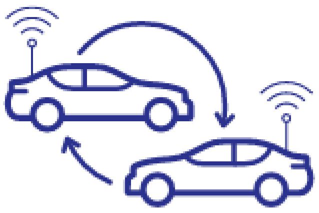
Faculty of Engineering

Computer Engineering Department

Helwan University

VEHICLE TO VEHICLE COMMUNICATION

Graduation Project Documentation



July 2022

Prepared by:

Omar Mohamed Mohamed

Nariman Mohamed Alaa

Rahma Mohamed Makram

Radwa Magdi Hassan

Abdel-Aziz Mohamed Abdel-Aziz

Under Supervision of:

Prof. Dr. El-Sayed Moustafa

ACKNOWLEDGMENT

All praises to Allah and His blessing for the completion of this project. We thank God for all the opportunities, trials and strength that have been showered on us to finish this project.

First and foremost, we would like to sincerely thank our supervisor **Prof. Dr. El-Sayed Moustafa** for his guidance, understanding, patience and most importantly, he has provided positive encouragement and a warm spirit to finish this project. It has been a great pleasure and honor to have him as our supervisor.

We are also grateful to all the Teaching staff in the Computer Engineering Department, for their consistent support and assistance. It was great sharing premises with all of you during the last five years.

ABSTRACT

The majority of accidents occur because the driver can only see as far as the cars directly in front of him/her, behind him/her, or on either side with the sensors and present electronic driver aids. A skilled driver may observe more than one automobile ahead or behind him, as well as the signal lights, and take preventative measures to avoid any unexpected actions or accidents. However, this isn't always enough. Any action done faster than the driver's response time, such as a vehicle approaching at a rapid speed next to him/her or discovering there's a large obstacle when the automobile is too close to take the necessary measures, may result in severe consequences. As a response, there must be another option in which the automobile detects the unexpected changes and warn the driver. There must also be a method to allow the driver to see more than two cars ahead or behind him/her and to inform the driver of changes that occur a bit further beyond his/her sight so that the driver may behave smoothly and predictably.

Vehicle-to-vehicle communication is one technical advancement that might solve this problem. This document will provide further information regarding V2V communication, its benefits, and its current market nowadays. Then comes the project's real implementation.

Table of Contents

CHAPTER 1: INTRODUCTION....5

1.1 Problem Statement....5
1.2 Project Objective....6
1.3 Motivation....7
1.4 Project Description (The Proposed Solution)....7
1.5 System Block Diagram....10

CHAPTER 2: HISTORY OF V2V COMMUNICATION....11

2.1 Literature Review....11
2.2 V2V Market....13

CHAPTER 3: MAIN IMPLEMENTATION TOOLS....14

3.1 Raspberry Pi....14
3.2 Python Programming Language....17
3.3 PyQt5 GUI Library....17

CHAPTER 4: CAR LOCALIZATION....19

4.1 Global Positioning System (GPS)....19
4.2 Camera Module....26
4.3 Speed Sensor Module....27
4.4 Lane Number Detection....28

CHAPTER 5: COMMUNICATION AND NETWORKING....30

5.1 Socket Programming in Python....30
5.2 Sockets in V2V Communication System....37

CHAPTER 6: APPLICATION LAYER....42

6.1 Application1: Emergency Hard Brake Detection....42
6.1.1 Description....42
6.1.2 Implementation....43
6.1.3 User Interface....47
6.2 Application2: Follow Me Information....49
6.2.1 Description....49
6.2.2 Implementation....50
6.2.3 User Interface....56
6.3 Application 3: Accident Detection Warning....59
6.3.1 Description....59
6.3.2 Implementation....59
6.3.3 User Interface....63

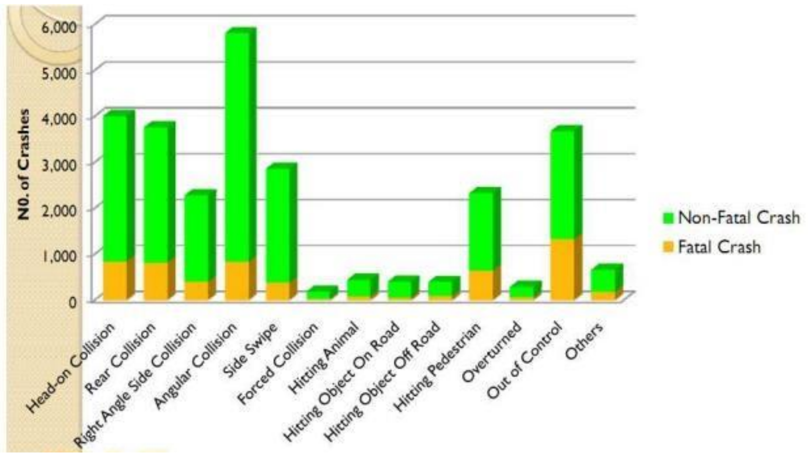
References....64

CHAPTER 1: INTRODUCTION

1.1 Problem Statement

Due to the rising yearly number of accidents and traffic congestion throughout the world, which has a negative impact on the economy. there is a pressing need to find a solution. Instead of seeking a solution, individuals begin to cause disorder on the road, resulting in traffic jams. During peak hours, more than 60% of the population is stuck on the road owing to excessive traffic. This tremendous congestion, in turn, has an impact on everyone's life. Aside from traffic congestion, there is an issue when people choose the incorrect way, resulting in a delay, or when a road is blocked. So, according to studies, these scenarios can be avoided if drivers are given a half-second notice so that they can take an alternate route to avoid traffic or road congestion.

The bar chart below shows the collision type that occur and the numbers of non-fatal and fatal crash resulting from the collision:



With the increasing of number of crashes every year, the number of fatal crashes also increase. When the number of crashes is higher, the possibility for fatal crashes to occur will be higher. Unfortunately, this problem will continue to increase because number of the increasing number of vehicles on the road.

So, according to the World Health Organization, 60% of accidents might be avoided if the driver was informed at least 0.5 to 1.5 seconds before the crash, and that could be done by using V2V communications. With this technology, vehicles can communicate with each other and could send along information about the vehicle situation and road conditions which help the driver to make the right decision in the right time.

1.2 Project Objective

Objective of Vehicle To Vehicle Communication System project is to avoid the roadway collisions and to drivers prior to the collision. Accidents are taking thousands of lives every day so with intent to reduce the adverse effects we implemented this project based on vehicle to vehicle communications systems, with the rapid advancement of wireless communication systems had paved path for the vehicle to vehicle wireless communication based warning systems.

We also wanted our project to fulfill the user's needs and expectations and satisfy…

- Flexibility and compatibility that it can implemented on any car type or model.
- Cheap and easy user interface that if we want to update the system anytime it will be so easy because it is programmable so it doesn't need any hardware exchange.
- Guaranteeing safety to people on the road.
- Regulation of cars motion so that no car cannot detect the other cars motion information in its pre-specified range.

1.3 Motivation

Our Thought During Idea Searching Phase

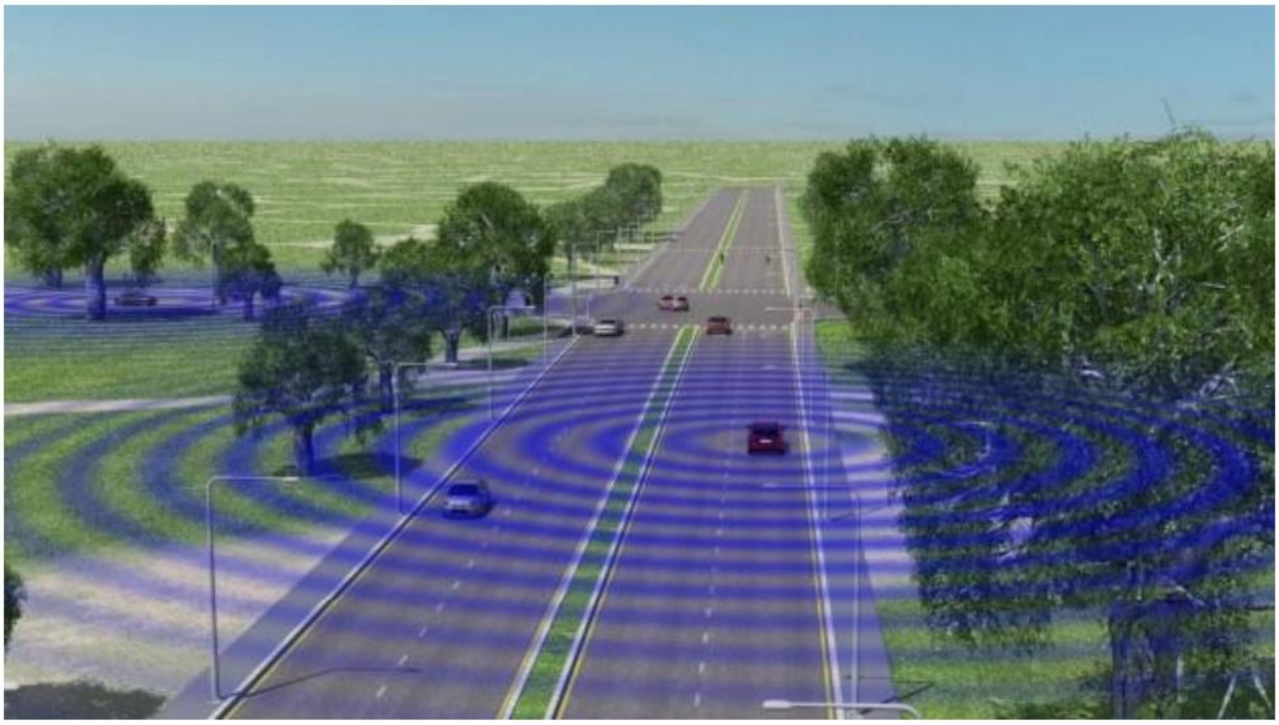
We thought about a project that meets our jobs requirements and meets our team's intended learning objective through the graduation project and we chose an idea which assembles different skills in our team. In addition, this project will help in solving economic problems and tragic problems caused by road accidents, it will provide safety to the drivers and it will improve the traffic management.

1.4 Project Description (The Proposed Solution)

The great ability of the machine on Focusing and not be distracted made us use this as a system, this system will provide a great help to reduce road accidents and congestion by making communication between vehicles to collect surrounding vehicles' motion information this information is one of the keys for accident prevention by helping the driver to make the right decision in the right time.

This is what Vehicle to Vehicle communication systems do, V2V communications are going to create new services by transmitting packets from vehicle to vehicle without the use of any deployed infrastructures. These services will enable the vehicle to transmit necessary data such as the current location, the motion's direction, and the speed directly to other vehicles. These vehicles would form a network, and pass information about road conditions, accidents, and congestion. A driver could be made aware of the emergency braking of a preceding vehicle or the presence of an obstacle in the roadway. It can also help vehicles negotiate critical points like blind crossings at the intersections or entries to highways, then vehicles can react to change driving situations and then instantly warn the drivers with emergency warning messages.

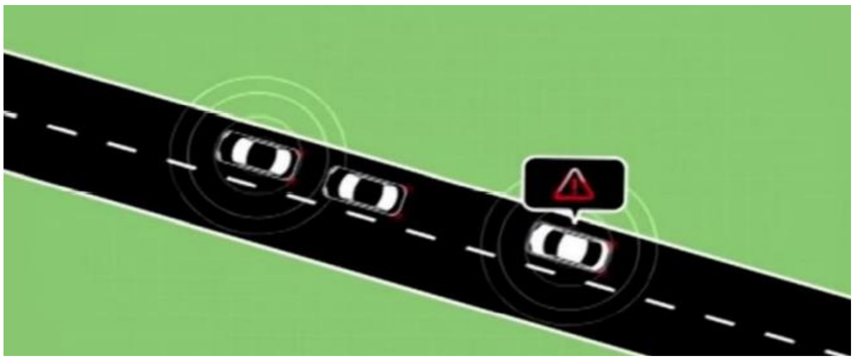
Using vehicle-to-vehicle (V2V) communication, a vehicle can detect the position and movement of other vehicles up to a quarter of a kilometer away. In area world where vehicles are equipped with a computer chip, GPS (Global Positioning System) Technology and some other sensors to detect vehicle information, your car will know where the other vehicles are, additionally other vehicles will Know where you are too whether it is in blind spots, stopped ahead on the highway but hidden from view, around a blind corner or blocked by other vehicles. The vehicles can anticipate and react to changing driving situations and then instantly warn the drivers with emergency warning messages.



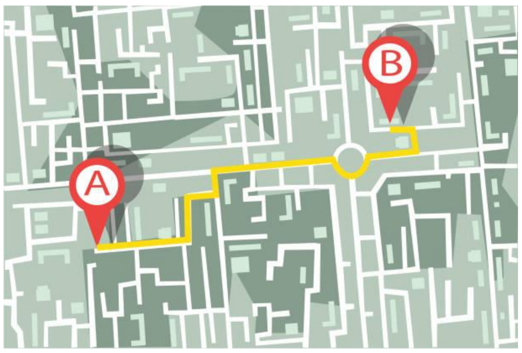
So, our project elaborates Implementation of a real time vehicle to vehicle communication system over a local WIFI network between cars where we broadcast car data and state to avoid potential collisions. Specifically handling 3 main applications:

1- <u>Emergency Hard Brake Detection</u>

Hard brake detection application enables a vehicle to broadcast a self-generated emergency brake event to surrounding vehicles. So, the vehicles in the same lane can avoid a sudden collision.



2- <u>Follow Me Information</u>



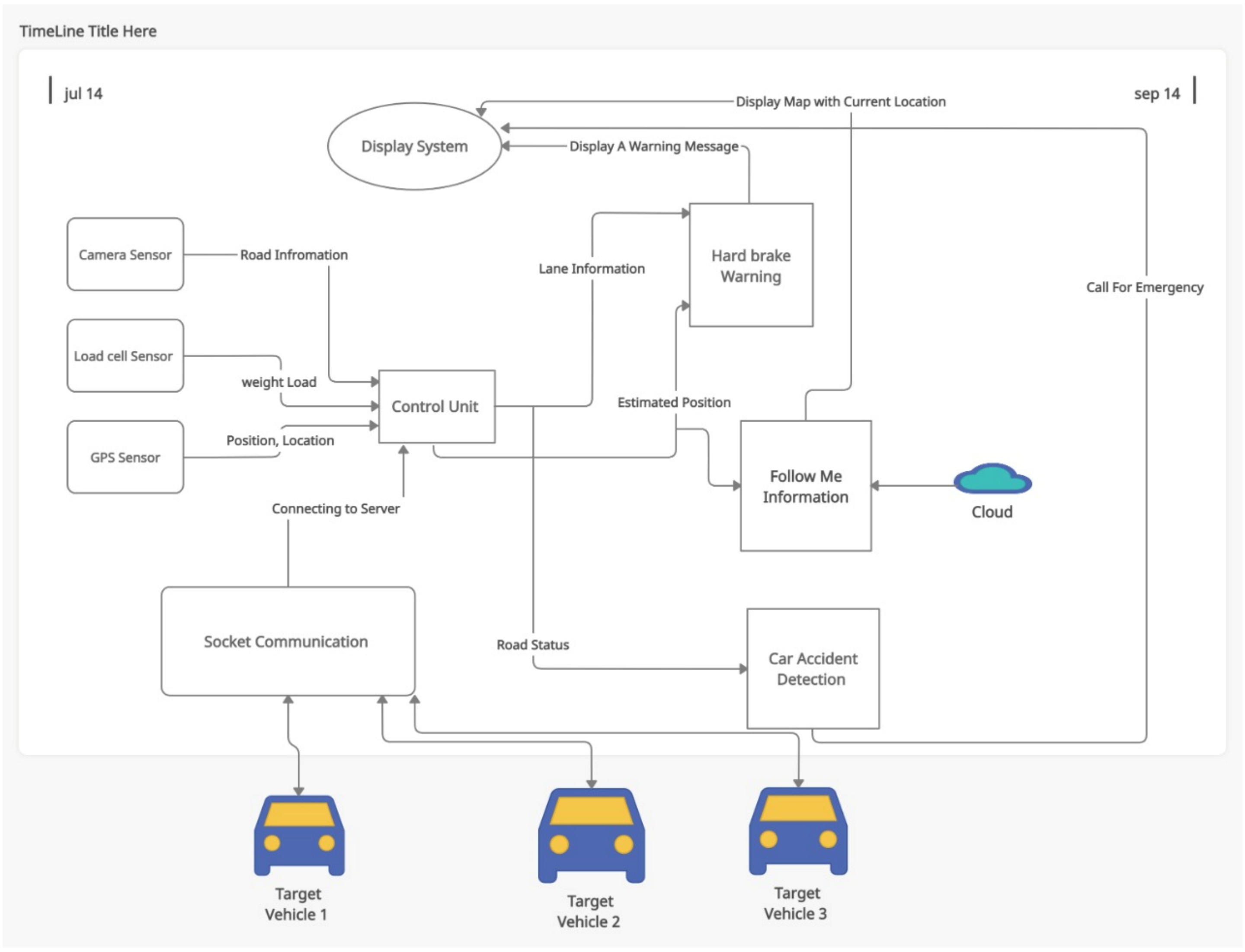
Follow-me information is a convenience feature for people travelling in groups of vehicles.
The received information can be used to plan or optimize a route.
The following vehicle and the follower vehicle locations will appear on the map for the following vehicle so it can take instructions from map to follow the follower vehicle.

<u>3- Accident Detection Warning</u>

Accident detection warning application will help in reducing the traffic jams by notifying the drivers by the location of an accident so that they can take another road to go wherever they want without time loss by getting blocked in accident road.

We will talk later about each application briefly in APPLICATION LAYER chapter…

1.5 System Block Diagram



CHAPTER 2: HISTORY OF V2V COMMUNICATION

2.1 Literature Review

Summary of what have been done in this area in the last 10 years, V2V communications research initially began under the Vehicle Infrastructure Integration Initiative in 2003, but its origins date back to the Automated Highway System (AHS) research of the 1990s, to show the evolution of v2v communication we choose some papers from different years, we will summarize them with their topics chronologically.

Lane Change Warning System based on V2V communication in 2014 [1]

This paper proposes a lane change warning system based on V2V communication which can be used for real road driving. Simulation model is built to test the performance of the system. Two meaningful conclusions are got as below:

1. The lane change warning system can coordinate both lane change safety and driving comfort to realize effective warning. The lane change warning system can make a comprehensive analysis of the collision risk between the changing-lane vehicle and its four surrounding vehicles. Furthermore, a driving style index is introduced to modify the warning timing to satisfy various drivers.

2. The analysis method of the minimum safe distance can precisely estimate the initial distance needed to safely complete the lane change process. To make the minimum safe distance more practical and specific, minimum safe distance of rear vehicle in target lane is analyzed according to the goal of both collision avoidance and vehicle following safety, the counterpart of front vehicle in target lane is analyzed to avoid emergency collision and counterparts of vehicles in original lane are analyzed to ensure vehicle following safety.

