EMERGENCY BRAKING SYSTEM

*A Project Report Submitted by*

**Keshav Vinayak Shanbhag**

**ABSTRACT**

Traffic accidents in India are a major source of deaths, injuries and property damage every year. According to Economic Times, more than 150,000 people are killed each year in traffic accidents in India. That’s about 400 fatalities a day and far higher than developed auto markets like the US. Various national and international researchers have found that the most common behavior of road drivers, which leads to accident, includes: over speeding, drunken driving, distractions to driver, red light jumping, avoiding safety gears like seat belts, and non-adherence to lane driving and overtaking in a wrong manner.

According to Economic Times, speeding caused almost 67 per cent of road accidents, which indicates that a special attention is required by manufacturers to incorporate a system that regulates speed. Globally, manufacturers do not usually add such safety systems until and unless they were forced to do so by mandatory government regulations.

The proposed project aims to design and develop an automatic braking system to minimize the chances of any accidents due to over speeding or any of above reasons. The proposed system regulates speed through automatic braking in cases where a driver fails to apply breaks at the right time in order to prevent or minimizes the accidents. The proposed system uses Raspberry PI as main component for controlling a vehicle. The system also uses sensors to detect any disaster within the vicinity and issue a warning to limit the speed to a safe level. However, the system automatically regulates speed when the speed is not limited to safe level. Additional features like sharing accidents locations with the nearest police stations through email or sms.

|  |  |
| --- | --- |
| **TABLE OF CONTENTS** |  |
| **Contents** | **Page** |
| Title Page | i |
| Abstract | ii |
| Table of contents | iii |
| List of figures | vi |
| CHAPTER 1 INTRODUCTION | 1-5 |
| 1.1 Overview | 3 |
| 1.2 Problem Statement | 4 |
| 1.3 Study Area | 4 |
| 1.4 Objectives | 4 |
| 1.5 Methodology | 4 |
| 1.6 Organization of the Report | 5 |
| CHAPTER 2 LITERATURE SURVEY | 7-9 |
| 2.1 Existing System | 7-8 |
| 2.2 Proposed System | 9 |
| CHAPTER 3 SYSTEM ANALYSIS & REQUIREMENTS | 10-19 |
| 3.1 System Analysis | 10 |
| 3.1.1 Relevance of Platform | 10 |
| 3.1.2 Relevance of Programming Language | 10 |
| 3.2 Requirement Analysis | 11-12 |
| 3.2.1 Scope & Boundary | 11 |
| 3.2.2 User Objective | 11 |
| 3.2.3 Inputs and Outputs | 11 |
| 3.2.3 Assumption & Dependencies | 12 |
| 3.2.4 General Constraints | 12 |

* 1. Functional Requirements
     1. Hardware Requirement
     2. Software Requirement
  2. Non-Functional Requirement

CHAPTER 4 SOFTWARE APPROACH

* 1. About VNC Viewer
  2. Android Blynk

CHAPTER 5 SYSTEM DESIGN

* 1. High Level Design Architecture
  2. Low Level Design Architecture
     1. Use Case Diagram
     2. Sequence Diagram
     3. Activity Diagram
     4. Flow Diagram

CHAPTER 6 SYSTEM IMPLEMENTATION

* 1. Design of User Interface
  2. MODULES

CHAPTER 7 SYSTEM TESTING

* 1. Introduction
  2. Unit Testing
  3. Integration Testing
  4. Test Cases

13-18

13

13

19

20-21

20

21

22-29

22-24

25-29

25

26

27

28-29

30-33

30

31-33

34-36

34

35

35

36

CHAPTER 8 RESULTS & DISCUSSIONS 37-40

[8.1 Screenshots](#_TOC_250004)

[CHAPTER 9 CONCLUSION AND FUTURE WORK 41-42](#_TOC_250003)

* 1. [Conclusion 41](#_TOC_250002)
  2. [Future Work 42](#_TOC_250001)

[REFERENCES 43](#_TOC_250000)

v

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| Figure 1.5.1 | Block Diagram of the System | 5 |
| Figure 3.3.1.1 | Raspberry PI 3b | 14 |
| Figure 3.3.1.2 | HC-SR04 Ultrasonic Sensor | 15 |
| Figure 3.3.1.3 | Motor | 16 |
| Figure 3.3.1.4 | IC L293D | 17 |
| Figure 5.1.1 | Raspberry pi and ultrasonic sensor circuit diagram | 24 |
| Figure 5.1.2 | LED Circuit connected to Raspberry PI | 24 |
| Figure 5.2.1 | Use Case Diagram | 25 |
| Figure. 5.2.2 | Sequence Diagram | 26 |
| Figure 5.2.3 | Activity Diagram | 27 |
| Figure 5.2.4 | Flow chart of the system | 29 |
| Figure 8.1.1 | VNC Viewer | 37 |
| Figure 8.1.2 | System Code stored in Raspberry Pi | 37 |
| Figure 8.1.3 | Console of Raspberry Pi | 38 |
| Figure 8.1.4 | Initialization Of Blynk Application | 38 |
| Figure 8.1.5 | Vehicle with Emergency Braking System | 39 |
| Figure 8.1.6 | Blynk UI for the driver | 39 |
| Figure 8.1.7 | Email Notifications | 40 |
| Figure 8.1.8 | Message Notifications | 40 |

vi

# CHAPTER 1 INTRODUCTION

Driving is a common activity for most of the people. The number of vehicles is increasing day by day. A traffic collision, also called a motor vehicle collision (MVC) among other terms, occurs when a vehicle collides with another vehicle, pedestrian, animal, road debris, or other stationary obstruction, such as a tree, pole or building. Traffic collisions often result in injury, death, and property damage.

