

**LOCALIZATION FOR AUTONOMOUS CAR USING
DEEP LEARNING
BY
LEE WOON SHIN**

**A REPORT
SUBMITTED TO
Universiti Tunku Abdul Rahman
in partial fulfilment of the requirements
for the degree of
BACHELOR OF INFORMATION SYSTEMS (HONOURS)
INFORMATION SYSTEMS ENGINEERING
Faculty of Information and Communication Technology
(Kampar Campus)**

JAN 2023

REPORT STATUS DECLARATION FORM

Title: LOCALIZATION FOR AUTONOMOUS CAR USING DEEP LEARNING

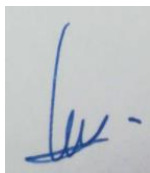
Academic Session: JAN 2023

I LEE WOON SHIN
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,



(Author's signature)

Sun Teik Heng
(Supervisor's signature)

Address:

5, Pesara Ioke Lim,

Taman Loke Lim,

30010 Ipoh Perak

Ts Sun Teik Heng @ San Teik Heng

Supervisor's name

Date: 25 APRIL 2023

Date: 25 APRIL 2023

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 21 JUNE 2011	Page No.: 1 of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

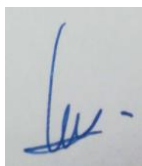
Date: 25 APRIL 2023

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that LEE WOON SHIN (ID No: 18ACB01136)
has completed this final year project/ dissertation/ thesis* entitled “LOCALIZATION
FOR AUTONOMOUS CAR USING DEEP LEARNING” under the supervision of Ts
Sun Teik Heng @ San Teik Heng (Supervisor) from the Department of Information
Systems, Faculty of Information and Communication Technology.

I understand that University will upload softcopy of my final year project / dissertation/
thesis* in pdf format into UTAR Institutional Repository, which may be made accessible
to UTAR community and public.

Yours truly,



(LEE WOON SHIN)

DECLARATION OF ORIGINALITY

I declare that this report entitled “**LOCALIZATION FOR AUTONOMOUS CAR USING DEEP LEARNING**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature :  _____

Name : LEE WOON SHIN

Date : 25 April 2023

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Ts Sun Teik Heng @ San Teik Heng and Ts Dr Chan Lee Kwun who has given me this bright opportunity to engage in a deep learning project. It is my first step to establish a career in deep learning field. A million thanks to you.

To a very special person in my life, Lee Foong Choi, for her patience, unconditional support, and love, and for standing by my side during hard times. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

ABSTRACT

Self-driving car, which is also known as autonomous vehicle or robotic car that all car manufacturer that wish to develop into a fully self-driving car. They are trying to develop and achieve higher level of automated driving. Localization is part of subsystem in the automated driving, which is most of the autonomous or robotic car manufacturer trying to improve and develop it into fully self-driving car. Localization in autonomous car can be defined as the process of locating the current position of vehicle. Localization able to let autonomous car know where it is at the current position of it and navigate the ways to destination. It can let user to drive less, avoid car accident when we are inside the car to our destination. To reach this goal, many cars manufacturer or technology company has worked in this area for years to solve the obstacles in the way to achieve next level of automated driving.

In this paper, it aims to focus on how the localization works with the proposed methods. The proposed method will use the OpenCV computer vision algorithms and other solutions to solve the localization problem.

TABLE OF CONTENTS

TITLE PAGE	i
REPORT STATUS DECLARATION FORM	ii
FYP THESIS SUBMISSION FORM	iii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii-ix
LIST OF FIGURES	x-xiii
LIST OF TABLES	xiv
LIST OF ABBREVIATIONS	xv
CHAPTER 1 INTRODUCTION	1
1.1 Problem Statement	2
1.2 Motivation	4
1.3 Project Scope	6
1.4 Project Objectives	7
1.5 Contributions	8
1.6 Report Organisation	9

CHAPTER 2 LITERATURE REVIEW	10
2.1 Previous works on Deep Learning	10
2.1.1 Semantic Segmentation	12
2.2 Limitation of Previous Studies	14
 CHAPTER 3 SYSTEM MODEL	 18
3.1 System Design Diagram/Equation	18
3.2 Use Case Diagram	20
3.3 Function Model Diagram	21
 CHAPTER 4 SYSTEM DESIGN	 22
4.1 System Block Diagram	22
4.2 System Components Specifications	24
4.2.1 Hardware	24
4.2.2 Software	30
4.3 Circuits and Components Design	32
4.4 System Components Interaction Operations	35
 CHAPTER 5 Experiment/Simulation	 36
5.1 Hardware Setup	36
5.2 Software Setup	38

5.3	Setting and Configuration	41
5.4	System Operation	44
5.5	Implementation Issues and Challenges	51
5.6	Concluding Remark	52
CHAPTER 6	SYSTEM EVALUATION AND DISCUSSION	53
6.1	System Testing and Performance Metrics	56
6.2	Testing Setup and Result	56
6.3	Project Challenges	59
6.4	Objectives Evaluation	60
6.5	Concluding Remark	61
CHAPTER 7	CONCLUSION AND RECOMMENDATION	62
7.1	Conclusion	62
7.2	Recommendation	63
REFERENCES		64
WEEKLY LOG		65-70
POSTER		71
PLAGIARISM CHECK RESULT		72-75
FYP2 CHECKLIST		76

LIST OF FIGURES

Figure Number	Title	Page
Figure 1.1.1	Sample diagram for four fisheye cameras	2
Figure 1.1.2	Diagram above showed how the four-fisheye camera views combined into a top-view image ("How Does 360 Car Camera Work Should You Install One?", 2021)	3
Figure 1.1.3	Diagram showed the unknown sticky liquid blocked the view of camera	3
Figure 1.2.1	Diagram showed example what LiDAR detected and mapped while it installed above a car	5
Figure 2.1.1.2	semantic segmentation	12
Figure 2.1.1.2	Diagram showed the framework of road marking detection in this studied paper	13
Figure 2.2.1	Diagram above show the price of the GNSS receiver which cost RM 16,800	14
Figure 2.2.2	Diagram above showed the LiDAR component	15
Figure 2.2.3	Diagram showed the principle of the segmentation	16
Figure 2.2.4	Sample of priori maps segmentation from a sample dataset	16
Figure 3.1.1	System Design	18
Figure 3.2.1	Diagram above show the Use Case Diagram of deep learning localization for autonomous vehicle	20
Figure 3.3.1	Diagram showed the function model diagram of the system	21

Figure 4.1.1	Block diagram of localization system using raspberry pi	22
Figure 4.2.1.1	Diagram above showed a sample camera module for raspberry pi	27
Figure 4.2.1.2	Diagram above showed a sample of GPS Module called “Neo 6M GPS module”	28
Figure 4.2.1.3	Diagram above showed an example of power bank which has the 12,000 mAh battery capacity	29
Figure 4.2.2.1	Visual Studio Code	30
Figure 4.2.2.2	OpenCV	31
Figure 4.2.2.3	VNC Viewer	31
Figure 4.3.1	Robot car chassis model circuits design	32
Figure 4.3.2	Conceptual diagram	33
Figure 5.1.1	L298N motor driver connects to jumper wires along with black and red wires of motors	35
Figure 5.1.2	L298N motor driver is well-connected with 4 DC motors and wheels	35
Figure 5.1.3	Jumper wires connected to Raspberry Pi 3B+	35
Figure 5.1.4	GPS module is connected at the Raspberry Pi 3B+ GPIO pins	35
Figure 5.1.5	Camera mounted at a board	36
Figure 5.1.6	Diagram showed how the camera mounted and a better angle of frame view	36

Figure 5.1.7	Diagram showed how the full robot car chassis is built	37
Figure 5.2.1	Diagram showed downloading python for Windows at python official website	38
Figure 5.2.2	Diagram showed Python extension at VS Code extension store	38
Figure 5.2.3	Diagram showed VS Code editor can be downloaded at different operating system at Visual Studio Code official website	39
Figure 5.2.4	Diagram showed Raspberry Pi Imager is available at the official website of Raspberry Pi	39
Figure 5.2.5	Raspberry pi operating system image	40
Figure 5.2.6	Wi-Fi receiver driver installed	40
Figure 5.2.7	OpenCV cv2 library version installed	40
Figure 5.3.1	Ip address of Raspberry Pi 3B+	41
Figure 5.3.2	Connection from VNC Viewer	41
Figure 5.3.3	CPU usage and temperature monitor	41
Figure 5.3.4	Raspberry Pi configuration page	42
Figure 5.3.5	Command of accessing config.txt	42
Figure 5.3.6	List of commands	42

Figure 5.3.7	Command pasted in config.txt	43
Figure 5.3.8	Location data after the command “sudo cat /dev/ttyAMA0” is typed in terminal	43
Figure 5.4.1	The right red boarder line of lane is detected, and show “left” as the moving direction	44
Figure 5.4.2	The two red boarder lines of lane are detected and show “right” as the moving direction	45
Figure 5.4.3	The two red boarder lines of lane are detected and show “straight” as the moving direction	45
Figure 5.4.4	S curve lane	46
Figure 5.4.5	Codes wrote in lane detection about the error handling	47
Figure 5.4.6	File called “try.py”	50
Figure 6.1.1	Picture of the red border lines between the lane is detected	53
Figure 6.1.2	Robot chassis car model is running on the designed path	54
Figure 6.1.3	Raspberry pi is attached with the GPS module	54
Figure 6.1.4	Result of location data	54
Figure 6.2.1	Performance of the lane detection	56
Figure 6.2.2	Result of the longitude and latitude of the current location	57
Figure 6.2.3	Pinned result of the longitude and latitude on map	57

LIST OF TABLES

Table Number	Title	Page
Table 4.2.1.1	Specifications of laptop	25
Table 4.2.1.2	Specifications of Raspberry pi 3 B+	26
Table 4.2.1.3	Specifications of Camera	27
Table 4.2.1.4	Specifications of GPS module	28
Table 4.2.1.5	Specifications of Power bank	29
Table 5.4.1	Source code of lane detection.	49

LIST OF ABBREVIATIONS

<i>AVM</i>	Around view monitoring
<i>CNN</i>	Convolutional neural network
<i>CSL</i>	Coarse-scale localization
<i>FOV</i>	Field of View
<i>FSL</i>	Fine-scale localization
<i>GNSS</i>	Global Navigation Satellite System
<i>GPS</i>	Global Positioning System
<i>GUI</i>	Graphical user interface
<i>LiDAR</i>	Light Detection and Ranging
<i>NDT</i>	Normal distribution transformation
<i>Radar</i>	Radio detection and ranging

Chapter 1

Introduction

Autonomous vehicle has become more important and popular in this recent decades because it believes that it is efficient to human as it can certify the driver and passenger safety, save time, and others. According to the Social Statistics Bulletin in 2019, there are 548,598 road accidents happened. 8,341 accidents that have people injuries and 6,284 deaths in Malaysia. These numbers showed that there are around 1,500 and more accidents cases and 22 people injured and 17 deaths from accidents every day! Some of the cases are due to driver faults. The number of the accidents can be decreased when the automated vehicle is developed. Autonomous car can avoid the human error like distracted or drunk driver as it is developed to finish the order given by the user. For example, it will follow, navigate, and find the shortest path to the destination that set by user safely. Thus, autonomous car will be able to recognize and tell the user current position of itself, destination, and others. It showed autonomous car has the potential and ability to save the passengers' lives, helps people who cannot drive and believe it able to reduce the number of accidents will decrease. Thus, in this article, it will show how the performance of the python script wrote using OpenCV (Open-Source Computer Vision) detecting the lane and get the real time data location.

1.1 Problem Statement

Generally, most of vehicle's position is determined by matching the features collected from image using prior map. Many cars manufacturer uses around view monitoring (AVM) which consists of four fisheye cameras that mounted around the car to provide the 360° surroundings as shown in Figure 1.1.1. For example, Benz, Lexus, Toyota, Audi, and others. Fisheye camera has limitation as it has a Field of View (FOV) of 180° for each fisheye camera. As shown in Figure 1.1.2, the image from fisheye camera normally is undistorted and combined into top-view image. The top-view image can show accurately the information of nearby lanes and it is not affected by neighbouring objects and weather. However, there one of limitation is the information it collected is not sufficient for the localization determination for a vehicle. For example, the image collected cannot differentiate the multipath of the same road due to the limitation of camera Field of View (FOV) range. It cannot detect the upcoming object, car, road sign and name or even roadways path due to short range of image detected from fisheye camera.



Figure 1.1.1 Sample diagram for four fisheye cameras [1].



Figure 1.1.2 Diagram above showed how the four-fisheye camera views combined into a top-view image [1].

Moreover, sometimes we might occur this scenario just like the picture shown in Figure 1.1.3, fisheye camera that mounted outside the car is blur as the camera is blocked with some unknown sticky liquid. It will cause the system of autonomous car cannot detect the location of the vehicle and localization system will have error.



Figure 1.1.3 Diagram showed the unknown sticky liquid blocked the view of camera [2].

1.2 Motivation

This project aims to develop a better localization system for autonomous vehicle for the future. It is important as it can greatly reduce the percentage of accidents. It shows there are 14 to 15 percent that saved from the accident due to automatic braking [3]. This shows that autonomous vehicle technologies can greatly reduce once it is fully automated.

There are always some questions that need to be answered in the localization system is how the autonomous know where it is, where is the items located beside it, how to reach destination from the current location, and what's driver up to. Thus, there are several solutions founded these few years. For example, using cutting-edge sensors, Global Positioning System (GPS), radio detection and ranging (radar), cameras, Global Navigation Satellite System (GNSS), Light Detection and Ranging (LiDAR), and others.

In autonomous vehicles, there are several combinations of sensors or methods are deployed to perform the tasks. First task is the environment detection. To detect the neighboring environment, several methods can be used such as normal camera, thermal camera, LiDAR, and radar.

Why LiDAR? What's that? LiDAR is a simplified form for Light Detection and Ranging. It is a remote sensing method that measures ranges or distances using light from a pulsed laser. It will generate a huge 3D map and we can know the environment that is more than 500 meters away from the current position. These used to track the on-road vehicle, road sign, lane marking, direction of the road and others. Some modern LiDAR even can differentiate between person who is standing or running, speed of bike and motorcycles and others.

Camera captured the digital images or video and even act as an eye for the computer to enables them deriving the meaningful and important information in real time and take action according to the captured information. Camera is important as it is the main component to let the system see what the world outside itself is. Based on the information it collected, it will go into machine learning like process it

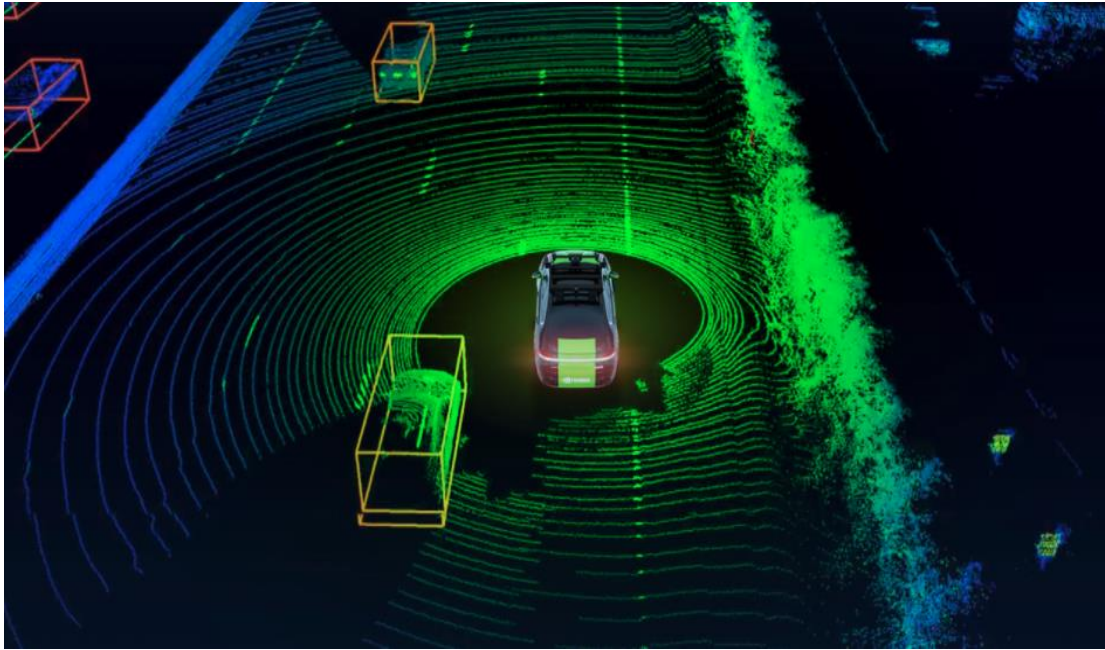


Figure 1.2.1 Diagram showed example what LiDAR detected and mapped while it installed above a car.

Second method is localization. Localization is to let the autonomous vehicle knows where its location, what items in its neighboring environment and where it wants to head up to. In many cases, it has been founded that deep learning can detect better due to its multiple layers. CNN is a deep learning algorithm which feed with an image, then it will pass through multi layers size of padding by turning into matrix pixel of values. It able to differentiate one from others with enough numbers of training and show how much percentage of accuracy. There are several algorithms tested and will be decided using which algorithm which show the highest accuracy of the results.

Furthermore, autonomous vehicle is very helpful to the people who unable to drive due to certain reasons such as brain injury, blind, broken leg, or arm, colour blindness, Parkinson's disease, and other reasons who caused someone unable to drive. These group of people will be benefited from autonomous vehicle as previously they can't drive due to their issues that related to health, disability, or age.

1.3 Project Scope

The scopes of the project include as below:

Build a car chassis model vehicle that has four 6v dc motors wheels connected with L298N 2A Motor Driver that has four AA battery with 1.5V each and connected with Raspberry Pi GPIO pins. The raspberry pi has a camera module connected, several GPIO pins that connected from the L298N 2A motor driver, and a power bank as a power supply for raspberry pi.

The purpose of this method is to let user or others able to know that the current location of the autonomous vehicle while the lane detection is working. Thus, the user will know what the current location and without driving it.

Besides, GPS module will be connected to the raspberry pi to get the current data location and tell user where they are now. With the latitude and longitude data, it able to tell user which position or location of any place on Earth's surface they are at.