Other authors in 2016, proposed to substitute the currently proposed standard (802.11p, DSRC/WAVE) to build an inter-vehicle communication system. They suggested using android smartphones along with Wi-Fi to replace the OBU (On-Board Unit) on consumer vehicles and to enable thereby V2I and V2V communications. The system is compatible with android devices with the 2.3 version and above. The users will then acquire an OBU that communicates with the nearby vehicles to broadcast an emergency alert or to warn about an accident. Experiment results showed the ease of use of the system, its efficiency, and its compatibility with Android devices.

Taking Advantage of V2V Communications for Traffic Management, in 2011 [2]
a paper was published with reference to a scenario where vehicles are equipped with devices collecting measurements and sending them to a remote control center through cellular networks, in this paper we discussed the advantage that vehicle to vehicle communications could provide. Numerical results, obtained through a simple analytical model, highlighted the significant reduction in data transmissions over the cellular networks that can be obtained even with a reduced number of equipped vehicles.

Demonstration of Forward Collision Avoidance Algorithm Based on V2V Communication a paper published in 2019 [3]
The system model is based on three scenarios including the duality between the autonomous mode system and the warning system for the driver mode. The basic scenario contains two vehicles which are Leading Vehicle (LV) and Following Vehicle (FV). Every vehicle is provided by a Global Positioning-System (GPS) device and Radio Frequency (RF). Antenna System is used to transfer positioning data between LV and FV and is used in couple of autonomous and warning system that uses Basic Safety Message (BSM)

Vehicle To Vehicle "V2V" Communication: Scope, Importance, Challenges, Research Directions and Future (article published 2020) [4]

The communication will be through the WIFI communication between nodes to simulate the real-life events. WIFI and GPS based vehicle location and tracking system will provide effective, real time vehicle location, mapping and reporting this information value and add by improving the level of service provided. A GPS-based vehicle tracking system will inform where your vehicle is and where it has been, how long it has been. The system uses geographic position and time information from the Global Positioning Satellites. The system has an "On-Board Module" which resides in the vehicle to be tracked.

2.2 V2V Market

V2V communication 's market is growing every year due to the enhancement of technology use in vehicles. A lot of investment is done in this field nowadays. Middle Eastern countries are considered a great potential for this technology due to the increase in population as well as the focus of many automobile companies on regions such as the Middle East and Africa. The development of this technology by automotive manufacturers, chip manufacturers as well as technology and solution providers are accelerating.

There are many companies interested in this technology such as BMW, Audi, Daimler, Volvo, and Ford-Applink. Among the solution providers Etrans Systems, Qualcomm Technologies Inc., Cisco Systems Inc., Delphi Automotive PLC, Autotalks Ltd., Denso, Arada Systems, Kapsch Group and Savari Inc., are included in the vehicle to vehicle communication market.

CHAPTER 3: MAIN IMPLEMENTATION TOOLS

3.1 Raspberry Pi

Python programming language is chosen to be the programming language for this project, so that The Raspberry Pi is chosen to be the main microcontroller for this project which is more compatible with Python.

The Raspberry Pi is a Low-cost high-performance, credit-card sized computer which has all its components, ports, and features out on display, it plugs into a computer monitor or TV and uses a standard keyboard and mouse. It is a capable little device that makes programming and computing easier. It can do everything you would expect, also, the Raspberry Pi has the ability to interact with the outside world and has been used in a wide array of digital maker projects.

Raspberry Pi OS:

It is a pocket PC size computer that enable to use an external operating system. The Raspberry Pi does not have a pre-installed operating system in which the user can start programming on, it needs a micro SD card inserted in it where the operating system is installed on it. The Raspberry Pi uses a lightweight operating system known as the Raspbian OS, it is a Linux operating system that comes with different applications and integrated development environment (IDE) pre-installed on it for programming which was used to write all the algorithms and hardware programs to control the pins.

- Raspberry Pi has so many versions, we chose to work with **Raspberry Pi 4 model B** because it is highly economical and has exceeds the previous versions in terms of speed and performance. It is the most powerful yet, getting a major increase in processing power, enhanced video output and peripheral connectivity, while maintaining the same low price and tiny size offered on past models.

Raspberry Pi 4 Model B Specifications and Features:

• **Processor:** Broadcom BCM2711, quad-core Cortex-A72 (ARM v8)
64-bit SoC @ 1.5GHz

It considered the brain of this device. It features everything needed to process input and store information. This CPU executes the number of instructions consisting of a computer program.

• **Memory:** 4GB LPDDR4 with on-die ECC

The RAM temporarily stores the information and with the removal of the power supply from the module, this memory is also wiped off.

• **Connectivity:** 2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac
wireless LAN, Bluetooth 5.0, BLE
Gigabit Ethernet
2 × USB 3.0 ports
2 × USB 2.0 ports.

• **GPIO:** Standard 40-pin GPIO header

External labels (from GPIO2 to GPIO27) come with the Broadcom (BCM) naming convention. This convention is useful when you are going to program with Python libraries.

• **Video & sound:** 2 × micro HDMI ports (up to 4Kp60 supported)

2-lane MIPI DSI display port

2-lane MIPI CSI camera port

4-pole stereo audio and composite video port

These ports support all of the available Raspberry Pi camera and display peripherals.

• **Multimedia:** H.265 (4Kp60 decode);

H.264 (1080p60 decode, 1080p30 encode);

OpenGL ES, 3.0 graphics

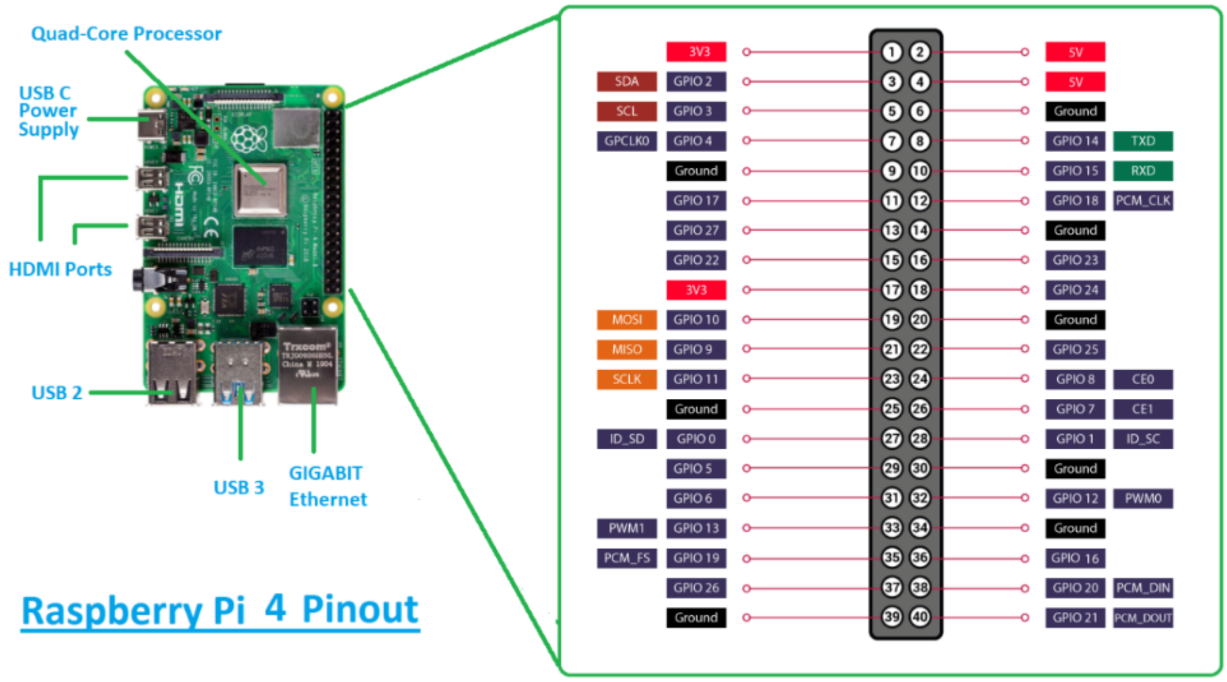
• **SD card support:** Micro SD card slot for loading operating system and data storage

• **Input power:** 5V DC via USB-C connector (minimum 3A1)

5V DC via GPIO header (minimum 3A1)

Power over Ethernet (PoE)–enabled

• **Environment:** Operating temperature 0–50ºC


Quad-Core Processor
USB C Power Supply
HDMI Ports
USB 2
USB 3
GIGABIT Ethernet
3V3 1 2 5V
SDA GPIO 2 3 4 5V
SCL GPIO 3 5 6 Ground
GPCLK0 GPIO 4 7 8 GPIO 14 TXD
Ground 9 10 GPIO 15 RXD
GPIO 17 11 12 GPIO 18 PCM_CLK
GPIO 27 13 14 Ground
GPIO 22 15 16 GPIO 23
3V3 17 18 GPIO 24
MOSI GPIO 10 19 20 Ground
MISO GPIO 9 21 22 GPIO 25
SCLK GPIO 11 23 24 GPIO 8 CE0
Ground 25 26 GPIO 7 CE1
ID_SD GPIO 0 27 28 GPIO 1 ID_SC
GPIO 5 29 30 Ground
GPIO 6 31 32 GPIO 12 PWM0
PWM1 GPIO 13 33 34 Ground
PCM_FS GPIO 19 35 36 GPIO 16
GPIO 26 37 38 GPIO 20 PCM_DIN
Ground 39 40 GPIO 21 PCM_DOUT

Raspberry Pi 4 Pinout

3.2 Python Programming Language

Python is an interpreted high-level general-purpose programming language. Python’s design philosophy emphasizes code readability with its notable use of significant indentation.

We choose Python to be the main programming language for our project, as it is simple and provided with many libraries that help us to write our code in an easy way, also it is compatible with the Raspberry Pi which is the main controller for our project.

3.3 PyQt5 GUI Library

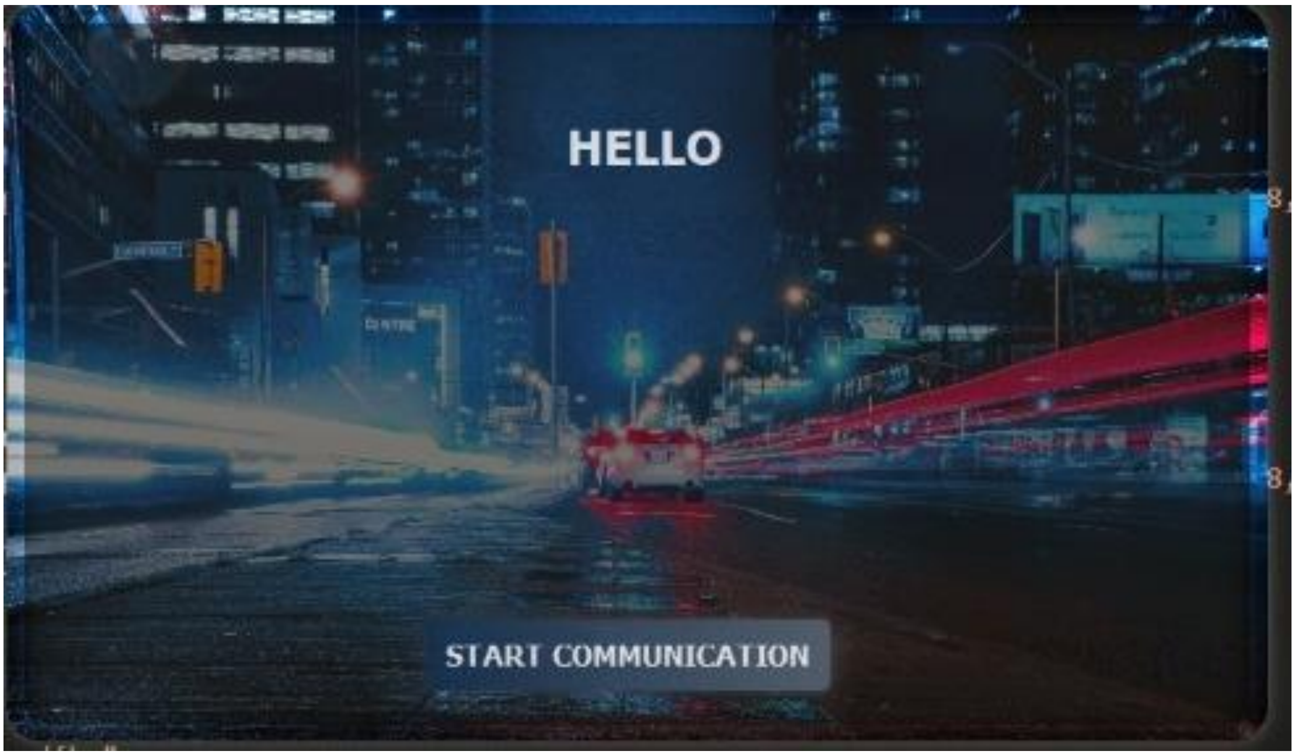
A GUI was needed for the users to be able to interface with our V2V communication system, so we created a simple GUI using PyQt5 library to make it easy to use our system.

PyQt is a toolkit for graphical user interface (GUI) widgets, it comes with many powerful and advanced widgets. It is extracted from the library of Qt. PyQt is the product of the combination of the Python language and the Qt library. PyQt is coming with the Qt Builder. We will use it to get the python code from the Qt Creator. With the support of a Qt designer, we can build a GUI, QT Designer makes designing and creating GUIs much easier. GUIs can easily be designed with QT designer, and their logic can be programmed using PyQT – this makes development faster and easier and gives PyQt an edge, and then we can get python code for that GUI.

PyQt supports all platforms including Windows, macOS and UNIX. PyQt can be used to create stylish GUIs, a modern and portable python framework.

PyQT gives you widgets to create complex GUIs. In fact, everything in PyQT (buttons, input fields, checkboxes, and Radio boxes) is a widget, so, It is widely used for creating large-scale GUI-based programs. It gives programmers the freedom to create GUIs of their choice.

And here is the starting window of our system that from it our vehicle can connect to the server and be in contact with all other vehicles that having the same system:



CHAPTER 4: CAR LOCALIZATION

Localization is how a car figures out where it is in the world. Determining the vehicle's precise position on the map is called "localization". By localizing itself, the vehicle can determine its precise relationship to all of the elements of the environment. Is the vehicle in the middle lane or the right lane? Where is the nearest car? And more questions can be answered through localization of the Car.
In our project we used different modules to achieve the objective of localization before sending the required data over communication protocol.

In this chapter we will discuss the modules we used in localization:

We used GPS to locate the car in the global map relative to latitude and longitude but the GPS accuracy wasn't enough for car localization, so in order to increase the accuracy of localization we used a camera module to capture the area around the car and from it we can classify and understand the area around it, and from the captured video from the camera we can know in which lane the car exactly in by using the lane number detection model, and we also use a speed sensor to get car data and use it in our algorithm.

4.1 Global Positioning System (GPS)

GPS is a satellite-based radio navigation system owned by the United States government and operated by the United States Space Force. It is one of the global navigation satellite systems (GNSS) that provides geolocation and time information to a GPS receiver anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites.

There are several different methods for obtaining a position using GPS. The method used depends on the accuracy required by the user and the type of GPS receiver available. The techniques can be broken down into three basic classes:

Autonomous Navigation:

Using a single stand-alone receiver. Position Accuracy is better than 100m for civilian users and about 20m for military users.

Differential Phase position:

Gives an accuracy of 0.5-20mm. Used for many surveying tasks, machine control etc.

Differentially corrected positioning:

More commonly known as DGPS, this gives an accuracy of between 0.5-5m. Used for inshore marine navigation, GIS data acquisition, precision farming.

In the project we will use the first class,

Autonomous Navigation…

This is the simplest technique employed by GPS receivers to instantaneously give a position and height and/ or accurate time to a user. The accuracy obtained is better than 100m (usually around the 30-50m mark) for civilian users and 5-15m for military users. Receivers used for this type of operation are typically small, highly portable handheld units with a low cost.

The GPS concept is based on time and the known position of GPS specialized satellites. The satellites carry very stable atomic clocks that are synchronized with one another and with the ground clocks. Any drift from true time maintained on the ground is corrected daily. In the same manner, the satellite locations are known with great precision. GPS receivers have clocks as well, but they are less stable and less precise.

Each GPS satellite continuously transmits a radio signal containing the current time and data about its position. Since the speed of radio waves is constant and independent of the satellite speed, the time delay between when the satellite transmits a signal and the receiver receives it is proportional to the distance from the satellite to the receiver. As presented in one of Isaac Newton's laws of motion is: **Distance = Velocity x Time**

A GPS receiver monitors multiple satellites and solves equations to determine the precise position of the receiver and its deviation from true time. At a minimum, four satellites must be in view of the receiver for it to compute four unknown quantities (three position coordinates (x, y, z) and clock deviation from satellite time).

The receiver location is expressed in a specific coordinate system, such as latitude and longitude using the WGS 84(World Geodetic System) geodetic datum or a country-specific system.

By getting the value of latitude, longitude and altitude we can define any point on the earth with these coordinates.

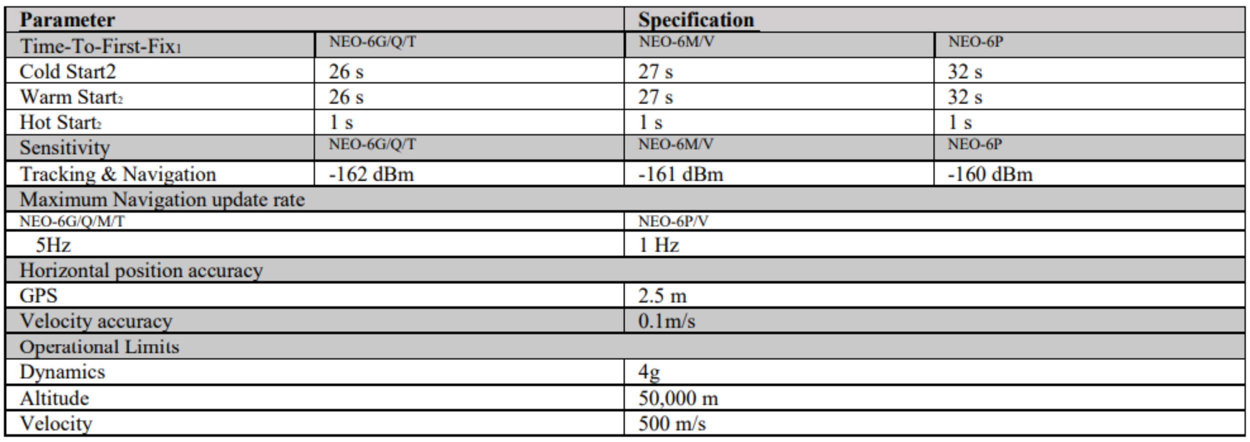
NEO-6 u-blox 6 GPS Modules…

The NEO-6 module series is a family of stand-alone GPS receivers featuring the high-performance u-blox 6 positioning engine. These flexible and cost-effective receivers offer numerous connectivity options in a miniature 16 x 12.2 x 2.4 mm package.