A number of factors contribute to the risk of collision, including vehicle design, speed of operation, road design, road environment, and driver skill, impairment due to alcohol or drugs, and behavior, notably speeding and street racing.

A 1985 study by K. Rumar, using British and American crash reports as data, suggested 57% of crashes were due solely to driver factors, 27% to combined roadway and driver factors, 6% to combined vehicle and driver factors, 3% solely to roadway factors, 3% to combined roadway, driver, and vehicle factors, 2% solely to vehicle factors, and 1% to combined roadway and vehicle factors. Reducing the severity of injury in crashes is more important than reducing incidence and ranking incidence by broad categories of causes is misleading regarding severe injury reduction. Vehicle and road modifications are generally more effective than behavioral change efforts with the exception of certain laws such as required use of seat belts, motorcycle helmets and graduated licensing of teenagers.

In 2013, 54 million people worldwide sustained injuries from traffic collisions. This resulted in 1.4 million deaths in 2013, up from 1.1 million deaths in 1990. About 68,000 of these occurred in children less than five years old.

Almost all high-income countries have decreasing death rates, while the majority of low- income countries have increasing death rates due to traffic collisions. Middle-income countries have the highest rate with 20 deaths per 100,000 inhabitants, accounting for 80% of all road fatalities with 52% of all vehicles. While the death rate in Africa is the highest (24.1 per 100,000 inhabitants), the lowest rate is to be found in Europe (10.3 per 100,000 inhabitants).

Now days, the technology has got vast changes which leads increase in speed. The speed plays a vital role to maintain time for longer distances. But, this speed also getting a major problem for causes of road accidents. The common braking is not sufficient for avoidance of accidents when driver is not active. Further improvement has to done in braking system in order to brake a vehicle when driver is not able to brake i.e., it may needs automatic braking system.

This automatic braking system allows the vehicle to brake without support of the driver. The main target of the autonomous braking system is that, vehicles should automatically brake when the sensors sense the obstacle. This is a technology for Automobiles to sense an imminent forward collision with another vehicle or an obstacle, and to break the vehicle accordingly, which is done by the braking circuit. This system includes two ultrasonic sensors viz. ultrasonic wave emitter and ultrasonic wave receiver. The ultrasonic wave emitter provided in front portion of an automatic braking system vehicle, producing and emitting ultrasonic waves in a predetermined distance in front of the vehicle. Ultrasonic wave receiver is also provided in front portion of the vehicle, receiving the reflected ultrasonic wave signal from the obstacle. The reflected wave (detection pulse) is measured to get the distance between vehicle and the obstacle.

The DC gear motor is connected to the wheels of vehicle and power input is given to it from Raspberry pi. Then PIC microcontroller is used to control the servo motor based on detection pulse information and the servo motor in turn automatically controls the

braking of the vehicle. Thus, this new system is designed to solve the problem where drivers may not be able to brake manually exactly at the required time, but the vehicle can stop automatically by sensing the obstacles to avoid an accident. In order to reduce the emission levels, more work is going on for the modification of engine work functions and all. There are several kinds of braking mechanism systems that would only can be applicable mechanically, to move the ideology more deep and brief the automatic braking system will be more sufficient and satisfactory in addition to mechanical braking system. In present generation, number of vehicles is coming into existence with newer technologies for implementation of human comfort and other conditioning. To extend the ideology in briefer manner and to take the step in different way, may automatic braking system would fulfill the methods of extension of technical existences.

In March 2016, the **National Highway Traffic Safety Administration** (NHTSA) and the **Insurance Institute for Highway Safety** announced the manufacturers of 99% of

U.S. automobiles had agreed to include automatic emergency braking systems as standard on virtually all new cars sold in the U.S. by 2022.

In Europe, there was a related agreement about **advanced emergency braking system** (AEBS) or **autonomous emergency braking** (AEB) in 2012.**United Nations Economic Commission for Europe** (UNECE) has announced that this kind of system will become mandatory for new heavy vehicles starting in 2015.NHTSA projected that the ensuing accelerated rollout of automatic emergency braking would prevent an estimated 28,000 collisions and 12,000 injuries. In India, Autonomous Emergency Braking system (AEB) could become mandatory on new cars by 2022.

### OVERVIEW

The primary use of automatic brakes is in pre crash and accident avoidance systems. The system is typically capable of warning the driver of an impending collision, tightening seatbelts, and taking other actions that can help prevent an accident or reduce the damage that occurs during a collision. It also sends exact location where the

proposed system implemented via email or message to the user. It also has feature of user controlling the system using application to control or monitor vehicle and get information about how the system is working.

### PROBLEM STATEMENT

The Problem is to design and develop an autonomous system that detects the far object on its way and helps to stop the vehicle avoiding accidents. It also helps to locate the position of the vehicle exactly where the proposed project implemented and also user controlling the system using application and get location through mail or message.

### STUDY AREA

Our area of studies includes Python, NOOBS OS.

### OBJECTIVE

The project has aimed to access systems based on:

1. To design and develop the vehicle with emergency braking system for avoiding collision.
2. To sense the obstacle in the environment using ultrasonic sensor and send as input to Raspberry Pi.

