

# **WARNING SYSTEM FOR DRIVER**

## **A PROJECT REPORT**

*Submitted by*

**Aditya Kumar Mishra (1834083511493)**

**Ajay Bodra (183408351494)**

**Nitish Kumar (183408351508)**

**Shubham Sharma (183408351514)**

*Under the guidance of*

**Mr. Saunak Bhattacharya**

(Designation, Department of Electronics & Communication Engineering)

*in partial fulfillment for the*

*award of the degree of*

**BACHELOR OF  
TECHNOLOGY**

in

**ELECTRONICS & COMMUNICATION  
ENGINEERING**

of

**CHAIBASA ENGINEERING COLLEGE**



**Bistumpur, Jhinkpani, West Singhbhum, Jharkhand 833215**

**2021**

## **DECLARATION**

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

.....

Signature of the Students

# CHAIBASA ENGINEERING COLLEGE



## CERTIFICATE

Certified that this project report titled “**WARNING SYSTEM FOR DRIVERS**” is the bonafide work of “Aditya Kumar Mishra (1834083511493), Ajay Bodra (183408351494), Nitish Kumar (183408351508), Shubham Sharma (183408351514)” who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

.....

### SIGNATURE

**Mr. SAUNAK BHATTACHARYA  
HEAD OF THE DEPARTMENT  
DEPT. OF ELECTRONICS &  
COMMUNICATION  
ENGINEERING**

.....

### SIGNATURE

**Mr. SAUNAK BHATTACHARYA  
HEAD OF THE DEPARTMENT  
DEPT. OF ELECTRONICS &  
COMMUNICATION  
ENGINEERING**

## **ABSTRACT**

An embedded system is a microprocessor-based computer hardware system with software that is designed to perform a dedicated function, either as an independent system or as a part of a large system. At the core is an integrated circuit designed to carry out computation for real-time operations. The main objective of this study is to develop an efficient WARNING system which contains an enriched dataset of Indian traffic signs. The developed technique is invariant in variable lighting, rotation, translation, and viewing angle and has a low computational time with low false positive rate. The development of the system has three working stages: image preprocessing, detection, and recognition. The system demonstration using a RGB colour The area under the receiver operating characteristic (ROC) curves was introduced to statistically evaluate the recognition performance. The accuracy of the developed system is relatively high and the computational time is relatively low which will be helpful for classifying traffic signs especially on high ways around Malaysia. The low false positive rate will increase the system stability and reliability on real-time application.

## **ACKNOWLEDGEMENT**

I would like to express my deepest gratitude to my guide, Mr. SAUNAK BHATTACHARYA his valuable guidance, consistent encouragement, personal caring, timely help and providing me with an excellent atmosphere for doing the project. All through the work, in spite of his busy schedule, he has extended cheerful and cordial support to me for completing this projectwork.

.....

**Signature of the Students**

# TABLE OF CONTENTS

<b>ABSTRACT</b>	<b>1</b>
<b>ACKNOWLEDGEMENTS</b>	<b>2</b>
<b>CONTENTS</b>	<b>6-7</b>
<b>LIST OF FIGURES AND TABLES</b>	<b>8</b>
<b>ABBREVIATIONS</b>	<b>9</b>
<b>CHAPTER-1: INTRODUCTION</b>	<b>10-14</b>
1. Motivation and Summary of Work	12
1.1. Sub Sections	13
1.2. Outline of Project Report...	14
<b>CHAPTER-2: LITERATURE SURVEY</b>	<b>15</b>
<b>CHAPTER-3: METHODOLOGY</b>	<b>16-25</b>
<b>CHAPTER-4: SYSTEM DESIGN</b>	<b>26-42</b>
<b>CHAPTER-5: RESULT AND ANALYSIS</b>	<b>43-47</b>
<b>CHAPTER-6: CONCLUSION</b>	<b>48-49</b>

<b>CHAPTER-7: FUTURE SCOPE</b>	<b>50</b>
<b>APENDEX- I: LINEARITY CALCULATIONS</b>	<b>51</b>
<b>APENDEX- II: PARALLELISM CALCULATION</b>	<b>52</b>
<b>BIBLIOGRAPHY:</b>	<b>53</b>

## **LIST OF FIGURES**

FIGURE 1	24
FIGURE 2	43
FIGURE3	44



## **LIST OF ABBREVIATION**

<b>TPS</b>	Throttle position
<b>IAT</b>	Intake air temperature
<b>MAF</b>	Mass air flow
<b>GPS</b>	Global positioning system
<b>OBD</b>	On board diagnostic

# CHAPTER 1

## INTRODUCTION

In order to solve the concerns over road and transportation safety, automatic traffic sign detection and warning system for drivers has been introduced. Along with a vehicle diagnosis system has been developed to monitor vehicle's health. Vehicles diagnosis is done by measuring coolant temperature, GPS speed, engine load, intake air temperature, MAF etc. An automatic traffic sign detection system can detect and recognize traffic signs from and within images captured by visual programming. In adverse traffic conditions, the driver may not notice traffic signs, which may cause accidents. In such scenarios, the proposed system comes into action. The main objective of the research on warning system is to improve the robustness and efficiency of the system. To develop a Warning system is a tedious job given the continuous changes in the environment and lighting conditions. Among the other issues that also need to be addressed are partial obscuring, multiple traffic signs appearing at a single time, and blurring and fading of traffic signs, which can also create problem for the detection purpose. For applying the Warning system in real time environment, a fast algorithm is needed. As well as dealing with these issues, a detection system should also avoid erroneous detection of non-signs. The aim of this research is to develop an efficient Warning system which can detect and classify traffic signs into different classes in real-time environment. For detecting the red traffic signs, a combination of color and shape based algorithm is presented which will make up the procedure of the detection stage. This paper is organized as follows: Section 2 presents the related works in the field of development of the WARNING system. In Section 3, the overall methodology is discussed. The experimental results and discussions are summarized in Section 4. In Section 5, the conclusion and some suggestions are made for future improvement on the field of automatic traffic sign detection and recognition.

Road traffic signs are divided in to two major categories of main signs and auxiliary signs. The main sign is divided into warning signs, prohibition signs, mandatory signs, guide signs, tourist signs and road construction and safety signs. Among them, prohibition signs mainly played a role in banning certain kinds of behavior, a total of 43 categories. Mandatory signs indicate the role of vehicles which is placed in the need to indicate vehicles, near the intersection. Warnings are mainly to alert drivers, vehicles

trians to beware of dangerous targets, a total of 45 categories. They all play an important role in traffic turning signs are of great significance for safe driving of drivers and therefore are the focus of the current research on traffic sign recognition.

Road and traffic signs considered in this thesis are those that use a visual/symbolic language about the road(s) ahead that can be interpreted by drivers. The terms are used interchangeably in this thesis, and elsewhere might also appear in combination, as “road traffic signs”. They provide the driver with pieces of information that make driving safe and convenient. A type of sign that is NOT considered in this thesis is the direction sign, in which the upcoming directions for getting to named towns or on numbered routes are shown not symbolically but essentially by text. Road and traffic signs must be properly installed in the necessary locations and an inventory of them is ideally needed to help ensure adequate updating and maintenance. Meetings with the highway authorities in both Scotland and Sweden revealed the absence of but a need for an inventory of traffic signs. An automatic means of detecting and recognising traffic signs can make a significant contribution to this goal by providing a fast method of detecting, classifying and logging signs. This method helps to develop the inventory accurately and consistently. Once this is done, the detection of disfigured or obscured signs becomes easier for human operator. Road and traffic sign recognition is the field of study that can be used to aid the development of an inventory system (for which real-time recognition is not required) or aid the development of an in-car advisory system (when real-time recognition is necessary). Both road sign inventory and road sign recognition are concerned with traffic signs, face similar challenges and use automatic detection and recognition. A road and traffic sign recognition system could in principle be developed as part of an Intelligent Transport Systems (ITS) that continuously monitors the driver, the vehicle, and the road in order, for example, to inform the driver in time about upcoming decision points regarding navigation and potentially risky traffic situations. Figure 1.1 depicts these relationships among the three fields. ITS focuses on integrating information technology into transport infrastructure and vehicles. These systems can include road sensors, in-vehicle navigation services, 2 electronic message signs, and traffic management and monitoring. The aim of intelligent transport systems is to increase transportation efficiency, road safety and to reduce the environmental impact with the use of advanced communication technologies

## 1.1 Motivation and Summary of The Work

Reckless driving is a significant factor in the increasing number of accidents on today's roads and has been extensively accepted. This proof has been verified by many researchers that have demonstrated ties between reckless driving and road accidents. Although it is hard to decide the exact number of accidents due to reckless driving, it is much likely to be underestimated. The above statement shows the significance of a research with the objective of reducing the dangers of accidents anticipated to reckless driving. So far, researchers have tried to model the behavior by creating links between reckless driving and certain indications related to the vehicle and to the driver.

Road accidents occur when the sign boards are not noticed by the driver and the speed of vehicle is not altered on time. So motivation for developing a warning system is concerned with pedestrian safety as well as the safety of the driver.

Looking at the problem of road and traffic sign recognition shows that the goal is well defined and it seems to be a simple problem. Road signs are located in standard positions and they have standard shapes, standard colours, and their pictograms are known. To see the problem in its full scale, however, a number of parameters that affect the performance of the detection system need to be studied carefully. Road sign images are acquired using a digital camera for the purpose of the current analysis. However, still images captured from a moving camera may suffer from motion blur. Moreover, these images can contain road signs which are partially or totally occluded by other objects such as vehicles or pedestrians. Other problems, such as the presence of objects similar to road signs, such as buildings or billboards, can affect the system and make sign detection difficult. The system should be able to deal with traffic and road signs in a wide range of weather and illumination variant environments such as different seasons, different weather condition e.g. sunny, foggy, rainy and snowy conditions. Different potential difficulties are depicted in one section of this chapter. Using the system in different countries can make the problem even worse. Different countries use different colours and different pictograms. The system should also be adaptive, which means it should allow continuous learning otherwise the training should be repeated for every country. To deal with all these constraints, road sign recognition should be provided with a large number of sign examples to allow the system to respond correctly when a traffic sign is encountered.

Transportation accident rate are still being a major challenge in many countries. There are many factors that could be cause transportation accident, especially in vehicle's internal system problem. To overcome this problem, OBD-II technology has been created to diagnostics vehicle's condition. OBD-II scanner plugged to OBD-II port or usually called Data Link Connector (DLC), and after that it sends the diagnostics to Raspberry Pi. Compared from another microcontrollers, Arduino, Raspberry Pi are chosen because it sustains the application to receive real-time diagnostics, process the diagnostics and send command to automobiles at the same time, rather than Arduino that must wait for another process finished to run another process. Outcome from this application is to enable automobile's user to diagnostics their own vehicles.

## 1.2 Outline of Project

The first work on automated traffic sign detection was reported in Japan in 1984. This attempt was followed by several methods introduced by different researchers to develop an efficient WARNING system and minimize all the issues stated above. An efficient WARNING system can be divided into several stages: preprocessing, detection, tracking, and recognition. In the preprocessing stage the visual appearance of images has been enhanced. Different colour and shape based approaches are used to minimize the effect of environment on the test images. The goal of traffic sign detection is to identify the region of interest (ROI) in which a traffic sign is supposed to be found and verify the sign after a large-scale search for candidates within an image. Different colour and shape based approaches are used by the researchers to detect the ROI. As the colour information can be unreliable due to illumination and weather change, shape based algorithm is introduced. The popular shape based approaches are Hough Transformation, Similarity Detection, Distance Transform Matching, and Edges with Haar-like features.

Car diagnostic tests scan your car's components and systems to check for issues with components like the engine, transmission, oil tank, throttle, and many more. Because car diagnostic tests require specific devices and expertise to read correctly, most tests are performed with mechanics or at dealer shops which becomes a time consuming process. However, you can also perform a car diagnostic test at home if you have the proper knowledge and equipment.

On-Board Diagnostic is a computer-based system developed by Automobile manufacturers for diagnosing car. It focuses on diagnosing the performance of car's engine to check any errors in car's engine components. OBD-II connect with DLC in the vehicle. Almost all the problem with the vehicle can be detect by OBD-II, such as exhaust manifold, ABS brake function, airbag (iSRS & SRS), inle performance, and so on. However, OBD was only for scanning the vehicle's diagnostics. To reading, processing, and showing the data, it needs Microcontroller. Microcontroller is a microprocessor system that installed in the chip. Microcontroller is different with microprocessor that used by PC because usually general microcontroller already filled with supporting minimal component system, such as memory and I/O interface. In another side, microprocessor usually only filled by CPU. Nowadays a lot of tools that can control device digitally. However, we have choosen two of the popular models and compared their feature. Those two models are, Arduino Uno and Raspberry Pi. Arduino is an open source microcontroller board that able to connect to computer and others devices via universal serial bus (USB). Arduino can be powered from any power supply such as computer, battery, and more. There a lot of various of Arduino to choose for every needs, to conform with our research we chose to use Arduino Uno

## 1.3 Aims and Objectives of the Project

The overall aim is to develop a system that can be used for traffic sign inventory. This system can assist local or national authorities in the task of maintaining and updating their road and traffic signs by automatically detecting and classifying one or more traffic signs from a complex scene (like the one shown in Figure 1.2) when captured by a camera from a vehicle. The main strategy is to find the right combination of colours in the scene so that one colour is located inside the convex hull of another colour and combine this with the right shape. If a candidate is found, the system tries to classify the object according to the rimpictogram combination and give the result of this classification.