It can track up to 22 satellites over 50 channels and achieve the industry's highest level of tracking sensitivity it can perform 5 location updates in a second with 2.5m horizontal position accuracy. The U-blox 6 positioning engine also has a Time-To-First-Fix (TTFF) of less than 1 second.

➢ GPS performance

This table illustrates performance measures for all The NEO-6 module series but will focus on NEO-6 M which will be used in our project.



Parameter		Specification	
Time-To-First-Fix[1]	NEO-6G/Q/T	NEO-6M/V	NEO-6P
Cold Start2	26 s	27 s	32 s
Warm Start[2]	26 s	27 s	32 s
Hot Start[2]	1 s	1 s	1 s
Sensitivity	NEO-6G/Q/T	NEO-6M/V	NEO-6P
Tracking & Navigation	-162 dBm	-161 dBm	-160 dBm
Maximum Navigation update rate			
NEO-6G/Q/M/T		NEO-6P/V	
5Hz		1 Hz	
Horizontal position accuracy			
GPS		2.5 m	
Velocity accuracy		0.1m/s	
Operational Limits			
Dynamics		4g	
Altitude		50,000 m	
Velocity		500 m/s	

➢ Power Management

u-blox receivers support different power modes. These modes represent strategies of how to control the acquisition and tracking engines in order to achieve either the best possible performance or good performance with reduced power consumption.

Maximum Performance Mode

During a Cold start, a receiver in Maximum Performance Mode continuously deploys the acquisition engine to search for all satellites. Once the receiver has a position fix (or if pre-positioning information is available), the acquisition engine continues to be used to search for all visible satellites that are not being tracked.

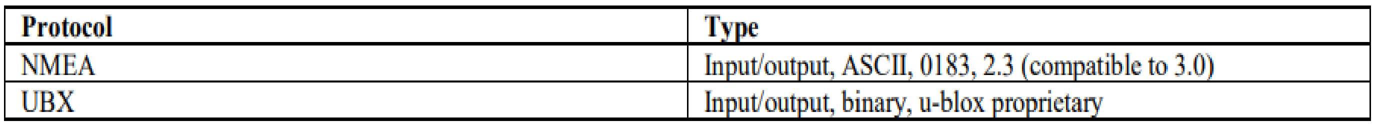
Eco Mode

During a Cold start, a receiver in Eco Mode works exactly as in Maximum Performance Mode. Once a position can be calculated and a sufficient number of satellites are being tracked, the acquisition engine is powered off resulting in significant power savings. The tracking engine continuously tracks acquired satellites and acquires other available or emerging satellites.

Power Save Mode

Power Save Mode (PSM) allows a reduction in system power consumption by selectively switching parts of the receiver on and off.

➢ Protocols and interfaces



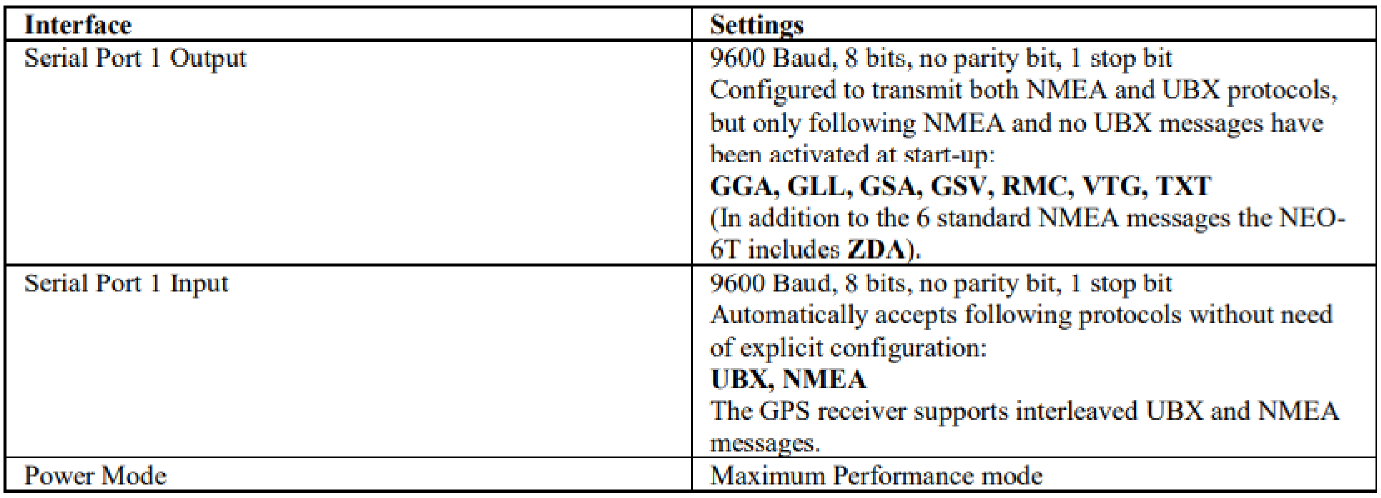
Protocol	Type
NMEA	Input/output, ASCII, 0183, 2.3 (compatible to 3.0)
UBX	Input/output, binary, u-blox proprietary

NEO 6 M chip supports the listed protocols and interfaces with UART, USB and SPI but in our project, we will depend on commercialized NEO-6 M module shown in the figure which supports an interface with UART only, NEO-6M modules include one configurable UART interface for serial communication.

➢ NEO-6M specifications

Receiver Type	50 channels, GPS L1(1575.42Mhz)
Horizontal Position Accuracy	2.5m
Navigation Update Rate	1HZ (5Hz maximum)
Capture Time	Cool start: 27sHot start: 1s
Navigation Sensitivity	-161dBm
Communication Protocol	NMEA, UBX Binary, RTCM
Serial Baud Rate	4800-230400 (default 9600)
Operating Temperature	-40°C ~ 85°C
Operating Voltage	2.7V ~ 3.6V
Operating Current	45mA
TXD/RXD Impedance	510Ω

➢ Default settings



Interface	Settings
Serial Port 1 Output	9600 Baud, 8 bits, no parity bit, 1 stop bit Configured to transmit both NMEA and UBX protocols, but only following NMEA and no UBX messages have been activated at start-up: **GGA, GLL, GSA, GSV, RMC, VTG, TXT** (In addition to the 6 standard NMEA messages the NEO-6T includes **ZDA**).
Serial Port 1 Input	9600 Baud, 8 bits, no parity bit, 1 stop bit Automatically accepts following protocols without need of explicit configuration: **UBX, NMEA** The GPS receiver supports interleaved UBX and NMEA messages.
Power Mode	Maximum Performance mode

• Once you a buy a module you can find it in of default configurations.

The module was on 9600 Baud and supports UART framing of 8 bits, no parity bit, 1 stop bit. Also, it was Configured to transmit both NMEA and UBX protocols.

• Design options:

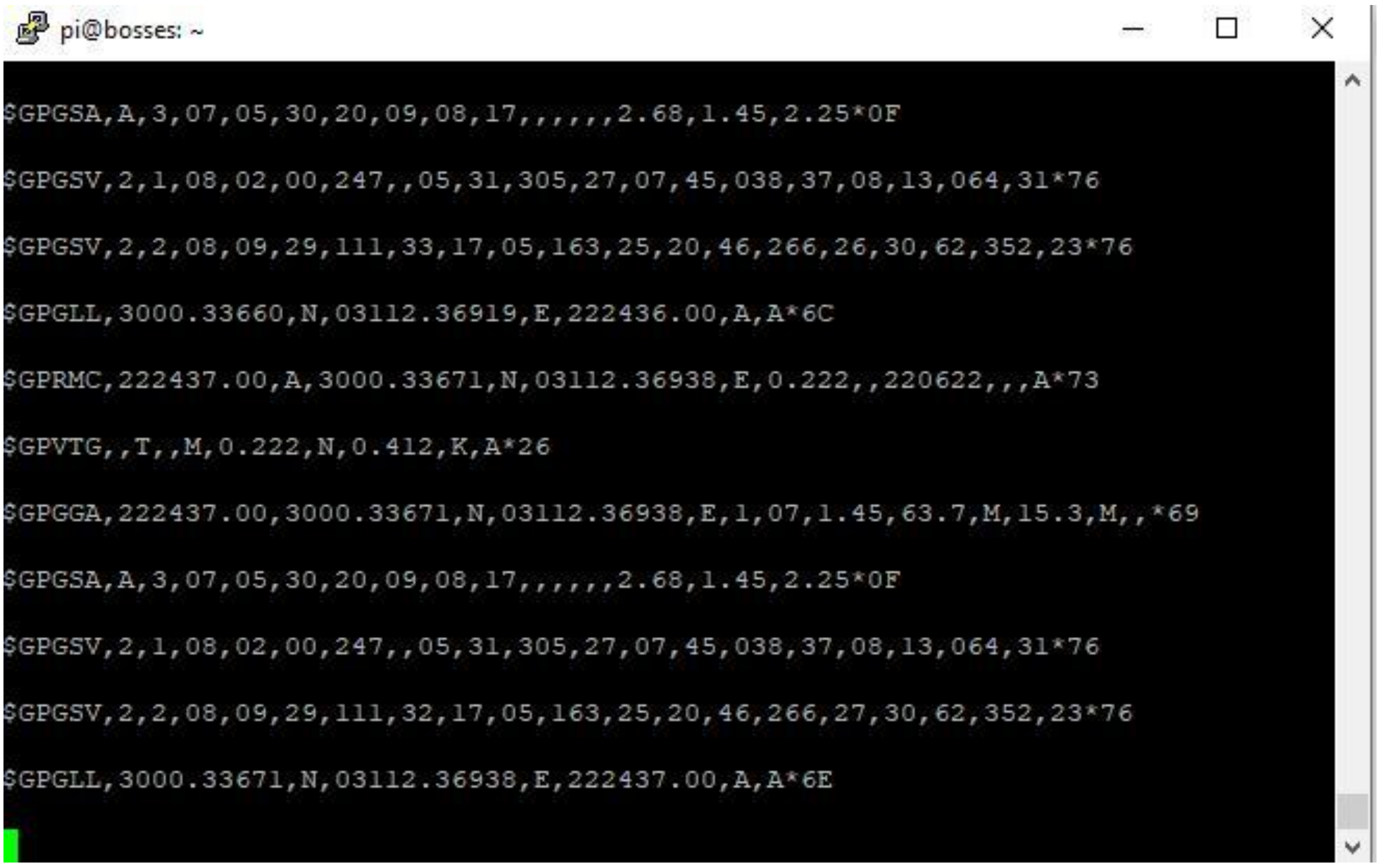
Depend on NMEA protocol frames.

Depend on UBX protocol frames.

• Our design option was to depend on NMEA protocol frames for the following reasons:

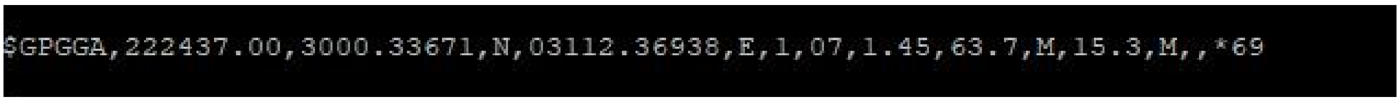
- NMEA frames are sent continuously with no need to request data from the receiver each time you need GPS frame.
- UBX protocol is bidirectional communication protocols, hence we need to send a request for the module and wait for it to respond with the needed frame which consumes double the time taken in NMEA frames reading.

• The GPS module actually receives 6 types of NMEA frames which is GGA, GLL, GSA, GSV, RMC, VTG.



```
$GPGSA,A,3,07,05,30,20,09,08,17,,,,,,2.68,1.45,2.25*0F
$GPGSV,2,1,08,02,00,247,,05,31,305,27,07,45,038,37,08,13,064,31*76
$GPGSV,2,2,08,09,29,111,33,17,05,163,25,20,46,266,26,30,62,352,23*76
$GPGLL,3000.33660,N,03112.36919,E,222436.00,A,A*6C
$GPRMC,222437.00,A,3000.33671,N,03112.36938,E,0.222,,220622,,,A*73
$GPVTG,,T,,M,0.222,N,0.412,K,A*26
$GPGGA,222437.00,3000.33671,N,03112.36938,E,1,07,1.45,63.7,M,15.3,M,,*69
$GPGSA,A,3,07,05,30,20,09,08,17,,,,,,2.68,1.45,2.25*0F
$GPGSV,2,1,08,02,00,247,,05,31,305,27,07,45,038,37,08,13,064,31*76
$GPGSV,2,2,08,09,29,111,32,17,05,163,25,20,46,266,27,30,62,352,23*76
$GPGLL,3000.33671,N,03112.36938,E,222437.00,A,A*6E
```

GGA (Global Positioning System Fixed Data) is an NMEA frame which consists of 70 to 75 characters contains the needed data and other data which is unneeded.



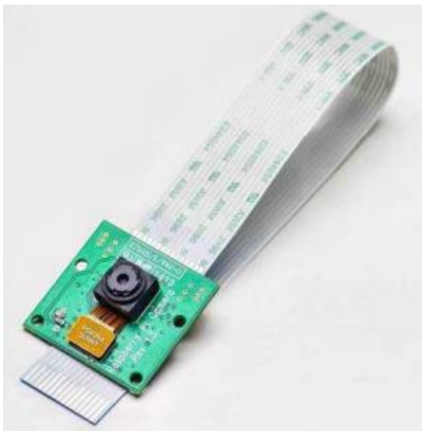
```
$GPGGA,222437.00,3000.33671,N,03112.36938,E,1,07,1.45,63.7,M,15.3,M,,*69
```

Latitude = 3000.33671 Longitude = 03112.36938

• We developed a code that decodes the GGA frame and extract data of it and started the testing of module alone. It contains the needed data which is Time, latitude and longitude. Based on this investigation we found that GGA frame supports all needed information and it would be enough for our application.

4.2 Camera Module

We used RP camera to capture the area around the vehicle, The Raspberry Pi Camera Module v2 is a high quality 8-megapixel image sensor custom designed add - on board for Raspberry Pi, featuring a fixed focus lens. It attaches to Pi by way of one of the small sockets on the board upper surface and uses the dedicated C Si interface, designed especially for interfacing to cameras.



The board itself is tiny, at around 25mm x 2 3 mm x 9mm. It also weighs just over 3g, making it perfect for mobile or other applications where size and weight are important.

RP Camera specifications:

• 8-megapixel camera capable of taking photographs of 3280 x 2464 pixels and static photos up to 25920 x1944 Pixel.

• has high resolution and small size that is specially designed for all versions of raspberry Pi.

• Capture video at 1080p30, 720p60 and 640x480p90 resolutions.

• All software is supported within the latest version of Raspbian Operating System.

• It is connected directly to Raspberry using small ribbon -comes with the camera - through the CSI (camera serial interface) port on your Raspberry- Pi.

4.3 Speed Sensor Module

Since v2v technology needs complete information about all the vehicles committed to it, it was necessary to calculate the speed of cars so that relative speed between vehicles on the road can be calculated and decisions can be taken according to that.

We used LM 393 speed sensor shown in the figure:

consists of two parts: the sensor part and the control part.

Sensor Part:

an Infrared LED and an NPN Photo Transistor. These two components are placed facing each other is a special housing made of black thermoplastic. This special housing ensures that the Photo Transistor receives light only from the Infrared LED and all the external source of light is eliminated.

Control Part:

LM393 Voltage Comparator and a few passive electronic components. The signal from the photo transistor is given to the LM393 and based on the presence or absence of an object between the Infrared LED and the Photo Transistor, the Output of the LM393 IC will either be HIGH or LOW. So in a nutshell it’s the IR LED always on and passes an IR ray if it meets the motor’s wheel slit so IR passes to the NPN photo transistor so the voltage comparator results HIGH, and if the IR ray meets blank areas between motor’s wheel slits it blocks the ray so it doesn’t reach the photo transistor so the voltage comparator results LOW.

4.4 Lane Number Detection

The lane number that the vehicle is traveling in is a key factor in intelligent vehicle fields. Many lane detection algorithms were proposed and if we can perfectly detect the lanes, we can directly calculate the lane number from the lane detection results.

Our code aims to detect the lane number that the vehicle is travelling on it that helping in collecting more information about every vehicle exact location and position.

We consider that a camera has been placed at a considerable height in every vehicle. This camera is responsible for relaying a live feed to the central servers, we were successfully able to derive the frames from the video feed that taken from the camera, the frames were processed and we were able to detect the lanes accurately using OpenCV in maximum cases (ie ideal cases).

Implementation Process:

The python code takes a video as input and extracts the frames from the video at regular intervals. These frames are then processed for lane detection and the driver's lane is identified accurately. This lane number is stored into a file so that we can use it later.

Our python code contains all the functions that are needed for image processing. The frames (images) are extracted from the video input (feed from the car's video camera). The lane in which the vehicle is driven upon will always be in the center of the image. We are not interested in our own lane, but the count of other lanes in the left and right area to find the lane number. Hence, the quadrilateral: lower_left → top_left_i → top_right_i → lower_right defines the area in which there will be canny edges of our own lane. All the image except this may contain the other desired lanes along with unwanted objects such as clouds, white coloured stop signs, etc which may give false line segments. To avoid this, we eliminate the upper quadrilateral too: (0,0) → (0,imshape[1]) → top_right → top_left. After eliminating these two regions, we get our desired region of interest, a concave hexagon: lower_left → top_left_i → top_right_i → lower_right → top_right → top_left. This region of image gives us all the desired lanes on the road.

We use two variables in our algorithm for classification first, “c_l” and the other “c_r”. We elicit the x-coordinates of the line segments drawn for the lanes in the region of interest and compare each of them with “c_l” and “c_r”. If the x-coordinates are less than c_l, that means the lane(line segment) must be a slower lane and we increase the count of left_lanes by 1. And if the x-coordinates are greater than c_r, that means the lane(line segment) discovered must be a faster lane, the count of right_lanes is incremented. Hence, in a 3 lane road, the value of lane_number is determined by: If right_lanes count is 0, lane_number must be 3 , as there will be 2 lanes detected in the left part of the image. Similarly if the count of left_lanes is 0, we must be in the 1st lane. And if both the count of left_lanes and right_lanes is greater than 0, we infer that we are in the middle lane (lane_number=2). Now, this lane_number is written to a file to be used later.

Steps of implementation:

1. Extract images(frames) from the video
2. Grey Filtering to get a grayscale image
3. Gaussian Blur to reduce noise of the image
4. Canny edge detection to get a greyscale image with marked boundaries
5. Scan boundary the masking to eliminate background and extract region of interest
6. Hough Transform returns line segments to be drawn and detect the edges endpoints and draw this lines in the original image.
7. the output image of these 6 steps that is image with given specific marks in the lanes then this image and lines will be input to a function that detect the lane number.