1.4 Project Objectives

The main objective of this project is to provide a solution in increasing the accuracy of localization of autonomous vehicle by using OpenCV. This project will be to develop a car chassis model that run by python OpenCV detecting the lane and get location data from GPS module. This will be done by coding and developing a python script that can filter out the color of line of lanes on the road and calculate the slope by finding the largest contour along with getting the real time location data from GPS module. The experiment will be done by determine the accuracy of the lane detection result and the accuracy of the GPS module used.

This will not only increase the localization accuracy of the autonomous vehicle, but it will also improve the traditional vehicle vision detection. Through this system, it believes that it can greatly reduce the numbers of the accidents.

1.5 Contribution

In this project, it will benefit the driver and passenger in terms of safety in the road. Firstly, with OpenCV based python script written running in raspberry pi car chassis model, it able to drive itself according to the lane detected. When the lane is detected, it will calculate the slope and make decision of whether turning left, right, or straight. Secondly, the GPS module that attached with raspberry pi will tell the user what the current position of the it since it is attached at the car chassis model. Thus, by having these functions stated above, it believes that it able to drive itself without driver by following the lane detected and while knowing the current location of it.

1.6 Report Organization

This report is organised into 7 chapters: Chapter 1 Introduction, Chapter 2 Literature Review, Chapter 3 System Methodology/Approach, Chapter 4 System Design, Chapter 5 System Implementation, Chapter 6 System Evaluation and Discussion, Chapter 7 Conclusion and Recommendation. The first chapter is the introduction of this project which includes problem statement, motivation, project scope, project objectives, contribution, and report organisation. The second chapter is the literature review carried out on several existing works on Deep Learning and its limitations. Both chapter 3 and 4 discuss the overall system methodology and system design of this project, of which chapter 3 focuses on the system design diagram, use case diagram, and function model diagram, and chapter 4 focuses on the system block diagram, system components specification, circuit and components design, along with system components interaction operations. The fifth chapter includes information on the specifics of how to use the system's design. Furthermore, the sixth chapter evaluates and discusses the system testing and performance, setup, result, project challenges, and objectives evaluation. For the last chapter, it will make a conclusion and recommendation to the proposed system and project.

Chapter 2

Literature Review

2.1 Previous works on Deep Learning

In recent years, there are many studies about the lane detection, road sign detection, and others using the deep learning. It will detect line marking, non-line marking, road sign, road name, direction, and others. There is one method of deep learning proposed by Liu is pixel-wise semantic segmentation for road marking and road boundary solution [4].

With the image captured by fisheye camera, it will differentiate the real and fake boundary points to exclude other items and detect the lines marking on the road. It will improve the localization accuracy of the autonomous vehicles. It also proposes the fine-scale localization (FSL) method and a coarse-scale localization (CSL) method for the around view monitoring (AVM) system-based localization.

Moreover, it implements the global positioning system (GPS) in around view monitoring (AVM) system. The experiments showed that these combination methods can highly increase the accuracy of the autonomous vehicle localization system.

Besides, there is another paper studied is proposing a localization system that using prior visual point cloud map which constrained in GNSS-challenged environments. In this studied paper, [5] proposed the localization system which using priori visual point cloud map and a stereo camera. The priori visual cloud map is segmented with the semi-global-block-matching (SGBM) algorithm.

It aims to forecast the visual point cloud of the image frame that captured by the stereo camera. Then, it will match with the sub-map using the normal distribution transformation (NDT). Normal distribution transformation (NDT) is a registration algorithm that used in many

field [6]. It was introduced by Biber and Strasser. The accuracy of the NDT is strongly relied on the size of cube used in the algorithm and it can avoid computing explicit correspondences.

2.1.1 Semantic Segmentation

In [4] proposed a research paper which was inspired by semantic segmentation. It is based on the convolutional neural network (CNN)- based semantic segmentation and consists of making prediction process for a whole input image. Semantic segmentation can produce pixels in an image and categorize them. It's important in autonomous vehicle system as it needs to understand each item in image and make dense prediction on the pixels of image. As the image shown below, I is image, the $\hat{y}_{x,y}$ is the output, (x, y) is the pixel position, represent the CNN model parameters. For L and L , it is label distribution and number of the predefined classes.

The task of semantic segmentation aims to produce pixel-level labels for an entire image. For a given image I , the label output $\hat{y}_{i,j}$ at pixel position (i, j) is computed by

$$\hat{y}_{i,j} = \arg \max_{l \in \{0, \dots, L-1\}} P(y_{i,j} = l | I; \theta) \quad (1)$$

where θ represents the CNN model parameters, $P(y_{i,j} = l | I; \theta)$ is the label distribution, and L is the number of the predefined classes. We classify road scene pixels into $L = 18$

Figure 2.1.1.1 semantic segmentation.

With this semantic segmentation, the localization system will exclude the unwanted elements from the images such as dynamic objects. As the studied paper main objectives is to detect both line and non-line road marking only to improve the localization of the autonomous car accuracy.

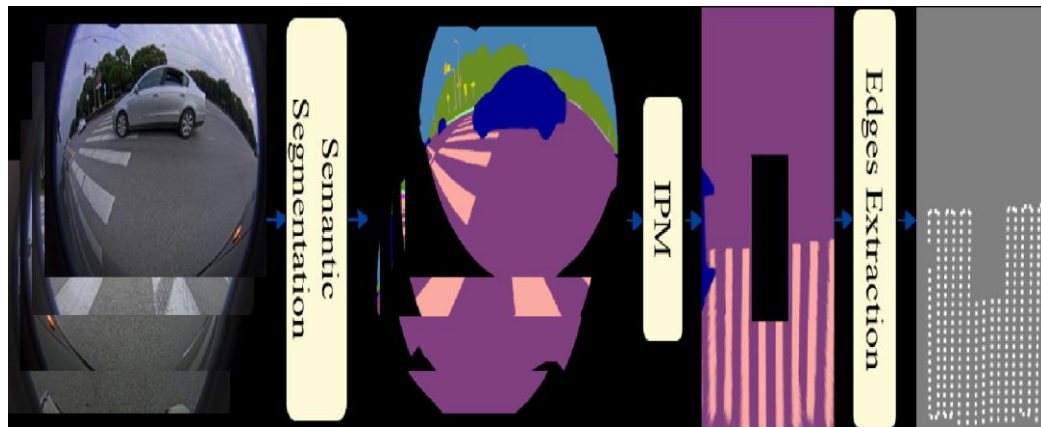


Figure 2.1.1.2 Diagram showed the framework of road marking detection in this studied paper.

However, with this semantic segmentation model, it may not be the good choice for implementing it to raspberry pi. It is because raspberry pi has low computation power. Thus, it is not suitable for perform real time detection on raspberry pi.

2.2 Limitation of Previous Studies

According to the previous first studied paper, there are several proposed methods and solution to solve the localization of autonomous vehicles. For examples, fisheye camera, semantic segmentation, coarse-scale localization (CSL), fine-scale localization (FSL), around view monitoring (AVM), and global positioning system (GPS) [4]. Although this method is resulted with **high accuracy**, but it still has many disadvantages such as the **budget of these methods proposed is too high** and limitations on the camera used which is fisheye camera. For example, the **information of images collected by the fisheye camera is not sufficient** for the localization determination system of a vehicle and others limitation that stated in problem statement Chapter 1.

Besides, the next studied paper, it used GNSS, LiDAR-camera based, Inertial Measurement Unit [5]. These solutions which used these methods proposed are very **expensive**. The price of the method proposed per unit is showed as the diagram bellow:

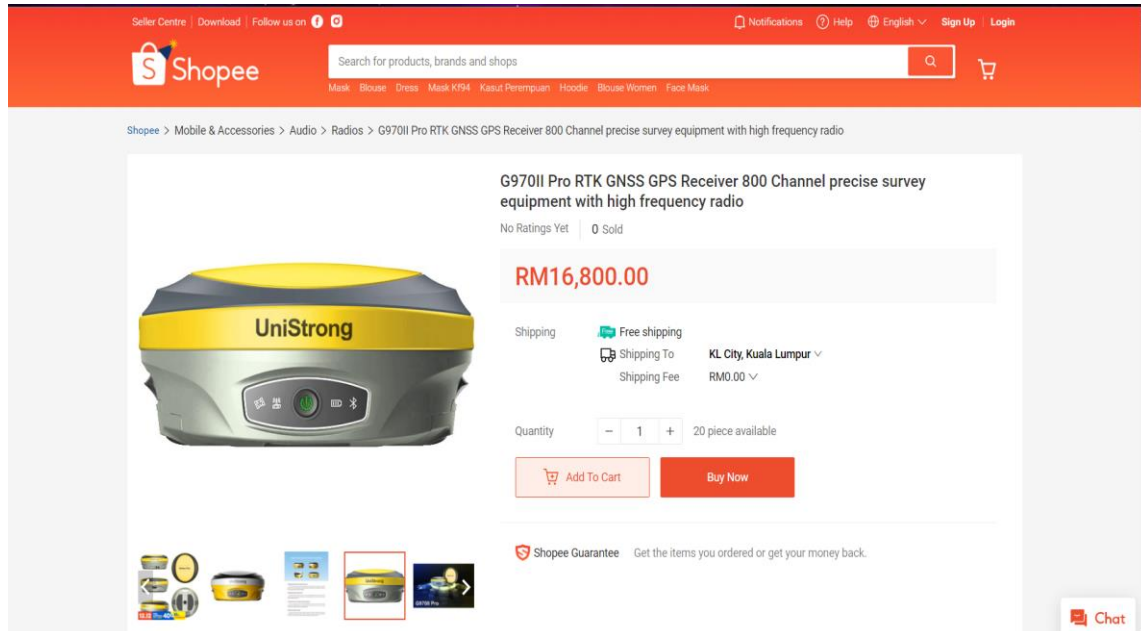


Fig. 2.2.1 Diagram above show the price of the GNSS receiver which cost RM 16,800



Figure 2.2.2 Diagram above showed the LiDAR component attached on Toyota Prius, and the model's name is called Velodyne HDL-64E LIDAR sensor and cost \$75,000 for one LiDAR sensor.

The Fig. 2.2.3 diagram above showed the first Google Self-Driving Car built. The most expensive component was the Velodyne HDL-64E LIDAR sensor. It is costed about \$75,00. This LiDAR model need to attach it on the vehicle so that it can scan the view around the car. However, although it has a good advantage on the long-distance view, it can't detect the up-close objects. Google mounted some radar sensor again to fix the problem [7].

Then, back to the studied paper, the results showed there are some problems of **low accuracy at the sharp turns** [5]. Prior visual point cloud map is the important methods in this experiment. It is because the proposed method by this paper is highly dependent on the prior visual point cloud map.

Visual Point Cloud Generation is retrieving the dense 3D surface reconstructions of wide-ranging streetscapes from vehicle-borne images and merged visual and laser odometry are known to as. LiDAR will generate visual point cloud maps prior to accuracy estimation. Depth of stereo images is estimated by the pyramid stereo matching network (PSMNet). Prior map Segmentation is very time-consuming as a whole prior visual point cloud map is generated by the stereo camera and it will keep matching to the visual point cloud. It means that it will keep segmenting the prior map every time the vehicle moves. The main idea of the studied papers is to create a sub-map and keep segmenting to the prior visual point cloud map according to the cube size. The principle of the segmentation is shown as diagram below:

$$\begin{cases} \text{If, } |L_{vehicle} - B_{edge}| > \tau, \text{continue} \\ \text{else, } B_{center} = L_{vehicle} \end{cases}$$

Figure 2.2.3 Diagram showed the principle of the segmentation.

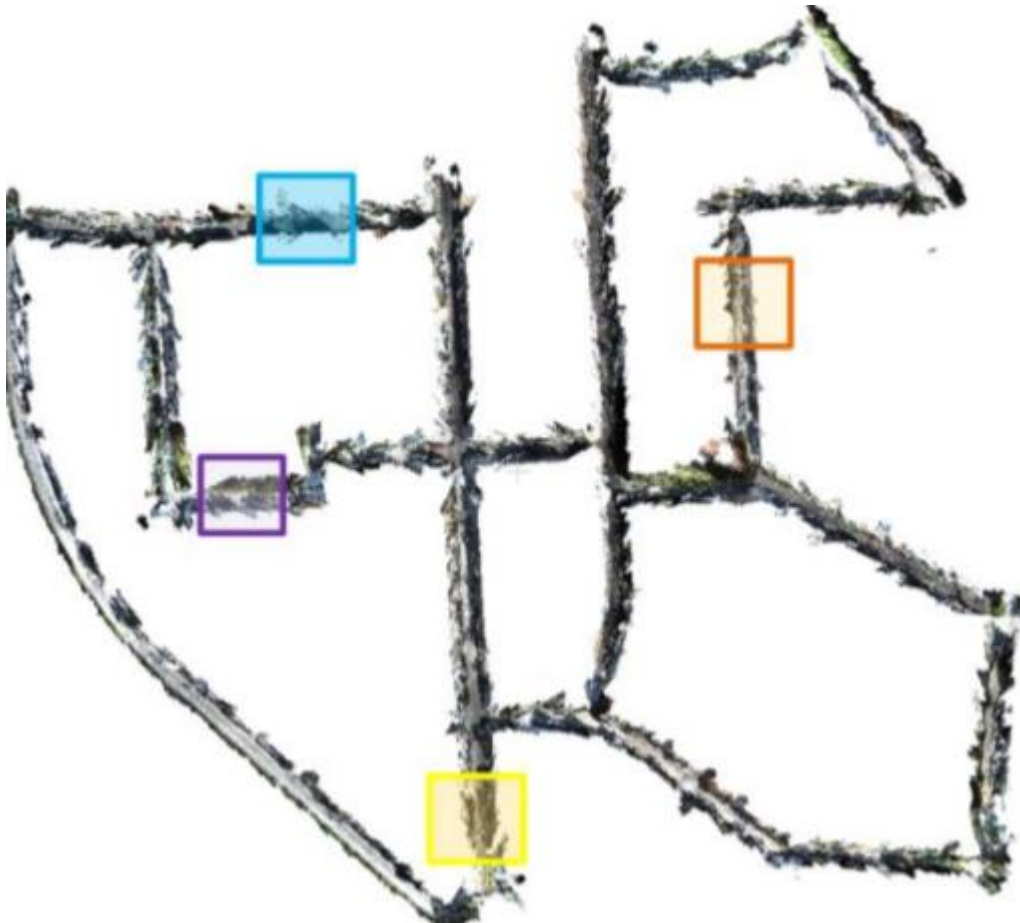


Figure 2.2.4 showed the sample of priori maps segmentation from a sample dataset.

Besides, using these methods that proposed in this studied paper also needs **big amount of compute power** as it needs to run those sensors and do all the calculations need to analyse ad make the autonomous vehicle make driving decisions. It also **cost massive consumptions of electricity and compute power**.

Chapter 3

System Model

3.1 System Design Diagram/Equation

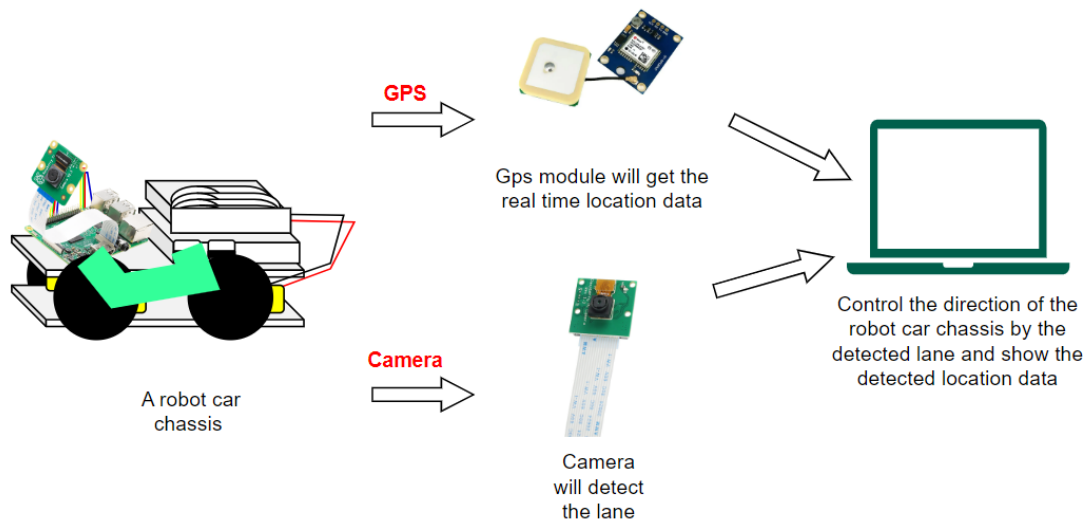


Figure 3.1.1 System Design

Referring to the system design figure provided above, the robot car chassis model consists of two main hardware components, namely the camera and GPS module. These hardware components are integrated with the Raspberry Pi 3B+ powered robot car chassis model, which acts as the control centre for the entire system. The camera is primarily used to capture real-time frames of the lane, which are then processed using the OpenCV computer vision library to detect lane. It is fixed at a strategic position on the robot car chassis model to provide an optimal view of the lane ahead.

On the other hand, the GPS module is responsible for providing real-time location data to the user. It able to let the user know where the current location of the robot chassis car model is.

Overall, the incorporation of these two key hardware pieces into the robot car chassis model offers a complete and trustworthy system for autonomous navigation and real-time tracking of the location and movements of the robot car chassis model.

Slope calculation

$$\text{slope} = \frac{vy}{vx}$$

‘vx’ and ‘vy’ are the components of a normalized vector that describes the direction of the line. ‘x’ and ‘y’ are the coordinates of a point on the line.

These can be retrieved in this line of code: “vx, vy, x, y = cv2.fitLine(largest_contour, cv2.DIST_L2, 0, 0.01, 0.01)”. To fit a straight line to the contour of the red line discovered in the current frame, the ‘fitLine’ function is used in the code. 4 values are returned by the function.

The slope describes how sharply the line angles. If the slope is positive, the line slopes to the right; if it is negative, the line slopes to the left; and if it is close to zero, the line is almost horizontal. The code values used to determine whether the line is slanted to the left, right, or straight are based on the typical range of slopes for each situation.