The objectives are thus:

1. To understand the properties of road and traffic signs and their implications for image processing for the recognition task.
2. To understand colour, colour spaces and colour space conversion.
3. To develop robust colour segmentation algorithms that can be used in a wide range of environmental conditions.
4. To develop a recogniser that is invariant to in-plane transformations such as translation, rotation, and scaling based on invariant shape measures.
5. To identify the most appropriate approach for feature extraction from road signs.
6. To develop an appropriate road sign classification algorithm.
7. To evaluate the performance of the aforementioned methods for robustness under different conditions of weather, lighting geometry, and sign.

## **2.LITERATURE SURVEY**

### **1.Mi-JinKim, Jong-Wook Jang, Yun-Sik Yu( September 2010): A Study on In-Vehicle Diagnosis System using OBD- II with Navigation**

In this study vehicle navigation system is a representative driver support system that is available for the present path search and guiding functions. Its usability has been increasing. Under the present competitive situation because of the expanding navigation market to meet the customers' needs with regard to new services, differentiated services are dramatically increasing. In addition, the dashboard indicates the statuses of many of the vehicle's functions, all of which the driver must be aware of.

### **2. Hwa-seon Kim(2015): A Study on Development of Engine Fault Diagnostic System**

This study implemented a mobile diagnosing system that provides user-centered interfaces for more precisely estimating and diagnosing engine conditions through communications with the self-developed ECU only for industrial CRDI engine use. For the implemented system, a new protocol was designed and applied based on OBD-II standard to receive engine data values of the developed ECU. The designed protocol consists of a message structure to request data transmission from a smartphone to ECU and a response message structure for ECU to send data to a smartphone.

### **3.Hasan Fleyeh,Dalarna University(2006): Road and traffic sign detection and recognition**

This paper presents an overview of the road and traffic sign detection and recognition. It describes the characteristics of the road signs, the requirements and difficulties behind road signs detection and recognition, how to deal with outdoor images, and the different techniques used in the image segmentation based on the colour analysis, shape analysis. It shows also the techniques used for the recognition and classification of the road signs. Although image processing plays a central role in the road signs recognition, especially in colour analysis, but the paper points to many problems regarding the stability of the received information of colours.

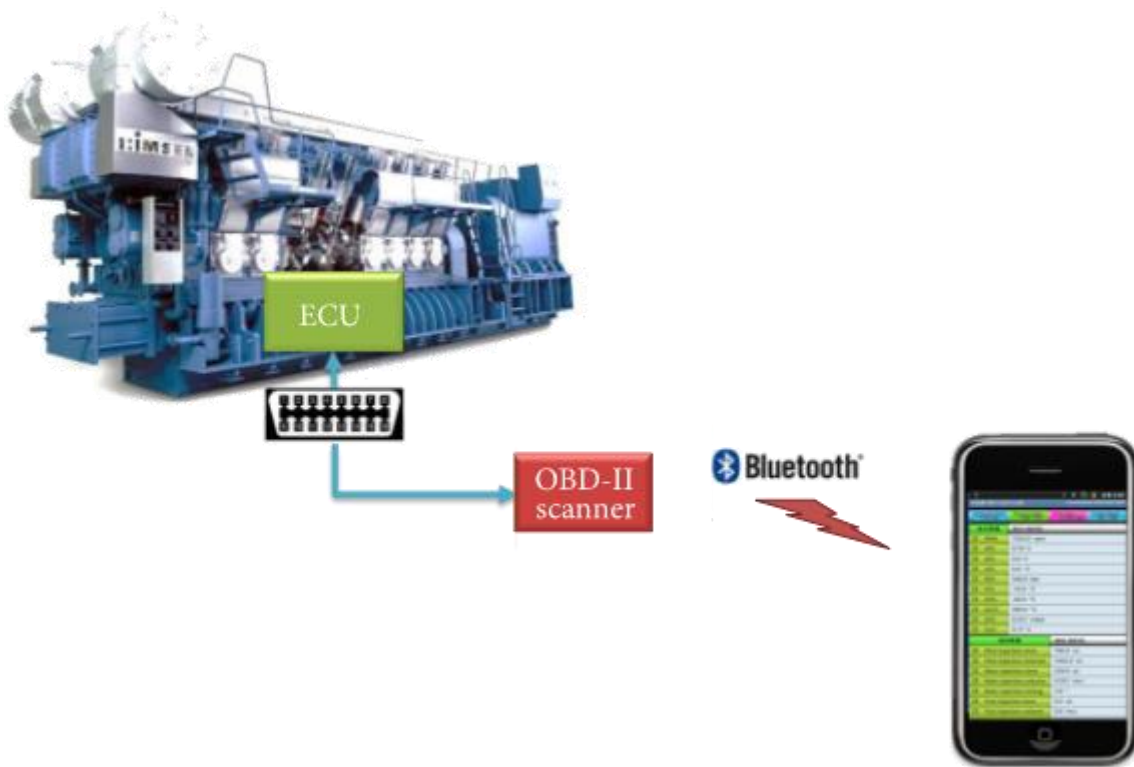
### **4.Akshata V.S and Subharn Panda(2019): TRAFFIC SIGN RECOGNITION AND CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS**

The traffic signs engraved on the roads these days enhance traffic safety by informing the driver of speed limits or any further possible dangers such as deep curvy roads, imminent repair road works.

### 3.METHODOLOGY

#### VEHICLE'S HEALTH DIAGNOSIS

This study developed a mobile diagnostic system based on OBD-II for the industrial CRDI engine. Figure 1 shows that this system is able to verify engine information and existence of malfunction therein by Bluetooth communication between the ECUs using OBD-II protocol.



With creation of the mobile application for engine diagnostic system, an administrator may confirm automotive information in real time without other devices. Drivers may always check the status of engine with smartphone which they always carry and may promptly respond to the malfunction in engine if any occurs.

For this experiment, an automotive simulator with which communication test could be conducted in the same way as an actual car was developed and tested.

This study developed a mobile diagnostic system based on OBD-II for the industrial CRDI engine.



ECUs using OBD-II protocol.

With creation of the mobile application for engine diagnostic system, an administrator may confirm automotive information in real time without other devices. Drivers may always check the status of engine with smartphone which they always carry and may promptly respond to the malfunction in engine if any occurs.

This study developed a mobile diagnostic system based on OBD-II for the industrial CRDI engine. This system is able to verify engine information and existence of malfunction therein by Bluetooth communication between the ECUs using OBD-II protocol.

For this experiment, an automotive simulator with which communication test could be conducted in the same way as an actual car was developed and tested. Figure 2 shows the OBD-II simulator which was personally manufactured. This consists of dongles for Bluetooth communication and OBD-II connector. Figure 3 is the screen which the developed ECU is equipped in the actual engine.

---

## 2.1 Design of Bluetooth OBD-II Protocol Structure

The OBD-II is a standard that visualizes the information on main system of vehicles or on failure transmitted from sensors attached to a vehicle to ECU from a center console or external device by using the serial communication function [3].

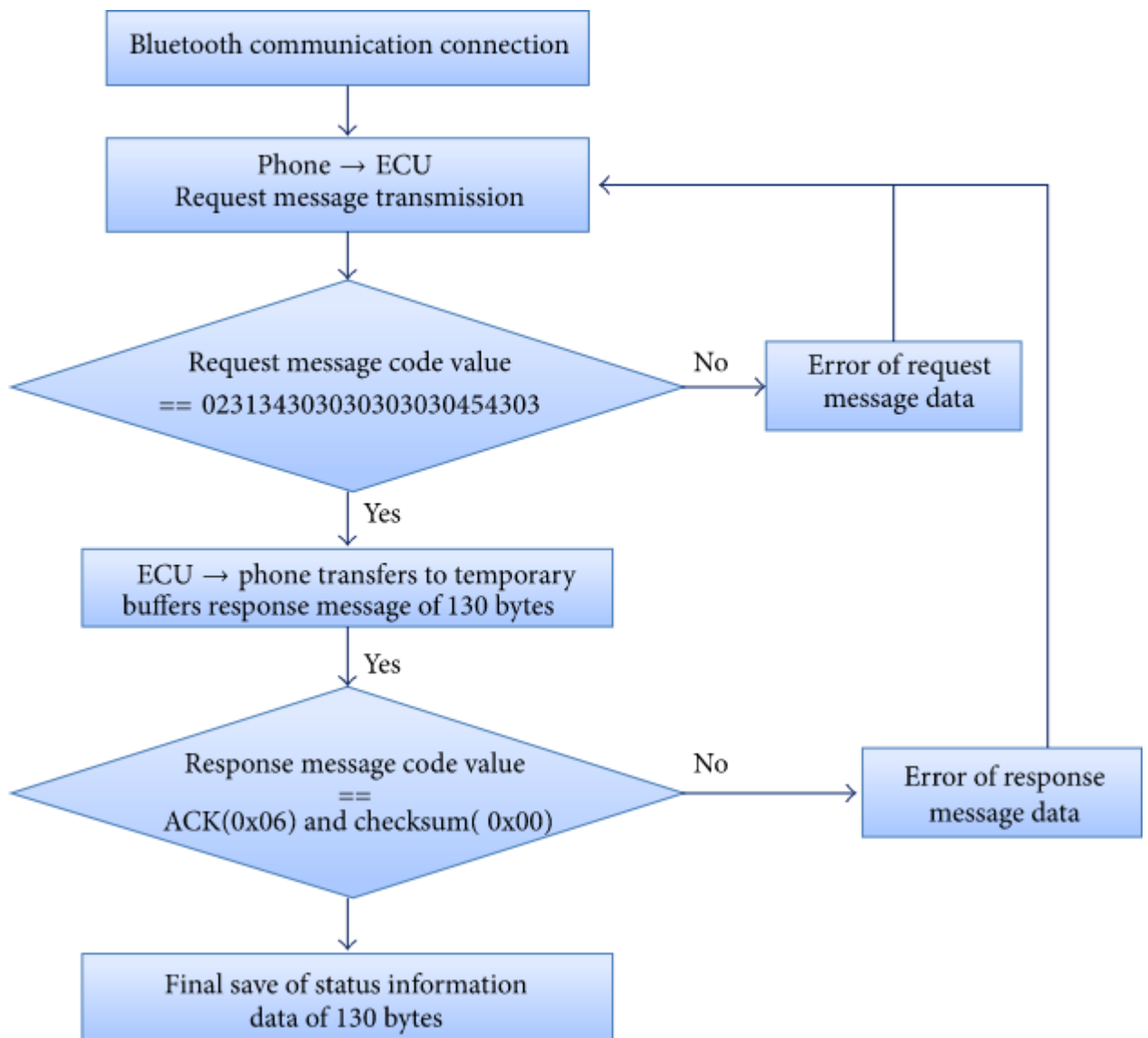
The OBD-II message format consists of 1-byte priority, target address, source address header, 7-byte data, and checksum and is basically used as a protocol for SAE-J1850 and ISO [4–6]. The CAN OBD message format consists of ID bits (11 or 29), DLC, 7 data bytes, and checksum (CRC-15 processing method) [7–9]. Table 1 shows OBD-II message format.

The developed OBD-II protocol has been manufactured based on the existing OBD-II standard. This differs from the existing OBD-II standard protocol structure. In case of the OBD-II protocol standard, automotive information of only one PID which was requested and can be read and responded. However, the developed industrial automotive OBD-II protocol read all the automotive information and transmits such information at once

---

## 2.2 OBD-II Protocol Structure for Obtaining Engine Status Information

The OBD-II message may be obtained from the automotive ECU by using automotive diagnostic tools. Figures 4 and 5 are proposed OBD-II protocol for this study. As seen in Table 2, the message consists of header, data, and checksum and saves 12 bytes of data in total and uses HEX codes.



## 2.3 OBD-II Protocol Structure for Obtaining Engine Trouble Code

There is a function to inform drivers that there is malfunction in the electronic control engine by lighting up the malfunction indicator lamp (MIL) and to set diagnostic trouble code (DTC) according to the details of malfunction and to automatically record such codes in the RAM of ECU if there is malfunction in electronic control engine or in exhaust gas related parts [13, 14]. This function was originally to set the OBD in order to easily verify the location to be inspected if automotive malfunction occurs but, thanks to speedy development of computers, it came to play a role of conducting ready-test (monitoring of exhaust gas equipment) as well as making freeze frame (function to record DTC on ECU) when malfunction occurs in input and output of ECU (computer) [15–17].

Therefore, a self-diagnosis function is the priority to be inspected when malfunction occurs in the car equipped with electronic control engine.

If disconnection or short circuit occurs in a sensor or actuator, ECU makes a comparison with preset voltage value and judges existence of malfunction and memorizes the preset DTC on the RAM. Such DTC information is memorized on the RAM of ECU (computer), so unless power provided for ECU is cut, such information is not deleted.

Table 5 shows the structure of DTC response message of ECU and Table 6 shows detailed code of OBD-II protocol DTC response message.