CHAPTER 5: COMMUNICATION AND NETWORKING

Vehicle-to-vehicle technology uses many technologies like WIFI, LTE and dedicated short-range communications (DSRC), the best is the DSRC but this module isn't available so we used WIFI in order to transfer frames between cars.

Python is chosen to be the main programming language for the networking and communication task, it is the preferable language because it comes with libraries for communication through servers and Wi Fi – “Socket”.
we used Socket programming in order to transfer frames between vehicles.

5.1 Socket Programming in Python

What are sockets?

Socket programming is one of the most fundamental technologies of computer network programming, it considers as the backbone of networking. Sockets are the endpoints of a bidirectional communications channel, they make the transfer of information between two different programs, devices or processes on the same machine, or between processes on different continents.
Sockets are interior endpoints of an inter-process communication flow across a computer network built for sending and receiving data by connecting two nodes on a network together to communicate with each other. A single network will have two sockets, one for each communicating device or program.
These sockets are a combination of an IP address and a Port. One socket listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. There are Different ports available for different types of protocols and different channel types (Unix domain sockets, TCP, UDP, and so on). A single device can have ‘n’ number of sockets based on the port number that is being used
Today, most communication between computers is based on the internet protocol; therefore, most network sockets are internet sockets. To create a connection between

machines, Python programs import the socket module, create a socket object, and call the object's methods to establish connections and send and receive data.

Sockets in Python:

Python provides two levels of access to network services. At a low level, you can access the basic socket support in the underlying operating system, which allows you to implement clients and servers for both connection-oriented and connectionless protocols. Python also has libraries that provide higher level access to specific application-level network protocols, such as FTP, HTTP, SMTP, and so on.
To achieve Socket Programming in Python, you will need to import the socket module or library. This module consists of built-in methods and specific classes that are required for creating sockets and help them associate with each other.
The Python Socket Programming has two sections:

- Python Server Socket Program
- Python Client Socket Program

The client and server can communicate by writing to or reading from their sockets. The server forms the listener socket while the client reaches out to the server.

Vocabulary that used in Sockets

Term	Description
Domain	The family of protocols that will be used as the transport mechanism. These values are constants such as **AF_INET**, **PF_INET**, **PF_UNIX**, **PF_X25**, and so on.
Type	The type of communications between the two endpoints, typically **SOCK_STREAM** for connection-oriented protocols and **SOCK_DGRAM** for connectionless protocols.

Protocol	Typically, zero, this may be used to identify a variant of a protocol within a domain and type.
Hostname	The identifier of a network interface: • A string, which can be a host name, a dotted-quad address, or an IPV6 address in colon (and possibly dot) notation • A string **"<broadcast>**", which specifies an **INADDR_BROADCAST** address. • A zero-length string, which specifies **INADDR_ANY**, or • An Integer, interpreted as a binary address in host byte order.
Port	Each server listens for clients calling on one or more ports. A port may be a fixnum port number, a string containing a port number, or the name of a service.

TCP Sockets creation

To create a socket, you must use the *socket. Socket ()* function available in *socket* module, which has the general syntax:

```
s = socket. Socket (socket family, socket type, protocol=0)
```

Here is the description of the parameters:

- **socket family:** This is either AF_UNIX or AF_INET
  (AF_INET refers to the address-family ipv4).
- **socket type:** This is either SOCK_STREAM or SOCK_DGRAM.
- **protocol:** This is usually left out, defaulting to 0.

You’re going to create a socket object using socket.socket(), specifying the socket type as socket.SOCK_STREAM because the protocol that is used is the Transmission Control Protocol (TCP).

Why TCP Protocol?

Networks are a best-effort delivery system. There’s no guarantee that your data will reach its destination or that you’ll receive what’s been sent to you.Network devices, such as routers and switches, have finite bandwidth available and come with their own inherent system limitations. They have CPUs, memory, buses, and interface packet buffers, just like your clients and servers. TCP relieves you from having to worry about packet loss, out-of-order data arrival, and other pitfalls that invariably happen when you’re communicating across a network. TCP is :

- reliable: Packets dropped in the network are detected and retransmitted by the sender.
- Has in-order data delivery: Data is read by your application in the order it was written by the sender.

- Once you have socket object, then you can use required functions to create your client or server program. Following is the list of **functions** required:

Server Socket Methods

Method	Description
bind ()	This method binds address (hostname, port number pair) to socket.
listen ()	This method sets up and start TCP listener.
accept ()	This passively accept TCP client connection, waiting until connection arrives (blocking).

Client Socket Methods

Method	Description
connect ()	This method actively initiates TCP server connection.

General Socket Methods

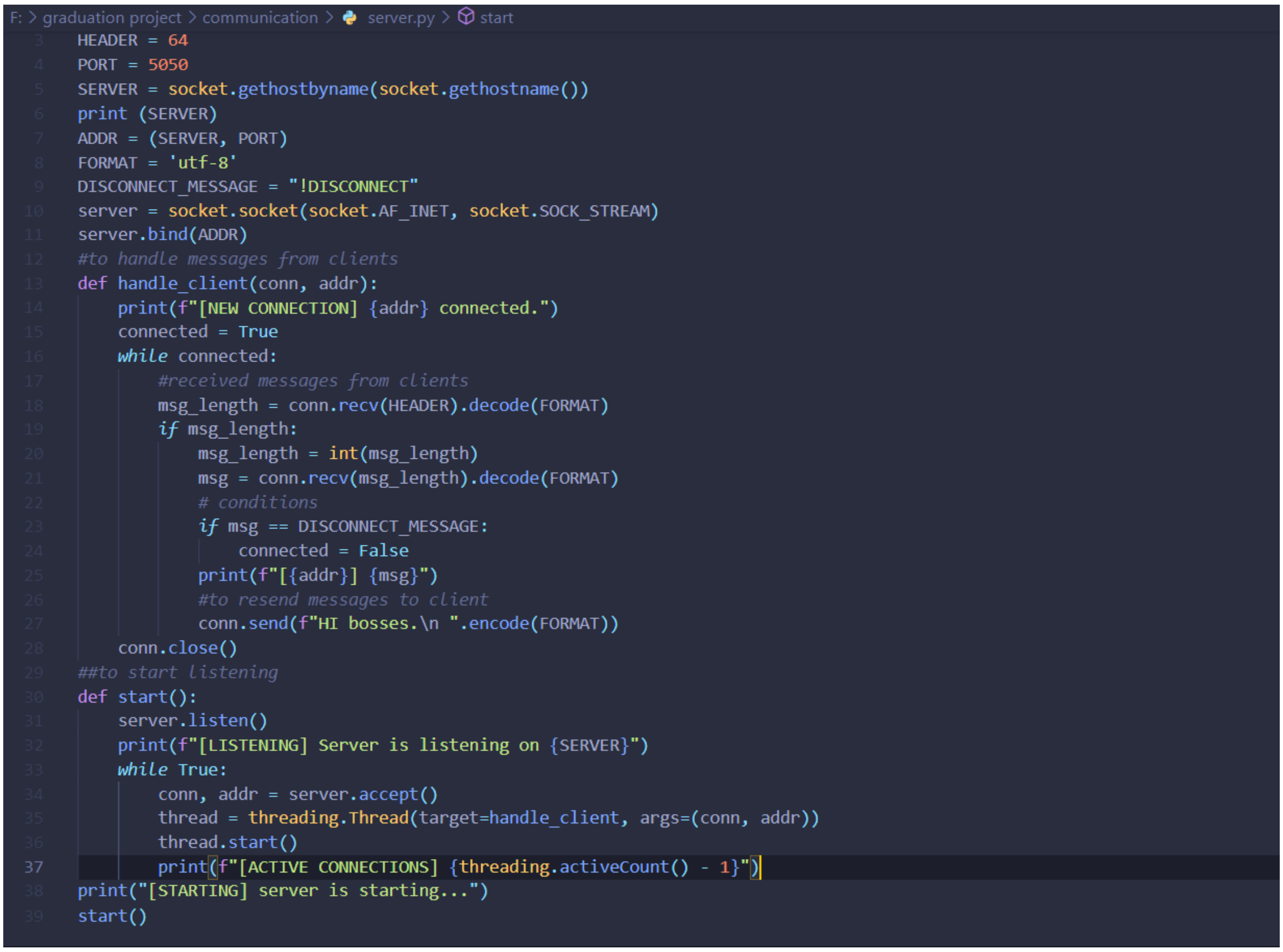
Method	Description
s.rvc ()	This method receives TCP message
s.send ()	This method transmits TCP message
s.close ()	This method closes socket
socket.gethostname ()	Returns the hostname.

Server Socket Program

A server is either a program, a computer, or a device that is devoted to managing network resources. Servers can either be on the same device or computer or locally connected to other devices and computers or even remote. There are various types of servers such as database servers, network servers, print servers, etc.

The Server Socket Program act as a Server and listening to client's request. When the Server Socket accept a request from the Client side, it reads the data from client and also it write the response to the connected client program.

Servers commonly make use of methods like socket.socket(), socket.bind(), socket.listen(), etc. to establish a connection and bind to the clients.


F: > graduation project > communication > server.py > start

```python
HEADER = 64
PORT = 5050
SERVER = socket.gethostbyname(socket.gethostname())
print (SERVER)
ADDR = (SERVER, PORT)
FORMAT = 'utf-8'
DISCONNECT_MESSAGE = "!DISCONNECT"
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(ADDR)
#to handle messages from clients
def handle_client(conn, addr):
    print(f"[NEW CONNECTION] {addr} connected.")
    connected = True
    while connected:
        #received messages from clients
        msg_length = conn.recv(HEADER).decode(FORMAT)
        if msg_length:
            msg_length = int(msg_length)
            msg = conn.recv(msg_length).decode(FORMAT)
            # conditions
            if msg == DISCONNECT_MESSAGE:
                connected = False
            print(f"[{addr}] {msg}")
            #to resend messages to client
            conn.send(f"HI bosses.\n ".encode(FORMAT))
    conn.close()
##to start listening
def start():
    server.listen()
    print(f"[LISTENING] Server is listening on {SERVER}")
    while True:
        conn, addr = server.accept()
        thread = threading.Thread(target=handle_client, args=(conn, addr))
        thread.start()
        print(f"[ACTIVE CONNECTIONS] {threading.activeCount() - 1}")
print("[STARTING] server is starting...")
start()
```

- After creating a socket object by **socket.socket()** as explained earlier, the socket object is then used to call other functions to setup a socket server.

The .bind() method is used to associate the socket with a specific network interface and port number. The values passed to .bind() depend on the address family of the socket. Here, we're using socket.AF_INET (IPv4). So it expects a two-tuple: (host, port), host can be a hostname, IP address, or empty string. port represents the TCP port number to accept connections on from clients. It should be an integer from 1 to 65535, as 0 is reserved, here we used a port number 5050.

Then, the **.listen() method** enables a server to accept connections. It makes the server a "listening" socket (This allows the server to listen to incoming connections)

Then, the **.accept() method** blocks execution and waits for an incoming connection. When a client connects, it returns a new socket object representing the connection and a tuple holding the address of the client. The new socket object from .accept() is important because it's the socket that will be used to communicate with the client. It's distinct from the listening socket that the server is using to accept new connections.

At last, we make a while loop and start to accept all incoming connections and close those connections after a message to all connected sockets.

Client Socket Program

A client is either a computer or software that receives information or services from the server. In a client-server module, clients request for services from servers.

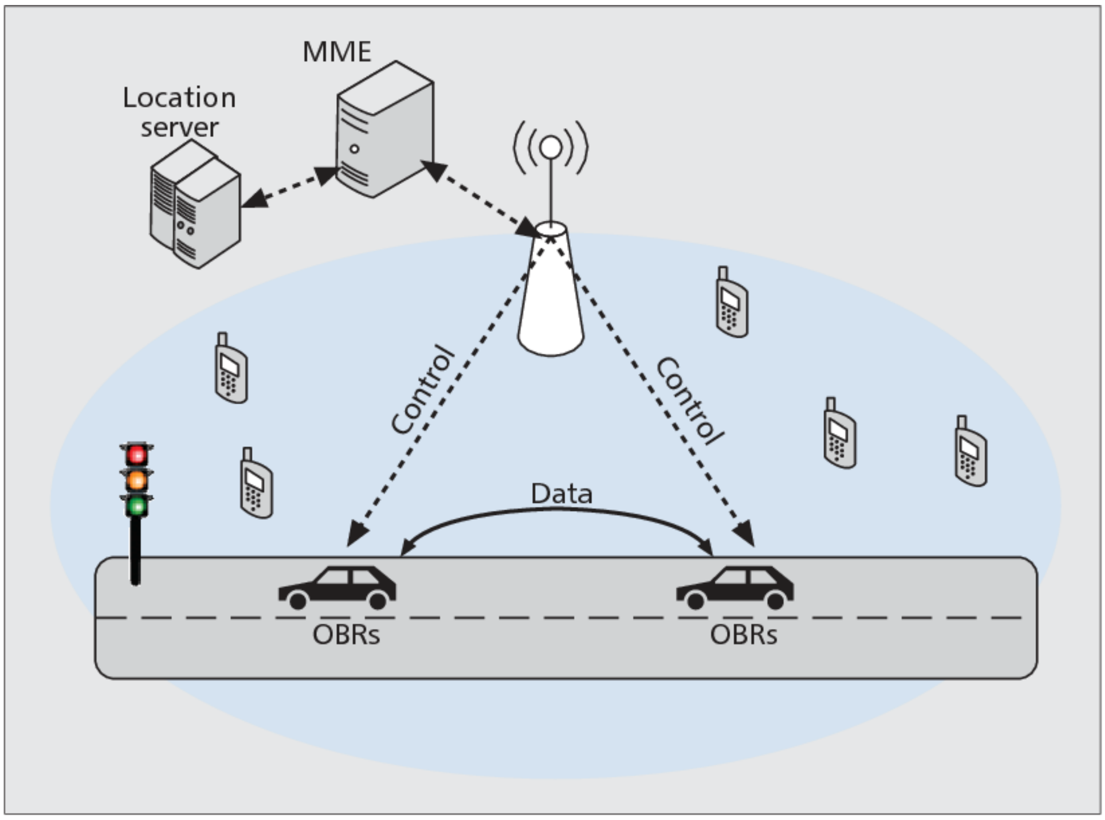
There is a very simple client program which will open a connection to a given port 5050 and given host, reads any available data from the socket, and then exits:

```
import socket host = socket.gethostname()

port=5050

s = socket.socket()

s.connect((host, port))

s.sendall(b'WelcomeUser!')

data = s.recv(1024)

s.close()

print(repr(data))
```

The client program creates a socket object, uses **.connect() method** to connect to the server by opening a TCP connection to hostname on the port, then, calls **.sendall() method** to send its message. Lastly, it calls **.recv() method** to read the server's reply and then prints it.

- We used **Multithreaded socket programs** to allow the server to communicate with more than on client concurrently, a **thread** is a sequence of such instructions within a program that can be executed independently of other code. A multithreaded program contains two or more parts that can run concurrently. Each part of such a program is called a thread, and each thread defines a separate path of execution. Multithreaded Socket Programming describes that a Multithreaded Socket Server can communicate with more than one client at the same time in the same network.

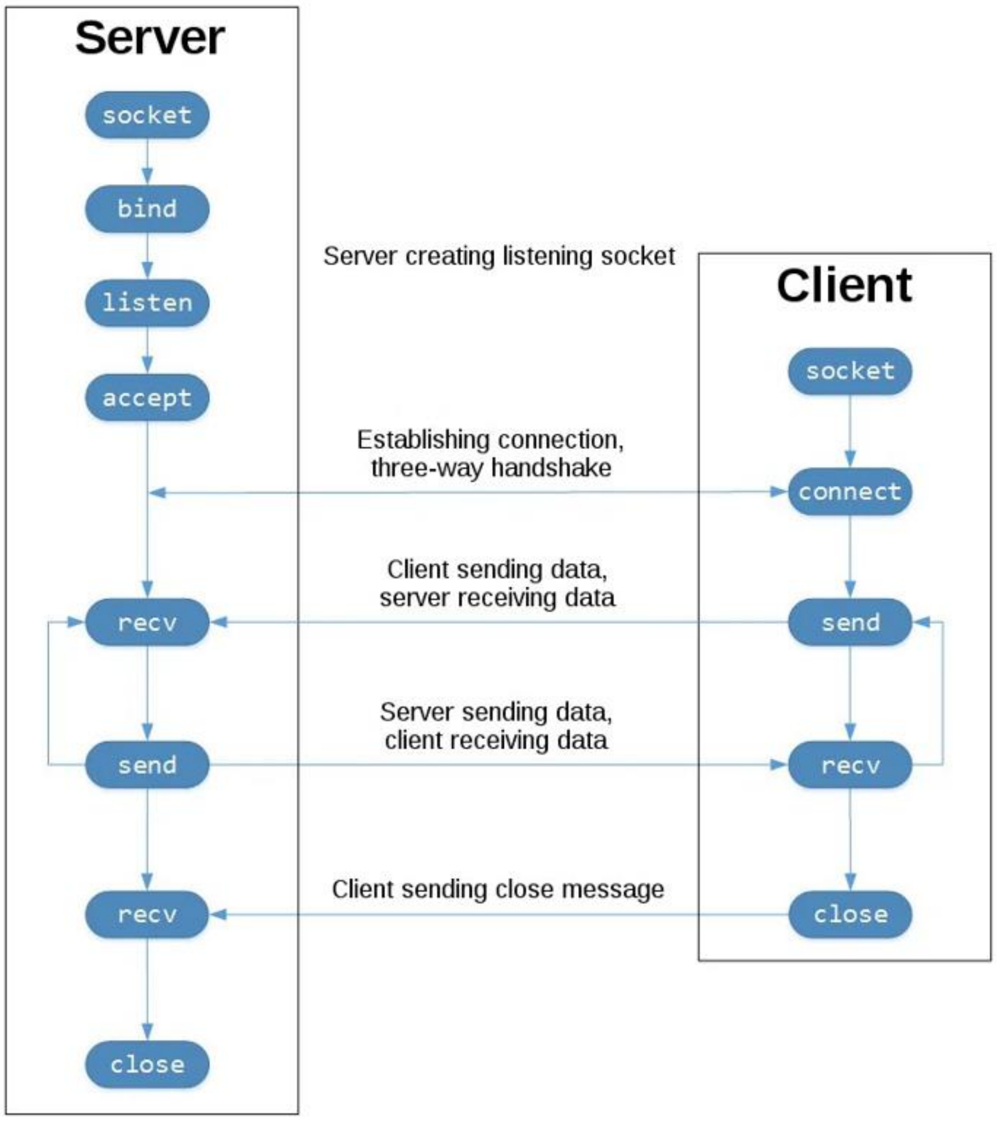
5.2 Sockets in V2V Communication System



In V2V communication system all vehicles are connected to the same server, these server handling the connection between all clients (vehicles) that enables them to communicate with each other by sending and receiving messages. After vehicles access to the server, they now can transfer data between each other across the server.