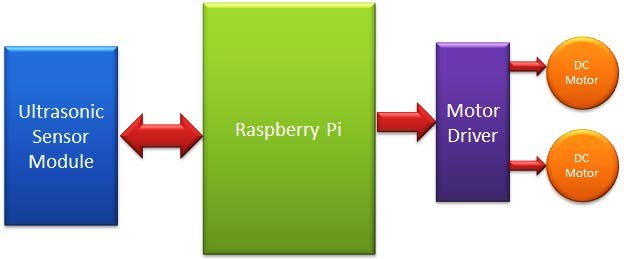
.

1. To send mail and message about location of the vehicle to the nearby Police station where the accident happened.

### METHODOLOGY

The main purpose of this project is to design the automatic braking system in order to avoid the accident. To develop a safety vehicle braking system using ultrasonic sensor and to design a vehicle with less human attention to the driving. This project is

necessary to be attached to every vehicle. By using this project the vehicle is stopped by automatic braking system. So we can avoid the accident. Raspberry PI as the main component of the system takes inputs and output and regulates the vehicle speed. Ultrasonic sensors detect the far object and sends these object details as input signals to the Raspberry Pi. As it receives the input of sensors, it will stop the vehicle by controlling the motors to stop at desired position at a certain distance from the obstacle. This way, the system helps the user to get out from collision and also it sends the message of location about the implementation of the system happened exactly at what location to the Police Station .It also has the feature of controlling the system through an application which controls the vehicle at user desired level and functions are known easily with the help of an application.



### Figure 1.5.1 Block Diagram of the System

* 1. **ORGANIZATION OF REPORT**

This Report is divided up into chapters, each dealing with different aspects of the project. Each chapter has a short introduction, explaining the subject of each chapter, and then the details each module is explained separately, and the following is a short overview of each of the chapters:

**Chapter 2** Outlines some of the research made on the project in the beginning. More research was made as this project report was being developed, as new areas have to be investigated. This research is summarized in the various chapters according to the different modules.

**Chapter 3** Specifies the software requirements specification, the existing and the proposed system along with analysis.

**Chapter 4** provides “Software approach” that gives introduction to the software used in the proposed system.

**Chapter 5** summarizes “System Design Workflow” through use case diagrams and sequence diagrams and also gives an outline of the design of the system.

**Chapter 6** describes “System Implementation” by how user can control the proposed system with an application and describes code implementation model-wise step-by-step that has implemented to match design chapter.

**Chapter 7** describes “System Testing” where test cases are checked against the developed results from implementation phase.

**Chapter 8** describes “Results and Discussion” regarding the working of the project and Screenshots of Software systems and UI design of Blynk application.

**Chapter 9** gives the brief “conclusion and future work” with references used by the system.

# CHAPTER 2 LITERATURE SURVEY

This chapter provides various details regarding to the areas of research with respect to the problem statement. It describes the work which has already been done and states the new scope. The scope has been clearly explained and the technology used to obtain this has been mentioned in the chapter.

### EXISTING SYSTEM

Michael Bommes , Adrian Fazekas , Tobias Volkenhoff , Markus Oeser [1] describes video based sensor systems can play a key role in delivering data for better road planning and traffic management. Smart road technologies will largely depend on data quality and quantity in the future. Video based detection systems, being an indispensable part of intelligent traffic systems (ITS), show huge potentials as they do not only offer a flexible way of data acquisition but are also being developed at a huge pace due to recent evolutions in hardware and software technology. In order to give a better understanding on the methods and potentials of this technology, a structured review is presented which not only includes current applications but also shows future use cases by analyzing the techniques of image processing and extrapolating their results to the future requirements of traffic engineering.

ATIBI Mohamed,ATOUF Issam,BOUSSAA Mohamed, BENNIS Abdellatif [2] describes vehicle detection system is based on two algorithms, a descriptor of the image type and a classifier type artificial neuron networks. In order to ensure rapidity in the calculation extracts features by the descriptor the concept of the integral image is used for the representation of the image. The learning of the system is performed on a set of positive images (vehicles) and negative images (non-vehicle), and the test is done on another set of scenes (positive or negative). To address the performance of the proposed system by varying one element among the determining parameters which is

the number of neurons in the hidden layer; the results obtained have shown that the proposed system is a fast and robust vehicle detector.

Manne Sai Sravana, Sudha Natarajanb, Eswar Sai Krishnac, Binsu J Kailathc [3] describes vehicle detection on –road by colour intensity segregation method.It consists of two steps.Firstly details such as pavements or lanes in the image frame are utilized to extract the region of interest. Secondly, a new filter is proposed that utilizes the intensity information to filter the illumination variations, shadows and cluttered backgrounds from the extracted region of interest and detect the vehicles subsequently. The proposed method is evaluated on videos from KITTI vision benchmark suite and on our videos recorded during cloudy and rainy days with variable resolution. The experimental results demonstrate the effectiveness of the proposed vehicle detection method by achieving 90% detection rate and also reduced computational load on hardware, making it suitable for real-time applications.

Bhushan Nemade [4] describes system looks into both day time as well as night time conditions to monitor traffic. Also it provides vehicle classification, traffic density,vehicle count, license plate detection and Incident detection. It combines many existing methods like background subtraction,kalman filter,2-linesalgorithm,headlight detection, license plate detection algorithm The proposed system implements 2-lines algorithm and vehicle classification using kalman filter for day time and headlight based detection for night time which helps in successful tracking of vehicles. The license plate detection uses Edge detection, Gaussian Analysis, Feature extraction and character recognition which makes it robust to detect license plates in both day and night conditions. Median error was reduces to 11% by use of 2-lines algorithm. Vehicle classification using kalman filters gives accuracy of 82%.The proposed system will give median error less than 10% and accuracy of more than 90% in counting and classifying vehicles. The proposed system will be tested on MIT traffic data sets, Media Lab LPR database.