## 2.4 ECU Information Collection Algorithm

### 2.4.1. Algorithm for Obtaining Engine Status Information

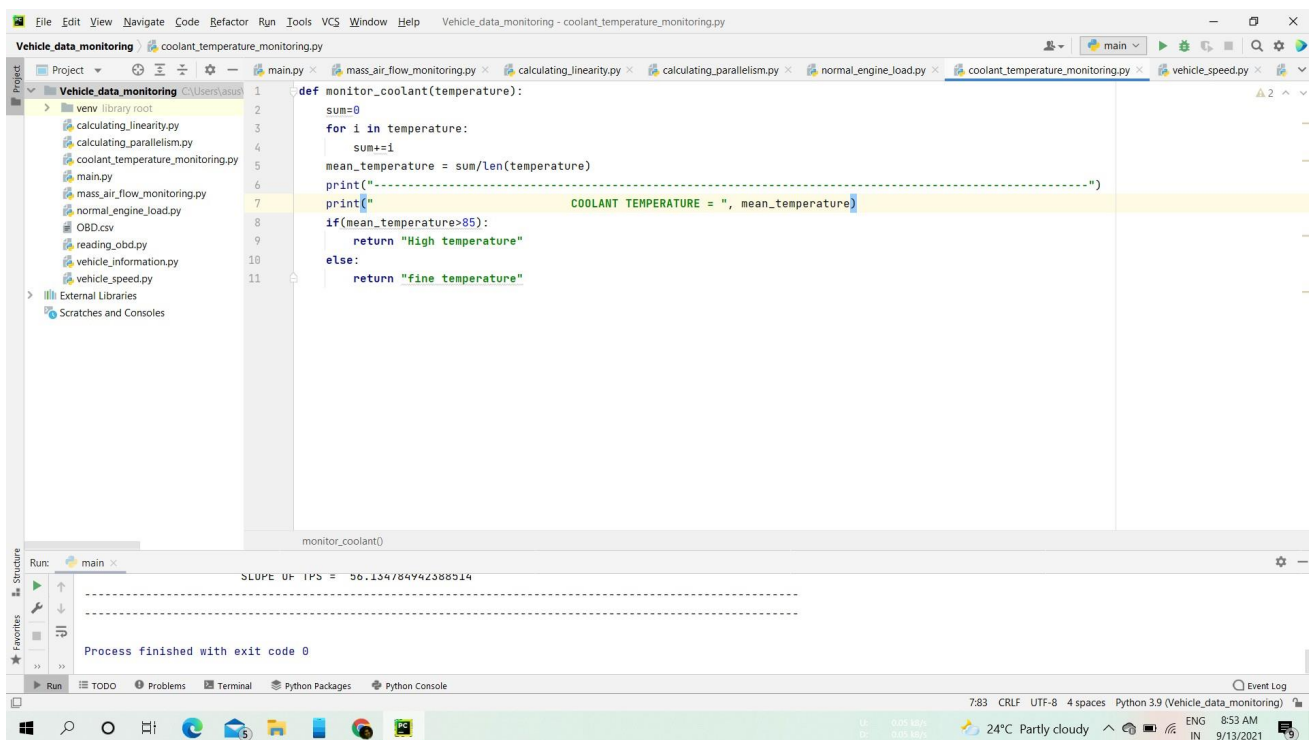
In order to collect the information of ECU, data are transmitted through the process as described in Figure 4.

First of all, if Bluetooth communication is connected, a data request message is transmitted to ECU. If the input request message is identical to 0231343030303030454303, ECU transmits OBD-II response message of 130 bytes to temporary buffers. All the data between STX and ETX are converted into HEX codes and sent. The calculation of checksum is of longitudinal redundancy check (LRC) and the lower byte of the sum of data between STX and ETX and checksum should be zero. If the response message is ACK(0x06) and the checksum value is 0x00, 31 automotive data of 130 bytes are finally saved.

## 2.4.2 Algorithm for Obtaining Engine Diagnostic Code

In order to collect DTC of ECU, such codes are transmitted through the process as described in Figure 5. Collection process of ECU diagnostic trouble codes is similar to the automotive information collection algorithm as explained above. First of all, if Bluetooth communication is connected, data request message is sent to ECU. If the input request message is identical to 023135303030303030454203, ECU transmits OBD-II response message of 14 bytes to temporary buffers. Then, if response message is ACK(0x06) and the checksum is 0x00, automotive information of 14 bytes becomes finally saved.

## COOLANT TEMPERATURE MONITORING CODE



```
1 def monitor_coolant(temperature):
2     sum=0
3     for i in temperature:
4         sum+=i
5     mean_temperature = sum/len(temperature)
6     print("-----")
7     print("COOLANT TEMPERATURE = ", mean_temperature)
8     if(mean_temperature>85):
9         return "High temperature"
10    else:
11        return "fine temperature"
```

```

1 import matplotlib.pyplot as plt
2 def speed_plotting(vehicle_speed, gps_speed):
3     plt.plot(gps_speed, color='b', label='gps_speed')
4     plt.ylabel('speed')
5     plt.plot(vehicle_speed, color='g', label='vehicle_speed')
6     plt.title('gps_speed and vehicle_speed')
7     plt.legend()
8     plt.show()
9
10 def speed_comparison(vehicle_speed, gps_speed):
11     sum = 0
12     for i in range(len(vehicle_speed)):
13         diff = vehicle_speed[i]-gps_speed[i]
14         if(diff<0):
15             diff=diff*-1
16         sum+=diff
17     mean_diff = sum/len(vehicle_speed)
18     print("-----")
19     print("      | MEAN DIFFERENCE IN VEHICLE SPEED AND GPS SPEED = ",mean_diff)
20     print()
21     if(mean_diff>6):
22         return " CHECK YOUR ENGINE "
23     else:
24         return "ENGINE IS FINE"
25
26 speed_comparison()

```

SLUPE UP IPS = 56.134784942388514

Process finished with exit code 0

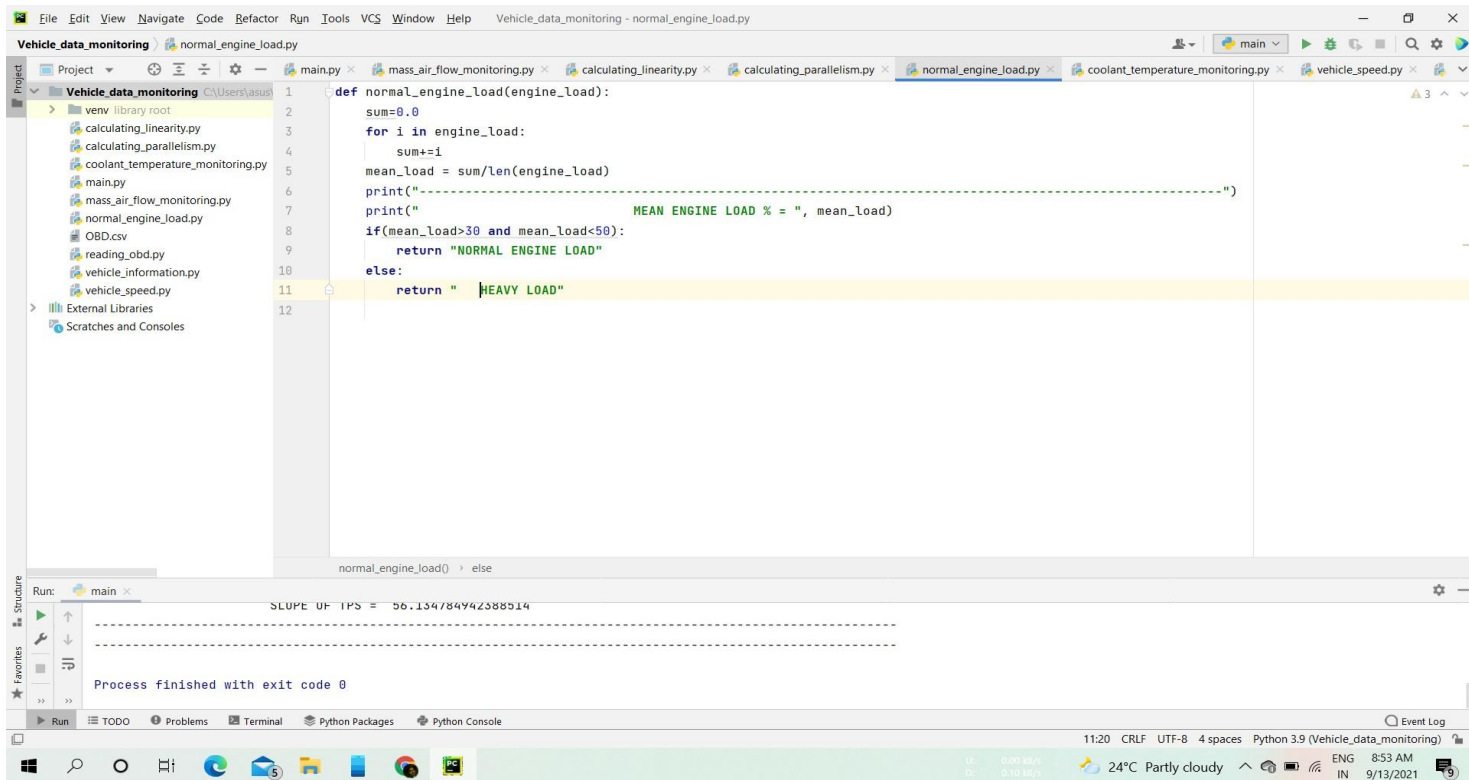
```

1 import csv
2 def read_csv_obd():
3     with open('OBD.csv') as csv_file:
4         csv_reader = csv.DictReader(csv_file, delimiter=',')
5         coolant_temperature = []
6         engine_rpm = []
7         gps_speed = []
8         intake_air_temperature = []
9         mass_air_flow = []
10        throttle_position = []
11        engine_load = []
12        for row in csv_reader:
13            #print(row)
14            coolant_temperature.append(float(row['Engine Coolant Temperature(°C)']))
15            gps_speed.append(float(row['GPS Speed (Meters/second)']))
16            engine_rpm.append(float(row['Engine RPM(rpm)']))
17            intake_air_temperature.append(float(row['Intake Air Temperature(°C)']))
18            engine_load.append(float(row['Engine Load(%)']))
19            mass_air_flow.append(float(row['Mass Air Flow Rate(g/s)']))
20            throttle_position.append(float(row['Throttle Position(Manifold)(%)']))
21            engine_load.append(float(row['Engine Load(%)']))
22            #print("coolant temperature : ",coolant_temperature)
23            #print("GPS speed : ",gps_speed)
24            #print("engine rpm : ",engine_rpm)
25
26 read_csv_obd()

```

SLUPE UP IPS = 56.134784942388514

Process finished with exit code 0



The screenshot shows an IDE window titled "Vehicle\_data\_monitoring - normal\_engine\_load.py". The project structure on the left includes files like "calculating\_linearity.py", "calculating\_parallelism.py", "coolant\_temperature\_monitoring.py", "main.py", "mass\_air\_flow\_monitoring.py", "normal\_engine\_load.py", "OBD.csv", "reading\_obd.py", "vehicle\_information.py", and "vehicle\_speed.py". The main editor displays the following Python code:

```
1 def normal_engine_load(engine_load):
2     sum=0.0
3     for i in engine_load:
4         sum+=i
5     mean_load = sum/len(engine_load)
6     print("-----")
7     print("MEAN ENGINE LOAD % = ", mean_load)
8     if(mean_load>30 and mean_load<50):
9         return "NORMAL ENGINE LOAD"
10    else:
11        return "HEAVY LOAD"
12
```

The bottom panel shows the "Run" output with the message "Process finished with exit code 0". The status bar at the bottom indicates the system time as 11:20, date as 9/13/2021, and weather as 24°C Partly cloudy.

## WARNING SYSTEM FOR DRIVER

### 1.1 Image Acquisition

The samples are collected from an inexpensive on board camera (Canon SX170 IS) which is connected to a laptop placed inside of a vehicle (Figure 1). The images were taken in different roads and highways in Malaysia under various weather conditions (Table 1) from 8:00 A.M. to 8:00 P.M. after every two seconds. The camera is placed in the left side of the dashboard so that it can capture the traffic sign of left side. The aim of this section is to create a database of traffic sign images under different variations.





**Figure 1**

**Model for sample collections (a) used car with a camera placed on the left side of the dashboard, (b) camera setup including a laptop, and (c) on road camera range and sign detection.**

## 1.2 Image Preprocessing

Image preprocessing is an important part of the TSDR system whose main idea is to remove low-frequency background noise, normalising the intensity of the individual particles images, removing reflections, and masking portions of images. Below is a description of selected image preprocessing techniques. The input image is divided into channels R, G, and B separately. In the proposed approach, filters are applied on each channel threshold to select those regions of the image where the values of the pixels fall in the range of our target object. For example, for traffic signs with a red background (such as stop signs), the threshold for channel R is pixels with values in the range of 90–255 and for channels G and B the range is 0–70. The region of interest (ROI) is the logical sum of the three filtered channels of R, G, and B.

## 1.3 Shape Matching Based Detection

The idea is to use colour characteristic of the preferred object to accelerate the procedure without employing model-based classifiers which is a time consuming process [40–42]. After filtering and analysing the features of the detected object, the candidates of the traffic sign are selected based on shape matching. The flow chart of the system is shown in Figure 2.