The server creates a listening socket, it listens for connections from clients. When a client connects, the server calls .accept() to accept, or complete, the connection. The client calls .connect() to establish a connection to the server and initiate the three-way handshake to ensures that each side of the connection is reachable in the network, in other words that the client can reach the server and vice-versa.

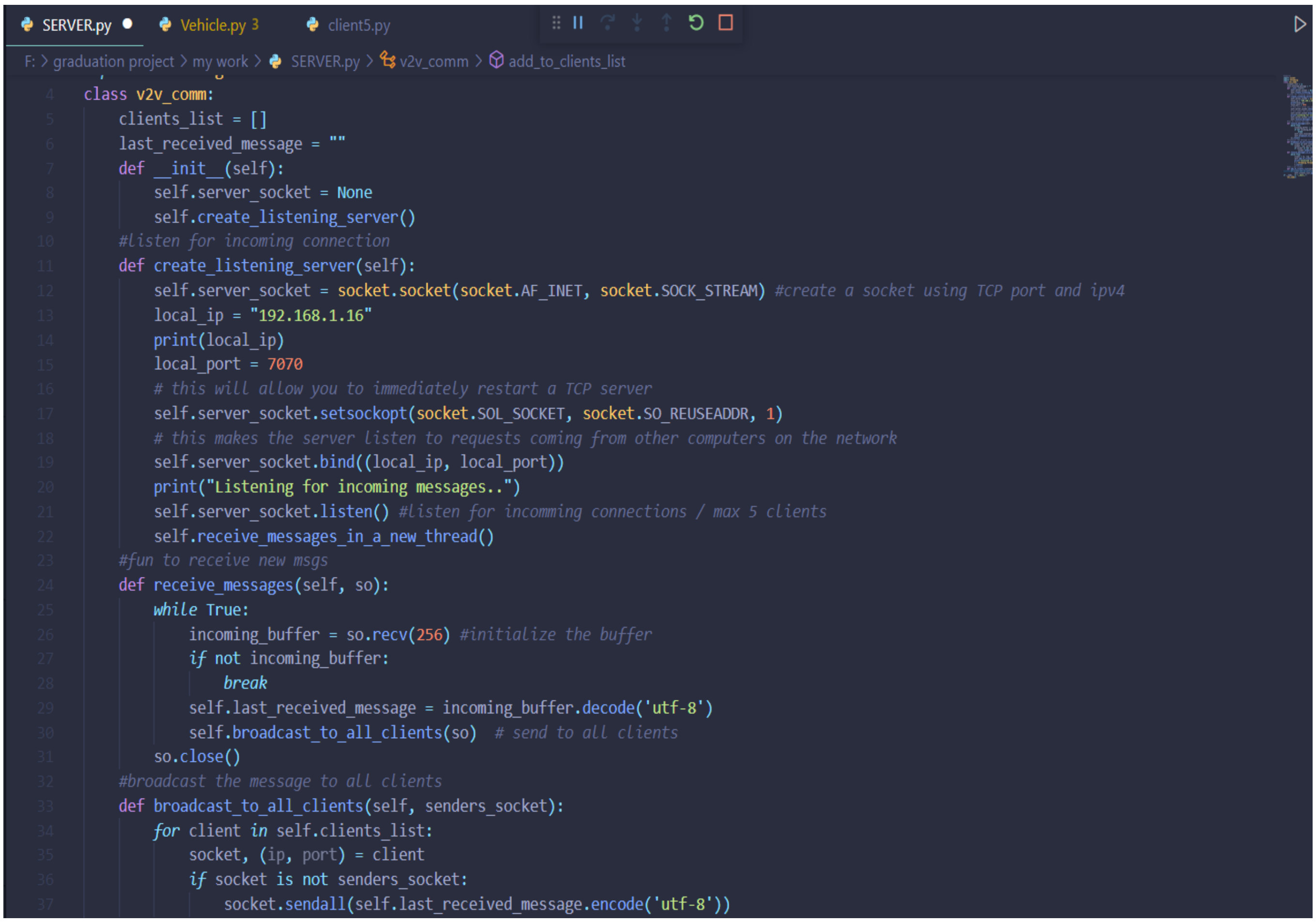
The following flow chart describes show how are vehicles can access to the server using server Ip and port:


Server
socket
bind
listen
accept
recv
send
recv
close
Server creating listening socket
Establishing connection, three-way handshake
Client sending data, server receiving data
Server sending data, client receiving data
Client sending close message
Client
socket
connect
send
recv
close

First,

we create a central server to listen to incoming vehicles and create a network to let vehicles communicate with each other. These server receives the message from the host vehicle as a package (list) contains information about the vehicle and information that the host vehicle want to inform the other vehicles about it, then the server takes these massage and encode it and broadcast these message to all vehicles connected to this server.

These code shows how we create the central server of the system:


SERVER.py
Vehicle.py 3
client5.py
F: > graduation project > my work > SERVER.py > v2v_comm > add_to_clients_list

```python
class v2v_comm:
    clients_list = []
    last_received_message = ""
    def __init__(self):
        self.server_socket = None
        self.create_listening_server()
    #listen for incoming connection
    def create_listening_server(self):
        self.server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #create a socket using TCP port and ipv4
        local_ip = "192.168.1.16"
        print(local_ip)
        local_port = 7070
        # this will allow you to immediately restart a TCP server
        self.server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        # this makes the server listen to requests coming from other computers on the network
        self.server_socket.bind((local_ip, local_port))
        print("Listening for incoming messages..")
        self.server_socket.listen() #listen for incomming connections / max 5 clients
        self.receive_messages_in_a_new_thread()
    #fun to receive new msgs
    def receive_messages(self, so):
        while True:
            incoming_buffer = so.recv(256) #initialize the buffer
            if not incoming_buffer:
                break
            self.last_received_message = incoming_buffer.decode('utf-8')
            self.broadcast_to_all_clients(so)  # send to all clients
        so.close()
    #broadcast the message to all clients
    def broadcast_to_all_clients(self, senders_socket):
        for client in self.clients_list:
            socket, (ip, port) = client
            if socket is not senders_socket:
                socket.sendall(self.last_received_message.encode('utf-8'))
```

Second,

we create our vehicles (clients) socket to connect to the server to communicate with other vehicles in the same network which created by the server.

We have two types of vehicles, the vehicle that sends the message to the server to warn or tell the other vehicles about a definite situation or information, These message is consists of a list that contains:

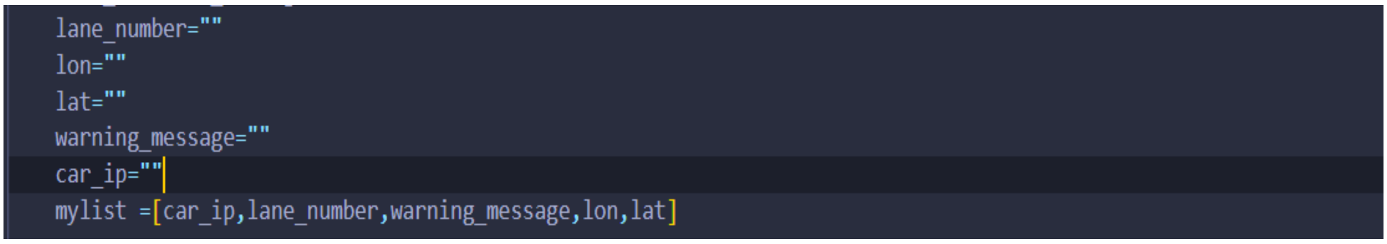
1- Car IP

2- Lane number (the car travelling in which lane)

3- Warning message contains the current state of the car

4- Longitude

5- Latitude



```
lane_number=""
lon=""
lat=""
warning_message=""
car_ip=""
mylist =[car_ip,lane_number,warning_message,lon,lat]
```

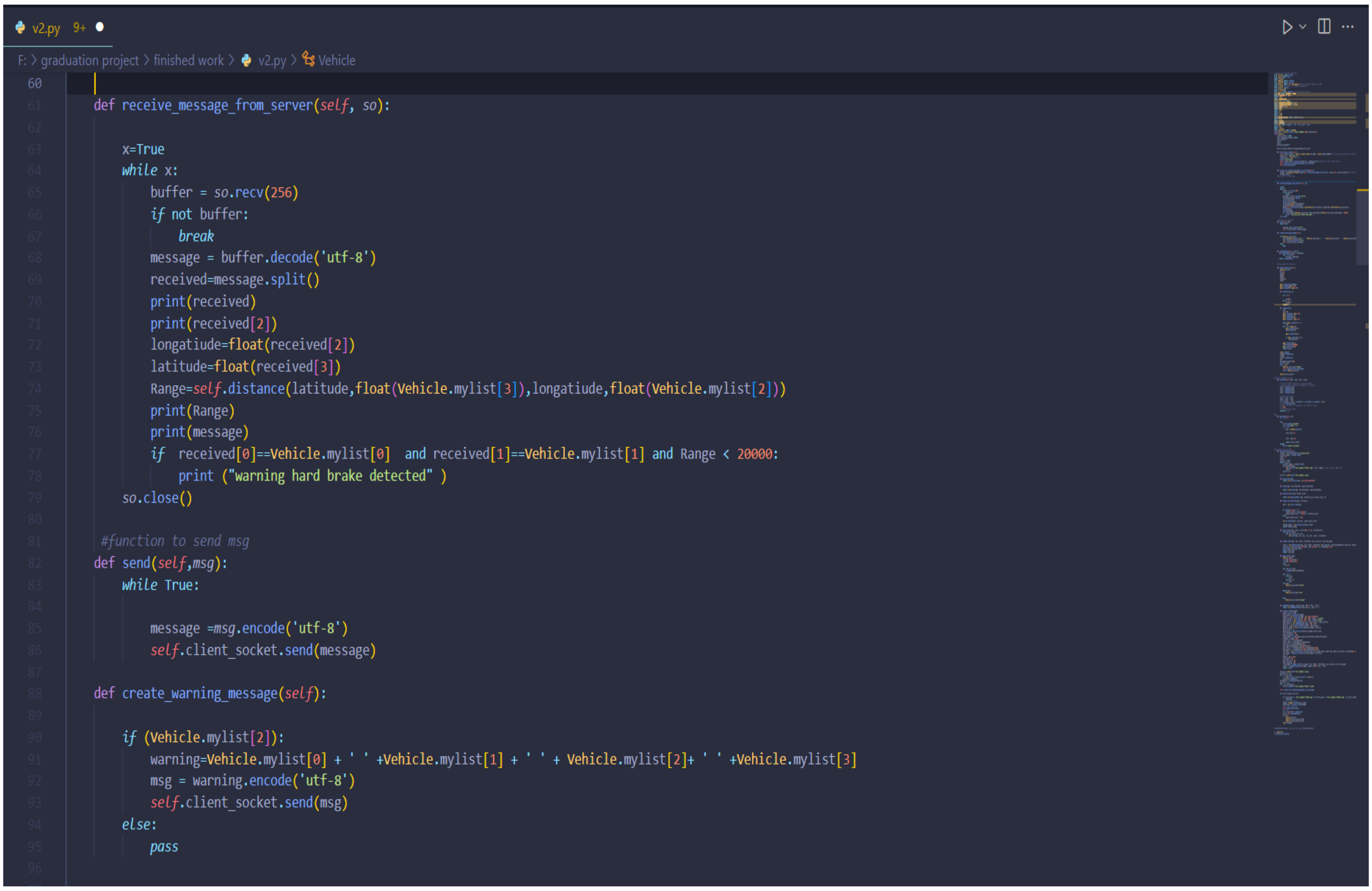
Then the server will broadcast these massage (list) to all other vehicles.

The other type of vehicles that will receive these message, they will receive the message as a list consists of number of characters separated with commas as the following:

192.168.4.1, second, braking, 4717.113990, 092725.0000

Then the client program will decode these message and splits the list to get the information and put them into variables related to their use. After that it will compare this information with the current vehicle information, if they are useful or will make a benefit to these vehicle, the system will receive it and shows to the user how he can benefit from it, if this information isn't useful for the current vehicle so the system will ignore it without intervention from the user.

These code show how we create our vehicles(clients) of the system:



```python

    def receive_message_from_server(self, so):

        x=True
        while x:
            buffer = so.recv(256)
            if not buffer:
                break
            message = buffer.decode('utf-8')
            received=message.split()
            print(received)
            print(received[2])
            longatiude=float(received[2])
            latitude=float(received[3])
            Range=self.distance(latitude,float(Vehicle.mylist[3]),longatiude,float(Vehicle.mylist[2]))
            print(Range)
            print(message)
            if  received[0]==Vehicle.mylist[0]  and received[1]==Vehicle.mylist[1] and Range < 20000:
                print ("warning hard brake detected" )
        so.close()

     #function to send msg
    def send(self,msg):
        while True:

            message =msg.encode('utf-8')
            self.client_socket.send(message)

    def create_warning_message(self):

        if (Vehicle.mylist[2]):
            warning=Vehicle.mylist[0] + ' ' +Vehicle.mylist[1] + ' ' + Vehicle.mylist[2]+ ' ' +Vehicle.mylist[3]
            msg = warning.encode('utf-8')
            self.client_socket.send(msg)
        else:
            pass

```

CHAPTER 6: APPLICATION LAYER

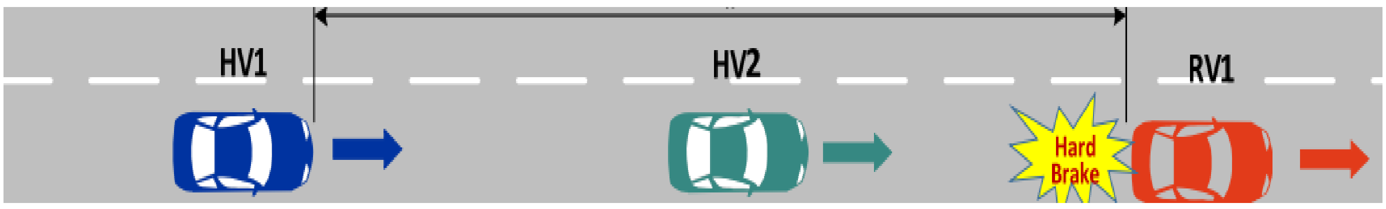
In this chapter we will explain the two applications we implemented to make our communication system useful, the following applications are some of the applications but there are a lot of other applications.

6.1 Application1: Emergency Hard Brake Detection

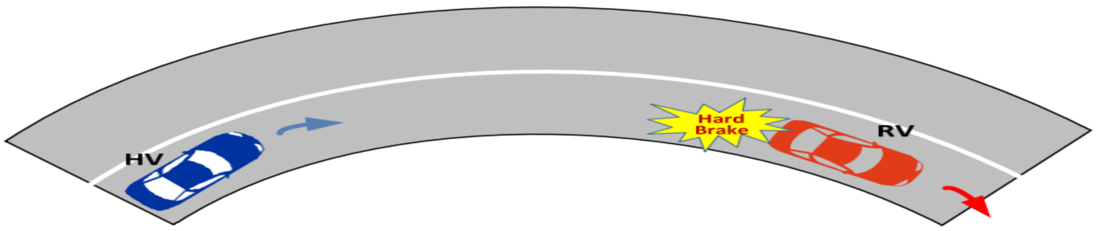
6.1.1 Description

Emergency Hard brake detection application enables a vehicle to broadcast a self-generated emergency brake event to surrounding vehicles. Upon receiving the event information, the receiving vehicle determines the relevance of the event and if appropriate provides a warning to the driver in order to avoid a crash. This application is particularly useful when the driver's line of sight is obstructed by other vehicles or due to bad weather conditions (e.g., fog, heavy rain).

The advantage of this application is achieved by saving of reaction time especially when several vehicles are moving in a row. For instance, in the scenario where the sight between vehicles HV1 and RV is blocked by a third vehicle (HV2), the driver of HV1 would be alerted only after the reaction time of HV2, which in practical scenarios may be up to 4 seconds, depending on the mental conditions of the driver. Each additional vehicle between HV1 and RV adds another significant amount to the reaction time. The saving in reaction time is offset by the signaling delay and processing time until the warning message shows up at HV1. If this latency can be limited to around 200 ms, there is still substantial gain compared to the possibly accumulated reaction time.



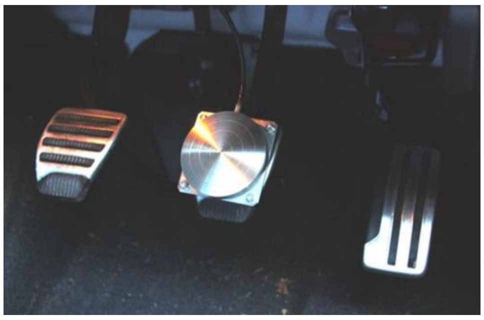
And also it will be very useful for on a rural road where the sight between HV and RV1 may be blocked due to a turn of the road, so HV driver can take a right decision before crashing the front vehicle from the received warning message.



6.1.2 Implementation

It is expected know which vehicles should receive a warning which are inside the warning zone while vehicles outside of the warning zone should not be disturbed by such warning indications.

So our tools divided into two parts, tools that detect the hard brake of the vehicle ahead (Load cell, Speed sensor) and tools that detect the warning zone (GPS, camera, Lane number detection Model).



The load cell will be placed on the brakes pedal and detect the weight applied on the brake pedal and the speed sensor will detect the speed of the car, if the load cell sensed 5 kg or more on the pedal and the speed sensor reading is less than 10 km/hr, therefore there is a hard brake occurs.

So, the system broadcast a warning message this message will be received only by the vehicles in the warning zone which is detect by GPS, every vehicle has its GPS so the system measures the distances between vehicles from the outputs of the GPS (longitude and latitude) and detect the lane number from the camera and lane number detection model, if the vehicle in the same lane and the distances between them not more than 1 km the vehicle will receive the message.

- we have explained briefly before the camera module and its usage with the lane number detection model and how they work to detect the lane number for the vehicle in Chapter 4: Car Localization, we have explained also in Chapter 4 the theory of work of the GPS and the Speed sensor. Now we will explain the theory of work of the load cell, how to calculate the distance by GPS latitude and longitude.