### PROPOSED SYSTEM

We aimed at developing a system for Indian traffic control. Considering low cost, traffic control and population problems, it’s a huge task for us to develop a system for better mankind. First we focused on collision control. We developed a system using raspberry Pi and ultrasonic sensors. The proposed system uses Raspberry PI as main component for controlling a vehicle. The system also uses ultrasonic sensors to detect any obstacle. The Sensors are placed at the front of the car and emits ultrasonic waves. Sensors detect the far obstacles and sends results back to the raspberry PI system.

Before transmitting the ultrasonic wave, transducer is used to generate the ultrasonic waves. The transducer is given a signal to intermittently produce ultrasonic waves. After that the ultrasonic transmitter sends the waves at a predetermined distance frontward. The maximum range for which obstacle can be detected depends on the range of ultrasonic sensors used.

If the ultrasonic wave detects the obstacle, it will produce a reflected wave. An ultrasonic receiver is used for receiving the ultrasonic waves reflected from the obstacle. The received ultrasonic wave is converted into a reception signal with the help of a transducer. The signal is amplified by an amplifier (operational amplifier). The amplified signal is compared with the reference signal, to detect components in amplified signal due to obstacles on the road.

Here, in raspberry PI system, with this input, a program runs and makes motor to get off at the time of collision. In this way, Emergency brake is applied if any obstacles occur in the environment. The next initiative on which we have focused is on location sharing and mailing. Here raspberry PI system maintains another set of program to locate vehicle location and send SMS or mail to the user in case if the vehicle coming in opposite direction without this system, hit the vehicle. The whole system is controlled manually by the driver using the Blynk application.

# CHAPTER 3 SYSTEM ANALYSIS AND REQUIREMENTS

Analysis is the process of breaking a complex topic into smaller to get better understanding of it. Here analysis had been done based on the three aspects: System analysis, Requirement analysis and Functional requirements. System analysis comprises of relevant platform and relevance programming language. Requirement analysis reveals input, output, scope and boundary, user objective, assumptions and constraint. Functional requirements specifies about hardware and software requirements.

### SYSTEM ANALYSIS

Here the analysis of the system is made with respect to relevance platform, Programming languages.

### Relevance of Platform

The System can work with all vehicles provided with high power supply, high capacity bearing, good mileage, high network connection, adapt to different roads/surfaces without friction.

### Relevance of Programming Language

Python is a widely used high-level programming language for general-purpose programming, created by Guido Van Rossum and first released in 1991. This language provides constructs intended to enable writing clear programs on both a small and large scale.

Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming and procedural styles. It has a large and comprehensive standard library.

Python interpreters are available for many operating systems allowing Python code to run on a wide variety of systems. Python is open based software and has a community based development model, as do nearly all of its variant implementations.

### REQUIREMENT ANALYSIS

Requirement analysis explains the scope and boundary of the project, user objectives, input and outputs of the project. It consists of assumptions and constraints.

### Scope and Boundary

The proposed system will meet the needs of the user by providing following features:

1. The user will be able to recognize the obstacle present in the environment.
2. This system work with vehicles of high capacity, high power, frictionless tyres.
3. The user can control the vehicle with an application provided user interfaces to manage the vehicle’s working.
4. The user will get a location updates through email/message if any opponent vehicle crashes with the vehicle with automatic braking system.

### User Objective

Main objective of the project is to recognize the obstacles in the environment. System detects the obstacle and stops the vehicle from crashing .As a user, he can control the vehicle with an application to manage the working of vehicle, to start the vehicle etc. System helps user to get location updates about vehicle’s accurate position about where the system got implemented through message/mail.

### Inputs and Outputs

Any obstacle that is present in the environment that makes disturbance to the way of vehicle which may cause damage to the vehicle in case if it moves in the obstacle direction is given as input. As and when the system detects an obstacle, it should stop the vehicle going in obstacle direction to resist collision happening by controlling the

motor functioning of the rear wheel makes output of the system.

### Assumptions and Dependencies

Assumptions and dependencies are:

1. Surface should be frictionless enough to move the car.
2. Only the vehicle detects the obstacle of front side.
3. Roads should be straight enough to make vehicle to go in linear direction.
4. Vehicle stops after detecting the obstacle. User has to start the vehicle again.
5. Vehicle should have internet connection to send location exactly to the user about where the system got implemented or vehicle got crashed due to opponent crashing.

### General Constraints

General Constraints are:

* + - 1. The Proposed system is limited to frictionless surfaces; system will fail with friction surface.
      2. The Vehicle starts moving in a linear direction and sense obstacles on its way. There should be limitation on roads that has to be straight enough to carry out the detection of obstacle.

### Functional Requirements

Functional requirements consist of two types. Those are software and hardware requirements.