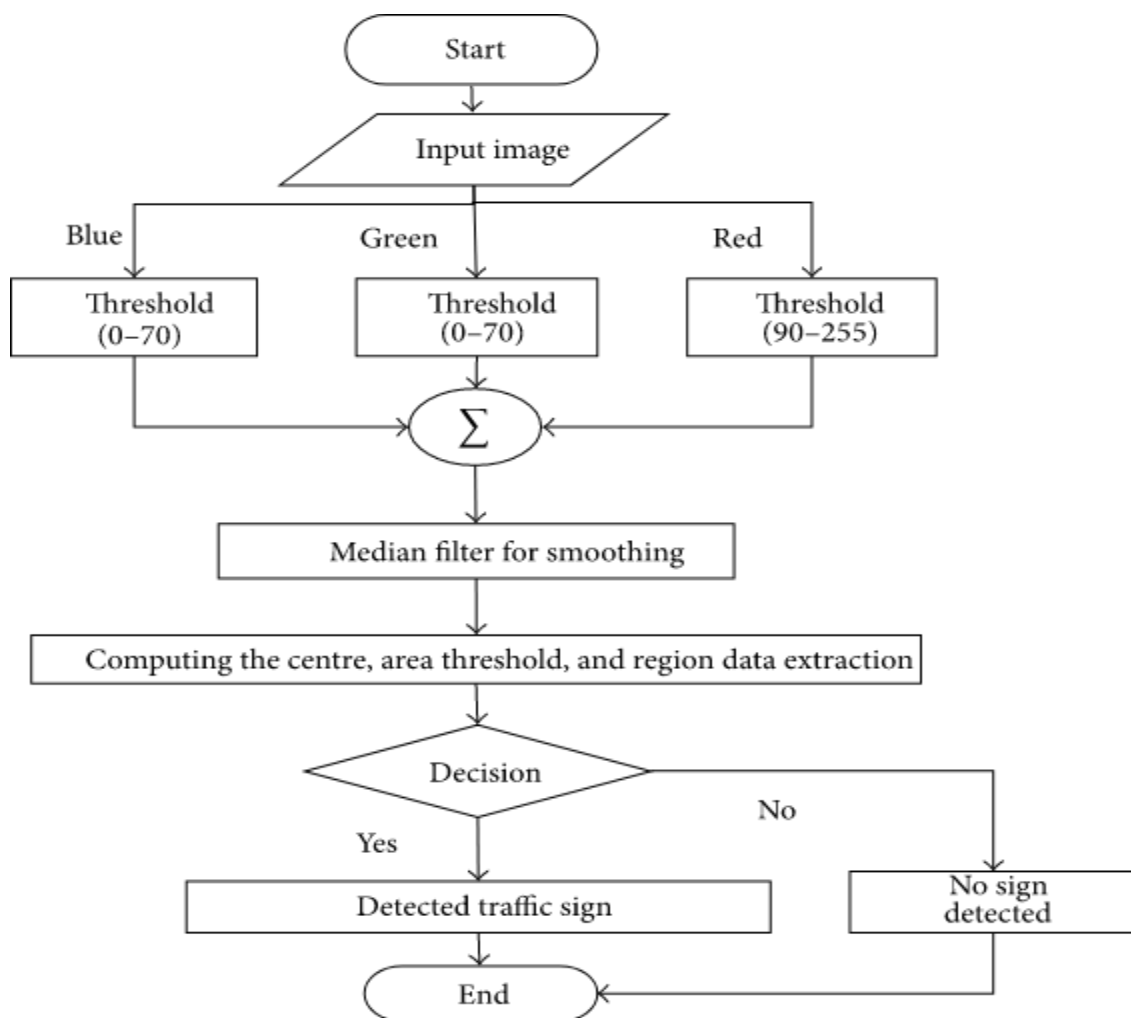


FIGURE 1



## 1.4 Shape Matching and Candidate Selection

As almost all traffic signs containing red colour are round or octagonal, the proposed method drew on these common shapes to detect hypothetical shapes which are close to traffic signs. Those regions with  $K$  in the range of 0.7–1.3 are accepted as candidates for traffic signs:

$$k = a/\pi * w^2.$$

where ‘a’ is the area of the region and ‘w’ is the longest width.

## 1.5 Traffic Sign Detection

The area range for road signs determines the distance in which the system can detect the traffic sign. Outside of this range, objects with the same range of pixels value cannot be traffic signs. In this level, crucial information such as centre, area, and longest width of each region is calculated. This information is used to decide whether or not each region is a traffic sign.

# SYSTEM DESIGN

## 1. Hardware Implementation

### 1.1 ECU

The use of the term ECU may be used to refer to an Engine Control Unit, however ECU also refers to an Electronic Control Unit, which is a component of any automotive mechatronic system, not just for the control of an engine.

In the Automotive industry, the term ECU often refers to an Engine Control Unit (ECU), or an Engine Control Module (ECM). If this unit controls both an engine and a transmission, it is often described as a **Powertrain Control Module (PCM)**.



Fundamentally, the engine ECU controls the injection of the fuel and, in petrol engines, the timing of the spark to ignite it. It determines the position of the engine's internals using a Crankshaft Position Sensor so that the injectors and ignition system are activated at precisely the correct time. While this sounds like something that can be done mechanically (and was in the past), there's now a bit more to it than that.

## 1.2 OBD II- protocol

Fundamentally, the engine ECU controls the injection of the fuel and, in petrol engines, the timing of the spark to ignite it. It determines the position of the engine's internals using a Crankshaft Position Sensor so that the injectors and ignition system are activated at precisely the correct time. While this sounds like something that can be done mechanically (and was in the past), there's now a bit more to it than that.

An internal combustion engine is essentially a big air pump that powers itself using fuel. As the air is sucked in, enough fuel has to be provided to create power to sustain the engine's operation while having a useful amount left over to propel the car when required. This combination of air and fuel is called a 'mixture'. Too much mixture and the engine will be full throttle, too little and the engine will not be able to power itself or the car.

Fundamentally, the engine ECU controls the injection of the fuel and, in petrol engines, the timing of the spark to ignite it. It determines the position of the engine's internals using a Crankshaft Position Sensor so that the injectors and ignition system are activated at precisely the correct time. While this sounds like something that can be done mechanically (and was in the past), there's now a bit more to it than that.

An internal combustion engine is essentially a big air pump that powers itself using fuel. As the air is sucked in, enough fuel has to be provided to create power to sustain the engine's operation while having a useful amount left over to propel the car when required. This combination of air and fuel is called a 'mixture'. Too much mixture and the engine will be full throttle, too little and the engine will not be able to power itself or the car.

OBD2 qualifies as a protocol for vehicle diagnostics communication (query-response communication). OBD2 is a part of emission legislation. Hence it follows the CARB (California Air Resources Board) initiated protocol mandates. OBD2 protocol monitors parameters in emission relevant ECUs as per the guidelines of CARB. On-board diagnostics 2 also follows CARB guidelines for data storage and data accessibility to be provided to the external scan tools.

The external scan tool requirements (defined in ISO 15031) mandate that the following modes should be supported.

### **OBD data collected**

docs.google.com/spreadsheets/d/1l-Pt0xufnCG8gVw6PSQ668GVVledrfmOOuGYIsKDEM/edit?gid=232033983

OBD

File Edit View Insert Format Data Tools Add-ons Help Last edit was 16 hours ago

90% 123 Arial 10 B I A

A1:E1	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
	GPS Speed (Meters/second)	Horizontal Dilution of Precision	Altitude	Bearing	G(x)	G(y)	G(z)	G(calibrated)	Engine Coolant Temperature	Engine RPM(rpm)	Intake Air Temperature	Engine Load(%)	Mass Air Flow Rate	Throttle Position(Manifold)(%)	
1	2.7951565	93	250	211.6	0.3504345	9.163701	-2.9478176	9.10E-04	56	2623.25	30	92.54901886	33.75999832	32.54901886	
2	0	98	255	0	-6.745979	4.3944516	-4.0395737	-0.06278998	56	3169.5	30	90.98039246	37.20000076	32.54901886	
3	0	105	247	0	-0.9232808	7.7367573	3.8708984	-0.0941295	56	2587.5	30	98.4313736	35.18000031	37.64706039	
4	0	101	225	0	-4.7923107	2.1726327	16.387754	0.7735128	57	2753	30	98.03921509	38.04000092	39.21568686	
5	0	101	225	0	-0.061368175	7.1908793	6.6864495	0.038574874	57	2957	30	96.86274719	39.34999847	38.82352829	
6	0	101	225	0	0.69519955	7.344568	6.993827	0.050246455	57	3158.5	30	97.2549057	41.93000031	40	
7	0	101	225	0	0.3312809	7.181762	7.118325	0.050316274	57	3430.25	30	95.29412079	48.09000015	40.39215851	
8	0	101	225	0	1.164463	7.574411	5.586036	-0.014310896	58	2927.25	30	95.49019623	4.010000229	16.47058868	
9	0	101	225	0	0.8963125	6.7795362	7.9610844	0.08882338	58	2267	30	77.64705658	27.78000069	29.41176605	
10	0	101	225	0	1.1836166	6.5496926	7.616319	0.050063074	58	2353.75	30	93.33333588	31.42000008	31.76470566	
11	0	101	225	0	1.3847296	7.4403358	8.037899	0.14437789	59	2325.75	30	94.90196228	31.54000092	32.15686417	
12	0	101	225	0	-0.2529043	6.645461	7.635473	0.051165044	59	2303	30	93.72549438	30.25	32.15686417	
13	0	101	225	0	0.3027803	5.475941	7.408311	0.058579795	60	1685.75	30	34.11764908	4.289999962	16.47058868	
14	0	101	225	0	0.11124419	5.7153616	7.3700037	-0.01134787	60	1258.5	30	34.90196228	4.260000229	16.47058868	
15	0	101	225	0	0.5805077	5.6770544	7.3412733	-0.021725193	60	1507.75	30	28.23529434	4.139999866	16.47058868	
16	0	101	225	0	-0.6261699	5.9643583	6.948624	-0.045349956	61	1686.5	30	63.52941132	15.42000008	24.70588303	
17	0	101	225	0	0.7816206	6.2133555	6.3835926	-0.0694378	61	1667	30	94.11764526	21.10000038	28.62745094	
18	0	101	225	0	-1.066703	5.533402	6.776242	-0.08260286	61	2378.75	30	100	31.87000084	37.64706039	
19	0	101	225	0	-1.1433175	5.2652516	8.471336	0.04240471	61	2489.75	30	95.29412079	33.84000015	37.25490189	
20	0	101	225	0	0.27404988	5.926051	8.442606	0.07083316	62	2089	30	40	6.179999828	16.07843208	
21	0	101	225	0	-0.79855245	5.3227124	8.298954	0.027306855	62	2012.5	30	48.23529434	11.89999962	20.7843132	
22	0	101	225	0	-1.0571262	4.1351886	7.4370413	-0.10691422	63	1779.5	30	24.70588303	3.279999971	17.25490189	
23	0	101	225	0	-0.4442106	5.025831	7.207198	-0.084187865	63	1708.25	30	22.74509811	3.25999999	16.07843208	
24	0	101	225	0	-0.23352085	4.93964	7.2838125	-0.08355957	63	1428	30	24.70588303	3.25999999	16.07843208	
25	0	101	225	0	-0.109022364	5.495095	8.567104	0.05657047	63	1376.5	30	25.88235283	3.210000038	16.07843208	
26	0	101	225	0											

Count: 5 Explore

OBD.csv LiquidCrystal\_I2C-1....zip FU7G07KIWMQTN....ino CAN-Bus\_Shield-m....zip CAN-0.3.1.zip vehicle\_diagnostic.pdf Show all

U: 0.19 KB/s D: 0.18 KB/s 24°C Partly cloudy ENG IN 11:56 PM 9/12/2021

Vehicle\_data\_monitoring - reading\_obd.py

Vehicle\_data\_monitoring reading\_obd.py

```

import csv
def read_csv_obd():
    with open('OBD.csv') as csv_file:
        csv_reader = csv.DictReader(csv_file, delimiter=',')
        coolant_temperature = []
        engine_rpm = []
        gps_speed = []
        intake_air_temperature = []
        mass_air_flow = []
        throttle_position = []
        engine_load = []
        for row in csv_reader:
            #print(row)
            coolant_temperature.append(float(row['Engine Coolant Temperature(°C)']))
            gps_speed.append(float(row['GPS Speed (Meters/second)']))
            engine_rpm.append(float(row['Engine RPM(rpm)']))
            intake_air_temperature.append(float(row['Intake Air Temperature(°C)']))
            engine_load.append(float(row['Engine Load(%)']))
            mass_air_flow.append(float(row['Mass Air Flow Rate(g/s)']))
            throttle_position.append(float(row['Throttle Position(Manifold)(%)']))
            engine_load.append(float(row['Engine Load(%)']))
        #print("coolant temperature : ",coolant_temperature)
        #print("GPS speed : ",gps_speed)
        #print("engine rpm : ",engine_rpm)
    read_csv_obd() with open('OBD.csv') as csv_file

```

SLUPE UF TPS = 56.134/84942388514

Process finished with exit code 0

4:63 CRLF UTF-8 4 spaces Python 3.9 (Vehicle\_data\_monitoring)

U: 0.00 KB/s D: 0.00 KB/s 24°C Partly cloudy ENG IN 8:53 AM 9/13/2021

## 1.2.1 Engine load

The engine load is the torque output of the engine.

An internal combustion engine is - approximately - a constant torque motor, meaning it can produce the same maximum torque at any rpm. The fact that it happens at a rpm or another, dictates how much power it produces.