Load Cell (Weight Sensor)

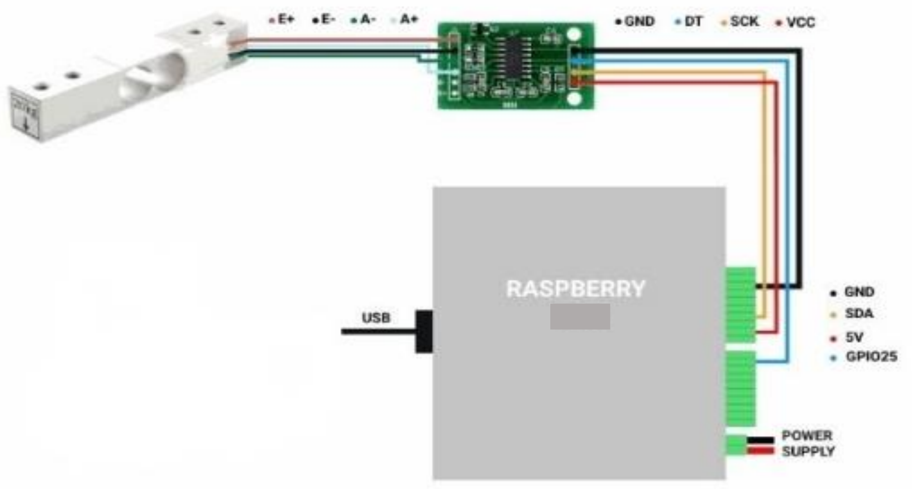
A load cell or weight sensor is one kind of sensor otherwise a transducer. A load cell works by converting mechanical force into digital values (electronic signal) that the user can read and record. The signal can be a change in voltage; current otherwise frequency based on the load as well as used circuit.

Theoretically, this sensor detects changes within a physical stimulus like force, pressure or weight and produces an output that is comparative to the physical stimulus. So, for a specific stable load otherwise weight size, this sensor provides an output value and that is comparative to the weight's magnitude.
It converts a force such as tension, compression, pressure, or torque into an electrical signal that can be measured and standardized. As the force applied to the load cell increases, the electrical signal changes proportionally.

HX711 module:

HX711 module is a Load Cell Amplifier breakout board for the HX711 IC that allows you to easily read load cells to measure weight. This module uses 24 high precision A/D converter chip HX711. It is a specially designed for the high precision electronic scale design, with two analog input channel, the internal integration of 128 times the programmable gain amplifier. The input circuit can be configured to provide a bridge type pressure bridge (such as pressure, weighing sensor mode), is of high precision, low cost is an ideal sampling front-end module. HX711 is an IC that allows you to easily integrate load cell into your project. No need of any amplifiers or dual power supply just use this board and you can easily interface it to any micro-controller to measure weight. The HX711 uses a two wire interface (Clock and Data) for communication. Compared with other chips, HX711 has added advantages such as high integration, fast response, immunity, and other features improving the total performance and reliability. Finally, it's one among the best choices for electronic enthusiasts. The chip lowers the cost of the electronic scale, at the same time, improving performance and reliability.

With the combination of the HX711 and the load cell, it is able to get measurable data the user can read.



Calculating distance from GPS output

To calculate the distance between 2 points in space we need to get latitude and longitude values of 2 GPS modules and use equations to get the distance.

Here we use the ‘haversine’ formula to calculate the great-circle distance between two points – that is, the shortest distance over the earth’s surface.

The great circle distance is the shortest distance between two points on a sphere.

In order to use this method, we need to have the co-ordinates of point A and point B.

First, convert the latitude and longitude values from decimal degrees to radians. For this divide the values of longitude and latitude of both the points by 180/pi. The value of pi is 22/7. The value of 180/pi is approximately 57.29577951. If we want to calculate the distance between two places in miles, use the value 3,963, which is the radius of Earth. If we want to calculate the distance between two places in kilometers, use the value 6, 378.8, which is the radius of Earth.

After conversion each latitude and longitude to radians, use the following formula to get the distance between point A and point B in miles:

Distance, d = 3963.0 * arccos[(sin(latA) * sin(latB)) + cos(latA) * cos(latB) * cos(longA – longB)]

If you want your value to be in units of kilometers, multiple d by 1.609344.

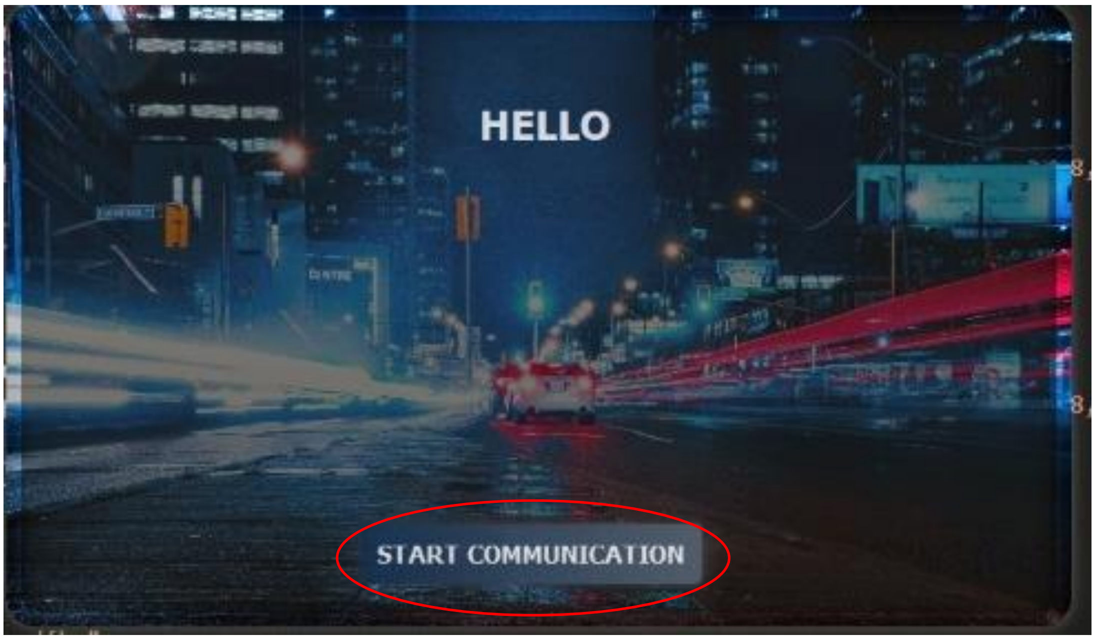
d in kilometers = 1.609344 * d in miles

Thus you can have the shortest distance between two places on Earth using the great circle distance approach.

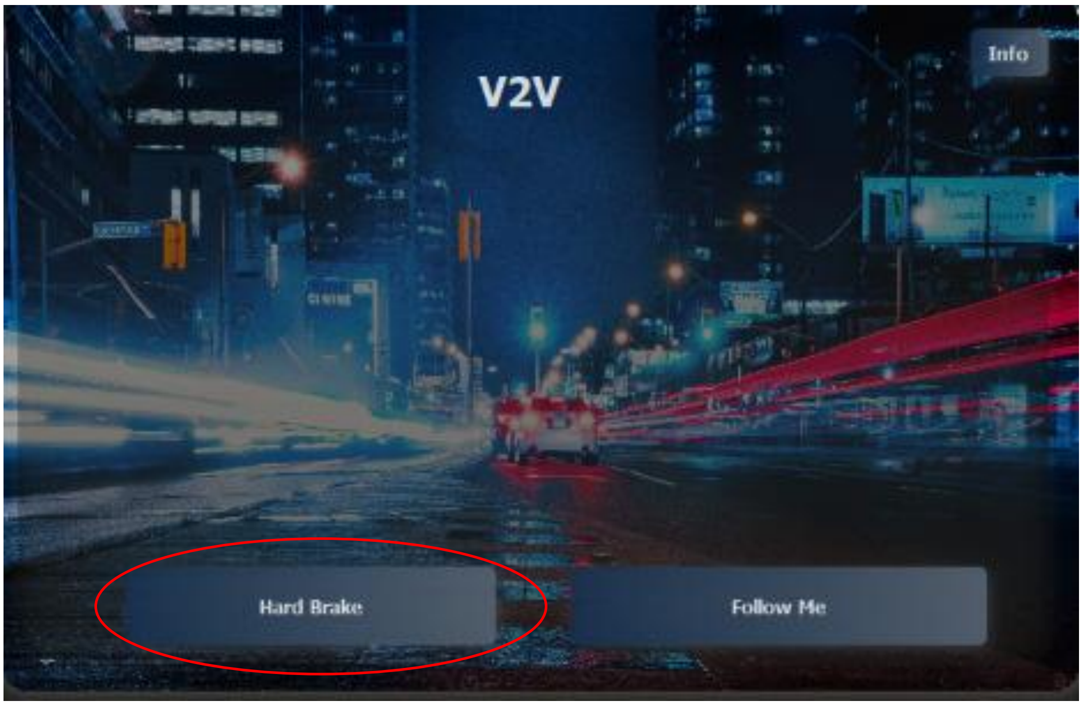
6.1.3 User Interface

In Our system the user will be able to use this application in such an easy way.

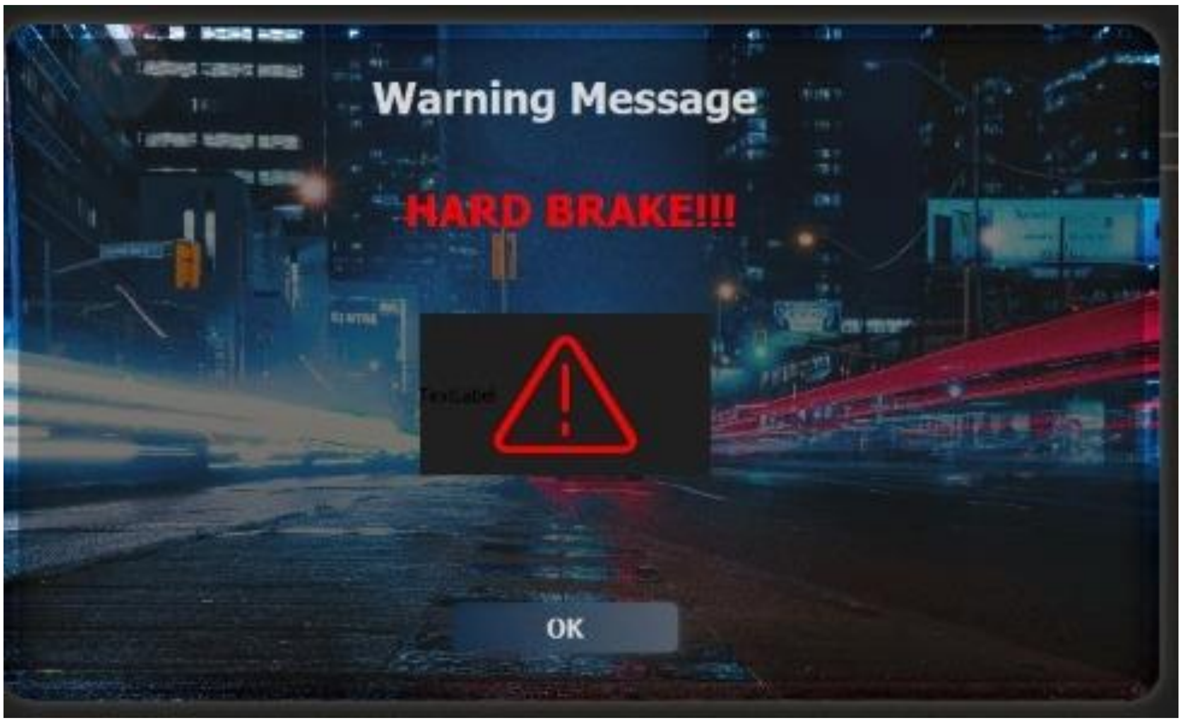
Firstly, the user should connect their cars with the server to be in contact with all other cars from the button shown in the system starting window.



Then, the system will move to another window that has a “Hard Brake” button which will enable emergency hard brake detection application in the system.



Now, If any vehicle, connected to the system server and enabled the emergency hard brake detection application, stopped suddenly all other vehicles in the warning zone will receive a warning message.



6.2 Application2: Follow Me Information

6.2.1 Description

Follow-me information is a convenience feature for people traveling in groups of vehicles. The received information can be used to plan or optimize a route.
A potential following vehicle can subscribe to the location information of a leading vehicle which is providing the service. After an authentication at and an authorization of the leading vehicle the position information shall be transmitted.
The transmission of that position shall be handled via direct communication.

The following feature will demonstrate that a following vehicle (RV) can be informed of the position of a leading vehicle (HV) via a direct link. The position of the HV is displayed in the GUI screen. The same functionality can be realized via a network link at larger distances between RV and HV.
If a vehicle enters the range of the direct link communication of a vehicle providing such an information, the communication shall switch to the direct link with a more accurate position and a higher, dynamic update rate.

Since the future world of connected cars, connected infrastructure and IoT will contain a lot of participants which are interested in specific information groups and are enabled to provide different types of data, a base service for positioning seems necessary. This application bonds a positioning service, which can be used for future applications, to an authentication.

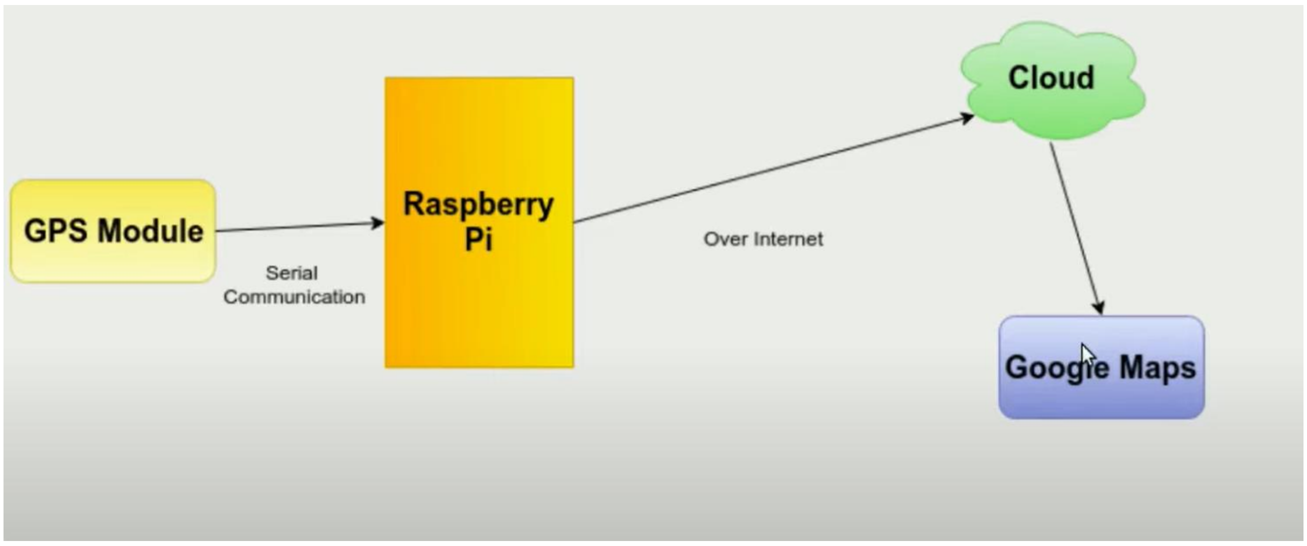
The visualization for the user shall distinguish between a position service using direct transmission and a service instance which handles the position updates via the network.

6.2.2 Implementation

Our application mainly depends on three main tools: GPS, Cloud (Thinkspeak) and Google Maps.

The GPS is used to get the car's location information which are the car's latitude and longitude, this information is sent to the cloud's database and saved on it, every time the car changes its location, its new location is updated on the cloud and saved into its database, this locations and information is sent to google maps to make the user able to track the needed car, and every time the cloud information is updated google maps is updated with the newest readings, and if the system is turned off google maps will read the last update from the cloud and store it until the system is turned on again and the cloud updated with new information.

Here in the cloud every car has its channel and its channel has two fields, field for latitude and the other field for longitude, the GPS write this information to the cloud by the Write API key function and then google maps read this information from the cloud by Read API key function. So, that any car can be able to follow (track) another car.



Thinkspeak (Cloud)

ThingSpeak is a Web Service (REST API) that lets you collect and store sensor data in the cloud and develop Internet of Things applications.

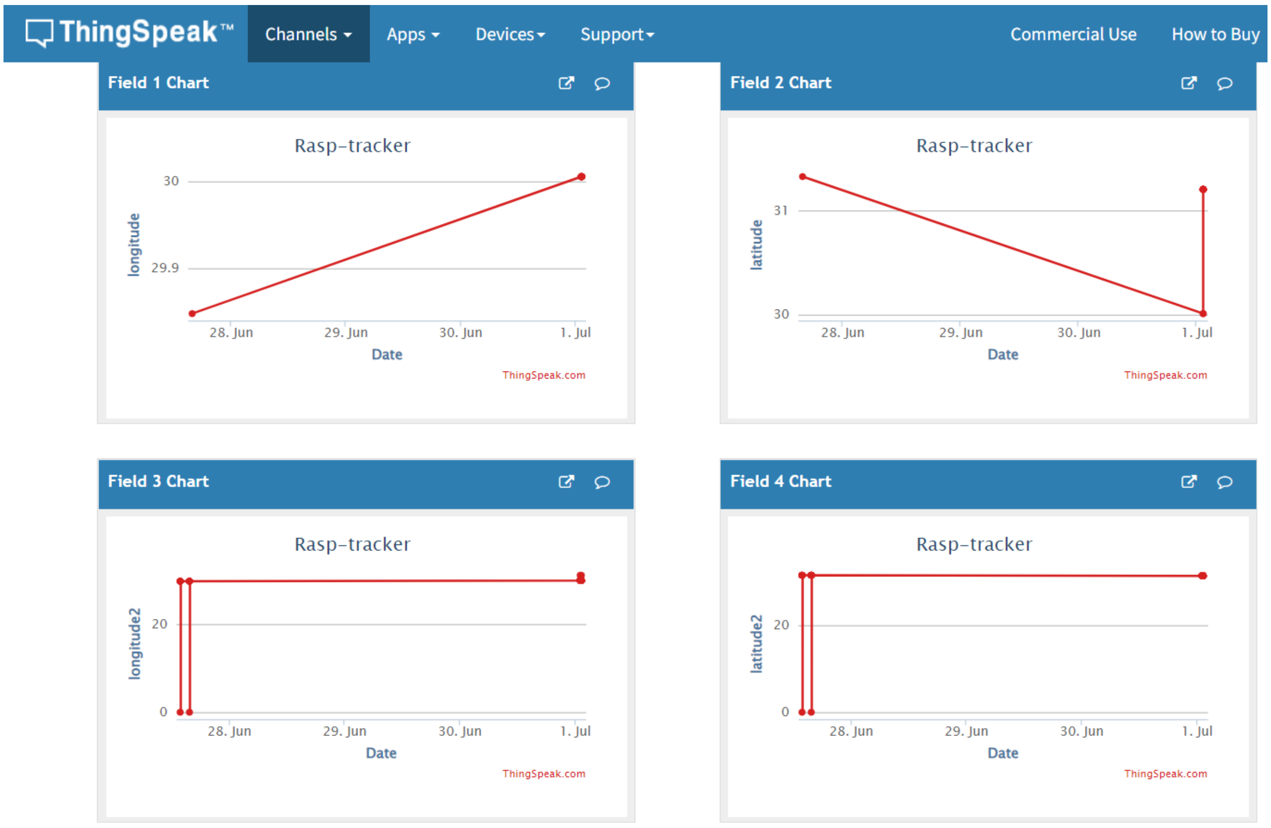
It works with Arduino, Raspberry Pi and MATLAB (premade libraries and APIs exists). But it should work with all kind of Programming Languages, since it uses a REST API.