### Hardware Requirements

1. Raspberry Pi 3b
2. HC-SR04 Ultrasonic Sensor
3. Motor
4. IC L293D
5. JioFi

### Software Requirements

1. Operating System: NOOBS OS
2. VNC Viewer
3. Blynk Application

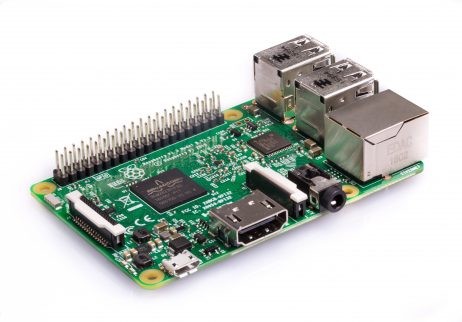
### Hardware Requirements

* + - 1. **Raspberry PI 3b**

The Raspberry Pi is a series of small single computers developed in the United Kingdom by the Raspberry pi foundation to promote teaching of basic computer science in schools and in developing countries. The Raspberry Pi Foundation provides Raspbian, a Debian-based Linux distribution for download, as well as third-party Ubuntu, Windows 10 IOT core, Risc OS, and specialised media center distributions. With an operating system it can work like a computer system.

The Raspberry Pi 3 Model B [5] is the third generation Raspberry Pi. This powerful credit-card sized single board computer can be used for many applications and supersedes the original Raspberry Pi Model B+ and Raspberry Pi 2 Model B. Whilst

maintaining the popular board format the Raspberry Pi 3 Model B brings you a more powerful processer, 10x faster than the first generation Raspberry Pi. Additionally it adds wireless LAN & Bluetooth connectivity making it the ideal solution for powerful connected designs.



### Figure 3.3.1.1 Raspberry PI 3b

* + - 1. **Ultrasonic sensor**

Ultrasonic ranging and detecting devices use high frequency sound waves called ultrasonic waves to detect presence of an object and its range. Normal frequency range of human ear is roughly 20Hz to 20,000Hz. Ultrasonic sound waves are sound waves that are above the range of human ear, and thus have frequency above 20,000Hz. An ultrasonic sensor [6] necessarily consists of a transducer for conversion of one form of

energy to another, a housing enclosing the ultrasonic transducer and an electrical connection. The HC-SR04 Ultrasonic Sensor (or any Ultrasonic Sensor for that matter), works on the principle that is similar to RADAR and SONOR i.e. transmits a signal and analyzes the target by capturing the reflected signals.

These sensors are of two types:

* Ultrasonic Transmitter – Before transmitting the ultrasonic wave, transducer is used to generate the ultrasonic waves. The transducer is given a signal to intermittently produce ultrasonic waves. After that the ultrasonic transmitter sends the waves at a predetermined distance frontward. The maximum range for which obstacle can be detected depends on the range of ultrasonic sensors used.
* Ultrasonic Receiver – If the ultrasonic wave detects the obstacle, it will produce a reflected wave. An ultrasonic receiver is used for receiving the ultrasonic waves reflected from the obstacle. The received ultrasonic wave is converted into a reception signal with the help of a transducer. The signal is amplified by an amplifier (operational amplifier). The amplified signal is compared with the reference signal, to detect components in amplified signal due to obstacles on the road.

### Figure 3.3.1.2 HC-SR04 Ultrasonic Sensor



* + - 1. **Motor**

The output of the Raspberry PI is the input to the motor. The motor [7] allows for precise control of angular position, velocity and acceleration. It consists of a motor coupled to a sensor for position feedback. Thus, it is a closed loop mechanism that uses position feedback to control its motion and final position.

The input to it is a signal, either analog or digital, representing the position commanded for the output shaft. The measured position of the output shaft is compared to the command position. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction as needed, to bring the output shaft to the appropriate position. As the required position approaches, the error signal reduces to zero and the motor stops.

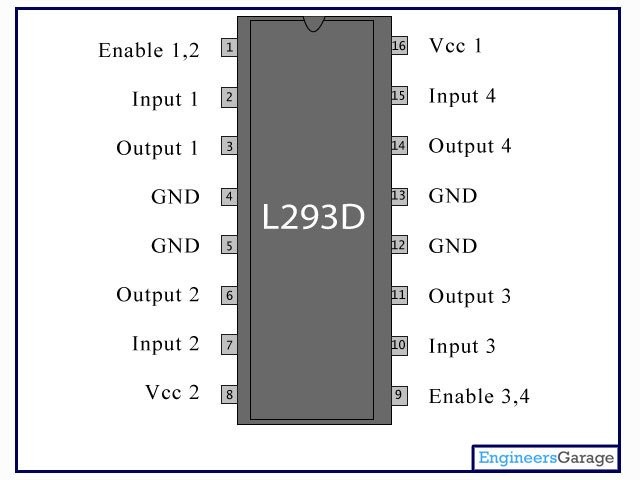
The output shaft of motor is capable of travelling somewhere around 180 degrees. A normal servo motor is used to control an angular motion between 0 and 180 degrees, and it is mechanically not capable of turning any farther due to a mechanical stop built on to the main output gear.

The angle through which the output shaft of the motor need to travel is determined according to the nature of the signal given to the motor as input from the Raspberry PI.

 **Figure 3.3.1.3 Motor**

* + - 1. **lC L293D**

IC-L293D is a dual H-bridge motor driver integrated circuit (IC) [8]. Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors.



### Figure 3.3.1.4 IC L293D

L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 &

15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively

.

Enable pins 1 and 9 (corresponding to the two motors) must be high for motors to start operating. When an enable input is high, the associated driver gets enabled. As a result,

the outputs become active and work in phase with their inputs. Similarly, when the enable input is low, that driver is disabled, and their outputs are off and in the high- impedance state.