Torque is proportional to the amount of force put on the piston, which is proportional to the amount of air being burnt in the combustion chamber, which is why you can relate the engine load to the quantity of air going into the engine, knowing the maximum amount of air that go in at a particular rpm, i.e. at WOT.

RPM does not really depends on throttle opening per say. When an engine is at constant speed, there is only the friction force going against the engine (from the bearings) and some others to run pumps and such, such that the engine can run. This represents the load at idle.

If you give enough heat (fuel combusted) to produce enough torque to go against this load, the engine will idle at constant speed.

The engine load is the torque output of the engine.

An internal combustion engine is - approximately - a constant torque motor, meaning it can produce the same maximum torque at any rpm. The fact that it happens at a rpm or another, dictates how much power it produces.

Torque is proportional to the amount of force put on the piston, which is proportional to the amount of air being burnt in the combustion chamber, which is why you can relate the engine load to the quantity of air going into the engine, knowing the maximum amount of air that go in at a particular rpm, i.e. at WOT.

RPM does not really depends on throttle opening per say. When an engine is at constant speed, there is only the friction force going against the engine (from the bearings) and some others to run pumps and such, such that the engine can run.

```
def normal_engine_load(engine_load):
    sum=0.0
    for i in engine_load:
        sum+=i
    mean_load = sum/len(engine_load)
    print("-----")
    print("MEAN ENGINE LOAD % = ", mean_load)
    if(mean_load>30 and mean_load<50):
        return "NORMAL ENGINE LOAD"
    else:
        return "HEAVY LOAD"
```

Run: main x

SLUPE UF TPS = 56.134784942388514

Process finished with exit code 0

### 1.2.2 Engine Coolant temperature

The engine load is the torque output of the engine.

An internal combustion engine is - approximately - a constant torque motor, meaning it can produce the same maximum torque at any rpm. The fact that it happens at a rpm or another, dictates how much power it produces.

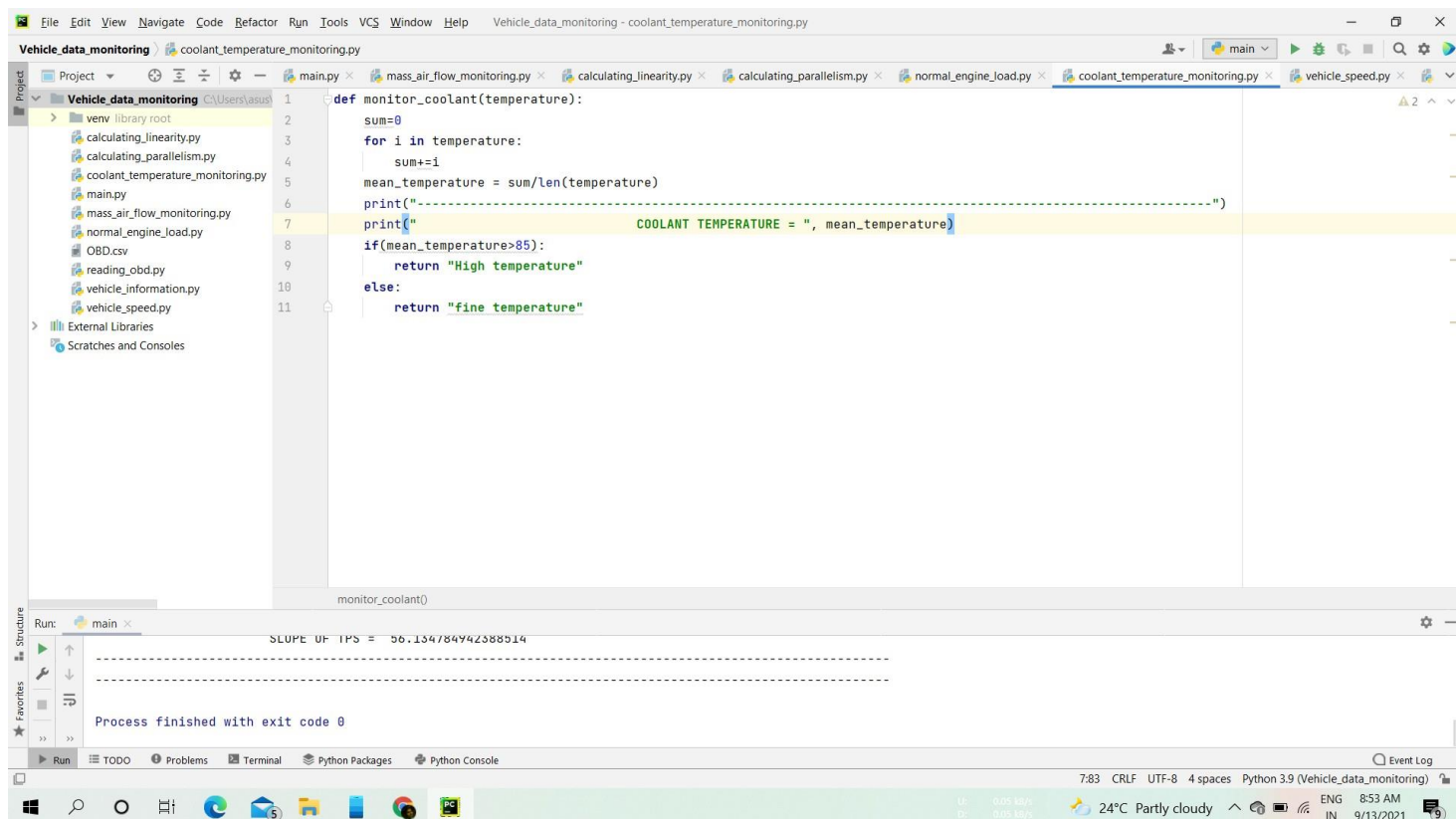
Torque is proportional to the amount of force put on the piston, which is proportional to the amount of air being burnt in the combustion chamber, which is why you can relate the engine load to the quantity of air going into the engine, knowing the maximum amount of air that go in at a particular rpm, i.e. at WOT.

RPM does not really depends on throttle opening per say. When an engine is at constant speed, there is only the friction force going against the engine (from the bearings) and some others to run pumps and such, such that the engine can run. This represents the load at idle.



If you give enough heat (fuel combusted) to produce enough torque to go against this load, the engine will idle at constant speed.

The engine cooling system is comprised of engine coolant, passages inside the engine block and cylinder head(s), a water pump to circulate the coolant, a thermostat to control the temperature of the coolant, a radiator to cool the coolant, a fan to pull air through the radiator, a radiator cap to control the pressure in the system, and interconnecting hoses to transfer the coolant from the engine to radiator, and also to the vehicle's heating system where hot coolant is used to warm the cabin of the vehicle.



The screenshot shows an IDE window titled "Vehicle\_data\_monitoring - coolant\_temperature\_monitoring.py". The left sidebar displays a project structure for "Vehicle\_data\_monitoring" with files like "main.py", "mass\_air\_flow\_monitoring.py", "calculating\_linearity.py", "calculating\_parallelism.py", "normal\_engine\_load.py", "coolant\_temperature\_monitoring.py", "vehicle\_speed.py", "OBD.csv", "reading\_obd.py", "vehicle\_information.py", and "vehicle\_speed.py". The main editor shows the code for "monitor\_coolant()":

```
1 def monitor_coolant(temperature):
2     sum=0
3     for i in temperature:
4         sum+=1
5     mean_temperature = sum/len(temperature)
6     print("-----")
7     print("COOLANT TEMPERATURE = ", mean_temperature)
8     if(mean_temperature>85):
9         return "High temperature"
10    else:
11        return "fine temperature"
```

The bottom of the IDE shows a "Run" window with the output "SLUPE UF TPS = 56.134784942388514" and a message "Process finished with exit code 0". The status bar at the bottom indicates "7:83 CRLF UTF-8 4 spaces Python 3.9 (Vehicle\_data\_monitoring)" and the system tray shows "24°C Partly cloudy" and "8:53 AM 9/13/2021".

### **1.1.1 MAF(Mass Air Flow)**

The engine cooling system is comprised of engine coolant, passages inside the engine block and cylinder head(s), a water pump to circulate the coolant, a thermostat to control the temperature of the coolant, a radiator to cool the coolant, a fan to pull air through the radiator, a radiator cap to control the pressure in the system, and interconnecting hoses to transfer the coolant from the engine to radiator, and also to the vehicle's heating system where hot coolant is used to warm the cabin of the vehicle.

A hot-wire mass air flow sensor has a small electrically heated wire (hot wire). A temperature sensor installed close to the hot wire measures the temperature of the air near the hot wire.

When the engine is idling, a small amount of air flows around the hot wire, so it takes a very low electric current to keep the wire hot when you press the gas, the throttle opens allowing more air to flow over the hot wire. The passing air cools the wire down. The more air flows over the wire, the more electrical current is needed to keep it hot. The electric current is proportional to the amount of air flow. A small electronic chip installed inside the air flow sensor translates the electric current into a digital signal and sends it to the engine computer (PCM). The PCM uses the air flow signal to calculate how much fuel to inject. The goal is to keep the air/fuel ratio at the optimal level.



## 1.1.2 GPS SPEED

How accurate is GPS for speed measurement?

As with positioning, the speed accuracy of GPS depends on many factors.

The government provides the GPS signal in space with a global average user range rate error (URRE) of  $\leq 0.006$  m/sec over any 3-second interval, with 95% probability.

This measure must be combined with other factors outside the government's control, including satellite geometry, signal blockage, atmospheric conditions, and receiver design features/quality, to calculate a particular receiver's speed accuracy.

How accurate is GPS for timing?

GPS time transfer is a common method for synchronizing clocks and networks to Coordinated Universal Time (UTC). The government distributes UTC as maintained by the U.S. Naval Observatory (USNO) via the GPS signal in space with a time transfer accuracy relative to UTC(USNO) of  $\leq 30$  nanoseconds (billionths of a second), 95% of the time. This performance standard assumes the use of a specialized time transfer receiver at a fixed location.

Is military GPS more accurate than civilian GPS?

The user range error (URE) of the GPS signals in space is actually the same for the civilian and military GPS services. However, most of today's civilian devices use only one GPS frequency, while military receivers use two.

Using two GPS frequencies improves accuracy by correcting signal distortions caused by Earth's atmosphere. Dual-frequency GPS equipment is commercially available for civilian use, but its cost and size has limited it to professional applications

### 1.1.3 Speedometer

Car engine rpm increases as you press the accelerator, as does power — at least to a point. An engine doesn't necessarily produce its maximum power at its highest rpm. Engine specifications typically present the peak horsepower figure followed by the rpm at which it occurs, such as 252 hp at 5,600 rpm. Torque, a measure of the engine's instantaneous twisting force, typically comes at lower rpm and may appear as a range in turbocharged or supercharged engines, such as 273 pounds-feet at 1,600-4,500 rpm.

Many cars have a tachometer to indicate engine rpm, usually measured in thousands. At the top of the tachometer range is a zone called the redline — usually highlighted literally, with a red line. Revving the engine beyond redline can cause damage. This is really a concern only for cars equipped with a manual transmission; vehicles with an automatic transmission are programmed to shift before the engine speed reaches that point. That will vary, too, depending on how hard you're pressing the accelerator pedal.

Suppose your motor vehicles speedometer not working and you need to know your vehicle speed for the given engine speed for a certain gear ratio. How you can do that , let's see..

Say, Engine rpm= 3000

Gear ratio=1.5

Axle ratio = 4

Tyre size = 12 inch radius

The formula for calculating vehicle speed is:

Car or automobile speed in consistent unit,

$$V = (\text{engine rpm} * \text{wheel tyre perimeter}) / (\text{gear ratio} * \text{axle ratio})$$

$$\text{Or, } V = (\text{engine rpm} * 3.14 * \text{tyre diameter}) / (\text{gear ratio} * \text{wheel axle ratio})$$

For the given values,

$$V = (3000 \text{ rpm} * 3.14 * 2 * 12 \text{ inch}) / (1.5 * 4)$$

$$\text{Or, } V (\text{kmph}) = (3000 * 60 \text{ rph} * 3.14 * 2 * 12 * 25.4 * 0.000001 \text{ km}) / (1.5 * 4)$$

$$\text{Or, } V = 57.42 \text{ kmph}$$

```

File Edit View Navigate Code Refactor Run Tools VCS Window Help Vehicle_data_monitoring - vehicle_speed.py
Vehicle_data_monitoring vehicle_speed.py
Project
  Vehicle_data_monitoring C:\Users\asus\
    venv library root
    calculating_linearity.py
    calculating_parallelism.py
    coolant_temperature_monitoring.py
    main.py
    mass_air_flow_monitoring.py
    normal_engine_load.py
    OBD.csv
    reading_obd.py
    vehicle_information.py
    vehicle_speed.py
  External Libraries
  Scratches and Consoles
main.py
1 import matplotlib.pyplot as plt
2 def speed_plotting(vehicle_speed, gps_speed):
3     plt.plot(gps_speed, color='b', label='gps_speed')
4     plt.ylabel('speed')
5     plt.plot(vehicle_speed, color='g', label='vehicle_speed')
6     plt.title('gps_speed and vehicle_speed')
7     plt.legend()
8     plt.show()
9 def speed_comparison(vehicle_speed, gps_speed):
10     sum = 0
11     for i in range(len(vehicle_speed)):
12         diff = vehicle_speed[i]-gps_speed[i]
13         if(diff<0):
14             diff=diff*-1
15         sum+=diff
16     mean_diff = sum/len(vehicle_speed)
17     print("-----")
18     print(" | MEAN DIFFERENCE IN VEHICLE SPEED AND GPS SPEED = ",mean_diff)
19     print()
20     if(mean_diff>6):
21         return " CHECK YOUR ENGINE "
22     else:
23         return "ENGINE IS FINE"
speed_comparison()
Run: main
SLUPE UF TPS = 56.134784942388514
-----
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
18:27 CRLF UTF-8 4 spaces Python 3.9 (Vehicle_data_monitoring)
24°C Partly cloudy 8:53 AM 9/13/2021

```

## 1.2 ELM-327 Adapter

The ELM327 is a programmed microcontroller produced by ELM Electronics for translating the on-board diagnostics (OBD) interface found in most modern cars. The ELM327 command protocol is one of the most popular PC-to-OBD interface standards and is also implemented by other vendors.