3.2 Use Case Diagram

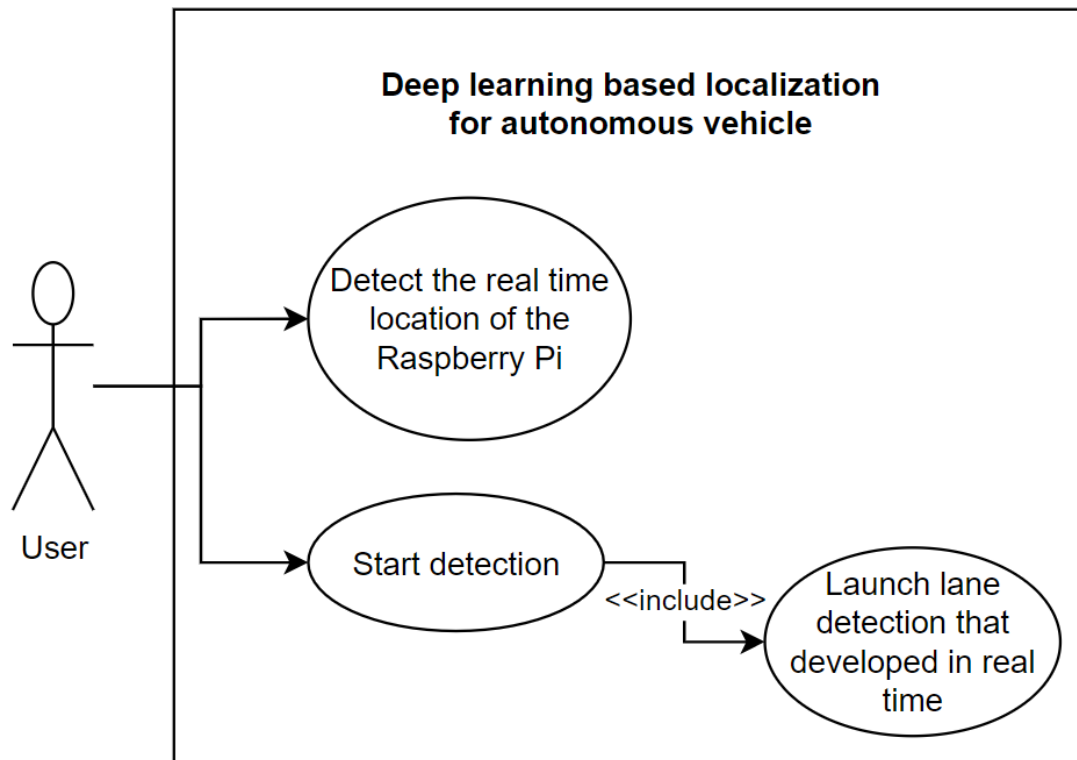


Figure 3.2.1 diagram above show the Use Case Diagram of deep learning localization for autonomous vehicle.

According to the use case diagram above, user will start the detection script that will launch the lane detection that scripted in real time. It will keep capturing the frame of the lane from camera and determine the appropriate direction of movement, whether it is a right turn, left turn, or straight ahead. Meanwhile, GPS module will concurrently detect the real time location of the Raspberry Pi at the robot car chassis model.

3.3 Function Model Diagram

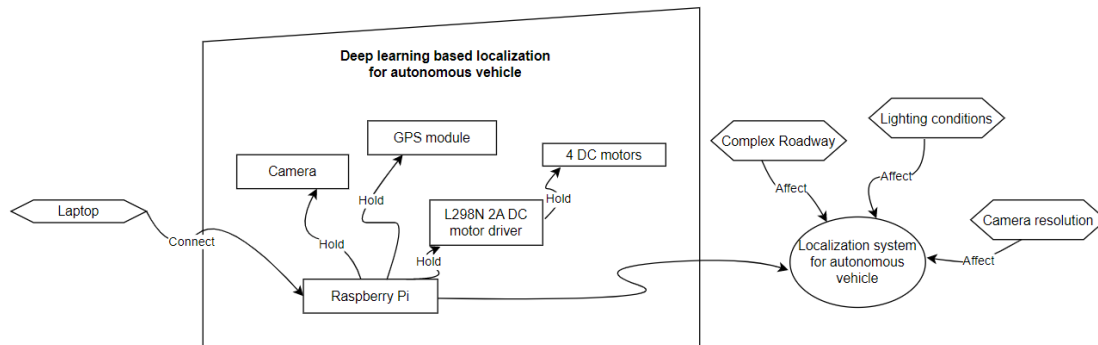


Figure 3.3.1 diagram showed the function model diagram of the system.

Based on the function model diagram above, laptop will connect to a robot chassis car model that built with main components, Raspberry Pi 3B+ through VNC Viewer application or Command Prompt SSH connection. The differences are through VNC Viewer application, it able to view the raspberry pi GUI just like other remote connection application such as TeamViewer, and AnyDesk application. It able to let user to view the real time camera frame detected. With Command Prompt SSH connection, it also can connect easily but it can't run the script that call the real time frame with this line "cv2.imshow("Contours", frame)". Thus, VNC Viewer application connection is better than the Command Prompt SSH connection. Then, Raspberry Pi 3B+ that holds camera module, GPS module and L298N 2A DC motor driver that connected with 4 DC motors. Camera is used to get the real time frame of detected lane back to raspberry pi. GPS module is used to retrieve the real time location back to raspberry pi. L298N 2A DC motor driver that connected 4 dc motors is used to control the output of power source from 4 AA battery holder. When the script runs in raspberry pi, it will send command to motor driver and it will control the direction of the motors go. Furthermore, there are several factors will affect the performance of this robot car chassis model such as complexity of roadway, the environment lighting conditions and camera resolution. These factors will greatly affect the performance of the robot car chassis model while making the correct direction and detecting correct lane with clearer frame.

Chapter 4 System Design

4.1 System Block Diagram

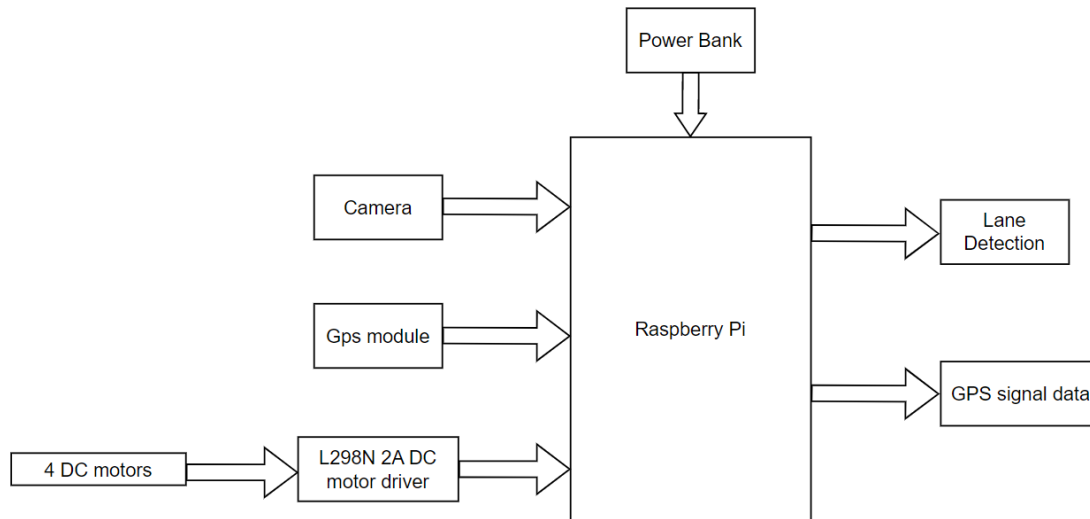


Figure 4.1.1 Block diagram of localization system using raspberry pi.

According to the block diagram above, it shows the various components that constitute the system. There are three primary input devices that are incorporated into the system, namely the camera module, GPS module, and L298N 2A DC motor driver. These input devices are connected to four DC motors, which are responsible for powering the system's movement.

The Raspberry Pi acts as the control centre for the system, and it is connected to the input devices and the power bank, which serves as the power source for the Raspberry Pi. The Raspberry Pi processes the input data obtained from the camera module, GPS module, and L298N 2A DC motor driver to generate the necessary output.

The output of the Raspberry Pi constitutes two primary data streams, which include the result of the lane detection function and GPS signal data. The lane detection function captures the frames of the lane from the camera module and analyses them in real-time using OpenCV to detect lane. Based on this analysis, the Raspberry Pi generates the necessary commands to control the DC motors, such as turning left, turning right, or moving straight ahead.

On the other hand, the GPS module is responsible for detecting the real-time location of the Raspberry Pi. In summary, the block diagram illustrates the complex interconnection of various hardware components that work together to facilitate the autonomous navigation of the system. The camera module, GPS module, and L298N 2A DC motor driver are the primary input devices that provide data to the Raspberry Pi, which processes the data to generate the necessary output required for the system.

4.2 System Components Specifications

4.2.1 Hardware

The project requires the integration of several hardware components, including a computer, a Raspberry Pi 3 B+, an L298N 2A DC Motor Driver, and a set of robot car chassis kits. The first step is to solder the black and red wires to the four DC motors, which represent the positive and negative terminals, respectively. The wires are then connected to the L298N 2A DC Motor Driver, which functions as the motors' control centre.

After connecting the motors to the driver, they are securely screwed onto the acrylic board. Using female-to-female jumper wires, the L298N 2A DC Motor Driver is then connected to the Raspberry Pi 3B+ GPIO Pins. The Raspberry Pi 3B+ is the primary testing and running platform for the Python scripts created for the project.

In summary, the project's hardware components include a computer, a Raspberry Pi 3 B+, an L298N 2A DC Motor Driver, and a robot car chassis kit. Soldering wires to the DC motors, connecting them to the L298N 2A DC Motor Driver, and securing the motors to the acrylic board are all steps in the hardware setup process. Female-to-female jumper wires are then used to connect the driver to the Raspberry Pi 3B+ GPIO pins. Finally, the Raspberry Pi 3B+ is used to test and run the Python scripts written for the project with the attached camera.

A] Laptop:

Laptop that will be used is Acer Aspire 5, with Intel Core i7-8550U, operating with Windows 11 Home, working with 12 GB DDR4 RAM, NVIDIA MX150 graphic card, and both 2 TB hard disk, SSD M.2 and HDD. Moreover, it also will be used to develop few python scripts such as keyboard-controlled driving, motor, lane detection, GPS location tracker and etc which run in the Raspberry pi 3 B+ model. Besides, it also will be used to test the scripts coded based on the recorded lane video by the robot chassis car.

Table 4.2.1.1 Specifications of laptop

Description	Specifications
Model	Acer Aspire 5 series
Processor	Intel Core i7-8550U
Operating System	Windows 10 Home
Graphic	NVIDIA GeForce MX150 2GB DDR3
Memory	12GB DDR4 RAM
Storage	1TB M.2 SSD, 1TB SATA HDD

B] Raspberry Pi:

Raspberry Pi that will be used is Raspberry Pi 3 B+, with ARMv7 Processor rev 4, operating with Raspberry Pi OS, working with 1 GB RAM, and 64 GB Micro SD card. It will be mainly used to retrieving the current location of the GPS module which attached with the Raspberry pi 3 B+ model. Then, the user able to see what the current location of it is in real time. Besides, it also needs to record the lane into mp4 format video.

Table 4.2.1.2 Specifications of Raspberry pi 3 B+

Description	Specifications
Model	Raspberry Pi 3 Model B Rev 1.2
Processor	ARMv7 Processor rev 4 (v71)
Operating System	Raspberry Pi OS
Memory	1GB RAM
Storage	64 GB Micro SD card

C] Cameras:

The cameras that will be used are called 5 Megapixel, Rev 1.3. It is a custom-designed Raspberry Pi add-on that is compatible with the majority of Raspberry Pi models. The resolution of the camera is 5 megapixels. It has a fixed focus lens on the board. This camera is able to capture 2592 X 1944-pixel image and it can support up to 1080p30.

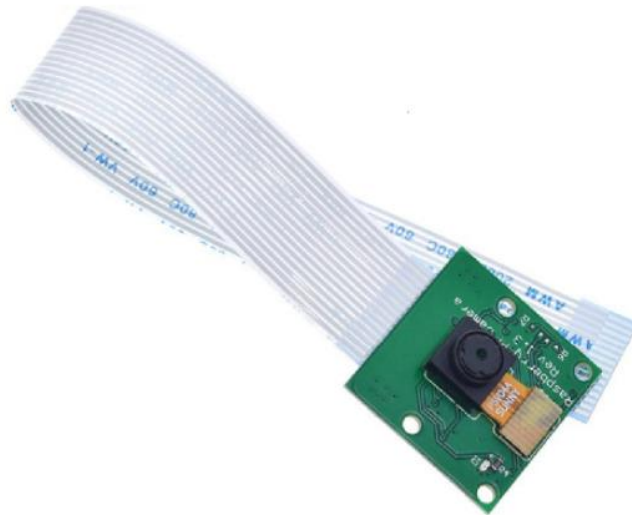


Figure 4.2.1.1 Diagram above showed a sample camera module for raspberry pi

Table 4.2.1.3 Specifications of Camera

Description	Specifications
Model	5 Megapixel, Rev 1.3
Size of board	25mm x 20mm x 9mm
Weight	3g
Connection	Short ribbon cable
Resolution	5 megapixels, 2592 X 1944 pixels static images
Supports quality	Up to 1080p30

D]GPS module:

GPS module will be used to trace on the status location of the raspberry pi. The GPS module will be used is called Neo 6M GPS module. It is cheap and easy to use by just plugging it in to the raspberry pi and it works. After that, we will write a script in python to retrieve the current location in longitude and latitude.



Figure 4.2.1.2 Diagram above showed a sample of GPS Module called “Neo 6M GPS module”.

Table 4.2.1.4 Specifications of GPS module

Description	Specifications
Model	Neo 6M GPS
Supply voltage	3.3V - 5V DC
Update rate	5Hz
Operating temperature range	-40 to +85 °C
Dimension	22 X 30 X 13 mm
Weight	19g

E] Power Bank:

Power bank is needed because the power source of the raspberry pi is using Micro-USB. The higher amount capacity of the power bank, the longer the raspberry pi can be lasted to. There is no specific model or brands needed as long as it is a power bank, support fast charging and high capacity, and low weight as it will make the robot car chassis easier in driving.

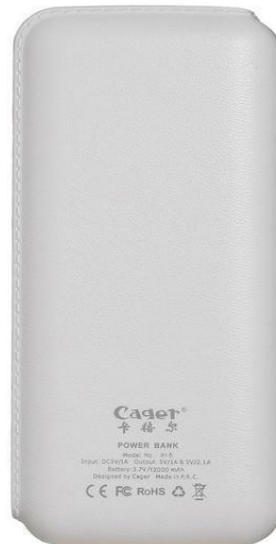


Figure 4.2.1.3 Diagram above showed an example of power bank which has the 12,000 mAh battery capacity.

Table 4.2.1.5 Specifications of Power bank

Description	Specifications
Model	Cager B17
Battery capacity	12000mAh/3.7V
Input voltage	Micro USB, DC5V-2A
Output voltage	USB, DC5V-1A / 5V-2.1A
Size	2.1 x 6.4 x 14cm
Weight	280g

4.2.2 Software

F] Software used:

Visual Studio Code (VS Code) is a free and open-sourced code editor created by Microsoft that is accessible on Windows, Linux, and macOS. It has become one of the most popular code editors due to its versatility, user-interface, and extensive library of extensions. VS Code offers support for many programming languages, including popular ones such as Python, JavaScript, HTML, C++ etc.

One of the main features of VS Code is its IntelliSense, which provides context-aware code completion suggestions and error highlighting as you type. It also includes a built-in debugger and version control integration, allowing developers to easily debug their code and track changes made to their codebase. Overall, VS Code is a great code editor that can be used for a variety of programming tasks ranging from simple scripts to complex projects.

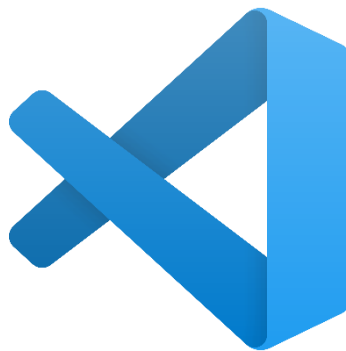


Figure 4.2.2.1 Visual Studio Code.

OpenCV (Open-Source Computer Vision Library) is a well-known open-source computer vision with machine learning library. It is written in C++ and has bindings for several programming languages, including Python. OpenCV for Python (cv2) is a Python library that provides a comprehensive set of functions and tools for image processing, video analysis, object detection and recognition, and more.

OpenCV-Python is easy to install, use. It offers a wide range of features and algorithms for computer vision tasks, including basic image processing, feature detection and extraction, object detection and tracking, face detection and recognition, and more. It also has tools for machine learning, including support for popular machine learning frameworks like TensorFlow and PyTorch. It able to read frames and applies image processing technique for detection.

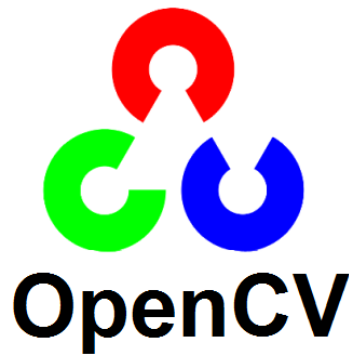


Figure 4.2.2.2 OpenCV.

VNC Viewer is a software application that allows remote access and control of a computer desktop or server from another computer or mobile device. To connect the two devices, it uses the Virtual Network Computing (VNC) protocol.

The user can view and interact with the remote desktop using VNC Viewer just as if they were physically in front of the computer. The software uses the remote computer's display to capture an image, which it then sends over the network to the local device. The user can then control the remote computer and carry out other operations using the keyboard and mouse of their local device.

VNC Viewer works with a variety of operating systems, including Windows, macOS, Linux, and mobile platforms such as iOS and Android. The software is available in both free and paid versions, each with its own set of features and capabilities.



Figure 4.2.2.3 VNC Viewer.

