junaedi and Habibullah Akbar [19].

4.3.1 Accuracy Analysis on Drowsy Behavior

In this section, the accuracy of drowsy behavior detection is analyzed. A set of video of people closing their eyes is being played and the Pi Camera is pointed at the monitor to capture the motion of the subject. The EAR value for drowsy behavior is set to 0.244 based on the analysis is the previous section. In this analysis, 25 videos were chosen. The sample video used in this part consist of subject with different gender and race. The accuracy of the system is calculated using Equation 3.1. Figure 4.8 shows portion images of sample video used to test for the accuracy of the system and Table 4.8 shows the results obtained.

Figure 4.8: Sample for drowsy behavior detection test

Table 4.8: Accuracy of drowsy behavior detection in this project

Total Video Tested	Number of Successful Detection	Number of Failed Detection	Accuracy
25	23	2	92%

From the result obtained, the system is able to detect drowsy behavior with an accuracy of 92%. Based on the results, the system is considered to be reliable to detect drowsiness. Reasons for this is because among the samples, there are subjects who wore spectacles during the analysis and the system is still able to identify the drowsy behavior. Besides, the fact that the tested subjects came consist of people from different ethics making they portrays different skin tone and eye sizes didn't cause

much error in the system's accuracy further shows that the system is reliable in detecting drowsiness.

4.3.2 Accuracy Analysis on Yawning Behavior

This section focuses on analyzing the accuracy of the system in detecting yawning behavior. A video is chosen from the Internet which consist of multiple subject yawning continuously. This video contains 25 different subject and the system is required to detect the yawning behavior of each subject continuously. Equation 3.1 was used to calculate the accuracy of the system in detecting yawning behavior. The yawn value for yawning behavior is set to 14 based on the analysis done in part 4.1.2. The result obtained is recorded in the Table 4.9 and Figure 4.9 shows the portion image of subject in this analysis.

Figure 4.9: Sample for yawning behavior detection test

Table 4.9: Accuracy of yawning behavior detection in this project.

Total Video Tested	Number of Successful Detection	Number of Failed Detection	Accuracy
25	25	0	100%

From the result obtained, the system is able to detect yawning behavior without any failure. The sample video used for this analysis includes subject from various age, gender and ethics. Each of these subject have different mouth size and the yawning level of them are different. With the value obtained after doing sufficient testing from part 4.1.2, the system is capable of detecting yawning behavior accurately.

CHAPTER 5

CONCLUSION AND FUTURE WORKS

5.1 Conclusion

In this project, image processing technique were successfully applied into the driver's monitoring system using a portable single board computer, the Raspberry Pi 4. The system is able to detect drowsy behavior and yawning behavior separately and simultaneously. Haar Cascade is the main algorithm use in this system to detect the human face. The algorithm is supported by Dlib's facial landmark predictor which can be found in OpenCV library to locate the eye and mouth. Two different algorithm is then used to compute the eye level and the mouth level for monitoring purpose. Beside identifying the driver's status, this system also aims to alert the driver to reduce the chances of accidents. This is achieved by connecting the system to a speaker. The alert sound is customized depending on which type of warning that the driver triggered.

This approach is better than classic buzzer alert because it can deliver specific messages to nudge the driver.

The system was also analyzed at the end of this project for improvement. To determine the state of driver, a threshold is set for both yawning and drowsy. These threshold act as the boundary between improper behavior and normal behavior. These threshold value were set after doing certain testing and analysis. To get the optimum value, still image was used as analysis target to reduce error in data. The accuracy of the system is tested and proven to hold a high accuracy in detecting drowsy and yawning behavior. In the test of accuracy, set of motion videos were used as input data to simulate drowsiness and yawning behavior.

IoT was also implemented onto this system. The system will not only detect and alert the driver, but also keeps track of the events record for future use. With Internet connection, the system is able to send footage captured by the camera to a cloud storage whenever the system displays certain warning. Dropbox is chosen as the online file storage platform because it is easy for personal or company usage. The uploaded footage is renamed by the time it is taken for the ease of reference. With this approach, it is easier to pinpoint the cause of certain road accidents. Even without accidents occurring, the data from the cloud storage can be used for other purposes such as to evaluate the driver's performance.