The original ELM327 is implemented on the PIC18F2480 microcontroller from Microchip Technology.

ELM327 is one of a family of OBD translators from ELM Electronics. Other variants implement only a subset of the OBD protocols

The ELM327 is a [PIC microcontroller](#) that has been customized with ELM Electronics' proprietary code that implements the testing protocols. When ELM Electronics sold version 1.0 of its ELM327, it did not enable the [copy protection](#) feature of the PIC microcontroller. Consequently, anyone could buy a genuine ELM327, and read ELM's proprietary binary microcontroller software using a [device programmer](#). With this software, pirates could trivially produce ELM327 clones by purchasing the same microcontroller chips and programming them with the copied code.<sup>[6][7]</sup> ELM327 copies were widely sold in devices claiming to contain an ELM327 device, and problems have been reported with the copies.<sup>[8]</sup> The problems reflect bugs that were present in ELM's version 1.0 microcode; those making the clones may continue to sell the old version.

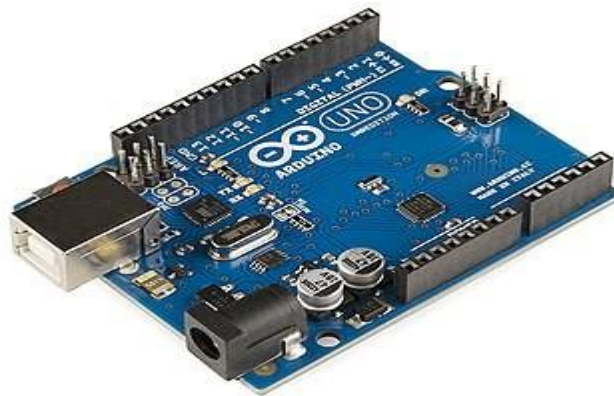


## 1.3 Arduino Uno

The **Arduino Uno** is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. It is similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available.

The Arduino project started at the Interaction Design Institute Ivrea (IDII) in Ivrea, Italy. At that time, the students used a BASIC Stamp microcontroller, at a cost that was a considerable expense for many students. In 2003, Hernando Barragán created the development platform *Wiring* as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas, who are known for work on the Processing language. The project goal was to create simple, low-cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a printed circuit board (PCB) with an ATmega168 microcontroller, an IDE based on Processing, and library functions to easily program the microcontroller. In 2003, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, added support for the cheaper ATmega8 microcontroller to Wiring. But instead of continuing the work on Wiring, they forked the project and renamed it *Arduino*. Early arduino boards used the FTDI

USB-to-serial driver chip and an ATmega168. The Uno differed from all preceding boards by featuring the ATmega328P microcontroller and an ATmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.



## **SIGN BOARD DETECTION**

### **COLOR CODING**

The most widely used color space is the color space, where a color point in the space is characterized by three color components of the corresponding pixel which are red (R), green (G), and blue (B). However since there exist a lot of color spaces, it is useful to classify them into fewer categories with respect to their definitions and properties. Vandenbroucke proposed the classification of the color spaces into the following categorie

<b>Colour</b>	<b>Meaning</b>
Red	Exclusively for STOP and YIELD signs, DO NOT ENTER signs, and it is used in the warning signs and forbidden signs.
Black	Used as information colour in some of the warning signs, prohibitory signs and information signs.
White	Used as background for route markers, guide signs, and certain regulatory signs, and as message colour on signs with brown, green, blue, black, and red backgrounds.
Orange	Used as background colour for construction and maintenance signs.
Yellow	Used as background colour for Warning signs and Prohibitory signs, and some of the supplementary signs.
Green	Used as background colour for express roads.
Blue	Used as background colour for regulatory, information signs, and supplementary signs.

## EDGE DETECTION

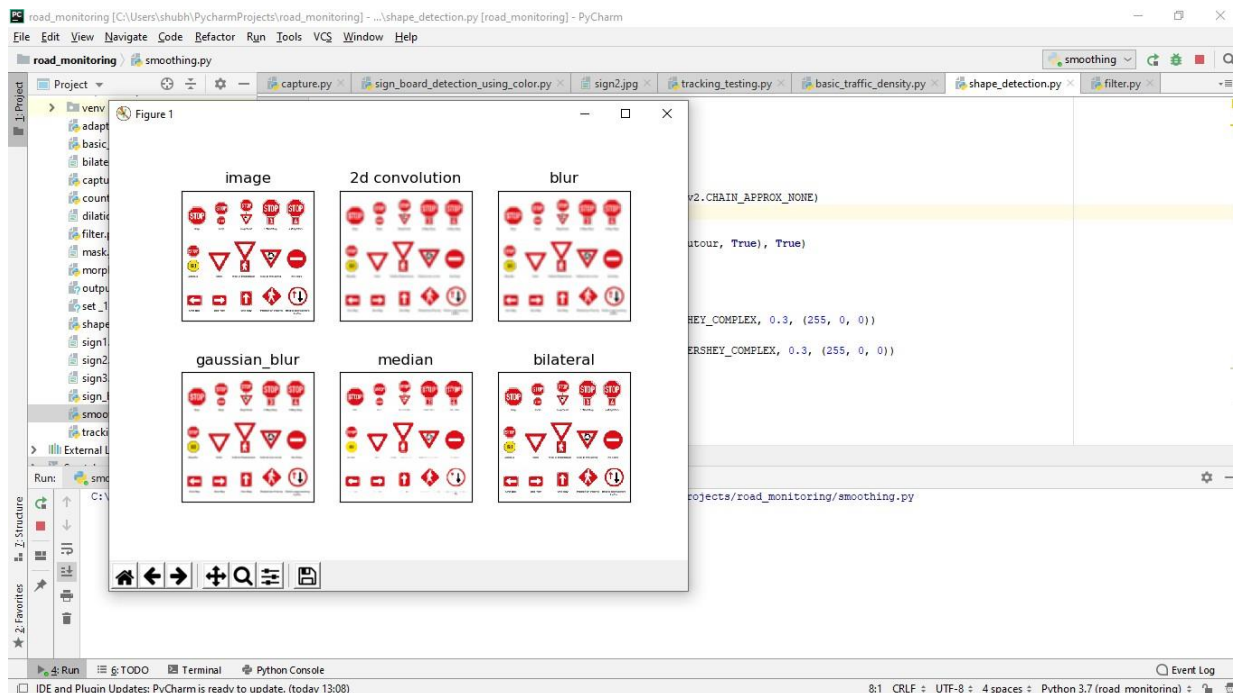
Edge detection is a type of image segmentation techniques which determines the presence of an edge or line in an image and outlines them in an appropriate way.

The main purpose of edge detection is to simplify the image data in order to minimize the amount of data to be processed. Generally, an edge is defined as the boundary pixels that connect two separate regions with changing image amplitude attributes such as different constant luminance and tristimulus values in an image and. There are different approaches and

algorithm to find out the edge in image processing that, in the meantime, canny operator due to high accuracy and low processing volume has a more favorable performance compared to other methods for our database.

## **MEDIAN FILTER**

In signal processing it is necessary to perform some kind of noise reduction on an image or signal. The median filter is a nonlinear digital filtering technique, used to remove noise. Also used to remove the noise of sign boards outline or edges. It is pre-processing step to improve the result of later processing e.g: edge detection on an image. The main idea of the median filter is to run through the signal entry by entry, replacing each entry with the media of neighboring entries. The pattern of neighbors is called the “window”, which slides, entry by entry, over the entire signal. For 1D signal, the most obvious window is just the first few preceding and following entries, whereas for 2D signals such as images, more complex window patterns are possible. If the window has an odd number of entries, then the median is simple to define: it is just the middle value after all the entries in the window are sorted numerically. For an even number of entries, there is more than one possible median.



## SHAPE DETECTION

Mathematical morphology is the branch of image processing that argues about shape and appearance of object in images. The erosion and dilation operators are basically operators of mathematical morphology that are used in this part to improve the edge detection image. At this step, first erosion action is applied in the edge detection image. After erosion action on image, the dilation action is done..





















## TRAFFIC SIGN

- **STOP Sign:** This is the most easily recognizable sign. It is important for every kid to know that a Stop sign is put up at places where the driver needs to bring his vehicle to a halt. This allows the kids to pass through the road without any risk of an accident. The kids should know to cross a road only where Stop sign is put and only after the vehicles have come to a stop to let the children make their way through the road.
- **School Crossing Sign:** As is self-explanatory, these signs are put where children cross the street right outside



the school. These signs are used to communicate to the driver to come to a stop in case kids are about to cross a road. Also, children should make sure to pass through the street only where these signs are put up.

- **Slow- Children At Play Sign:** Again, as is self-explanatory, this sign tells the driver to slow down and be alert of children that might end up on a street from the nearby play-zones. However, children should also be taught to be careful while playing. Parents should ask their kids to not play on the road as there is an increased risk of an accident on a public street
- **Seat Belt- Buckle Up Sign:** Seat belts are compulsory for both driver and front passenger in a vehicle. However, children are at a much higher risk of getting hurt than adults and hence, it's advisable for all kids to buckle up .

				
STOP	GIVE WAY	STRAIGHT PROHIBITOR NO ENTRY	PEDESTRIAN PROHIBITED	HORN PROHIBITED
				
NO PARKING	NO STOPPING OR STANDING	SPEED LIMITED 50	RIGHT HAND CURVE	LEFT HAND CURVE
				
RIGHT HAIR PIN BEND	LEFT HAIR PIN BEND	NARROW ROAD AHEAD	NARROW BRIDGE	PEDESTRIAN CROSSING
				
SCHOOL AHEAD	ROUND ABOUT	DANGEROUS DIP	HUMP OR ROUGH	BARRIER AHEAD

## OPEN CV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene

# CAMERA

An infrared camera is a non-contact device that detects infrared energy (heat) and converts it into an electronic signal, which is then processed to produce a thermal image or video, on which you can perform temperature calculations. Heat sensed by an infrared camera can be very precisely quantified, or measured, allowing you to not only monitor thermal performance, but also identify and evaluate the relative severity of heat-related problems