* + - 1. **JioFi**

JioFi acts as router to connect Raspberry PI system with nearby computer system. As Raspberry PI itself a system, it requires hardware assistance like keyboard, mouse etc. In order to do this, we are creating a virtual environment in our system that shows raspberry PI system. Plugging all the devices to the Raspberry PI will be cumbersome This way JioFi works as bridge connector to the whole project..

### NON FUNCTIONAL REQUIREMENTS

Non-functional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs. Also known as system qualities, non-functional requirements are just as critical as functional Epics, Capabilities, Features, and Stories. They ensure the usability and effectiveness of the entire system.

Performance:

Better component design to get better performance at peak time. Sensors detect obstacle in fractions as it is ultrasonic sensors, and which helps motor to stop functioning immediately, giving faster output rate.

Reliability:

With sufficient power and internet connection, system will run with no disrupts.

Availability:

System can be made use of at any time hence available 24\*7 with internet connection.

Maintainability:

Maintenance is easy and economical.

Portability:

System can run on any vehicles.

# CHAPTER 4 SOFTWARE APPROACH

### ABOUT VNC VIEWER

VNC (Virtual Network Computing) is a graphical desktop sharing system that allows you to remotely control the desktop interface of one computer (running VNC Server) from another computer or mobile device (running VNC Viewer). VNC Viewer transmits the keyboard and either mouse or touch events to VNC Server, and receives updates to the screen in return. The desktop of the Raspberry Pi can be seen inside a window on computer. We control it as though we were working on the Raspberry Pi itself.VNC Connect from RealVNC is included with Raspbian. It consists of both VNC Server, that controls Raspberry Pi remotely, and VNC Viewer, controls desktop computers remotely from Raspberry Pi.Cloud connections are convenient and encrypted end-to-end. They are highly recommended for connecting Raspberry Pi over the internet. There's no firewall or router reconfiguration, and no need to know the IP address of Raspberry Pi.First we have to sign up for RealVNC new account and use the same credentials to sign in both VNC Server and VNC Viewer.After connecting from the compatible VNC Viewer app from RealVNC, enter the user name and password used to log into account on the Raspberry Pi. By default, these credentials are pi and raspberry.In this way, we can use Raspberry Pi computer virtually to work and run programs using VNC Viewer in user system.

### ABOUT BLYNK

Blynk App allows creating amazing interfaces for projects using various widgets available. After downloading the Blynk App, we created a New Blynk account. After successfully logged into the account, we started creating a new project. We selected hardware model used in our project i.e. Raspberry Pi 3b. Auth Token is a unique identifier which is needed to connect our hardware to Smartphone. We got Auth Token automatically on our email after project creation. Next step is to create the project. First we added one button on/off for motor and led functioning. Button properties have to be set using widget settings. Blynk App allows creating amazing interfaces for projects using various widgets available. Then we have to select the pin to which led is connected for hardware. After running the program in PC, when we click on the button, motor and led start functioning. In this way, user can control the system with help of Blynk application.

# CHAPTER 5 SYSTEM DESIGN

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. They are high level design and low level design as they are explained below.

### HIGH LEVEL DESIGN ARCHITECTURE

The distance of an object using HC-SR04 Ultrasonic Sensor is calculated. In order to send the 40 KHz Ultrasound, the TRIG Pin of the Ultrasonic Sensor must be held HIGH for a minimum duration of 10µS. After this, the Ultrasonic Transmitter, will transmits a burst of 8-pulses of ultrasound at 40 KHz. immediately; the control circuit in the sensor will change the state of the ECHO pin to HIGH. This pin stays HIGH until the ultrasound hits an object and returns to the Ultrasonic Receiver. Based on the Time for which the Echo Pin stays HIGH, you can calculate the distance between the sensor and the object.

The Raspberry Pi needs to read the Echo pin to calculate the time and hence the corresponding GPIO pin on the Raspberry Pi must be configured as Input So, before connecting the Echo Pin to the Raspberry Pi, it must be given to a level converter.

Connect the Trig Pin of the HC-SR04 Ultrasonic Sensor to the Physical Pin 16 i.e. GPIO23 of the Raspberry Pi. Use a combination of 680Ω and 1.5 KΩ Resistor to convert the Echo pin to 3.3V Logic (approximately) and connect it to Physical Pin 18 i.e. GPIO24 of the Raspberry Pi.Finally, provide the +5V and GND connections to the Ultrasonic Sensor from the Raspberry Pi Pins.