5.2 Future Works

As the technology rises, there will be a day when man are no longer needed to control a vehicle. But looking at today's technologies, we are still far from the fully autonomous driving era. The nearest trend to us is the partial driving automation

technology where man are still required to operate the vehicles but the vehicle is capable of performing certain action on different occasion by its own.

With this in the context, the driver monitoring system is becoming more crucial. In order for the vehicle to perform action on its own, it needs to monitor not only the condition of the road but also the condition of the driver. The driver monitoring system can be equipped with certain safety maneuver such as reducing the vehicle speed when the drivers shows inappropriate behavior. The monitoring system can also be upgraded to monitor more aspects of the driver's condition. For example, the system can fully analyze the focus level of the driver, the health condition of the driver and many more.

As a conclusion, the driver monitoring systems in this project holds a great potential in current time. With the rise of IoT technology, the usage of driver monitoring system can be amplified. More work can be done in improving the system so that it is possible to be implemented on all vehicles.

REFERENCES

[1] M. H. Alsibai and S. A. Manap, "A study on driver fatigue notification systems," *ARPN J. Eng. Appl. Sci.*, vol. 11, no. 18, pp. 10987–10992, 2016.

[2] A. S. Aghaei *et al.*, "Smart Driver Monitoring: When Signal Processing Meets Human Factors: In the driver's seat," *IEEE Signal Process. Mag.*, vol. 33, no. 6, pp. 35–48, 2016.

[3] I. G. Daza, L. M. Bergasa, S. Bronte, J. Javier Yebes, J. Almazán, and R. Arroyo, "Fusion of optimized indicators from advanced driver assistance systems (ADAS) for driver drowsiness detection," *Sensors (Switzerland)*, vol. 14, no. 1, pp. 1106–1131, 2014.

[4] E. C. Orthopaedics, "Road Traffic Accidents– Burden on Society," vol. 2, no. January, pp. 30–33, 2018.

[5] K. A. Admassu, "Dataset Analysis and Trend Learning of the NHTSA 2015 Traffic Fatalities Data Dataset Analysis and Trend Learning of the NHTSA 2015 Traffic Fatalities Data," no. May, 2019.

[6] H. Wang, J. Hu, and W. Deng, "Face Feature Extraction: A Complete Review,"

IEEE Access, vol. 6, pp. 6001–6039, 2018.

[7] F. Jiménez, J. E. Naranjo, J. J. Anaya, F. García, A. Ponz, and J. M. Armingol, "Advanced Driver Assistance System for Road Environments to Improve Safety and Efficiency," in *Transportation Research Procedia*, Jan. 2016, vol. 14, pp. 2245–2254, 2016.

[8] D. G. Costa, "On the Development of Visual Sensors with Raspberry Pi," pp. 19–22, 2018.

[9] V. K. Bhanse and M. D. Jaybhaye, "Face Detection and Tracking Using Image Processing on Raspberry Pi," in *Proceedings of the International Conference on Inventive Research in Computing Applications, ICIRCA 2018*, Dec. 2018, pp. 1099–1103,2018.

[10] K. Goyal, K. Agarwal, and R. Kumar, "Face detection and tracking: Using OpenCV," in *Proceedings of the International Conference on Electronics, Communication and Aerospace Technology, ICECA 2017*, 2017, vol. 2017-January, pp. 474–478, 2017.

[11] A. E. Lörincz, M. N. Risteiu, A. Ionica, and M. Leba, "Driver monitoring system for automotive safety," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 294, no. 1, 2018.

[12] K. G. S. B. P. P, "Intelligent Driver Monitoring and Vehicle Control System Intelligent Driver Monitoring and Vehicle Control System," no. December, 2017.

[13] M. Omidyeganeh *et al.*, "Yawning Detection Using Embedded Smart

Cameras," *IEEE Trans. Instrum. Meas.*, vol. 65, no. 3, pp. 570–582, 2016.

[14] M. Jabbar, "Intelligent driver monitoring system," vol. 2020, no. June 2014, 2020.