4.3 Circuits and Components Design

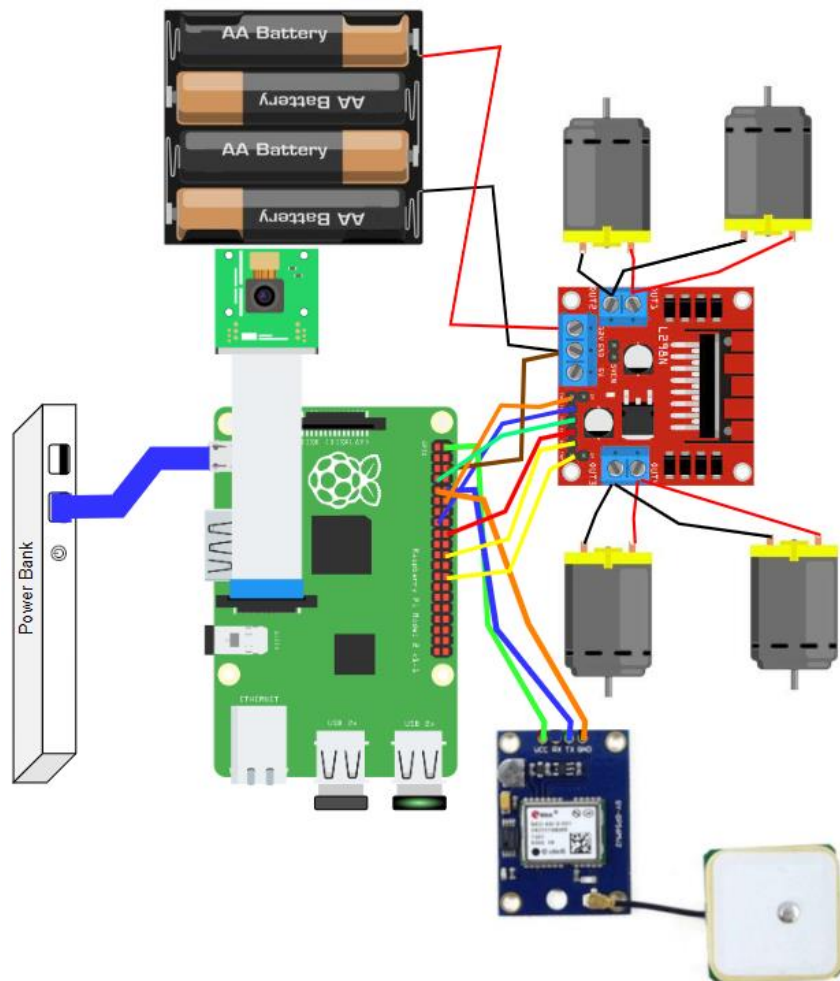


Figure 4.3.1 showed the robot car chassis model circuits design.

The Raspberry Pi 3B+ is the primary component of the system, based on the circuit design for the robot car chassis model shown above. Several other components, including the L298N 2A motor driver and GPS module, are connected to this device via female-to-female jumper wires.

The L298N 2A motor driver is in charge of controlling the four DC motors that power the robot car chassis model, which are powered by a 4 AA battery holder. The GPS module is also linked to the Raspberry Pi 3B+ to provide real-time location data.

A camera module is connected to the Raspberry Pi 3B+'s camera port to capture real-time camera frames. A power bank with a capacity of 12,000 mAh is used to power the Raspberry Pi 3B+.

In summary, the circuit design for the robot car chassis model involves the use of several components, with the Raspberry Pi 3B+ serving as the primary device. Other components include the L298N 2A motor driver, GPS module, camera module, and power bank. These components are interconnected using female-to-female jumper wires and are responsible for powering and controlling the various functions of the robot car chassis model.

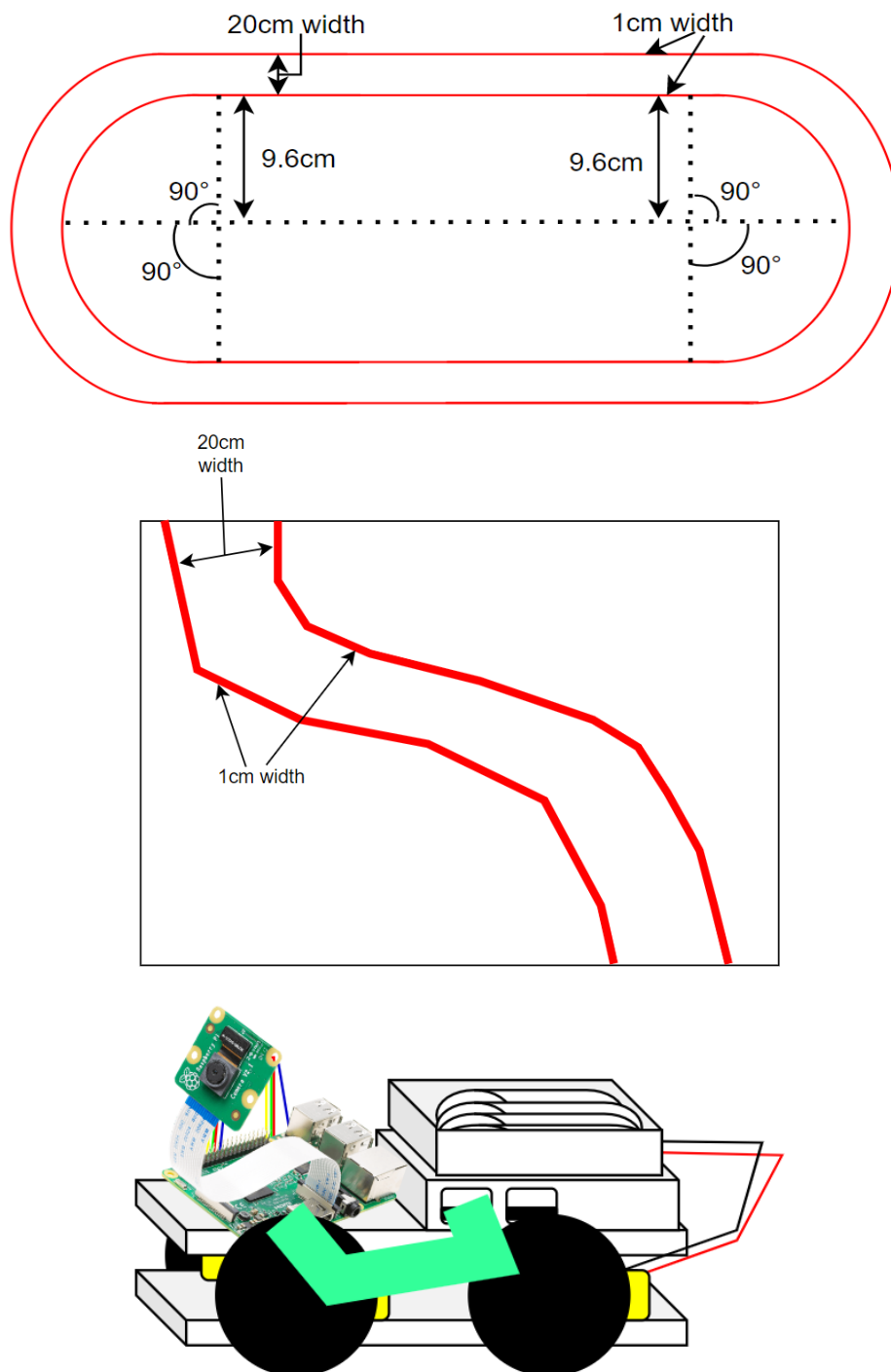


Figure 4.3.2 Conceptual diagram

The diagram above showed 2 types of lanes are designed which are S curve lane and oval lane along with the design diagram of robot car chassis model will be developed. First, both type of lanes is drawn on Mahjong paper with 20 cm width for lane and the boarder lines are in 1cm width. For the oval lane, both U-turn turning point of the lane has 20cm width lane, 1cm boarder lines between the lane and 9.6 cm radius excluding the boarder lines. For the S curve lane, it also has 20 cm width of white lane and 1cm for the red boarder lines. Moreover, the car chassis model is built by a raspberry pi which connected with camera module to detect the lane, L298N 2A dc motor driver that connected with 4 DC motor wheels and 4 AA battery holder.

4.4 System Components Interaction Operations

A robot car chassis model consists of multiple components that interact with each other to enable the car to detect and stay within its lane. The system typically includes hardware components such as camera module, 4 DC motors, L298N 2A motor driver, wheels, 4 AA battery holder Raspberry Pi 3B+ and power bank, as well as software components such as Visual Studio Code (VS Code) and OpenCV. First, the camera captures frame of the lane ahead and sends them to the processor for analysis in real time. The OpenCV based scripts algorithms will process the frame to detect red boarder lines between the lane by filtering the colour of the boarder lines that interested in this line “cv2.inRange(hsv, lower_red, upper_red)”. Then, it will be used to detect the red boarder lines between the lane and calculating the slope value of it. Once the boarder lines are detected and slope value is calculated, the motor control system takes action to make decision whether to turn left, right, or straight. These are made based on the result of the slope value calculated from the detected boarder lines. Then, it will call the “motor.py” scripts which contain the codes of controlling the 4 DC motors make decision such as turning left, right, or straight. It defined the number of GPIO pins on Raspberry Pi board and send signal to the L298N 2A motor driver that connected with the motors to control the direction of the wheels should move. Meanwhile, it also interacts with other components in the car such as the GPS module to get the current location data of it. To make a test on the scripts before running in the Raspberry Pi 3B+, the code will run on the Visual Studio Code (VS Code) code editor with the recorded lane video as frame input. Overall, the interactions between the various components of the robot car chassis model are critical to the successful operation of the system. By analysing and optimizing these interactions, the robot car chassis model can be built successfully.

Chapter 5

Experiment/Simulation

5.1 Hardware Setup

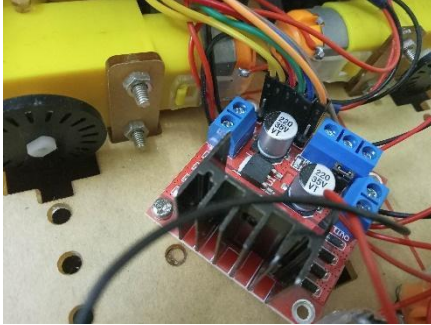


Figure 5.1.1 showed a L298N motor driver connects to jumper wires along with black and red wires of motors.



Figure 5.1.2 showed the L298N motor driver is well-connected with 4 DC motors and wheels.

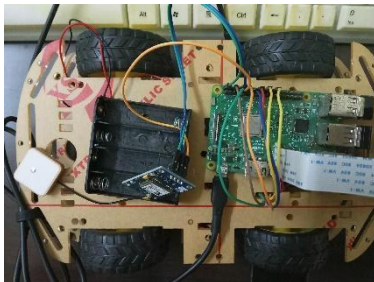


Figure 5.1.3 showed the jumper wires connected to Raspberry Pi 3B+.



Figure 5.1.4 showed GPS module is connected at the Raspberry Pi 3B+ GPIO pins.



Figure 5.1.5 showed the camera mounted at a board.

Figure 5.1.6 showed how the camera mounted and a better angle of frame view.

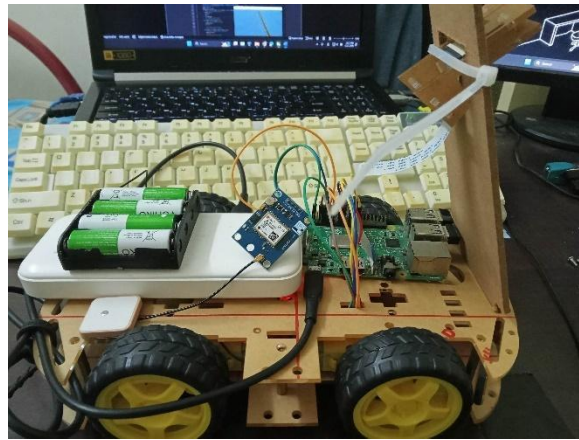


Figure 5.1.7 showed how the full robot car chassis is built.

According to the figures above showed, it shows how whole robot car chassis model is built. With this design of robot chassis car model, it able to connect Wi-Fi through the Wi-Fi USB receiver. Then, with the remote connection to Raspberry Pi 3B+, it able to let user controls the DC motors, camera module, and GPS module with the python-based script. The 4 AA battery holder and power bank are the power source of the robot car chassis model.

5.2 Software Setup

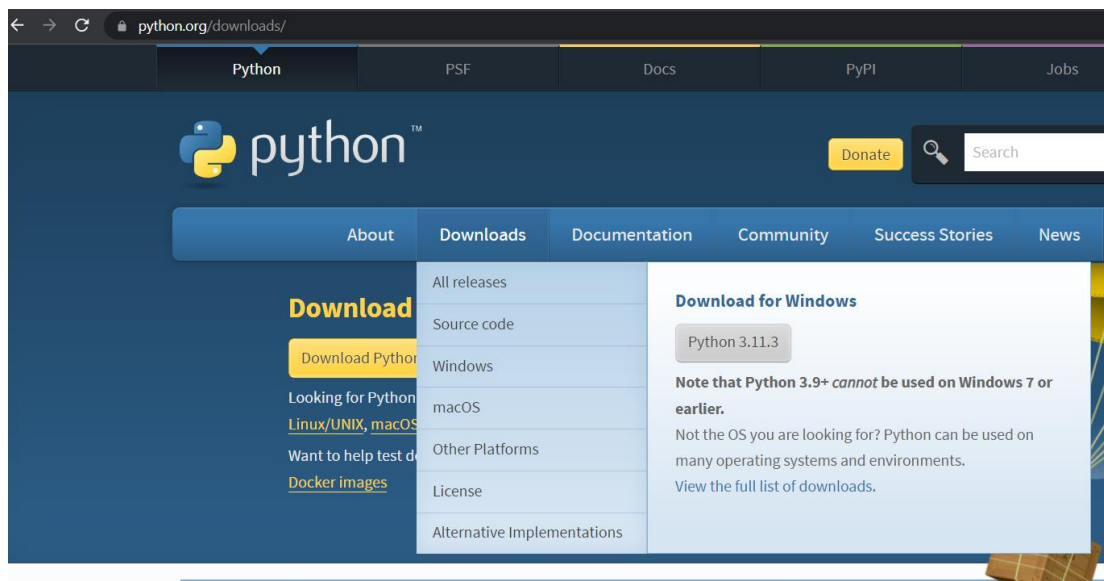


Figure 5.2.1 showed downloading python for Windows at python official website.

Based on the Figure 5.2.1, it showed where the python can be downloaded at official website. It is needed as the script need to be coded and tested in Windows 10 operating system laptop.

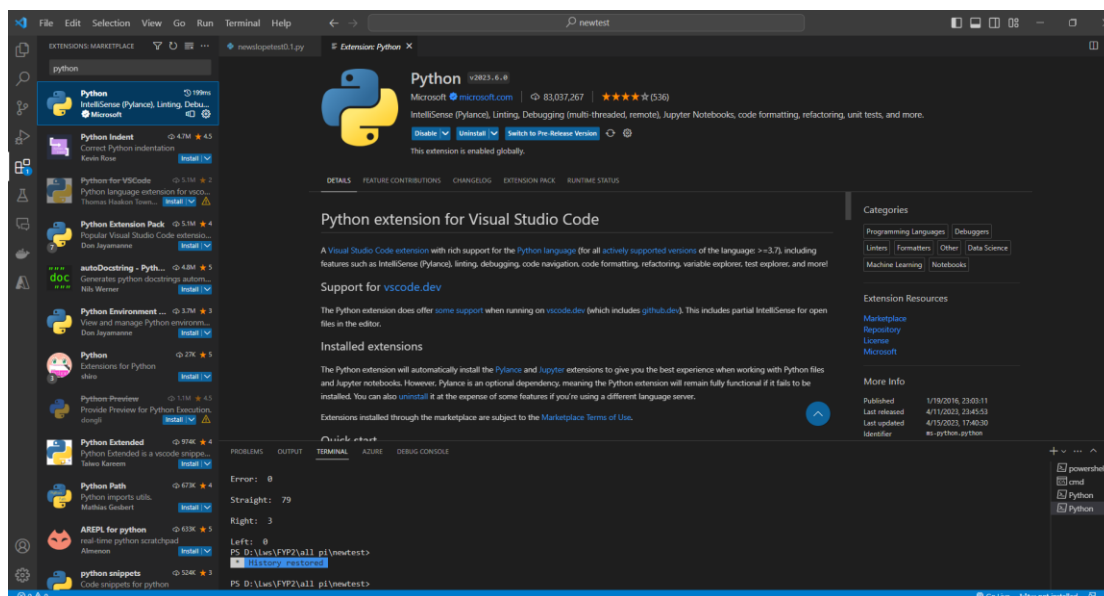


Figure 5.2.2 showed Python extension at VS Code extension store.

Based on the figure 5.2.2, it showed that python extension also available in the VS Code editor extension store. It can be installed from there too in order to code in python programming language.

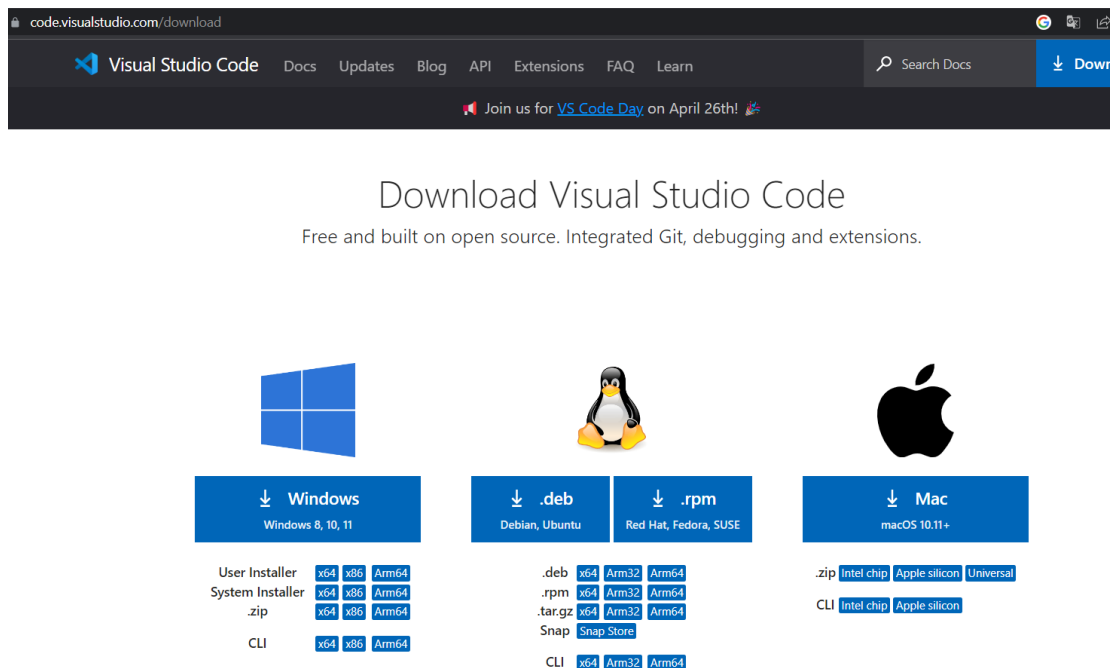


Figure 5.2.3 showed VS Code editor can be downloaded at different operating system at Visual Studio Code official website.