CAMERA



IR CAMERA SENSOR

# RESULT AND ANALYSIS

## 1. ANALYSIS OF MEAN DIFFERENCE BETWEEN GPS SPEED AND SPEEDOMETER

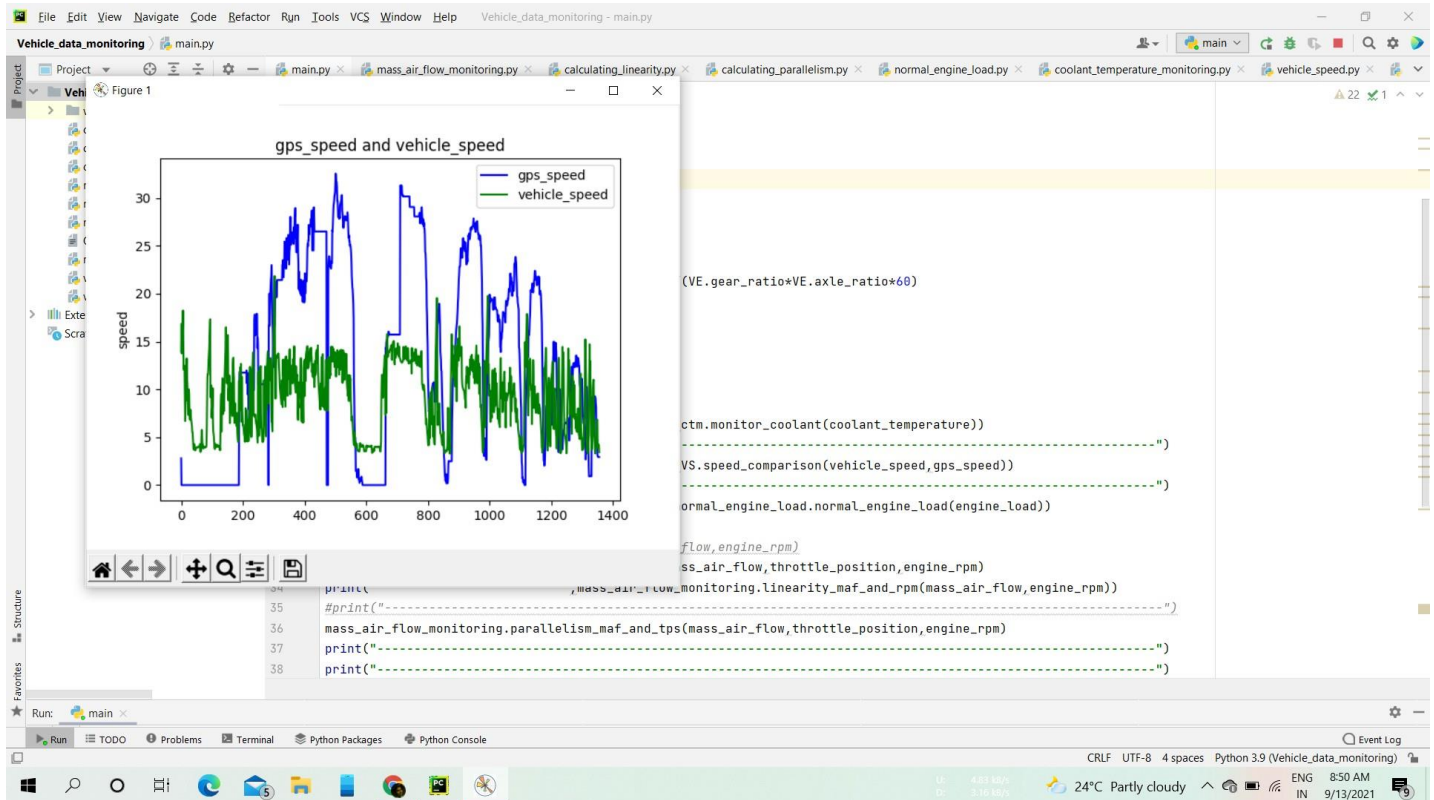


FIGURE 2

## 2. Analysis and calculation of parallelism between MAF and TPS

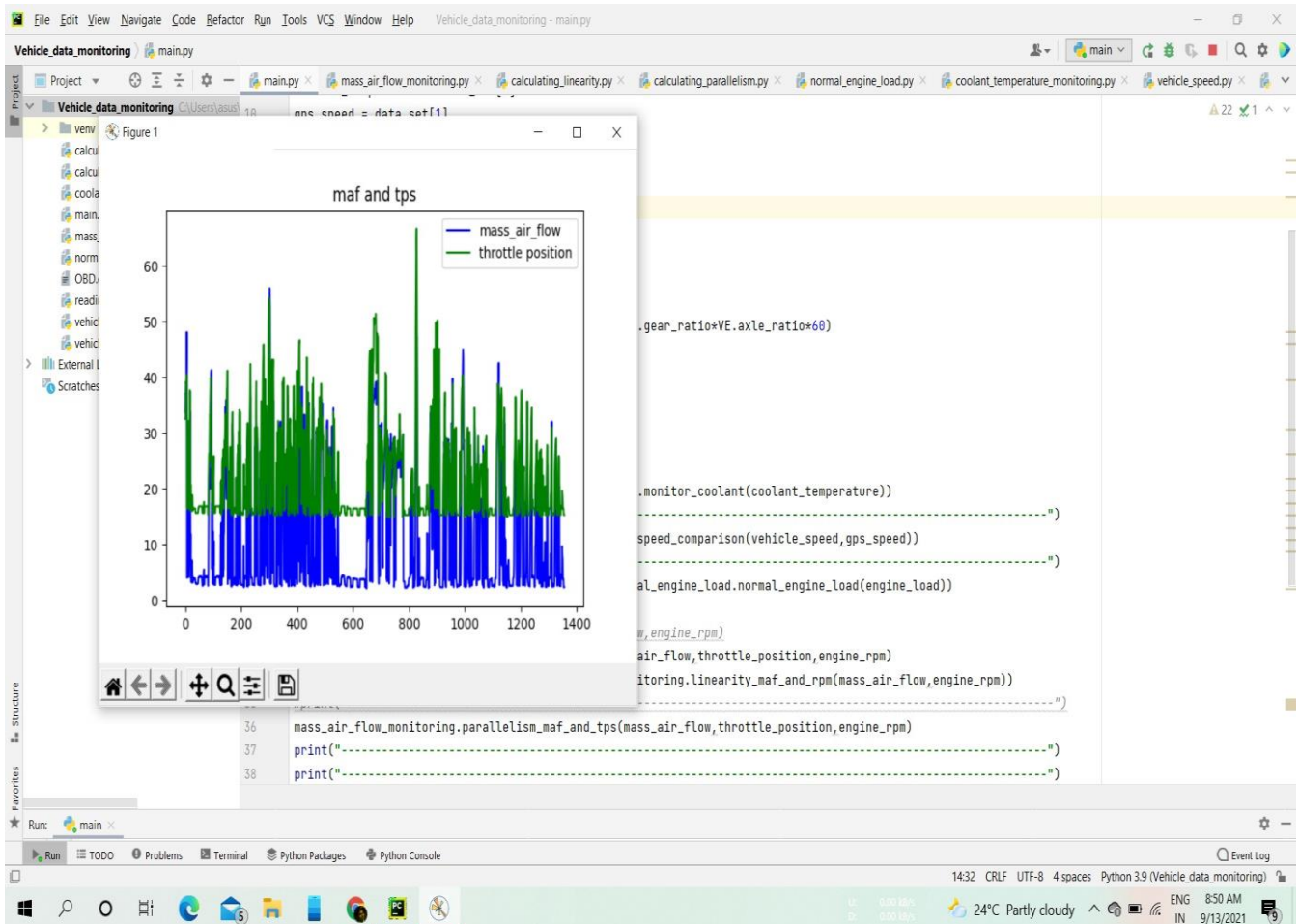
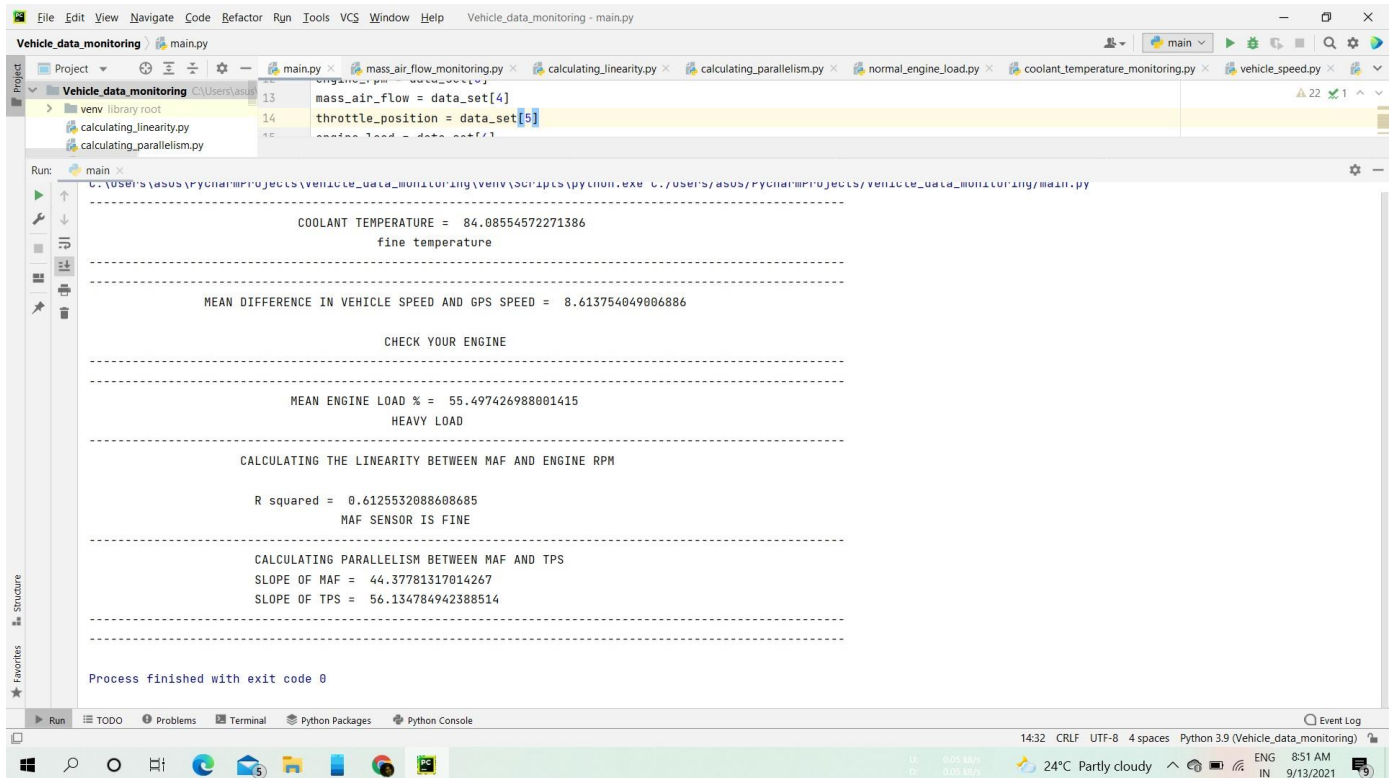


FIGURE 3

### 3. Final Result:

This is the required output for coolant temperature monitoring , engine load monitoring , linearity between MAF and Engine RPM and Parallelism between MAF and TPS