[15] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "COMPUTER VISION AND IMAGE UNDERSTANDING Active Shape Models-Their Training and Application," *Comput. Vis. Image Underst.*, vol. 61, no. 1, pp. 38–59, 1995.

[16] H. Khalaf and S. Juboori, "A Real-Time Monitoring System for the Drivers Using PCA and SVM Husam Khalaf Salih Juboori," *Int. Res. J. Eng. Technol.*, vol. 4, no. 6, pp. 2970–2974, 2017.

[17] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, 2004.

[18] N. Hazim Barnouti, S. Sameer Mahmood Al-Dabbagh, W. Esam Matti, and M. Abdul Sahib Naser, "Face Detection and Recognition Using Viola-Jones with PCA-LDA and Square Euclidean Distance," 2016.

[19] S. Junaedi and H. Akbar, "Driver Drowsiness Detection Based on Face Feature and PERCLOS," *J. Phys. Conf. Ser.*, vol. 1090, no. 1, 2018.

[20] M. Naveenkumar, "OpenCV for Computer Vision Applications," no. March 2015, pp. 52–56, 2016.

[21] M. Khan, S. Chakraborty, R. Astya, and S. Khepra, "Face Detection and Recognition Using OpenCV," *Proc. - 2019 Int. Conf. Comput. Commun. Intell.*

Syst. ICCCIS 2019, vol. 2019-Janua, pp. 116–119, 2019.

[22] K. D. Pandya, "Face Detection – A Literature Survey," *Int. J. Comput. Tech. -*–, vol. 3, no. 1, pp. 67–70, 2016.

[23] J. Cech and T. Soukupova, "Real-Time Eye Blink Detection using Facial Landmarks," *Cent. Mach. Perception, Dep. Cybern. Fac. Electr. Eng. Czech Tech. Univ. Prague*, pp. 1–8, 2016.

[24] O. Of and M. Carriers, "PERCLOS : A Valid Psychophysiological Measure of Alertness As Assessed by Psychomotor Vigilance," *October*, vol. 31, no. 5, pp. 1237–1252, 1998.

[25] B. Mohana and C. M. Sheela Rani, "Drowsiness Detection Based on Eye Closure and Yawning Detection," *Int. J. Recent Technol. Eng.*, no. 4, pp. 2277–3878, 2019.