Based on the figure above showed where the VS Code editor can be downloaded at official website, it has various version for different operating system such as Windows, Linux, and Mac operating system. It is needed to be install in laptop for coding purpose.

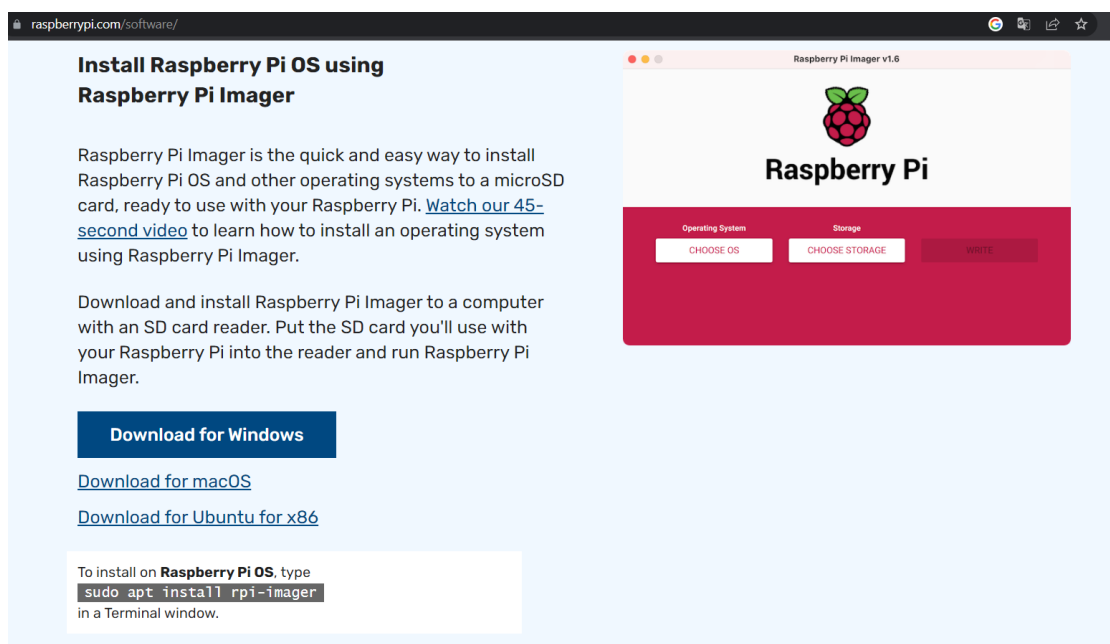
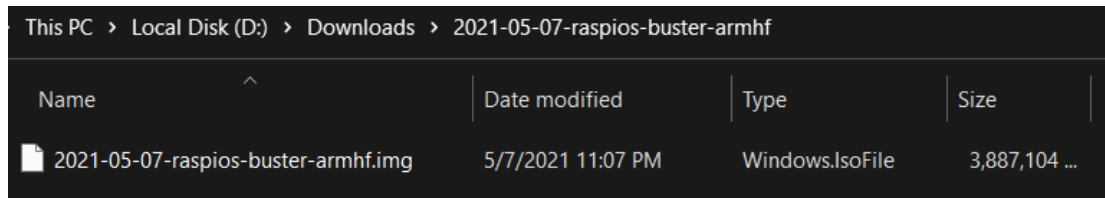


Figure 5.2.4 showed Raspberry Pi Imager is available at the official website of Raspberry Pi.

Based on the figure 5.2.4 showed where the Raspberry Pi Imager can be downloaded and installed from its official website. It is used to install the raspberry pi operating system images write into SD card which used to boot the Raspberry Pi 3B+.



This PC > Local Disk (D:) > Downloads > 2021-05-07-raspbian-buster-armhf

Name	Date modified	Type	Size
2021-05-07-raspbian-buster-armhf.img	5/7/2021 11:07 PM	Windows.IsoFile	3,887,104 ...

Figure 5.2.5 showed the raspberry pi operating system image.

Based on the figure 5.2.5 showed above, it is a raspberry pi operating system image that used to boot the raspberry pi. This is the Raspberry Pi operating system version. It is because the Wi-Fi adapter used as Wi-Fi receiver need to download its driver and it can't support the latest raspberry pi operating system. Thus, this specific version of raspberry pi operating system is needed to compatible with the Wi-Fi receiver driver.

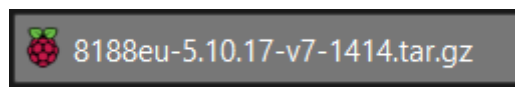
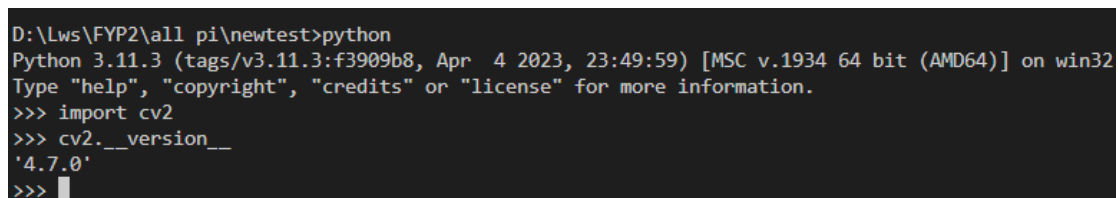


Figure 5.2.6 showed the Wi-Fi receiver driver installed.

Based on the figure 5.2.6 showed above, it showed the Wi-Fi receiver driver used. It is installed to let the Wi-Fi USB receiver able to scan and connect to the Wi-Fi. This is the specific version of driver used for the Wi-Fi USB receiver.



```
D:\Lws\FYP2\all pi\newtest>python
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'4.7.0'
>>>
```

Figure 5.2.7 showed the OpenCV cv2 library version installed.

Based on the figure 5.2.7 showed above, it is the OpenCV cv2 library version used. It is needed when writing the lane detection script as it is a powerful library of computer vision and image processing algorithms.

5.3 Setting and Configuration

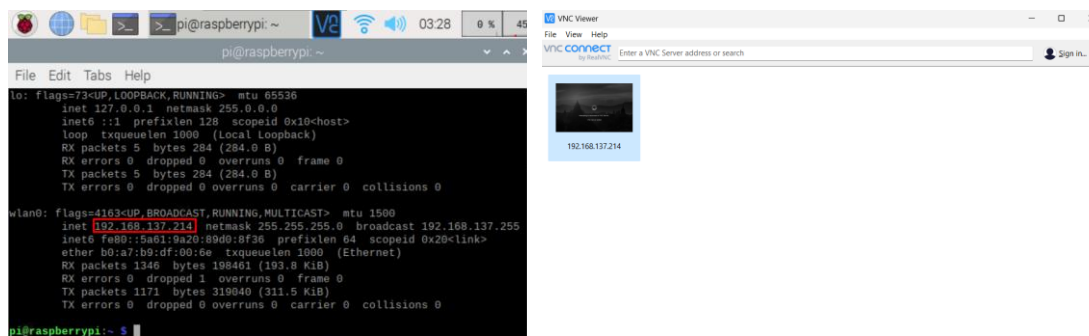


Figure 5.3.1 showed the Ip address of Raspberry Pi 3B+ Figure 5.3.2 showed the connection from VNC Viewer

Based on the figures above showed, to have a remote control from local laptop, Raspberry Pi need to connect to the same local network that also connected by the local laptop. Then, find out the Ip address of raspberry pi at the terminal with the command of “ifconfig”. Then, type the Ip address of raspberry pi to the VNC Viewer application there and type the raspberry pi’s password. Once it is done, the remote control of raspberry pi from laptop is completed.

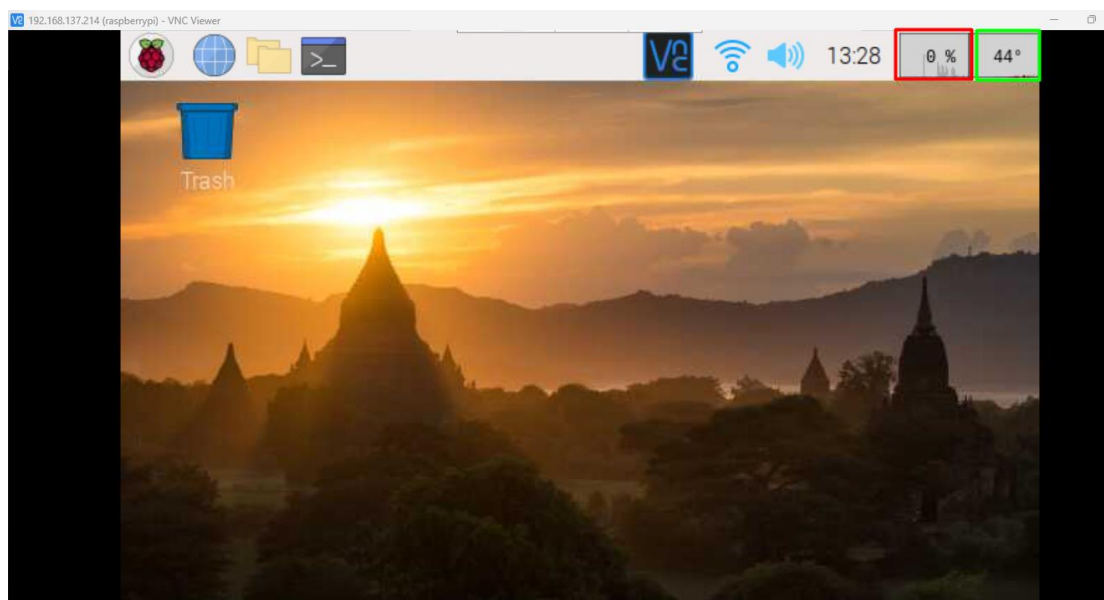


Figure 5.3.3 showed the CPU usage and temperature monitor.

Based on the figure 5.3.3 showed above, it is important to turn on the CPU usage and temperature monitor to keep monitoring it. Thus, it will show how the raspberry pi's performance every time run the Python script.

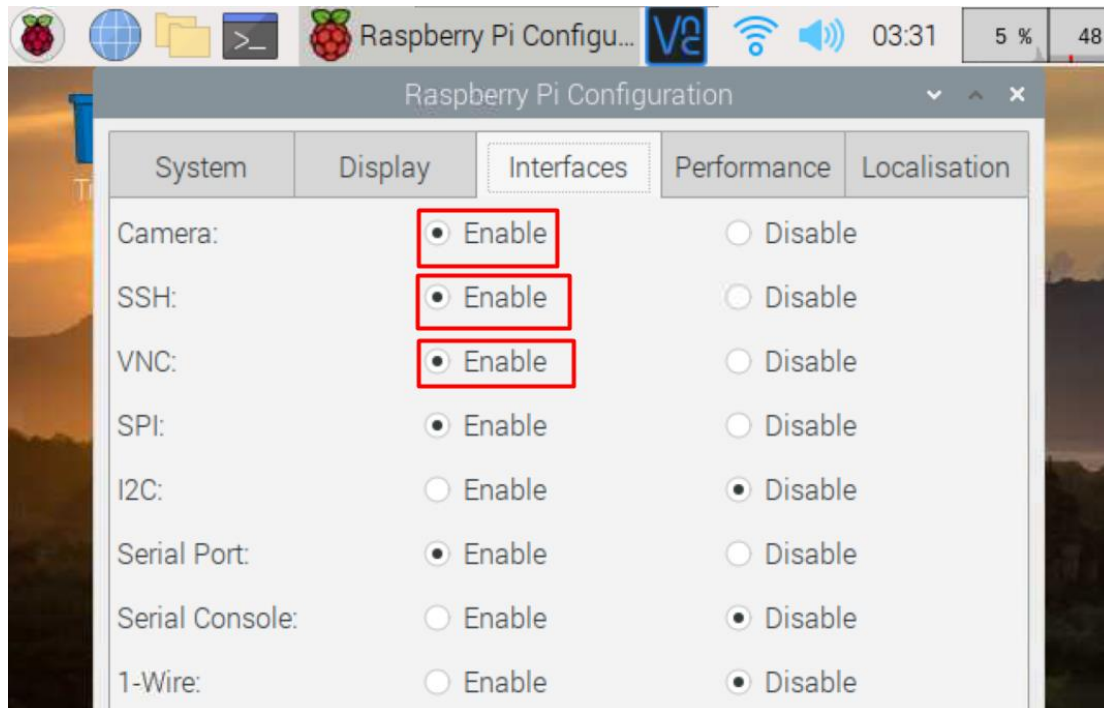


Figure 5.3.4 showed the Raspberry Pi configuration page.

Based on the figure 5.3.4 showed above, there are few features needed to turn on once the raspberry pi operating system is installed such as camera, SSH connection and VNC connection. Thus, these features able to use and work normally.

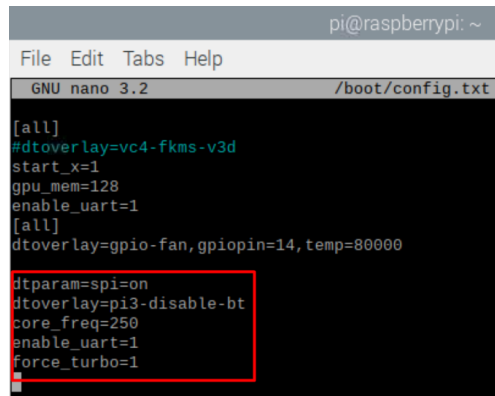
GPS module

```
sudo nano /boot/config.txt
```

Figure 5.3.5 showed the command of accessing config.txt.

```
dtoverlay=spi=on
dtoverlay=pi3-disable-bt
core_freq=250
enable_uart=1
force_turbo=1
```

Figure 5.3.6 showed the list of commands.



```

pi@raspberrypi: ~
File Edit Tabs Help
GNU nano 3.2 /boot/config.txt

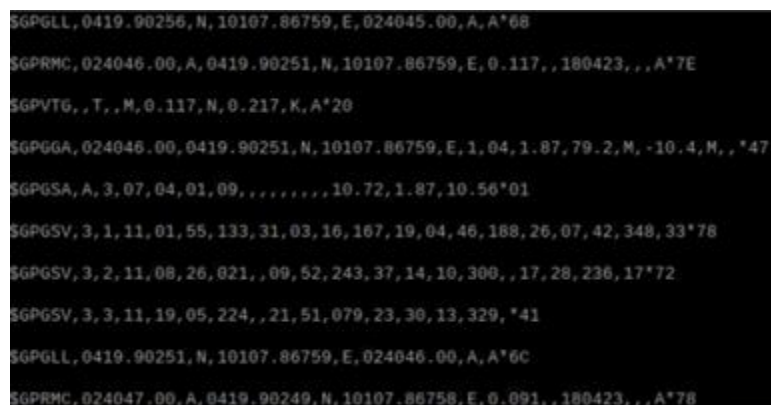
[all]
#dtoverlay=vc4-fkms-v3d
start_x=1
gpu_mem=128
enable_uart=1
[all]
dtoverlay=gpio-fan,gpiopin=14,temp=80000

dtparam=spi=on
dtoverlay=pi3-disable-bt
core_freq=250
enable_uart=1
force_turbo=1

```

Figure 5.3.7 showed the command paste in config.txt.

Based on the figures showed above, in order to setup the GPS module after connecting with Raspberry Pi 3B+ with jumper wires to the correct raspberry pi's GPIO pins, the command showed in the figure 5.3.5 need to be type in terminal and click 'enter'. It will access to the config.txt that stored at '/boot' folder. Then, paste the list of commands showed in figure 5.3.6 at the bottom of the config.txt file. Next, close the text editor by clicking 'ctrl' + 'x' and 'y' for saving the config.txt after pasting it just like the figure 5.3.7 showed. Lastly, type "sudo reboot" in the terminal and click 'enter' to reboot the raspberry pi. After rebooting, open the terminal and type "sudo cat /dev/ttyAMA0" and it will show lots of location data that need to express by using python script.



```

$GPGLL,0419.90256,N,10107.86759,E,024045.00,A,A*68
$GPRMC,024046.00,A,0419.90251,N,10107.86759,E,0.117,,180423,,A*7E
$GPVTG,,T,,M,0.117,N,0.217,K,A*20
$GPGGA,024046.00,0419.90251,N,10107.86759,E,1.04,1.87,79.2,M,-10.4,M,,*47
$GPGSA,A,3,07,04,01,09,,,,,,,,,10.72,1.87,10.56*01
$GPGSV,3,1,11,01,55,133,31,03,16,167,19,04,46,188,26,07,42,348,33*78
$GPGSV,3,2,11,08,26,021,,09,52,243,37,14,10,300,,17,28,236,17*72
$GPGSV,3,3,11,19,05,224,,21,51,079,23,30,13,329,*41
$GPGLL,0419.90251,N,10107.86759,E,024046.00,A,A*6C
$GPRMC,024047.00,A,0419.90249,N,10107.86758,E,0.091,,180423,,A*78

```

Figure 5.3.8 showed the location data after the command "sudo cat /dev/ttyAMA0" is typed in terminal.