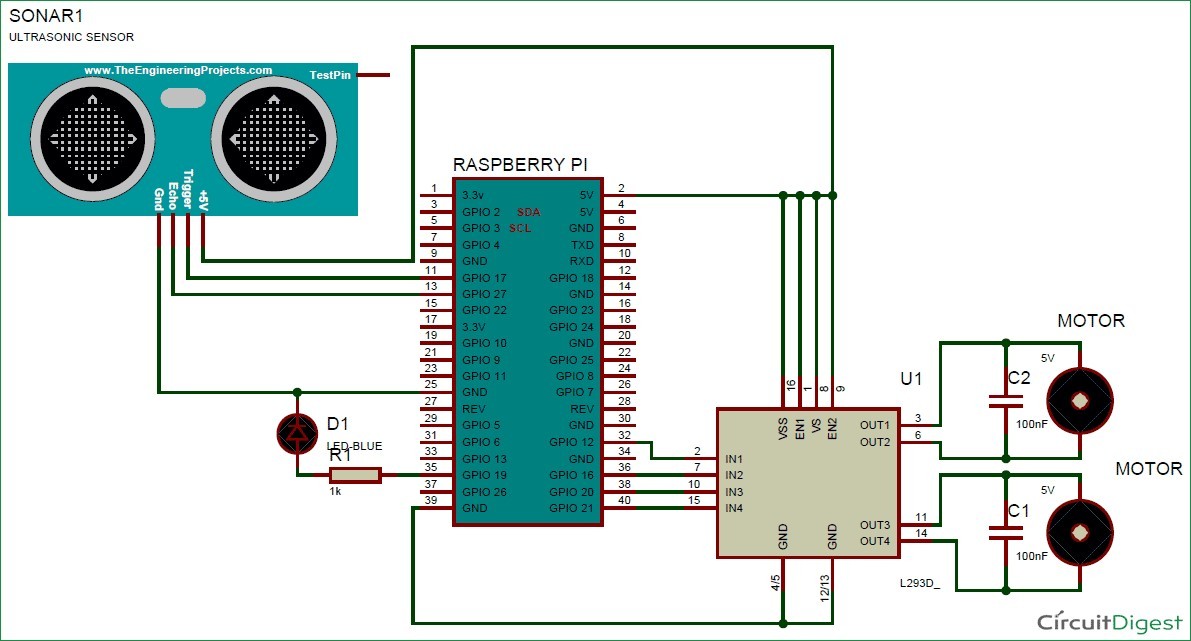
IC L293D is connected to Raspberry PI as shown in the below figure 1.5[9] .For this

,motor is connected to the output pins of IC L293D.When Raspberry PI gives output, motor runs clockwise or anti-clockwise based on the input of sensors. When the system finds an obstacle, it gives output to the motor to stop the working. The code written in python works on input of sensors and gives output to the motor. First code ,finds if any obstacles are there around 50cm .If it finds then stops the motor and the vehicle will come to rest around 15cm from the obstacle. The code makes the vehicle to lower the speed when it detects the obstacle at certain distance.

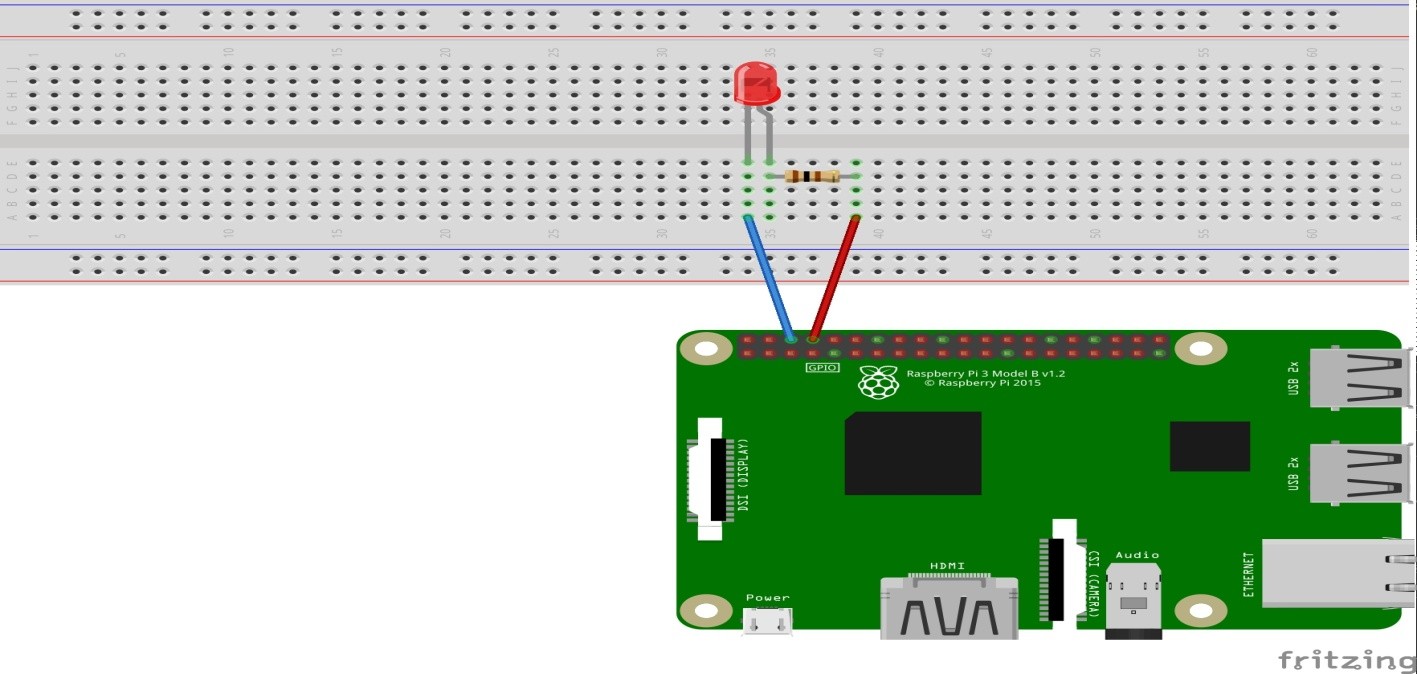
Then we did the LED circuit controllable from the Raspberry Pi by connecting the circuit to the general purpose input/output (GPIO) pins on the Raspberry Pi as shown in figure1.6 [10].