It's considered as an IoT analytics platform service that lets you collect and store sensor data in the cloud and develop Internet of Things applications.

Thinkspeak with GPS

Each GPS stores each data which considered in two terms (latitude, longitude) in channel in cloud which we can use as a database for that car, I can know every location of running car and store it on cloud that is considerable as car tracking for emergency.

As we see in the next figure each GPS stores its latitude and longitude in each field, so every change in the value is recorded and stored. Which lead us to use this feature to apply an application.



Google Maps

Google Maps is a free web mapping service by Google that provides various types of geographical information. Using Google Maps, you can:

- Search for places or get directions from one place to another.
- View and navigate through horizontal and vertical panoramic street level images of various cities around the world.
- Get specific information like traffic at a particular point.

Google Maps provides an API using which you can customize the maps and the information displayed on them.

Steps to Load the Map on a website…

Step 1: Create an HTML Page Create a basic HTML page with head and body tags.

Step 2: Load the API

Load or include the Google Maps API using the script tag as shown below:

```
<script src="http://maps.googleapis.com/maps/api/js"></script>
```

Step 3: Create the Container

To hold the map, we have to create a container element, generally the tag (a generic container) is used for this purpose. Create a container element and define its dimensions as shown below:

```
<div id="sample" style="width:900px;height:580px;"></div>
```

Step 4: Map Options

Before initializing the map, we have to create a mapOptions object (it is created just like a literal) and set values for map initialization variables. A map has three main options (centre, zoom, and maptypeid)

- centre − Under this property, we have to specify the location where we want to centre the map. To pass the location, we have to create a LatLng object by passing the latitude and longitudes of the required location to the constructor.
- zoom − Under this property, we have to specify the zoom level of the map.
- maptypeid − Under this property, we have to specify the type of the map we want. Following are the types of maps provided by Google –

  o ROADMAP (normal, default 2D map)

  o SATELLITE (photographic map)

  o HYBRID (photographic map + roads and city names)

  o TERRAIN (map with mountains, rivers, etc.)

Within a function, say, loadMap(), create the mapOptions object and set the required values for center, zoom, and mapTypeId as shown below:

```
function loadMap() {
   var mapOptions = {
   center:new google.maps.LatLng(17.240498, 82.287598),
   zoom:9,
   mapTypeId:google.maps.MapTypeId.ROADMAP
  };
```

Step 5: Create a Map Object

You can create a map by instantiating the JavaScript class called Map. While instantiating this class, you have to pass an HTML container to hold the map and the map options for the required map. Create a map object as shown below

```
var map=new google.maps.Map(document.getElementById("sample"),mapOptions);
```

Step 6: Load the Map

Finally load the map by calling the loadMap() method or by adding DOM listener.

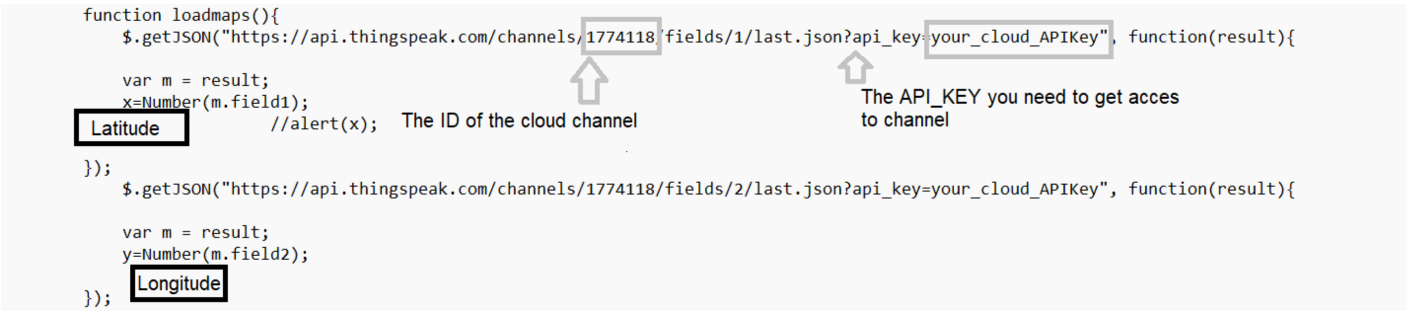
```
google.maps.event.addDomListener(window, 'load', loadMap);
                      or
<body onload="loadMap()">
```

Why Embedding a Google Map on A Website?

To have the best benefit of google map, You can include a Google map of your business, location, and reviews on your website. The map will show a Google pin with a view of your business location, address, reviews, review stars, and directions. If embedded correctly, this map can be a great addition to your website.

It is a tool that allows visitors to your website to get an overview of your business in a matter of seconds.

We used website because it's the best way to get benefit of google map and thinkspeak together, like we see in the following figure:

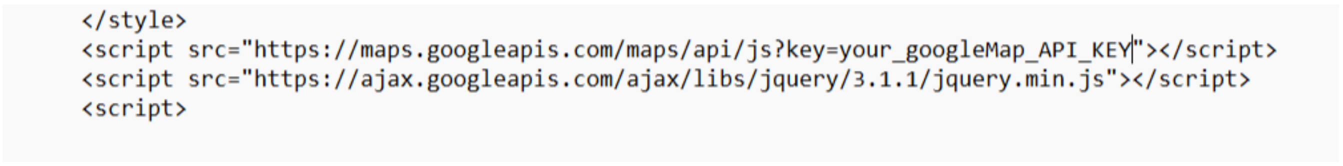


```
function loadmaps(){
    $.getJSON("https://api.thingspeak.com/channels/1774118/fields/1/last.json?api_key=your_cloud_APIKey", function(result){

    var m = result;
    x=Number(m.field1);
              //alert(x);

});
    $.getJSON("https://api.thingspeak.com/channels/1774118/fields/2/last.json?api_key=your_cloud_APIKey", function(result){

    var m = result;
    y=Number(m.field2);

});
```

Latitude
The ID of the cloud channel
The API_KEY you need to get acces to channel
Longitude

in the above figure, we can extend two values latitude and longitude from the cloud by having the id and the API_KEY of the channel and you can have many fields as you can in your cloud and you can call them into your code.

And the same for the google map in the following figure:

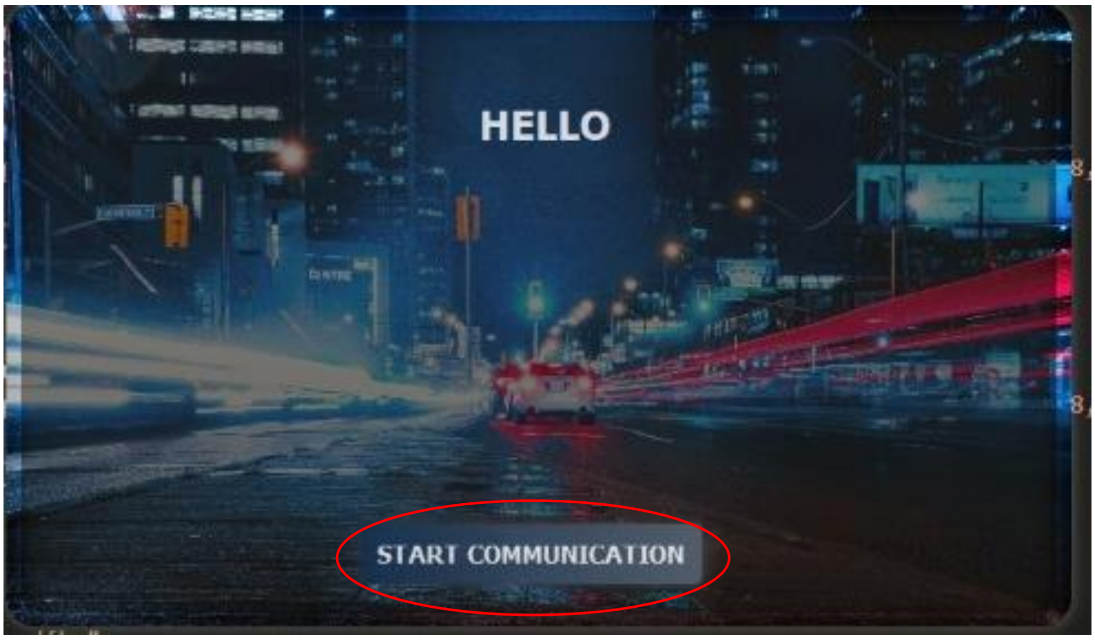


```
</style>
<script src="https://maps.googleapis.com/maps/api/js?key=your_googleMap_API_KEY"></script>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
<script>
```

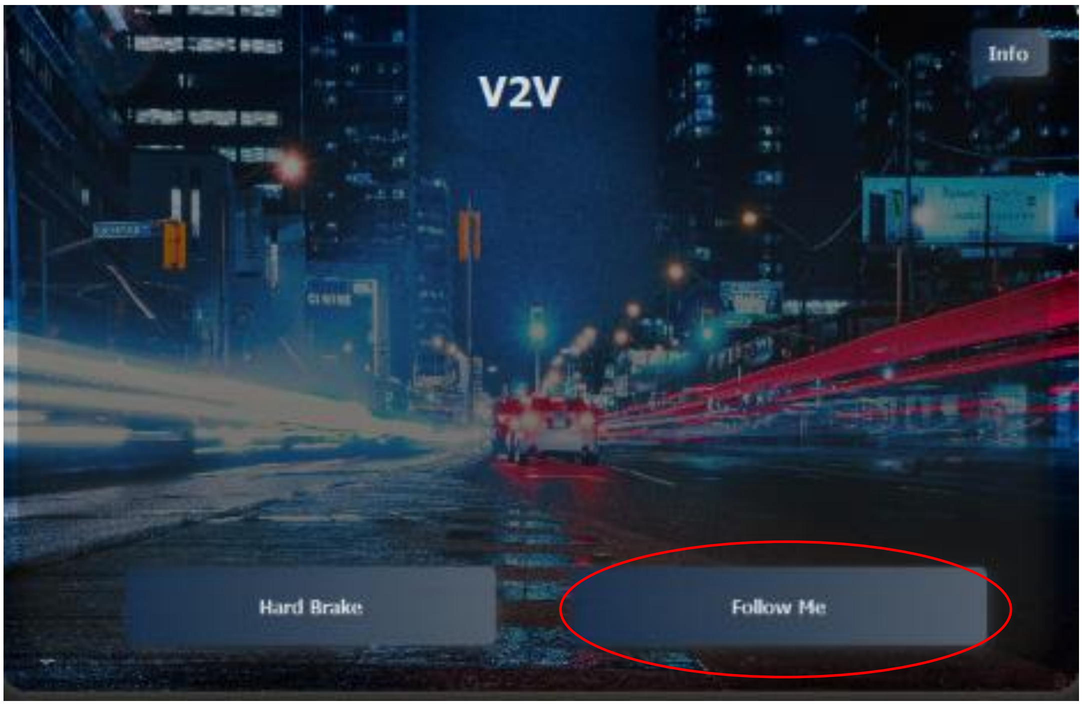
6.2.3 User Interface

In Our system the user will be able to use this application in such an easy way.

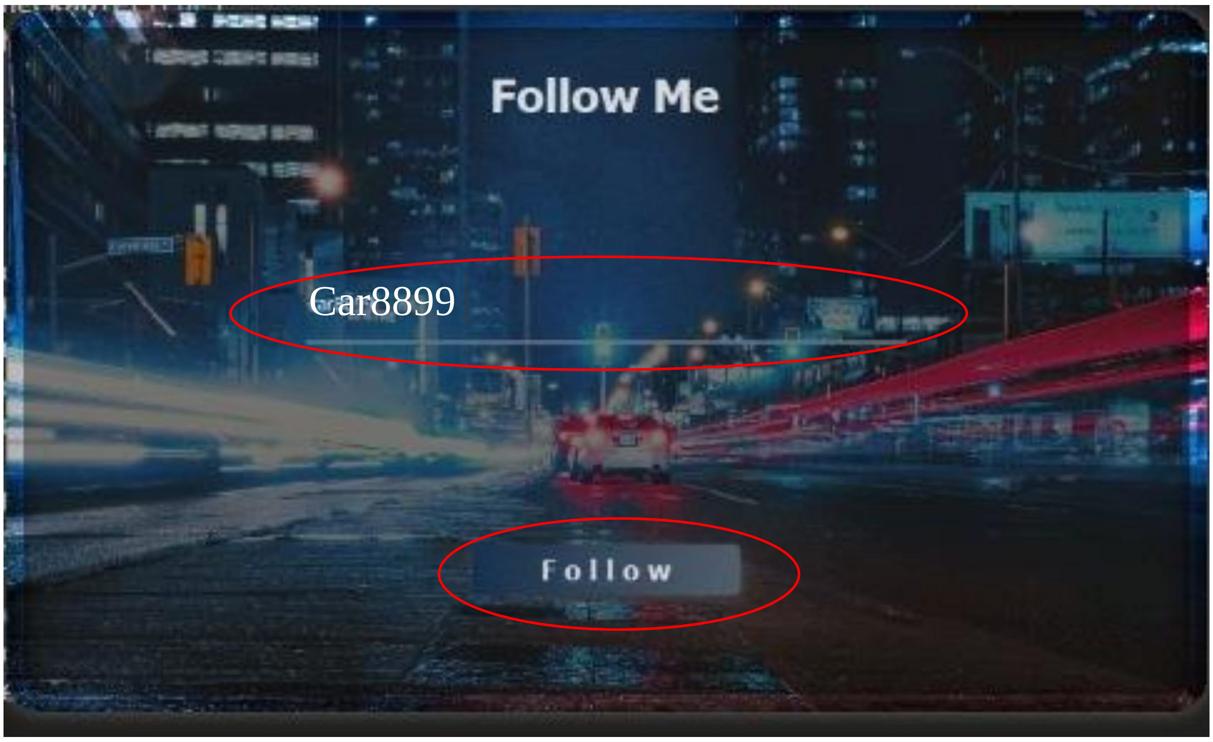
Firstly, the following and the leader users should connect their cars with the server to be in contact with all other cars from the button shown in the system starting window.



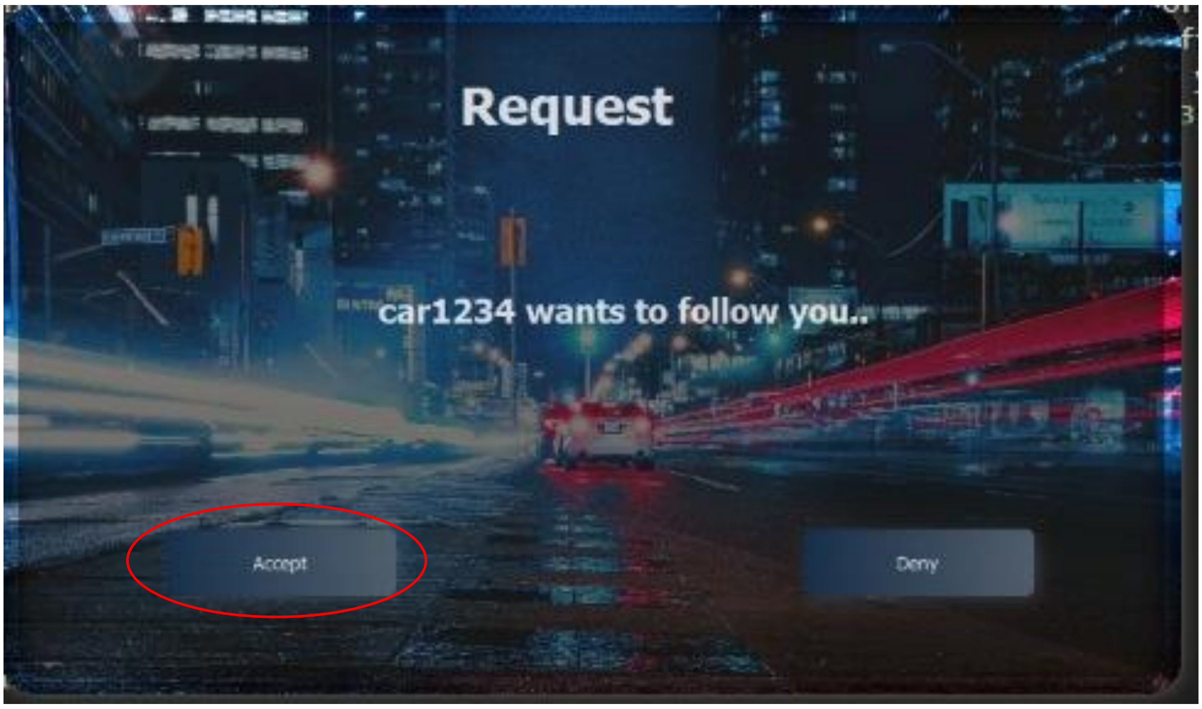
Then, the system will move to another window that has a “Follow Me” button which will make the following user able to use this application.



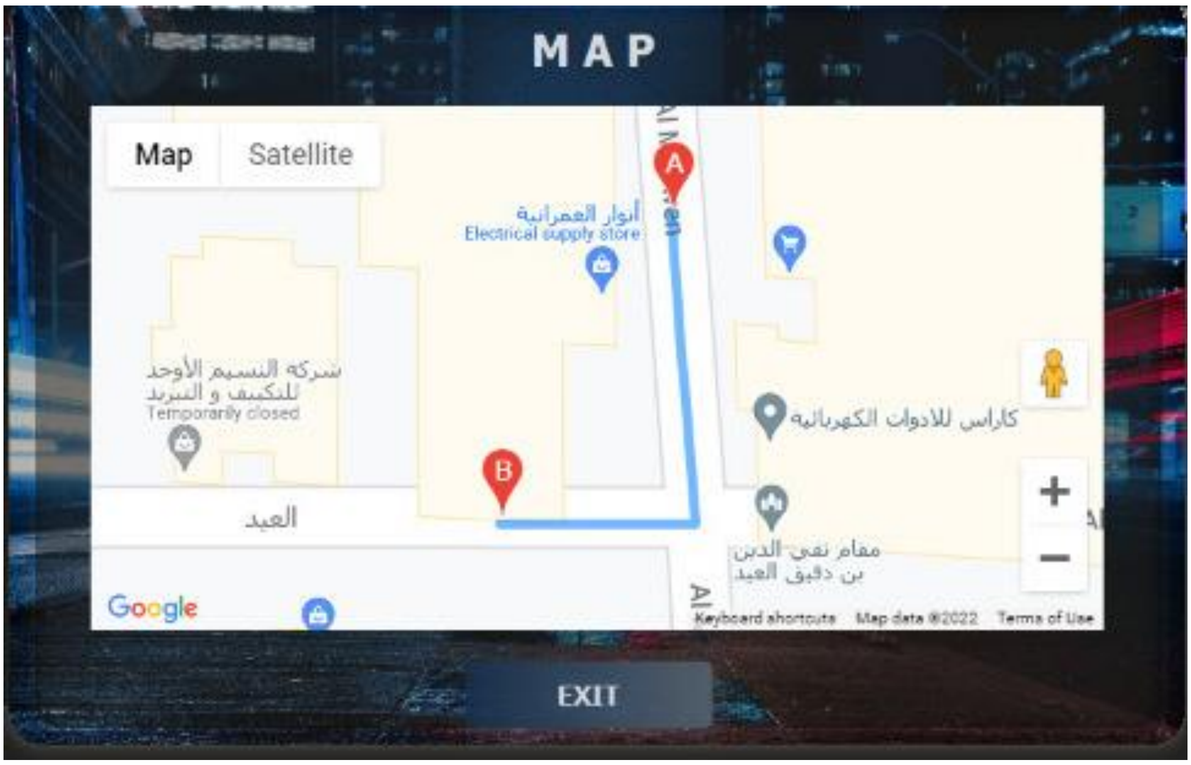
After that, The following user will enter the ID or username of the car he wants to follow (leader Car) and press in the "Follow" button.