5.4 System Operation

Lane Detection – Oval Lane

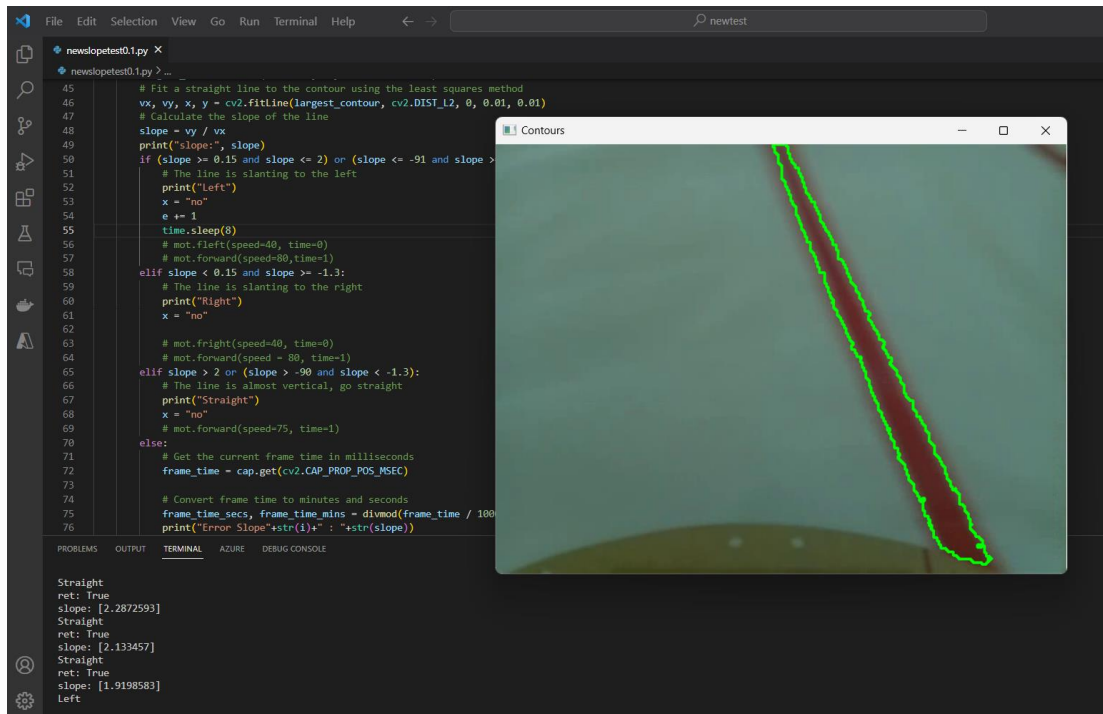


Figure 5.4.1 showed the right red boarder line of lane is detected and show “left” as the moving direction.

Based on the figure showed above, when the frame of video is fed as input, the red boarder line between the lane is detected and make responses of direction left after calculating the slope value of the lines.

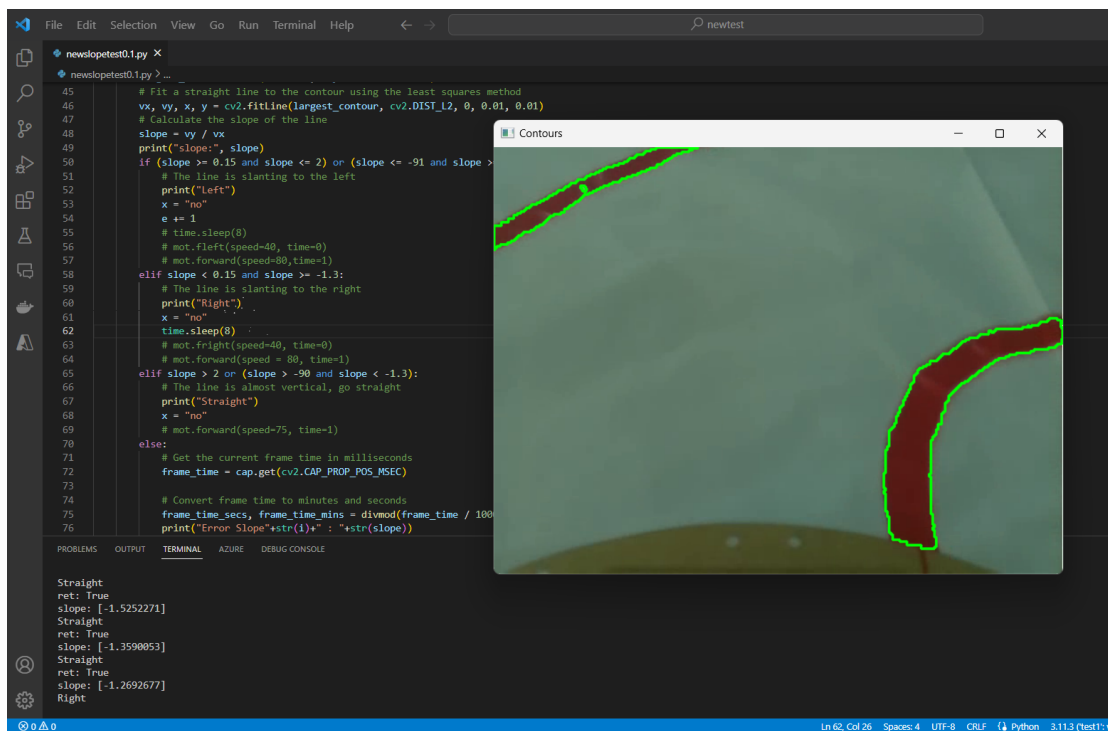


Figure 5.4.2 showed the two red boarder lines of lane are detected and show “right” as the moving direction.

Based on the figure showed above, when the frame of video is fed as input, the red boarder lines between the lane are detected and make responses of direction right after calculating the slope value of the lines.

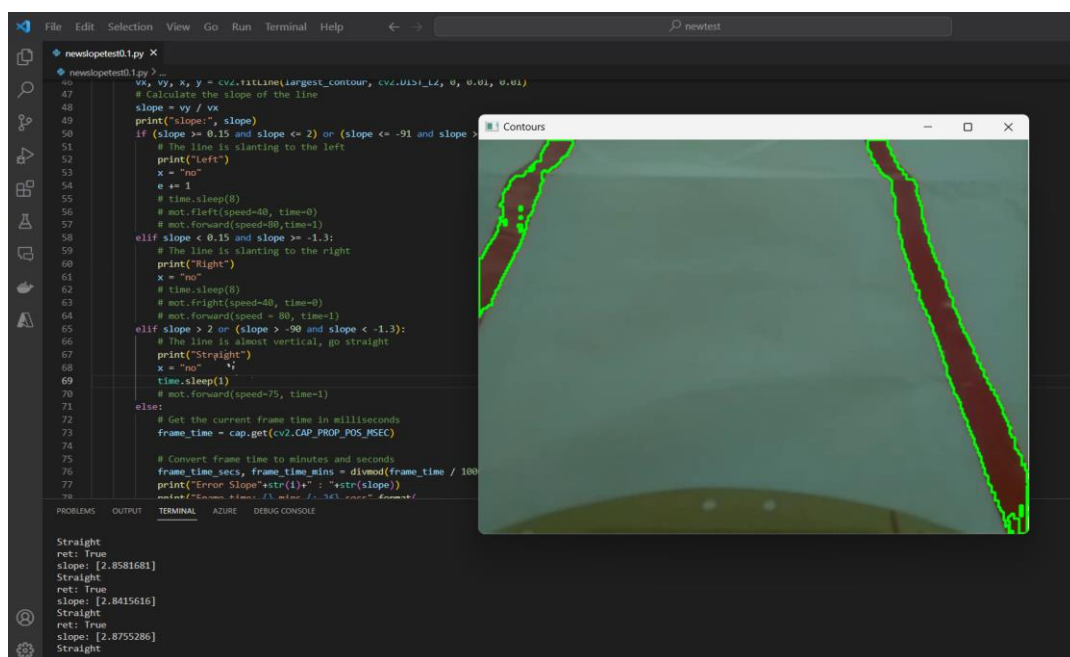


Figure 5.4.3 showed the two red boarder lines of lane are detected and show “straight” as the moving direction.

Based on the figure showed above, when the frame of video is fed as input, the red boarder lines between the lane are detected and make responses of direction straight after calculating the slope value of the lines.

Lane Detection – S curve Lane

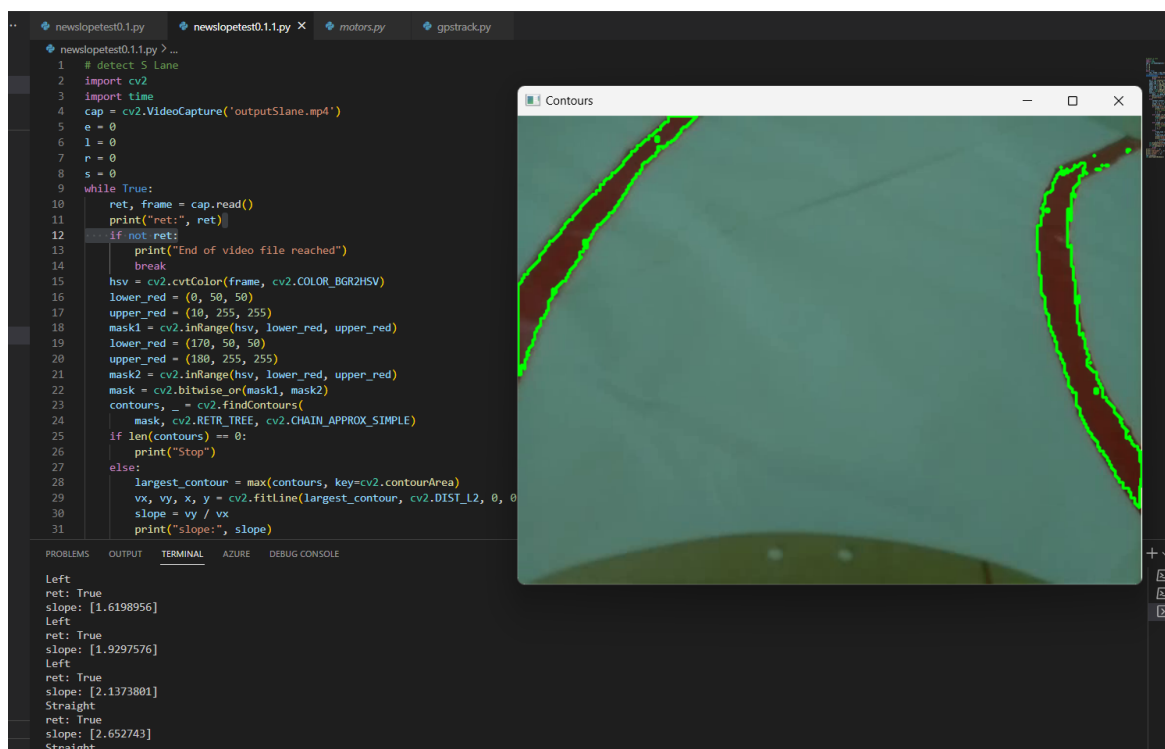


Figure 5.4.4 S curve lane.

Based on the figure showed above, it is a S curve lane. It has 20 cm width of white lane and 2 red boarder lines between it are detected. It will calculate the slope value and determined the direction it should turn in result of slope value and left, right, or straight.

```

75     else:
76         # Get the current frame time in milliseconds
77         frame_time = cap.get(cv2.CAP_PROP_POS_MSEC)
78         e += 1
79         # Convert frame time to minutes and seconds
80         frame_time_secs, frame_time_mins = divmod(frame_time / 1000, 60)
81         print("Error Slope"+str(e)+" : "+str(slope))
82         print("Frame time: {} mins {:.2f} secs".format(
83             int(frame_time_mins), frame_time_secs))
84
85         x = "Error"
86         # time.sleep(8)
87         # time.sleep(0.1)
88
89         # Show the frame with the contours
90         cv2.drawContours(frame, contours, -1, (0, 255, 0), thickness=2)
91         cv2.imshow("Contours", frame)
92
93         # Exit the loop if the 'q' key is pressed
94         if cv2.waitKey(1) & 0xFF == ord('q'):
95             break
96         # Release the VideoCapture object and close all windows
97         print("Error: ", e)
98         print("\nStraight: ", s)
99         print("\nRight: ", r)
100        print("\nLeft: ", l)
101        cap.release()
102        cv2.destroyAllWindows()
103

```

Figure 5.4.5 showed the codes wrote in lane detection about the error handling.

Based on the figure showed above, error handling lines are coded so that it is easy to debug by showing the slope value and frame time in minutes and seconds that has error when error occurred. Then, it will calculate the amount of error in total.

```

import cv2
cap = cv2.VideoCapture('output.mp4')
e = 0
l = 0
r = 0
s = 0
while True:
    ret, frame = cap.read()
    print("ret:", ret)
    if not ret:

```

```

    print("End of video file reached")
    break
hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
lower_red = (0, 50, 50)
upper_red = (10, 255, 255)
mask1 = cv2.inRange(hsv, lower_red, upper_red)
lower_red = (170, 50, 50)
upper_red = (180, 255, 255)
mask2 = cv2.inRange(hsv, lower_red, upper_red)
mask = cv2.bitwise_or(mask1, mask2)
contours, _ = cv2.findContours(
    mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
if len(contours) == 0:
    print("Stop")
else:
    largest_contour = max(contours, key=cv2.contourArea)
    vx, vy, x, y = cv2.fitLine(largest_contour, cv2.DIST_L2, 0, 0.01, 0.01)
    slope = vy / vx
    print("slope:", slope)
    if (slope >= 0.15 and slope <= 2) or (slope <= -91 and slope >= -267):
        print("Left")
        x = "no"
        l += 1
    elif slope < 0.15 and slope >= -1.3:
        print("Right")
        x = "no"
        r += 1
    elif slope > 2 or (slope > -90 and slope < -1.3):
        print("Straight")
        x = "no"
        s += 1
    else:
        frame_time = cap.get(cv2.CAP_PROP_POS_MSEC)

```

```

    e += 1
    frame_time_secs, frame_time_mins = divmod(frame_time / 1000, 60)
    print("Error Slope"+str(e)+" : "+str(slope))
    print("Frame time: { } mins {:.2f} secs".format(
        int(frame_time_mins), frame_time_secs))
    x = "Error"
    cv2.drawContours(frame, contours, -1, (0, 255, 0), thickness=2)
    cv2.imshow("Contours", frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
print("Error: ", e)
print("\nStraight: ", s)
print("\nRight: ", r)
print("\nLeft: ", l)
cap.release()
cv2.destroyAllWindows()

```

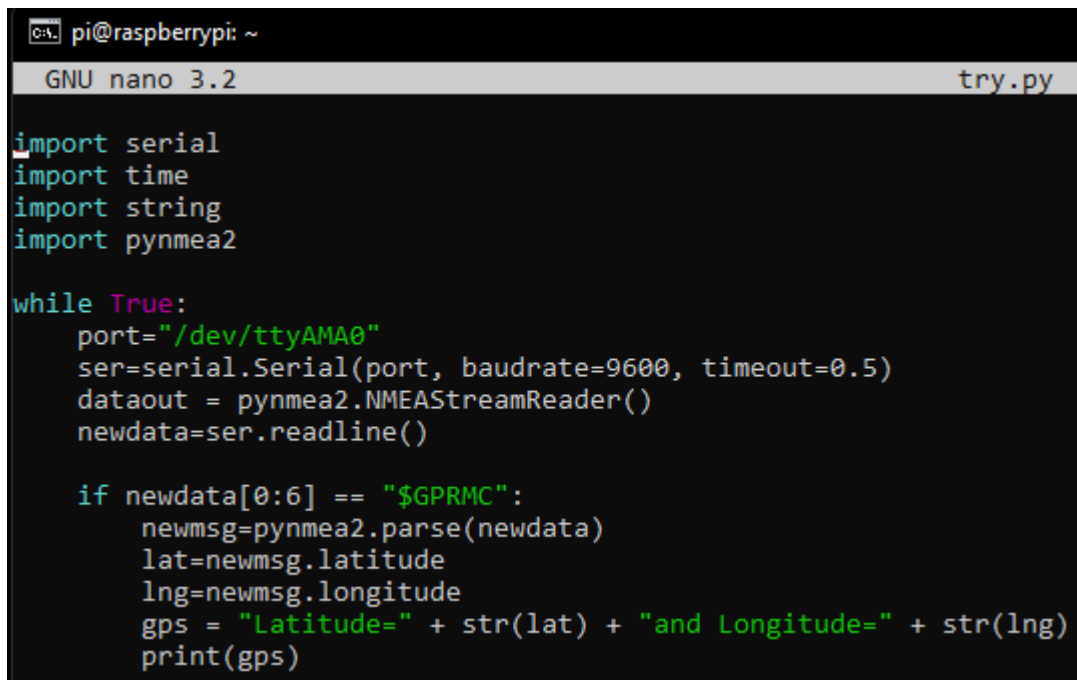
Table 5.4.1 Source code of lane detection.

Based on the table 5.4.1 above, it showed the whole source code wrote in this final year project 2. First, it imports the cv2 libraries and read the recorded lane video called “out.mp4”. Then, declare the variables to store the increment value later. Moreover, it will read each frame of video while it is true. Furthermore, a list of code to filter out the red colour in order to detect the interested colour of red boarder lines and calculate the slope value of it. with the condition set, it will print the result “left”, “right”, or “straight” based on the slope value. It also prints “stop” if the contours are equal to 0. Besides, it also has an error handling for detect the error occurred by showing its frame time and slope value. Then, it will stop when the ‘q’ as quit is pressed or the video is ended. Finally, it will print out the amount of error, straight, left, and right as result for the video.

Location detection

Moreover, we setup the raspberry pi with the GPS module by connecting them using jumper wire. Then, we install the raspberry pi operating system into SD card by using

Raspberry Pi Imager. Next, all the library or setup are installed to run the GPS module. Then, we wrote a simple function in python code to get the longitude and latitude of the current location.



```

pi@raspberrypi: ~
GNU nano 3.2 try.py

import serial
import time
import string
import pynmea2

while True:
    port="/dev/ttyAMA0"
    ser=serial.Serial(port, baudrate=9600, timeout=0.5)
    dataout = pynmea2.NMEAStreamReader()
    newdata=ser.readline()

    if newdata[0:6] == "$GPRMC":
        newmsg=pynmea2.parse(newdata)
        lat=newmsg.latitude
        lng=newmsg.longitude
        gps = "Latitude=" + str(lat) + "and Longitude=" + str(lng)
        print(gps)

```

Figure 5.4.6 above showed a file called “try.py”.

Based on the figure 5.4.5 above, a file called “try.py” is written in python language to grab the results of GPS module.

In GPS module implementation, it involves some setup in raspberry pi setting, coding on website and functions written in python language and testing on location.

In this project, it aims to propose some solutions for better localization system and overcome the main limitation of methods that proposed by the studied papers. The solution that will be implement a python OpenCV based lane detection script and show the current location of the raspberry pi using GPS module. The methods are using 1 raspberry pi 3 model B+, 1 camera attached with the raspberry pi, GPS tracker, power bank with 20,000mAh, and python OpenCV based lane detection scripts. It believes that it can highly improve the accuracy of the autonomous vehicles by using these methods although it is cheaper than the proposed methods in studied papers.