APPENDICES

Appendix A: Code for system

```
import numpy as np
import dlib
import cv2
import imutils
import argparse
from threading import Thread
from temp.tempimage import TempImage
from imutils.video import VideoStream
from imutils import face_utils
from imutils.video import FPS
import datetime
import dropbox
import time
import warnings
import os
import json
ap = argparse.ArgumentParser()
ap.add_argument( "-c", "--conf", required=True,
        help="/home/pi/Desktop/project/conf.json")
args = vars(ap.parse_args())
```

```
warnings.filterwarnings("ignore")
conf = json.load(open(args["conf"]))
client = None
def euclidean_dist(ptA, ptB):
    return np.linalg.norm(ptA - ptB)
def alarm(msg):
   global alarm_status
   global alarm_status2
   global saying
   while alarm_status:
      print('call')
      s = 'espeak "' +msg+ '"'
      os.system(s)
   if alarm_status2:
      print('call')
      saying = True
      s = 'espeak "' +msg+ '"'
      os.system(s)
      saying = False
def eye_aspect_ratio(eye):
   A = euclidean_dist(eye[1], eye[5])
   B = euclidean_dist(eye[2], eye[4])
   C = euclidean_dist(eye[0], eye[3])
   ear = (A + B) / (2.0 * C)
   return ear
def lip_distance(shape):
   top_lip = shape[50:53]
```

```
    top_lip = np.concatenate((top_lip, shape[61:64]))
    low_lip = shape[56:59]
    low_lip = np.concatenate((low_lip, shape[65:68]))
    top_mean = np.mean(top_lip, axis=0)
    low_mean = np.mean(low_lip, axis=0)
    distance = abs(top_mean[1] - low_mean[1])
    return distance
if conf["use_dropbox"]:
    dbx = dropbox.Dropbox(conf["dropbox_access_token"])
    print("[SUCCESS] dropbox account linked")
EYE_AR_THRESH = 0.24
EYE_AR_CONSEC_FRAMES = 36
YAWN_THRESH = 14
alarm_status = False
alarm_status2 = False
saying = False
print ("[INFO] loading facial landmark predictor...")
detector = cv2.CascadeClassifier("/home/pi/Desktop/classifiers/haarcascade_frontalface_default.xml")
predictor = dlib.shape_predictor("/home/pi/Desktop/classifiers/shape_predictor_68_face_landmarks.dat")
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["right_eye"]
print("[INFO] starting video stream thread...")
vs = VideoStream(usePiCamera=0).start()
fps = FPS().start()
time.sleep(1.0)
lastUploaded = datetime.datetime.now()
```

```
flag=0
while True:
    start_time = time.time()
    frame = vs.read()
    frame = imutils.resize(frame, width=320)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    rects = detector.detectMultiScale(gray, scaleFactor=1.1,
                        minNeighbors=5, minSize=(30,30),
                        flags=cv2.CASCADE_SCALE_IMAGE)
    for (x, y, w, h) in rects:
      timestamp = datetime.datetime.now()
      rect = dlib.rectangle(int(x), int(y), int(x + w),
                    int(y + h))
      shape = predictor(gray, rect)
      shape = face_utils.shape_to_np(shape)#converting to NumPy Array
      leftEye = shape[lStart:lEnd]
      rightEye = shape[rStart:rEnd]
      leftEAR = eye_aspect_ratio(leftEye)
      rightEAR = eye_aspect_ratio(rightEye)
      ear = (leftEAR + rightEAR) / 2.0
      distance = lip_distance(shape)
      leftEyeHull = cv2.convexHull(leftEye)
      rightEyeHull = cv2.convexHull(rightEye)
      cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
      roi_gray = gray[y:y+h, x:x+w]
      roi_color = frame[y:y+h, x:x+w]
      cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
```

```
cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
lip = shape[48:60]
cv2.drawContours(frame, [lip], -1, (0, 255, 0), 1)
ts = timestamp.strftime("%A %d %B %Y %I:%M:%S%p")
if ear < EYE_AR_THRESH:
   flag += 1
   if flag >= EYE_AR_CONSEC_FRAMES:
      print ("***ALERT!***")
      if alarm_status == False:
         alarm_status = True
         t = Thread(target=alarm, args=('wake up',))
         t.deamon = True
         t.start()
      if conf["use_dropbox"]:
         t = TempImage()
         cv2.imwrite(t.path, frame)
         print("[UPLOAD] {}".format(ts))
         path = "/{base_path}/{timestamp}.jpg".format(
            base_path=conf["dropbox_base_path"], timestamp=ts)
         dbx.files_upload(open(t.path, "rb").read(), path)
         t.cleanup()
      cv2.putText(frame, "***ALERT!***", (7,210),
         cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
      #print ("Drowsy")
else:
   flag = 0
   alarm_status = False
```

```
        if (distance > YAWN_THRESH):
            cv2.putText(frame, "Yawn Alert", (10, 35),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            if alarm_status2 == False and saying == False:
              alarm_status2 = True
              t = Thread(target=alarm, args=('take some rest',))
              t.deamon = True
              t.start()
        else :
          alarm_status2 = False
        fpsInfo = "FPS: " + str(1.0 / (time.time() - start_time)) # FPS = 1 / time to process loop
        cv2.putText(frame, "{:.9s}".format(fpsInfo), (10, 20),
        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 100), 2)
        cv2.putText(frame, "EAR: {:.3f}".format(ear), (175, 30),
        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        cv2.putText(frame, "YAWN: {:.2f}".format(distance), (175, 60),
        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF
    if key == ord("a"):
      break
fps.update()
fps.stop()
print("[INFO] elasped time: {:.2f}".format(fps.elapsed()))
print("[INFO] approx. FPS: {:.4f}".format(fps.fps()))
cv2.destroyAllWindows()
vs.stop()
```