Then, The leader car will get a follow request that there is a car wants to follow him and gives him its ID or username, the user of the leader car has to accept this request to make the following user able to know its location up-to-date.



Finally, After the leader car's user accepts the "Follow me" request, the system will open website have a google map of the leader location marked as leader and your location, then you can follow him by google routing option.



6.3 Application 3: Accident Detection Warning

6.3.1 Description

Accident detection warning application will help in reducing the traffic jams due the occurrence of an accident which leads to roads blocking for several hours, and once you entered the road that has the accident we’re blocked in this road and can’t change your direction to another non crowded road.

So, our system helps the driver to not enter the road path that has any accident by notifying the driver with warning message that there is an accident in his area and sends the location of the accident so he can take another path to go where ever he wants without any loss of time.

That happens when a vehicle that already entered the accident road, it will detect the accident at once and then the system send warning massages to help all other system users in this area to not enter this road, so that will help in saving time, money and ease traffic.

6.3.2 Implementation

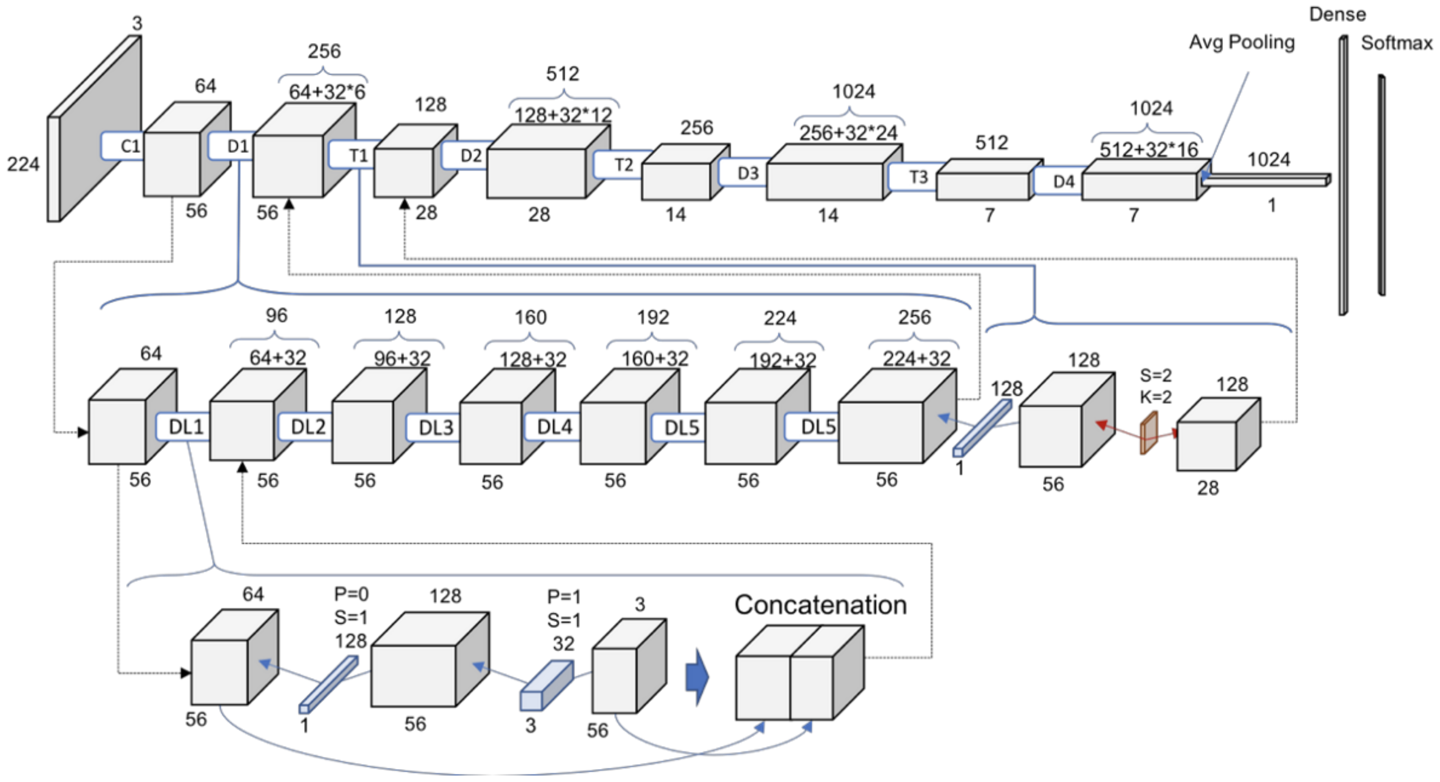
Our application mainly tools: GPS, Camera and a machine learning model.

The camera will capture the road in front of the vehicle and sends the video to the machine learning model this model continuously detects if there is an accident on the road or the road is in a normal situation and send the results to the vehicle system, Once the system gets an output from the model that there is an accident it will send a warning message to the vehicles in the same area, with the help of the GPS to know the location of the accident which will be the same location of the vehicle that sends the warning, the GPS also help to make sure that the message will be received by the vehicles within the same area only and not disturb the vehicles out of range with these message.

Model Overview

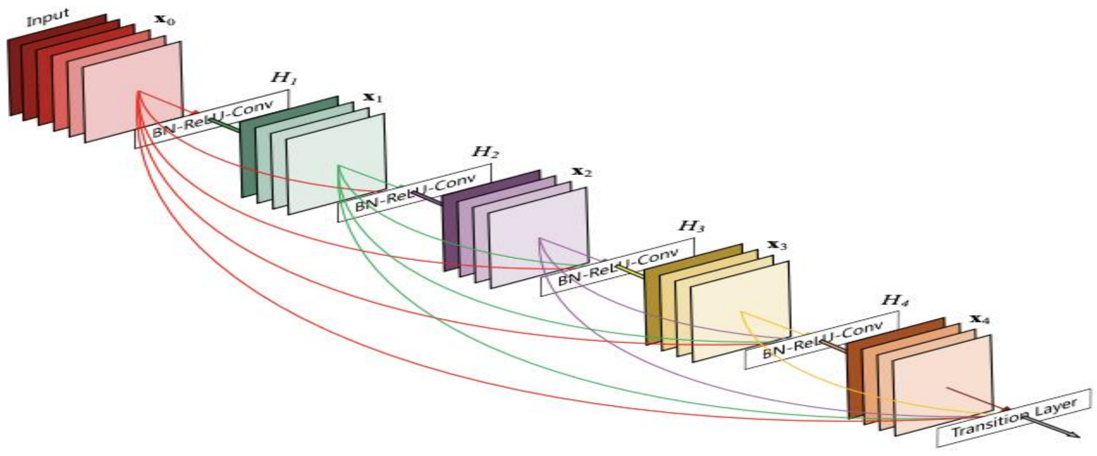
For this project we have tweaked Densenet-161 architecture.

Dense Convolutional Network (DenseNet), connects each layer to every other layer in a feed-forward fashion. Whereas traditional convolutional networks with L layers have L connections - one between each layer and its subsequent layer - our network has L(L+1)/2 direct connections. For each layer, the feature-maps of all preceding layers are used as inputs, and its own feature-maps are used as inputs into all subsequent layers. DenseNets have several compelling advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.



The nest figure shows a 5-layer dense block with a growth rate of k = 4. Each layer takes all preceding feature-maps as input.

An Architecture that distills this insight into a simple connectivity pattern: to ensure maximum information flow between layers in the network, we connect all layers (with matching feature-map sizes) directly with each other. To preserve the feed-forward nature, each layer obtains additional inputs from all preceding layers and passes on its own feature-maps to all subsequent layers.



In the following table: DenseNet architectures for ImageNet. The growth rate for all the networks is k = 32. Note that each "conv" layer shown in the table corresponds the sequence BN-ReLU-Conv.

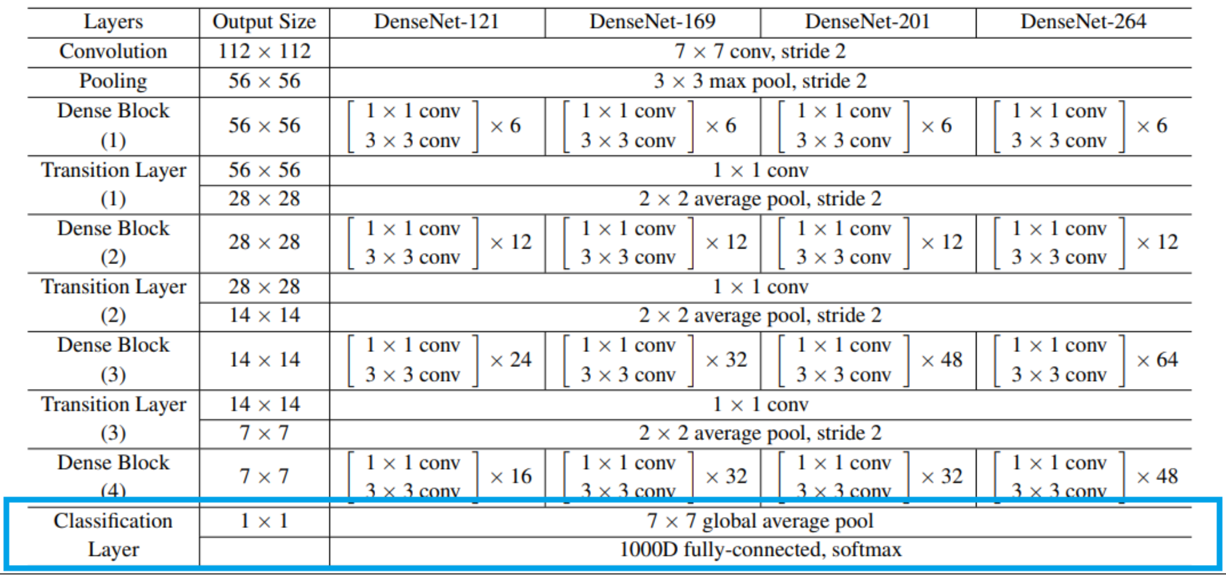
Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Obtain state-of-the-art results on the datasets that we tested on. One explanation for this is that each layer has access to all the preceding feature-maps in its block and, therefore, to the network's "collective knowledge". One can view the feature-maps as the global state of the network. Each layer adds k feature-maps of its own to this state. The growth rate regulates how much new information each layer contributes to the global state. The global state, once written, can be accessed from everywhere within the network and, unlike in traditional network architectures, there is no need to replicate it from layer to layer.

Transfer Learning for our model

It is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.

It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems.



Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112×112	7×7 conv, stride 2			
Pooling	56×56	3×3 max pool, stride 2			
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56×56	1×1 conv			
	28×28	2×2 average pool, stride 2			
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28×28	1×1 conv			
	14×14	2×2 average pool, stride 2			
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14×14	1×1 conv			
	7×7	2×2 average pool, stride 2			
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1×1	7×7 global average pool			
		1000D fully-connected, softmax			

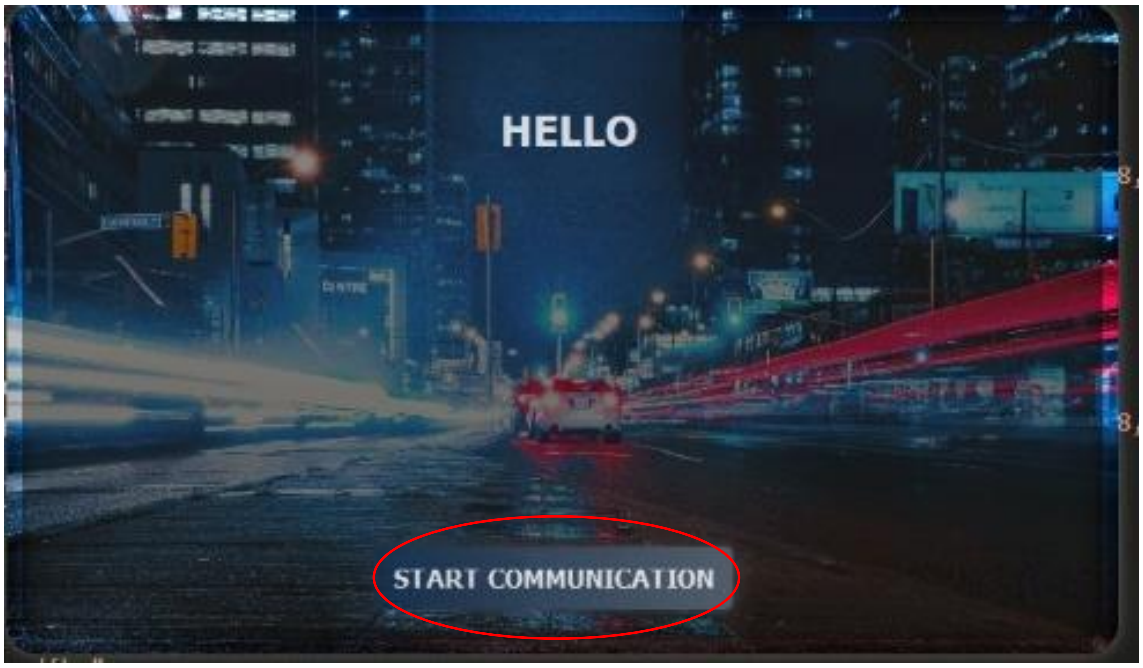
We will replace the last layer like in the table by our layer contains our data that we want to train on it.

```
model.classifier = nn.Sequential(nn.Linear(2208, 1000),
                                 nn.ReLU(), #enable elnetwork to learn during backword propagation
                                 nn.Dropout(0.2),
                                 nn.Linear(1000, 2),
                                 nn.LogSoftmax(dim=1))
```

6.3.3 User Interface

In Our system the user will be able to use this application in such an easy way.

Firstly, the user should connect their car with the server to be in contact with all other cars from the button shown in the system starting window.



Now, the accident detection warning application is automatically enabled once the system is connected to the server so whenever there is an accident in the vehicle's area the system will warn the driver and send to him the location of the accident.

References

[1] Hu, F. (2018). Vehicle-to-vehicle and vehicle-to-infrastructure communications.

[2] CAR 2 CAR Communication Consortium. Car-2-car.org. (2022). Retrieved 25 Jan 2022, from https://www.car-2-car.org/.

[3] Dang, Ruina & Ding, Jieyun & Su, Bo & Yao, Qichang & Tian, Yuanmu & Li, Keqiang. (2014). A lane change warning system based on V2V communication. 2014 17th IEEE International Conference on Intelligent Transportation Systems.
From
https://www.researchgate.net/publication/283113807_A_lane_change_warning_system_based_on_V2V_communication

[4] Bazzi, Alessandro & Masini, Barbara. (2011). Taking advantage of V2V communications for traffic management.
From
https://www.researchgate.net/publication/234126603_Taking_advantage_of_V2V_communications_for_traffic_management

[5] Hosny, Ahmed & Yousef, Mohamed & Gamil, Wessam & Badwi, Mohamed & Darweesh, M.. (2019). Demonstration of Forward Collision Avoidance Algorithm Based on V2V Communication.
From
https://www.researchgate.net/publication/333925720_Demonstration_of_Forward_Collision_Avoidance_Algorithm_Based_on_V2V_Communication

[6] Zorkany, M. & Yasser, Ahmed & Galal, Ahmed. (2020). Vehicle To Vehicle "V2V" Communication: Scope, Importance, Challenges, Research Directions and Future. The Open Transportation Journal.
From
https://www.researchgate.net/publication/342051269_Vehicle_To_Vehicle_V2V_Communication_Scope_Importance_Challenges_Research_Directions_and_Future

[7] Halfacree, G. (2020). The official Raspberry Pi beginner's guide (4th ed.).

[8] Ltd, R. (2022). Raspberry Pi 4 Model B specifications – Raspberry Pi. Raspberry Pi. Retrieved 1 July 2022,
from https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/.

[9] Python, R. (2022). Socket Programming in Python (Guide) – Real Python. Realpython.com. Retrieved 11 April 2022,
from https://realpython.com/python-sockets/.

[10] Python, R. (2022). Python and PyQt – Real Python. Realpython.com. Retrieved 30 May 2022,
from https://realpython.com/python-pyqt-gui-calculator/

[11] NEO-6 seriesVersatile u-blox 6 GPS modules – u-box.com. Retrieved 1 July 2022,
from https://www.u-blox.com/en/product/neo-6-series

[12] u-blox 6 GPS ModulesData Sheet– u-box.com. Retrieved 1 July 2022,
from https://www.u-blox.com/en/product/neo-6-series

[13] P. Chetprayoon, F. Takahashi and Y. Uchida, "Prediction of Lane Number Using Results From Lane Detection," 2020 IEEE 9th Global Conference on Consumer Electronics (GCCE), 2020,
from https://ieeexplore.ieee.org/document/9291808

[14] Pasha, S. (2016). Thingspeak Based Sensing and Monitoring System for IoT with Matlab Analysis. (pp. 19-23). Retrieved 7 June 2022,
from https://www.ijntr.org/download_data/IJNTR02060018.pdf.

[15] Morley, S., Sullivan, J., Carve, M., & Kippen, R. (2017). Energetic particle data from the global positioning system constellation. (pp. 283-289) United States. Dept. of Energy.

[16] Maureira, M.A. (2014). ThingSpeak – an API and Web Service for the Internet of Things.
from

[17] W. Chang, L. Chen and K. Su. DeepCrash: A Deep Learning-Based Internet of Vehicles System for Head-On and Single-Vehicle Accident Detection With Emergency Notification. in IEEE Access. (vol. 7)

[18] N. Agarwal et al. (2021). Camera-based Smart Traffic State Detection in India using Deep Learning Models. 2021 International Conference on COMmunication Systems & NETworkS (COMSNETS)

[19] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang. Adanet: Adaptive structural learning of artificial neural networks.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. FeiFei. Imagenet: A large-scale hierarchical image database. In CVPR, 2009.