5.5 Implementation Issues and Challenges

During the final year project 2, there are many issues and challenges faced such as high time consumption, and invalid and incorrect information on online guideline.

First, high time consumption of writing and testing the scripts. It needs to keep trying the scripts wrote to detect the boarder lines of the lane and figuring out the errors occurred in the scripts. It can be due to many reasons, raspberry pi setup problem, environment light condition or invalid libraries installed while self-learning on these libraries used. It is because the environment light condition, and other issues can cause bad detection on the boarder lines of the lane. Besides, there are many errors too in setting up the GPS module in Raspberry Pi such as bad soldering of male pin header on GPS module, invalid version of raspberry pi operating system, and others. Luckily, most of the issues and challenges that faced is solved one by one although it took long period of time. It is benefit because the more errors occurred, the more solutions are learned while solving it.

5.6 Concluding Remark

In shorts, location detection and lane detection are built successfully. With the GPS module, it believes able to retrieve the correct location data continuously. Besides, with the robot car chassis model, it believes that it able to make detection on the frame of lane using camera and control the 4 DC motors to make correct direction based on what detected.

Chapter 6 System Evaluation and Discussion

6.1 System testing and Performance Metrics

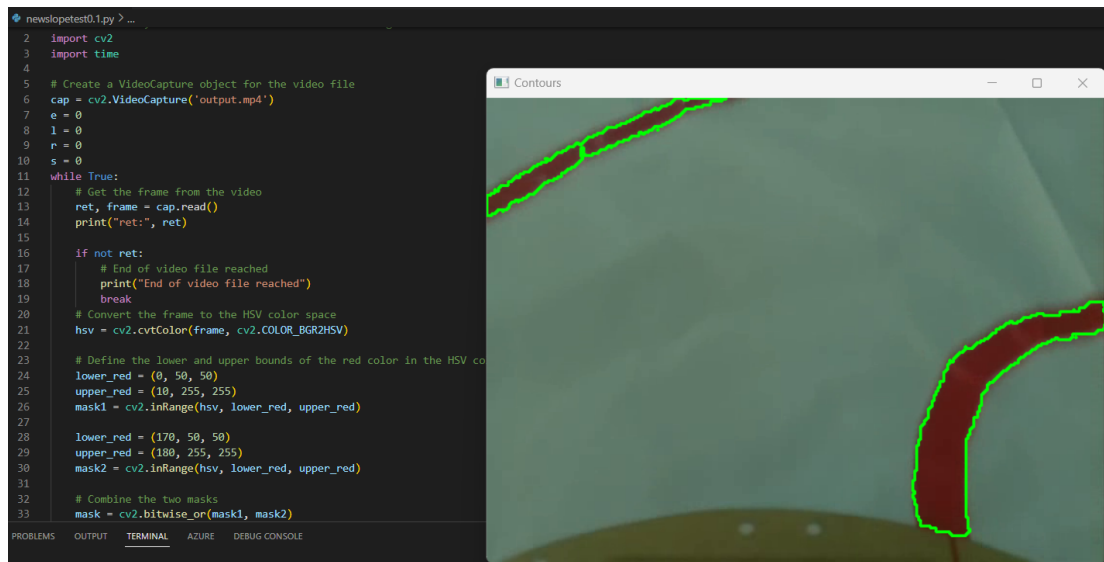


Figure 6.1.1 showed the picture of the red border lines between the lane is detected.

Based on the figure showed above, it is trying to detect the lane from the video frame. The boarder lines between the lane are detected correctly. When the lines are detected, it will mask the lines with the green lines around it to show the boarder lines are detected. The performance of the lane detection script is good while it takes the recorded lane video as input and runs with Visual Studio Code (VS Code) in Windows 10 Laptop.



Figure 6.1.2 showed the robot chassis car model is running on the designed path.

Based on the figure above, it shows the robot car chassis model is going straight out of the lane, but the frame showed is still at the previous frame which showing the turning right. It can be concluded that Raspberry Pi 3B+ is not very good in doing the real time lane detection as its frame still freezes at the previous frame, not the latest frame showed.

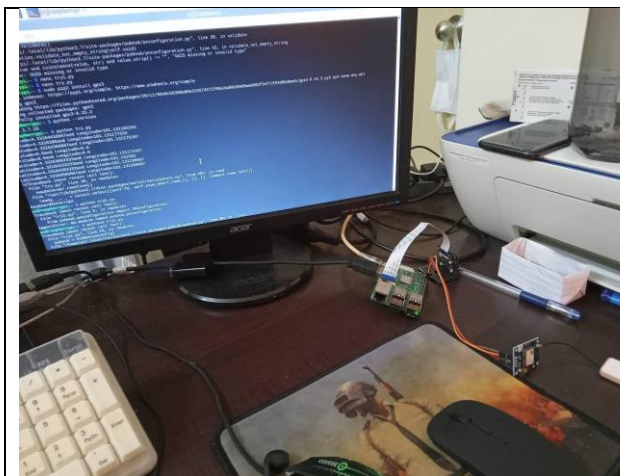


Figure 6.1.3 showed the raspberry pi is attached with the GPS module.

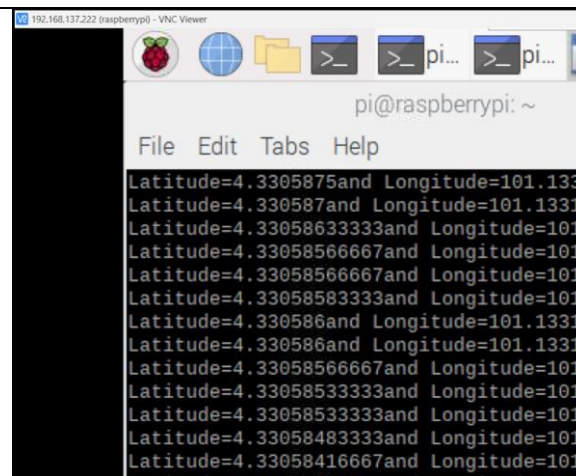


Figure 6.1.4 showed the result of location data.

Based on the figure 6.1.3 and 6.1.4, the raspberry pi is attached with GPS module and the monitor is showing the result after running the file that coded in python. It showed the longitude

and latitude that detected from GPS module. With the value retrieved, it able to know the current location of the robot car chassis model.

6.2 Testing Setup and Result

The image shows a Visual Studio Code editor window with a Python script named `newslopetest0.1.py`. The script is a lane detection program using OpenCV. It captures video frames, finds contours, and calculates the error, slope, and frame time. The terminal output shows the results of the script execution.

```

newslopetest0.1.py X
newslopetest0.1.py > ...
78     frame_time = cap.get(CV2.CAP_PROP_FPS_MSEC)
79     e += 1
80     # Convert frame time to minutes and seconds
81     frame_time_secs, frame_time_mins = divmod(frame_time / 1000, 60)
82     print("Error Slope"+str(e)+" : "+str(slope))
83     print("Frame time: {} mins {:.2f} secs".format(
84         int(frame_time_mins), frame_time_secs))
85
86     x = "Error"
87     # time.sleep(8)
88     # time.sleep(0.1)
89
90     # Show the frame with the contours
91     cv2.drawContours(frame, contours, -1, (0, 255, 0), thickness=2)
92     cv2.imshow("Contours", frame)
93
94     # Exit the loop if the 'q' key is pressed
95     if cv2.waitKey(1) & 0xFF == ord('q'):
96         break
97
98     # Release the VideoCapture object and close all windows
99     print("Error: ", e)
100    print("\nStraight: ", s)
101    print("\nRight: ", r)
102    print("\nLeft: ", l)
103    cap.release()
104    cv2.destroyAllWindows()
105

```

The terminal output shows the results of the script execution:

```

Right
ret: False
End of video file reached
Error: 0
Straight: 1307
Right: 1036
Left: 4
PS D:\Lws\FYP2\all pi\newtest>

```

Figure 6.2.1 showed performance of the lane detection.

Based on the figure showed above, it is the lane detection script that run in Visual Studio Code editor. It can see that based on the printed results, the error handling code is detected 0 error,

1307 frames in recorded lane video are detected to go straight, 1036 frames are detected to turn right, and 4 frames are detected to turn left.

```

File "/home/pi/.local/lib/python3.7/site-packages/pubnub/pnconfiguration.py", line 4
    PNConfiguration.validate_not_empty_string(self.uuid)
File "/home/pi/.local/lib/python3.7/site-packages/pubnub/pnconfiguration.py", line 4
    assert value and isinstance(value, str) and value.strip() != "", "UUID missing"
AssertionError: UUID missing or invalid type
pi@raspberrypi:~ $ nano try1.py
pi@raspberrypi:~ $ nano try.py
pi@raspberrypi:~ $ sudo pip2 install gps3
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting gps3
  Downloading https://files.pythonhosted.org/packages/20/c1/9548cb5388a85e31557d1f1f6
Installing collected packages: gps3
Successfully installed gps3-0.33.3
pi@raspberrypi:~ $ python --version
Python 2.7.16
pi@raspberrypi:~ $ python try.py
Latitude=4.33164416667and Longitude=101.131183333
Latitude=4.3316395and Longitude=101.131177333
Latitude=4.33163366667and Longitude=101.131175167
Latitude=0.0and Longitude=0.0
Latitude=0.0and Longitude=0.0
Latitude=4.33163683333and Longitude=101.131174167
Latitude=4.33164233333and Longitude=101.131183
Latitude=4.33164583333and Longitude=101.131196667
Latitude=4.33164566667and Longitude=101.131209667
^CTraceback (most recent call last):
  File "try.py", line 10, in <module>
    newdata=ser.readline()
  File "/usr/lib/python2.7/dist-packages/serial/serialposix.py", line 483, in read
    ready, _ = select.select([self.fd, self.pipe_abort_read_r], [], [], timeout.time_left())
KeyboardInterrupt
pi@raspberrypi:~ $ python try1.py
Traceback (most recent call last):
  File "try1.py", line 5, in <module>
    from pubnub.pnconfiguration import PNConfiguration
ImportError: No module named pubnub.pnconfiguration
pi@raspberrypi:~ $ python3 try1.py
Traceback (most recent call last):
  File "try1.py", line 16, in <module>
    pubnub = Pubnub('pubnub')

```

Figure 6.2.2 above showed the result of the longitude and latitude of the current location.

Based on the figure 6.2.2 above, it showed the result of longitude and latitude that detected from the GPS module after running the “try.py”. It showed the current location of where it is located after the detection.

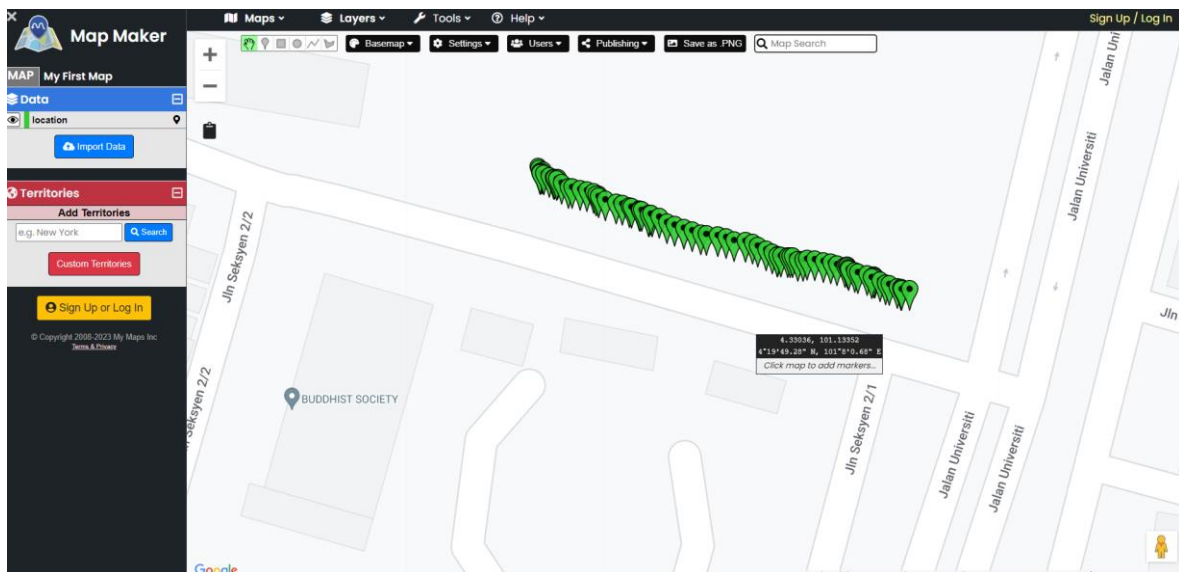


Figure 6.2.3 above showed the pinned result of the longitude and latitude.

Based on the figure 6.2.3 above, it showed the pinned result of longitude and latitude that detected from the GPS module after running the “try.py” while the robot car chassis running. It showed the latest current location of where it is located after the detection and able to show in map with the location data detected.

6.3 Project Challenges

In this final year project 2, there are many challenges faced in the project development. One of the project challenges is bad performance on Raspberry Pi 3B+ CPU. It is because when the lane detection script runs, the frame showed starts lagging although it still able to print the correct direction it should goes. Then, to ensure it is because of the low or bad performance of Raspberry Pi CPU, a recorded lane video is made by a keyboard-controlled scripts which able to run with keyword's keys such as 'w', 'a', 's', 'd', 'q', 'e' for motors direction and 'r' and 't' are for start and stop recording. Then, the lane detection script is tested using Windows 10 Visual Studio Code (VS Code). The result showed that the lane is perfectly detected, and the responses of motors direction are good too.

6.4 Objectives Evaluation

Based on the objective of this project mentioned above is to provide a solution in increasing the accuracy of localization of autonomous vehicle by using OpenCV by building a car chassis model that run by python OpenCV **detecting the lane** and **get location data from GPS module**. Based on the results stated in 6.2, the lane detection script is well-performed in lane detection and location data from the GPS module is retrieved successfully. However, the lane detection script is performed badly while running in the robot car chassis model. It is because of the low or bad CPU of Raspberry Pi 3B+. It not able to handle heavy task as the lane detection script needs to get the frame from camera and make detection on the boarder lines between the lane and make decision after calculating the slope value. Thus, lane detection script is done using Visual Studio Code (VS Code) code editor in Windows 10 laptop to make sure the code in script is working well.

This will not only increase the localization accuracy of the autonomous vehicle, but it will also improve the traditional autonomous car lane detection. Through this system, it believes that it can greatly reduce the numbers of the accidents. It can be concluded that the objectives set is achieved.

6.5 Concluding Remark

In shorts, it can be concluded that the lane detection script performs well when executed in Visual Studio Code (VS Code) code editors, and real-time location data of the robot car chassis can be obtained successfully. However, when the script runs on the robot car chassis, it lags and delivers poor results due to high CPU consumption on the Raspberry Pi 3B+. This issue results in an inability to display the latest camera frame on the Raspberry Pi 3B+.

Chapter 7 Conclusion and Recommendation

7.1 Conclusion

In conclusion, the aim of this final year project is to enhance the localization of autonomous vehicles with the use of OpenCV. To achieve this, a robot car chassis has been developed, which utilizes a Python-based OpenCV script to detect lanes and collect real-time location data from a GPS module. However, due to the limited CPU performance of the Raspberry Pi, the robot car chassis only used to record the lane in MP4 format. The recorded lane is then used as input for the script that runs on a Windows 10 laptop using Visual Studio Code (VS Code). The robot car chassis is also capable of retrieving location data from the GPS module. The project's goal is to explore innovative computer vision techniques to improve the safety and efficiency of autonomous vehicles' localization.

7.2 Recommendation

For the future recommendation, better hardware component such as Nvidia Jetson Nano can be used to replace the Raspberry Pi 3B+ as it has much more capable GPU and CPU than Raspberry Pi 3B+. Thus, the robot car chassis model can be run more smoothly during lane detection while retrieving the real time current location data. Higher output for DC motors should be replaced so that it can run easily as the robot car chassis model is quite heavy. Better design of camera holder is recommended so that it has a better view. It can be done by 3D printer to print out customized camera holder on the robot car chassis model after calculating the angle and size of the camera holder.

REFERENCES

- [1] How Does 360 Car Camera Work | Should You Install One? (2021). Accessed: August. 26, 2022

- [2] Qualitative results of restored images using our proposed CycleGAN architecture. (2021). Accessed: August. 26, 2022

- [3] R. Consumer, "Avoiding Crashes with Self-Driving Cars", Avoiding crashes with self-driving cars - consumer reports, 2014. [Online]. Available: <https://www.consumerreports.org/cro/magazine/2014/04/the-road-to-self-driving-cars/index.html>. [Accessed: 27- Aug- 2022].

- [4] L. Deng, M. YANG, h. Bing, T. Li, H. Li and C. Wang, "Semantic Segmentation-Based Lane-Level Localization Using Around View Monitoring System", Ieeexplore.ieee.org, 2022. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/8764443>. [Accessed: 27- Aug- 2022].

- [5] X. Lin, F. Wang, B. Yang, and W. Zhang, "Autonomous Vehicle Localization with Prior Visual Point Cloud Map Constraints in GNSS-Challenged Environments," Remote Sensing, vol. 13, no. 3, p. 506, Jan. 2021, doi: 10.3390/rs13030506. [Online]. Available: <http://dx.doi.org/10.3390/rs13030506>

- [6] M. Attia, Normal Distribution Transform with Point Projection for 3D Point Cloud Registration. 2017, p. 1. [Online]. Available: http://ipco-co.com/PET_Journal/vol_csp_2017/ID-87.pdf. [Accessed: 27- Aug- 2022].

- [7] R. Amadeo, "Google's Waymo invests in LIDAR technology, cuts costs by 90 percent", Ars Technica, 2022. [Online]. Available: <https://arstechnica.com/cars/2017/01/googles-waymo-invests-in-lidar-technology-cuts-costs-by-90-percent/>. [Accessed: 27- Aug- 2022].

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 2
Student Name & ID: LEE WOON SHIN 18ACB01136	
Supervisor: Ts Sun Teik Heng @ San Teik Heng	
Project Title: Localization for autonomous car using deep learning	

1. WORK DONE

Prepare a list of robot car chassis model components that needed to buy.
Build the robot car chassis model.

2. WORK TO BE DONE

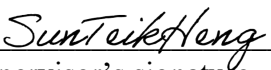
Learn how to write the GPS tracker script.
Learn how to write a colour filter with OpenCV.
Design the lane for robot car chassis to run.
Calculate each length value of lane.
Draw a lane for robot car chassis model.
Draw bigger lane for robot car chassis model.
Test run the lane with keyboard-controlled script.
Write a lane detection script and test run.
Fixing error that occurred.
Rewrite the script for better performance.