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Vehicle_data_monitoring - main.py
Vehicle_data_monitoring CAUsers\asus
main.py mass_air_flow_monitoring.py calculating_linearity.py calculating_parallelism.py normal_engine_load.py coolant_temperature_monitoring.py vehicle_speed.py
venv library root
calculating_linearity.py
calculating_parallelism.py
Run: main
C:\Users\asus\python\projects\vehicle_data_monitoring\venv\scripts\python.exe C:/Users/asus/python/projects/vehicle_data_monitoring/main.py
-----
COOLANT TEMPERATURE = 84.08554572271386
fine temperature
-----
MEAN DIFFERENCE IN VEHICLE SPEED AND GPS SPEED = 8.613754049006886
-----
CHECK YOUR ENGINE
-----
MEAN ENGINE LOAD % = 55.497426988001415
HEAVY LOAD
-----
CALCULATING THE LINEARITY BETWEEN MAF AND ENGINE RPM
R squared = 0.6125532088608685
MAF SENSOR IS FINE
-----
CALCULATING PARALLELISM BETWEEN MAF AND TPS
SLOPE OF MAF = 44.37781317014267
SLOPE OF TPS = 56.134784942388514
-----
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
14:32 CRLF UTF-8 4 spaces Python 3.9 (Vehicle_data_monitoring)
24°C Partly cloudy ENG 8:51 AM IN 9/13/2021
```

**We found the following results :**

**Coolant temperature = 84.08554572271386**

**Mean difference in vehicle speed and gps speed = 8.61375404**

**Mean engine load% = 55.4974269**

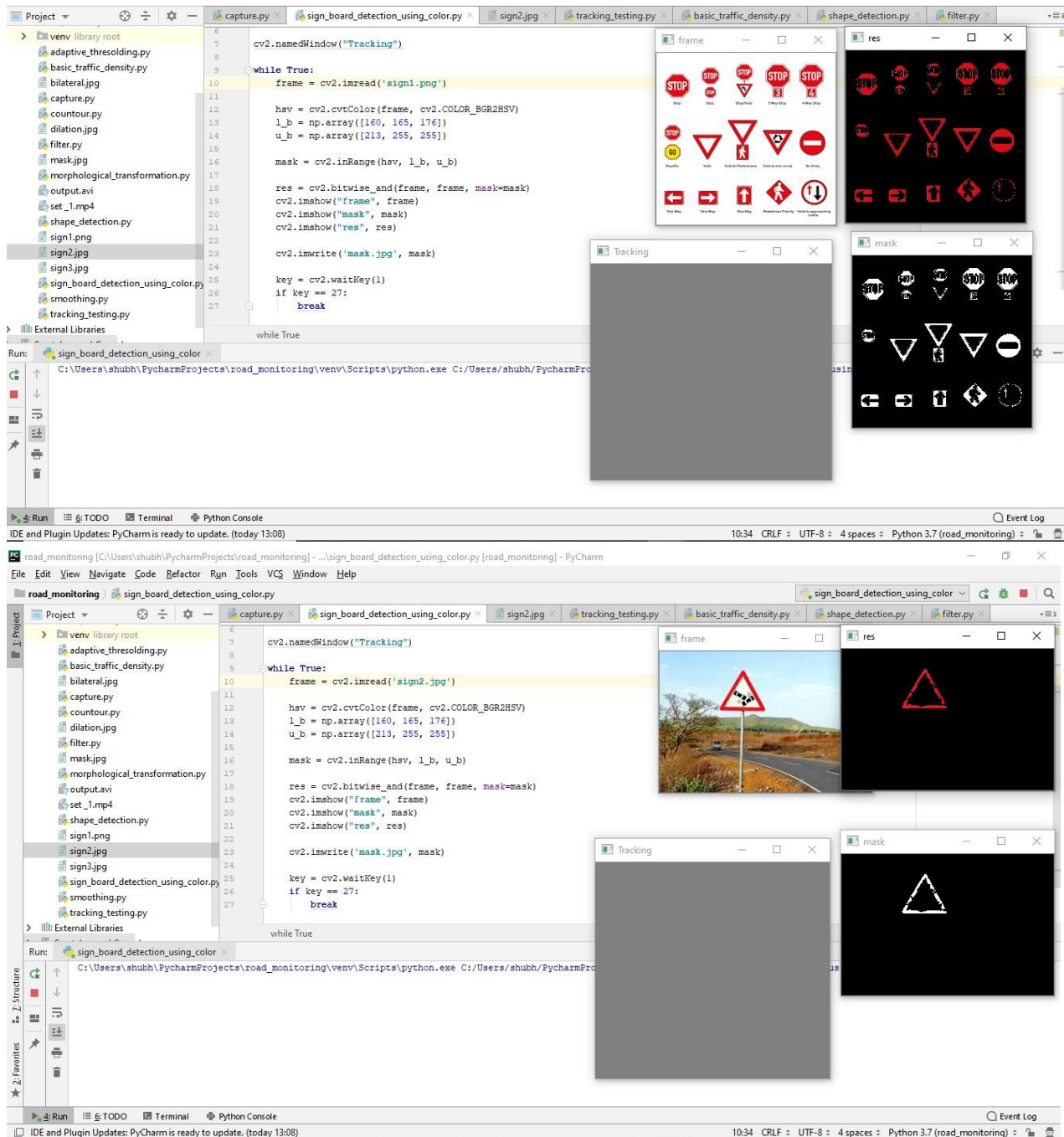
**Calculating the linearity between MAF and Engine Rpm**

**Slope of MAF = 44.3778**

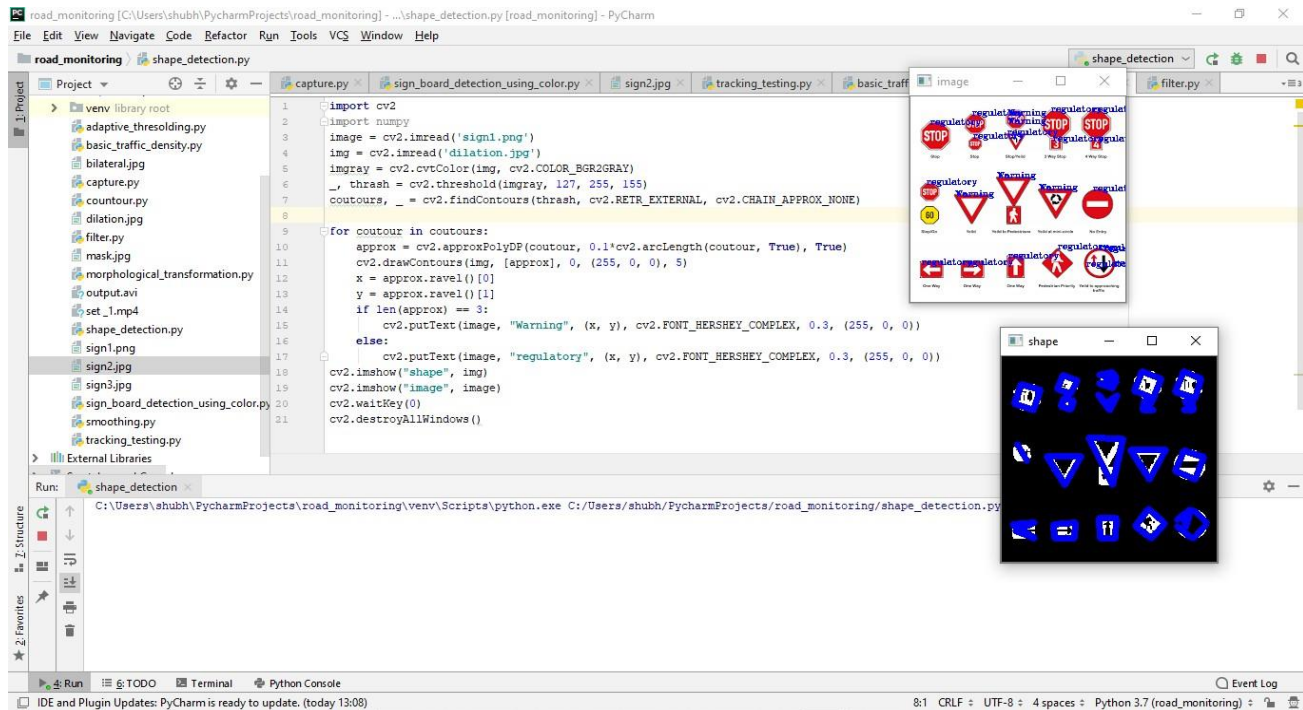
**Slope of TPS = 56.13478**



**BGR colour code combination is used to identify the traffic sign board. Code for traffic sign identification using BGR colour code combination is as follows:**

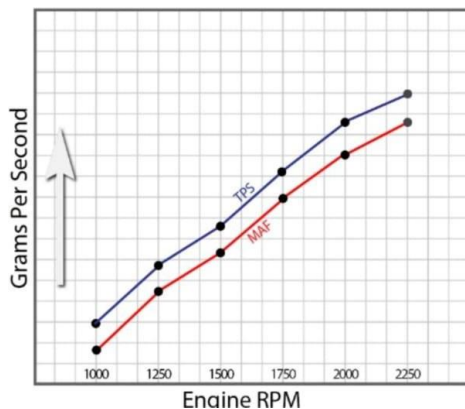


Shape detection focusses on identifying mostly the triangular and circular shape of the traffic sign board because most of the traffic sign board are either in triangular or circular shape.



# CONCLUSION

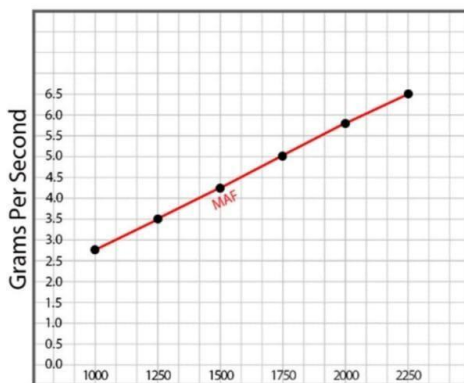
1. We found the acceptable range for an engine to run at coolant temperature between 80°c and 110°c.
2. Coolant temperature above 110° is referred as too hot. This means the engine is beginning to overheat. Too cold is anything under 80 degree celsius.
3. maf sensor conclusion



## Example 4

Another approach would be to graph both the **TPS** and the **MAF** sensor because they should essentially produce a plotted graph rising parallel to each other.

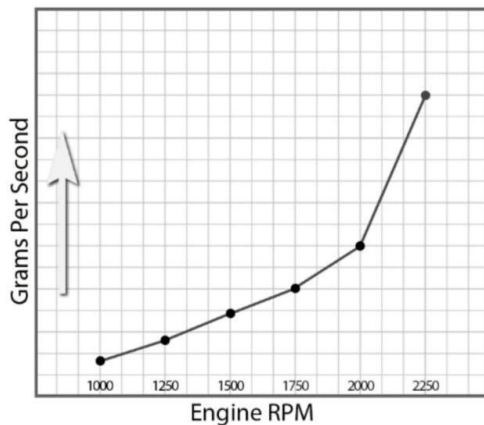
3.



## Example 1

A good MAF sensor should show a steady linear rise from 1000 to 2250.

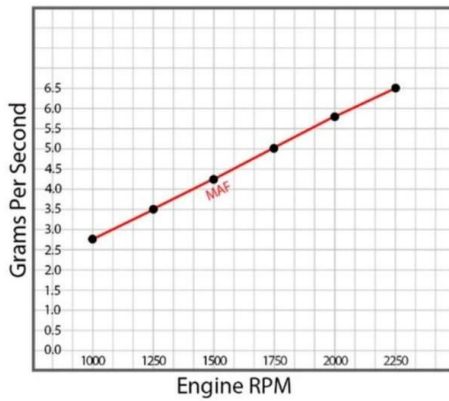
(Sample from 2009 Ford Focus 2.0L)



## Example 3

A sensor with a dirty sensing wire or not measuring all the air (air leak downstream) may rise too slowly.





### Example 1

A good MAF sensor should show a steady linear rise from 1000 to 2250.

(Sample from 2009 Ford Focus 2.0L)



4. Normal Engine Load percentage found is between 30 percent to 50 percent for best working.
5. In detection of traffic sign board the color code(BGR) could be use from [160,165,176] to [215,255,255] range.
6. Median filter found very useful to smooth raw image data.
7. The shape of red part of traffic sign boards are generally Triangle, Octagonal or Circle. That can be detected by using shape detection processes.

## **FUTURE SCOPE**

- Embedded vehicle diagnostics has the potential to drive benefits throughout the automotive value chain. At the OEM end of the chain, they can gain hugely through Reduction of costs associated with vehicle recalls, Reduction of No-Trouble-Found (NTF) backed warranty claims, all made possible through advanced and guided diagnostics.
- The Service dealerships will benefit by improving the Fix-First-Visit ratios, reducing service time and cost through remote/guided diagnostics, achieving higher technician efficiencies and productivity, and last but by no means the least, the Customer (Vehicle owner) will enjoy higher levels of convenience and comfort through the service dealerships new found abilities around preventive/predictive maintenance, resulting in Improved vehicle up-times.
- In the future, the recognition phase can be sped up using dimension reduction of feature vectors. The number and parameters of Gabor filters will greatly influence the results of the classification results. It is necessary to research the optimization of Gabor parameters using optimization algorithms, to improve the efficiency and accuracy of traffic sign detection and classification. Finally, we accelerated our method by a GPU and further improved the efficiency.

# APPENDIX

## A.1 LINEARITY CALCULATIONS

### Measurement System Linearity – Type A Uncertainty

Linearity looks at the accuracy of the measurements over the full range of the device.

- In order to measure the linearity of a device, we must take repeated measurements of parts or samples that cover its entire range.
- So that we don't introduce reproducibility error into the picture, the same operator must make all the measurements.

To check linearity, measure at least 5 samples that cover the full range of the instrument.

- Reference measurements for each of the samples (made by your quality group or by an outside laboratory) will be needed to determine linearity.
- The reference measurements will be compared to the results from the instrument whose linearity is being studied.
- Measure each of the samples randomly at least 10 times.
- For each of the parts, calculate the average and the range of the measurements made. The sample averages and ranges will be used with the reference values to determine linearity either graphically or by calculations.

Graphical Method:

- Plot the average measured values (on the y-axis) for each sample against the reference value (on the x-axis). If the resulting line is approximates a straight line with a 45-degree slope, the measurement device is linear.
- If the measured values do not form a straight line, or the line diverges from the optimal 45-degree slope, you may have a problem with linearity.

Calculating Linearity:

- A technique does exist that provides a precise mathematical evaluation of the linearity.
- The evaluation is based on the equation of a line that defines the relationship between the bias and the reference values of the parts or samples.
- The bias is the value of the sample measurement minus the reference measurement.
- To calculate the line of best fit, use the equation:

$$y = ax + b$$

where:

y = bias value

a = slope of the line

x = reference value

b = the y-intercept

- To calculate the slope, a:

$$a = \frac{\sum xy - \frac{(\sum x)(\sum y)}{n}}{\sum x^2 - \frac{(\sum x)^2}{n}}$$

where:

n = total number of measurements made

- To calculate the y-intercept, b:

$$b = \bar{y} - a\bar{x}$$

- With values for a and b, we can complete the regression equation (y = ax + b); it gives us the line of best fit.
- Using the results of the regression equation, we can determine the "goodness of fit" by calculating the Coefficient of Determination, R<sup>2</sup>.

$$R^2 = \frac{\left[ \sum xy - \sum x \left( \frac{\sum y}{n} \right) \right]^2}{\left[ \sum x^2 - \frac{(\sum x)^2}{n} \right] \times \left[ \sum y^2 - \frac{(\sum y)^2}{n} \right]}$$

- $R^2$  lets us know what amount of the variation in the bias values the regression line explains.
- If  $R^2$  is 0.6 (60%) or more, the regression line is an adequate representation of the line of best fit.

## A.2 PARALLELISM CALCULATION

We can determine from their equations whether two lines are parallel by comparing their slopes. If the slopes are the same and the  $y$ -intercepts are different, the lines are parallel. If the slopes are different, the lines are not parallel.

- To calculate the line of best fit, use the equation:
- To calculate the slope,  $a$ :

$$y = ax + b$$

where:

$y$  = bias value

$a$  = slope of the line

$x$  = reference value

$b$  = the  $y$ -intercept

$$a = \frac{\sum xy - \frac{(\sum x)(\sum y)}{n}}{\sum x^2 - \frac{(\sum x)^2}{n}}$$

where:

$n$  = total number of measurements made

## Bibliography

### JOURNALS:-

Cloud-Based Driver Monitoring and Vehicle Diagnostic with OBD2 Telematics

1Malintha Amarasinghe, 2Sasikala Kottegoda, 3Asiri Liyana Arachchi, 4Shashika Muramudalige, 5H. M. N. Dilum Bandara, and 6Afkham Azeez

Department of Computer Science and Engineering ,University of Moratuwa, Katubedda, Sri Lanka, 10400 \*WSO2 Lanka Inc. ,Colombo 03, Sri Lanka,\*[6azeez@wso2.com](mailto:6azeez@wso2.com)

Indian Sign Board Recognition Using Image Processing Techniques,1. G.Revathi

2. Dr.G.Balakrishnan, Department of Computer Science &Engineering Department of Computer Science &Engineering, publish date - 15/03/2016

[2] Gauri A. Tagunde,"Detection,Classification and Recognition of Road Traffic Signs Using Color and Shape Features". International Journal of Advanced Technology & Engineering Research, ISSN No: 2250-3536, Volume 2, Issue 4, July 2012.

### BOOKS:-

[1] Bernd Jahne, Digital Image Processing, Springer Science & Business Media, 2013.

[2] Keith McCord, Automotive Diagnostic Systems: Understanding OBD I and OBDII;CarTech Inc, 2011