3. PROBLEMS ENCOUNTERED

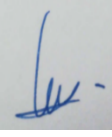
No problem faced.

4. SELF EVALUATION OF THE PROGRESS

Still ok.



 Supervisor's signature



 Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 4
Student Name & ID: LEE WOON SHIN 18ACB01136	
Supervisor: Ts Sun Teik Heng @ San Teik Heng	
Project Title: Localization for autonomous car using deep learning	

1. WORK DONE

Learn how to write the GPS tracker script.
Learn how to write a colour filter with OpenCV.

2. WORK TO BE DONE

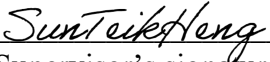
Design the lane for robot car chassis to run.
Calculate each length value of lane.
Draw a lane for robot car chassis model.
Draw bigger lane for robot car chassis model.
Test run the lane with keyboard-controlled script.
Write a lane detection script and test run.
Fixing error that occurred.
Rewrite the script for better performance.

3. PROBLEMS ENCOUNTERED

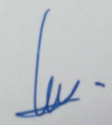
No error faced.

4. SELF EVALUATION OF THE PROGRESS

Ok.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 6
Student Name & ID: LEE WOON SHIN 18ACB01136	
Supervisor: Ts Sun Teik Heng @ San Teik Heng	
Project Title: Localization for autonomous car using deep learning	

1. WORK DONE

Design the lane for robot car chassis to run.
Calculate each length value of lane.

2. WORK TO BE DONE

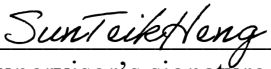
Draw a lane for robot car chassis model.
Draw bigger lane for robot car chassis model.
Test run the lane with keyboard-controlled script.
Write a lane detection script and test run.
Fixing error that occurred.
Rewrite the script for better performance.

3. PROBLEMS ENCOUNTERED

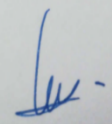
No error faced.

4. SELF EVALUATION OF THE PROGRESS

Ok.



 Supervisor's signature



 Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 8
Student Name & ID: LEE WOON SHIN 18ACB01136	
Supervisor: Ts Sun Teik Heng @ San Teik Heng	
Project Title: Localization for autonomous car using deep learning	

1. WORK DONE

Draw a lane for robot car chassis model.
 Draw bigger lane for robot car chassis model.
 Test run the lane with keyboard-controlled script.
 Write a lane detection script and test run.

2. WORK TO BE DONE

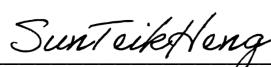
Fixing error if it occurred.

3. PROBLEMS ENCOUNTERED

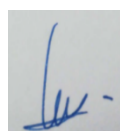
The performance of the lane detection is bad for most of the first version of scripts wrote.
 Keep finding out the reasons of bad performance happened.

4. SELF EVALUATION OF THE PROGRESS

Delayed, due to some bad performance of scripts tested.



 Supervisor's signature



 Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 10
Student Name & ID: LEE WOON SHIN 18ACB01136	
Supervisor: Ts Sun Teik Heng @ San Teik Heng	
Project Title: Localization for autonomous car using deep learning	

1. WORK DONE

Rewrote the scripts based on the error occurred and keep debugging.
 Redesign a new design of lane that in red boarder lines.
 Retest the lane detection scripts wrote.

2. WORK TO BE DONE

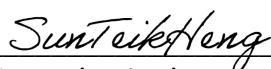
Fixing error if it occurred.

3. PROBLEMS ENCOUNTERED

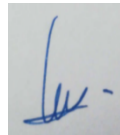
Still bad performance in lane detection.
 DC motors easily running out of power.

4. SELF EVALUATION OF THE PROGRESS

Slow.



 Supervisor's signature



 Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 12
Student Name & ID: LEE WOON SHIN 18ACB01136	
Supervisor: Ts Sun Teik Heng @ San Teik Heng	
Project Title: Localization for autonomous car using deep learning	

1. WORK DONE

Rewrote the scripts based on the error occurred and keep debugging.
 Retest the lane detection scripts wrote.
 Changed to 4 AA rechargeable batteries.
 Lane detection script is well-coded.

2. WORK TO BE DONE


Fixing error if it occurred.

3. PROBLEMS ENCOUNTERED

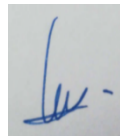
Found out that Raspberry Pi 3B+ can't run the lane detection script smoothly due to not enough of CPU it has.
 It able to response but the frame still freezes at the previous frame.

4. SELF EVALUATION OF THE PROGRESS

Slow, due to error keep occurred.



 Supervisor's signature



 Student's signature

POSTER



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

LOCALIZATION FOR AUTONOMOUS CAR USING DEEP LEARNING

Introduction

Autonomous vehicle has become important recently as it believe it is benefit to human as it can certify driver and passenger safety, save time and others. It also able to reduce the number of accidents as it able to avoid human error like distracted or drunk driver. Localization system able to tell user the current position of autonomous vehicle, destination that input by driver, and surroundings.

Objectives

- To increase accuracy of localization using OpenCV
- To detecting lane and retrieve location data

Proposed Methods

01

Build a robot car chassis model

AND

Write lane detection scripts using OpenCV

02

Attach a GPS module to Raspberry Pi

AND

Retrieve real time location data

Why proposed methods in this project is better than the existing methods?

- Cheaper in budget cost
- Lower amount of computation power needed
- Lower electricity power
- Faster in real time detection compared with Visual Point Cloud Map

Conclusion:

A robot car chassis model with OpenCV based lane detection and GPS module are the proposed solution to improve the localization system. It believes that with these methods proposed able to provide solution in increasing the localization system in autonomous vehicle.

Project Developer: Lee Woon Shin

Project Supervisor: Ts Sun Teik Heng @ San Teik Heng

PLAGIARISM CHECK RESULT

4/25/23, 5:10 PM

Turnitin - Originality Report - 18ACB01136_FYP2

Turnitin Originality Report Processed on: 25-Apr-2023 17:08 +08 ID: 2068332834 Word Count: 8947 Submitted: 5 18ACB01136_FYP2 By Woon Shin Lee	
Similarity Index 10%	Similarity by Source Internet Sources: 7% Publications: 5% Student Papers: N/A

1% match (Internet from 15-Dec-2022) http://eprints.utar.edu.my/4640/1/fyp_CS_2022_CGQ.pdf
1% match (Internet from 30-Mar-2023) http://eprints.utar.edu.my/4715/1/fyp_IA_2022_CWS.pdf
1% match (Liyuan Deng, Ming Yang, Bing Hu, Tianyi Li, Hao Li, Chunxiang Wang. "Semantic Segmentation-Based Lane-Level Localization Using Around View Monitoring System", IEEE Sensors Journal, 2019) Liyuan Deng, Ming Yang, Bing Hu, Tianyi Li, Hao Li, Chunxiang Wang. "Semantic Segmentation-Based Lane-Level Localization Using Around View Monitoring System", IEEE Sensors Journal, 2019
1% match (Xiaohu Lin, Fuhong Wang, Bisheng Yang, Wanwei Zhang. "Autonomous Vehicle Localization with Prior Visual Point Cloud Map Constraints in GNSS-Challenged Environments", Remote Sensing, 2021) Xiaohu Lin, Fuhong Wang, Bisheng Yang, Wanwei Zhang. "Autonomous Vehicle Localization with Prior Visual Point Cloud Map Constraints in GNSS-Challenged Environments", Remote Sensing, 2021
< 1% match (Internet from 15-Dec-2022) http://eprints.utar.edu.my/4632/1/fyp_CT_2022_YTL.pdf
< 1% match (Internet from 15-Dec-2022) http://eprints.utar.edu.my/4653/1/fyp_CS_2022_LCS.pdf
< 1% match (Internet from 15-Dec-2022) http://eprints.utar.edu.my/4629/1/fyp_CT_2022_SNX.pdf
< 1% match (Internet from 09-Oct-2022) http://eprints.utar.edu.my/2914/1/CN%2D2018%2DCKC%2D1503270.pdf
< 1% match (Internet from 30-Mar-2023) http://eprints.utar.edu.my/4725/1/fyp_IA_2022_YJW.pdf
< 1% match (Internet from 11-Oct-2022) http://eprints.utar.edu.my/4339/1/18ACB06319_FYP2.pdf
< 1% match (Internet from 28-Nov-2020) http://eprints.utar.edu.my/1952/1/CN-2016-1102867-1.pdf
< 1% match (Internet from 15-Dec-2022) http://eprints.utar.edu.my/4670/1/fyp_CS_2022_TXE.pdf
< 1% match (Internet from 08-Oct-2022) http://eprints.utar.edu.my/1550/1/18%2D2015%2D1102695%2D1.pdf
< 1% match (Internet from 03-Mar-2023) http://eprints.utar.edu.my/4731/1/fyp_IA_2022_YCH.pdf
< 1% match (Internet from 30-Mar-2023) http://eprints.utar.edu.my/4738/1/fyp_IB_2022_LZH.pdf
< 1% match (Internet from 09-Oct-2022) http://eprints.utar.edu.my/3451/1/fyp_IA_2019_TKW_1402934.pdf
< 1% match (Internet from 08-Oct-2022) http://eprints.utar.edu.my/1846/1/FYP_2_Full_Report.pdf
< 1% match (Internet from 30-Mar-2023) http://eprints.utar.edu.my/4743/1/fyp_IA_2022_TJT.pdf
< 1% match (Internet from 15-Dec-2022) http://eprints.utar.edu.my/4621/1/fyp_%2D_CN_%2D_LWJ_%2D_1702593_.pdf
< 1% match (Internet from 12-Jun-2022) http://eprints.utar.edu.my/1551/1/CN-2015-1202013-1.pdf
< 1% match (Internet from 30-Mar-2023) http://eprints.utar.edu.my/4720/1/fyp_IA_2022_LJ.pdf
< 1% match (Internet from 10-Nov-2021) https://fict.utar.edu.my/documents/FYP/FYP2_template/FYP2_Report_Template_CS.docx

https://www.turnitin.com/newreport_printview.asp?eq=0&eb=0&esm=0&oid=2068332834&sid=0&n=0&m=2&svr=25&r=30.159803350172343&lang=en... 1/8

PLAGIARISM CHECK RESULT

4/25/23, 5:10 PM

Turnitin - Originality Report - 18ACB01136_FYP2

< 1% match (Internet from 10-Nov-2021) https://fict.utar.edu.my/documents/FYP/IIPSPW_template/IIPSPW_Report_Template_IA.docx
< 1% match (Internet from 06-Nov-2022) https://www.codingdict.com/sources/py/cv2/4425.html
< 1% match ("16th International Conference on Information Technology-New Generations (ITNG 2019)", Springer Science and Business Media LLC, 2019) "16th International Conference on Information Technology-New Generations (ITNG 2019)", Springer Science and Business Media LLC, 2019
< 1% match ("Communication Software and Networks", Springer Science and Business Media LLC, 2021) "Communication Software and Networks", Springer Science and Business Media LLC, 2021
< 1% match (Internet from 24-Jan-2023) https://www.mdpi.com/2072-4292/13/6/1111/htm
< 1% match (Internet from 01-Mar-2023) https://www.coursehero.com/file/127882413/compv7py/
< 1% match (Internet from 06-Jan-2023) https://python.hotexamples.com/examples/cv2/-/fitUne/python-fitline-function-examples.html
< 1% match (Internet from 06-Jan-2023) https://python.hotexamples.com/de/examples/cv2/-/approxPolyDP/python-approxpolydp-function-examples.html
< 1% match (M. Nivedita, Priyanka Chandrashekar, Shibani Mahapatra, Y. Asnath Vicky Phamila, Sathish Kumar Selvaperumal. "Image Captioning for Video Surveillance System using Neural Networks", International Journal of Image and Graphics, 2021) M. Nivedita, Priyanka Chandrashekar, Shibani Mahapatra, Y. Asnath Vicky Phamila, Sathish Kumar Selvaperumal. "Image Captioning for Video Surveillance System using Neural Networks", International Journal of Image and Graphics, 2021
< 1% match (Yahya Mohammed Al-Naggar, Norlida Jamil, Mohd Firdaus Hassan, Ahmad Razlan Yusoff. "Condition monitoring based on IoT for predictive maintenance of CNC machines", Procedia CIRP, 2021) Yahya Mohammed Al-Naggar, Norlida Jamil, Mohd Firdaus Hassan, Ahmad Razlan Yusoff. "Condition monitoring based on IoT for predictive maintenance of CNC machines", Procedia CIRP, 2021
< 1% match (Internet from 12-Jan-2023) https://www.programcreek.com/python/example/70452/cv2.DIST_L2
< 1% match (Yang Xing, Chen Lv, Dongpu Cao. "Road Perception in Driver Intention Inference System", Elsevier BV, 2020) Yang Xing, Chen Lv, Dongpu Cao. "Road Perception in Driver Intention Inference System", Elsevier BV, 2020
< 1% match (Internet from 05-Apr-2023) https://github.com/lopepardo/InvisibilityCloak/blob/master/README.md
< 1% match (Sufyan bin Uzayr. "Optimizing Visual Studio Code for Python Development", Springer Science and Business Media LLC, 2021) Sufyan bin Uzayr. "Optimizing Visual Studio Code for Python Development", Springer Science and Business Media LLC, 2021
< 1% match (Internet from 10-Sep-2017) https://www.cytron.com.my/featured
< 1% match (Internet from 25-May-2022) https://forums.developer.nvidia.com/t/green-screen-when-using-raspberry-pi-camera-v2-attached-to-jetson-nano-and-cv2-opencv/173596
< 1% match (Internet from 24-May-2019) http://circuitdesolator.blogspot.com/2018/12/test.html
< 1% match (Internet from 19-Apr-2023) https://dspace.aiub.edu/jspui/bitstream/123456789/868/1/2022.2.31%20Book.pdf
< 1% match (Internet from 03-Oct-2022) https://monarch.qucosa.de/api/qucosa%3A20751/attachment/ATT-0/
< 1% match () Jun Wang, Moudao Li, Weibin Jiang, Yanwei Huang, Ruiquan Lin. "A Design of FPGA-Based Neural Network PID Controller for Motion Control System", Sensors (Basel, Switzerland)
< 1% match (Arvind Mukundan, Yu-Ming Tsao, Fen-Chi Lin, Hsiang-Chen Wang. "Portable and Low-cost Hologram Verification Module That Uses Hyperspectral Imaging", Research Square Platform LLC, 2022) Arvind Mukundan, Yu-Ming Tsao, Fen-Chi Lin, Hsiang-Chen Wang. "Portable and Low-cost Hologram Verification Module That Uses Hyperspectral Imaging", Research Square Platform LLC, 2022
< 1% match (Thomas Bräunl. "Embedded Robotics", Springer Science and Business Media LLC, 2022) Thomas Bräunl. "Embedded Robotics", Springer Science and Business Media LLC, 2022
< 1% match (Internet from 24-Nov-2020) https://code.visualstudio.com/docs/python/environments
< 1% match (Internet from 01-Nov-2022) https://dokumen.pub/security-and-safety-interplay-of-intelligent-software-systems-esorics-2018-international-workshops-issa-2018-and-csits-2018-barcelona-spain-september-67-2018-revised-selected-papers-1st-ed-978-3-030-16873-3-030-16874-2.html

https://www.turnitin.com/newreport_printview.asp?eq=0&eb=0&esm=0&oid=2068332834&sid=0&n=0&m=2&svr=25&r=30.159803350172343&lang=en... 2/8

PLAGIARISM CHECK RESULT

4/25/23, 5:10 PM

Turnitin - Originality Report - 18ACB01136_FYP2

< 1% match (Internet from 06-Feb-2023) https://patents.google.com/patent/US9534910B2/en
< 1% match (Internet from 27-Sep-2022) https://vtechworks.lib.vt.edu/bitstream/handle/10919/87582/Doyle_JE_T_2019.pdf?isAllowed=y&sequence=1
< 1% match (Internet from 03-Feb-2023) https://www.researchgate.net/publication/341002916_Hyperspectral_Classification_of_Cyperus_esculentus_Clones_and_Morphologically_Simila
< 1% match (Internet from 30-May-2019) https://www.tweaking4all.com/hardware/pir-sensor/
< 1% match ("Internet of Things and Its Applications", Springer Science and Business Media LLC, 2022) "Internet of Things and Its Applications", Springer Science and Business Media LLC, 2022
< 1% match (Uzma Maroof, Arash Shaghaghi, Sanjay Jha. "PLAR", Proceedings of the 2nd International ACM Workshop on Security and Privacy for the Internet-of-Things, 2019) Uzma Maroof, Arash Shaghaghi, Sanjay Jha. "PLAR". Proceedings of the 2nd International ACM Workshop on Security and Privacy for the Internet-of-Things, 2019

PLAGIARISM CHECK RESULT

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	LEE WOON SHIN
ID Number(s)	18ACB01136
Programme / Course	IA
Title of Final Year Project	Localization for autonomous car using deep learning

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>10</u> % Similarity by source Internet Sources: <u>7</u> % Publications: <u>5</u> % Student Papers: <u>N/A</u> %	
Number of individual sources listed of more than 3% similarity: <u>0</u>	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Sun Teik Heng
Signature of Supervisor

Signature of Co-Supervisor

Name: Ts Sun Teik Heng @ San Teik Heng

Name: _____

Date: 25 APRIL 2023

Date: _____



UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	18ACB01136
Student Name	LEE WOON SHIN
Supervisor Name	Ts Sun Teik Heng @ San Teik Heng

TICK (✓)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
	Front Plastic Cover (for hardcopy)
✓	Title Page
✓	Signed Report Status Declaration Form
✓	Signed FYP Thesis Submission Form
✓	Signed form of the Declaration of Originality
✓	Acknowledgement
✓	Abstract
✓	Table of Contents
✓	List of Figures (if applicable)
✓	List of Tables (if applicable)
	List of Symbols (if applicable)
✓	List of Abbreviations (if applicable)
✓	Chapters / Content
✓	Bibliography (or References)
✓	All references in bibliography are cited in the thesis, especially in the chapter of literature review
✓	Appendices (if applicable)
✓	Weekly Log
✓	Poster
✓	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
✓	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

(Signature of Student)

Date: 25 April 2023