A simple LED circuit consists of a LED and resistor. The resistor is used to limit the current that is being drawn and is called a current limiting resistor. Without the resistor the LED would run at too high of a voltage, resulting in too much current being drawn which in turn would instantly burn the LED, and likely also the GPIO port on the Raspberry Pi.

Here LED bulbs connected in front of the vehicle starts when vehicle starts moving. As vehicle detects obstacle and stops running, suddenly bulbs will stop.



### Figure 5.1.1. Raspberry pi and ultrasonic sensor circuit-diagram



**Figure 5.1.2 LED Circuit connected to Raspberry PI**

### LOW LEVEL DESIGN ARCHITECTURE

The low level design architecture is concerned with architecture that how exactly the concept of high level design has been implemented.

### Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

Use case diagrams are in fact twofold- they are both behavior diagrams, because they describe behavior of the system, and they are also structure diagrams –as a special case of class diagrams where classifiers are restricted to be either actors or use cases related to each other with associations. A use case diagram can portray the different types of users of a system and various ways that they interact with the system as shown in Fig. 1.7.

**Open Blynk Application**

**PRESS “ON” BUTTON**

**TO START THE VEHICLE**

**DRIVE THE VEHICLE**

**PRESS “OFF” BUTTON**

**TO STOP THE VEHICLE**

**CLOSE Blynk Application**

**DRIVER**

### Fig 5.2.1: Use Case Diagram

* + 1. **Sequence Diagrams**

A sequence diagram is a kind of interaction diagram that shows how process operates with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence.

Open Blynk App

Click “On” Button

ON Selected

Check Obstacle

No Obstacle

Start Vehicle

Check Obstacle

Obstacle Found

Stop Vehicle

Check Obstacle

No Obstacle

Start Vehicle Again

Click “OFF” Button

Close Blynk App

OFF Selected

Stop Vehicle

**SENSOR**

**MOTOR**

**MOTOR**

**SENSOR**

**RASPBERRY PI**

**RASPBERRY PI**

**BLYNK APP**

**BLYNK APP**

**DRIVER**

**DRIVER**

### Figure 5.2.2. : Sequence Diagram

* + 1. **Activity Diagrams**

Activity diagrams are graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Fig. 5.2.2 we show the activity diagram for different modules.

**START VEHICLE**

**OBSTACLE DETECTION**

**STOP VEHICLE**

**OPPONENT COLLISION**

**EMERGENCY EMAIL TRANSMISSION**



### Figure 5.2.3.: Activity Diagram



* + 1. **Flow Diagrams**

A **flowchart** is a type of diagram that represents an algorithm, workflow or process. Flowchart can also be defined as a diagrammatic representation of an algorithm (step by step approach to solve a task).

The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.



**Start**

**Initialize Auth-Token**

**Initialize Blynk Button**

**Initialize TRIG, ECHO,AverageDISt=0**

**Initialize m11, m12, m21, m22**



**B**

**If Value==1**

**If ECHO==0**

**M11=0,m12=1,**

**m21=0,m22=1**

**Pulse-Start=time.time()**

**If ECHO==1**

**END**

**Pulse-End=time.time()**

**A**

**Get Button Value**





**If AverageDISt<50**

**AverageDISt=AverageDISt+Distance**



**A**

**Pulse-duration=(Pulse-End)-(Pulse-Start)**

**Distance=Pulse-duration\*17150**

**B**

**M11=0,m12=1,m21=0,**

**m22=1**

**M11=0,m12=1,m21=0,m22=1**

**Figure 5.2.4: Flow chart of the system**

# CHAPTER 6 SYSTEM IMPLEMENTATION

The basic goal in system implementation is to specify the logic for the different modules that have been specified during system design. This chapter gives the implementation details of the entire system.

### Design of User Interface

The Blynk app basically connects to the Blynk server which automates the raspberry pi computer. First user has to log-in by giving the credentials and he will get an Authorization token for future use. The same token can be used to connect the Raspberry Pi and Blynk app. First we have to design a user interface for the driver to automate the vehicle with a single touch. First step is to create a project by selecting the Raspberry Pi 3b and add elements needed for interface. We are using a button (ON/OFF) to start and stop the vehicle. After selecting the button, we have to text labels here ON & OFF. Then we have to initialize the output pins(Here it is V3). After doing all features, we have to run the project. Before running the program, we have to initialize the Blynk app in the raspberry Pi. As we run the code and give input from the application, raspberry pi will get input of button as 1 or 0 which specifies on or off of the system. This will make the vehicle to run or stop the motor functioning. This is the very first step in the system process. This will help the driver to automate the vehicle in one single touch.

### MODULES

The whole project is mainly divided into 6 critical modules based on functionalities of the actors.

### Module 1: Initialization of Libraries

First of all, we include required libraries, initialize variables and define pins for ultrasonic sensor, motor and components.

import requests//to get ubidots to send message import math//to do mathematical calculations import RPi.GPIO as GPIO//to get inputs and ouputs import time//to add sleep function GPIO.setwarnings(False) GPIO.setmode(GPIO.BCM)

TRIG = 17

ECHO = 27

led = 22 m11=16 m12=12 m21=21 m22=20

### Module 2: Initialization of Blynk Application

The first step is to connect between Blynk and raspberry pi with the help of authorize token and pin. Next step is get value of the button. Based on value of button the system will get started or get stopped.

Initialization: export PATH=$PATH:/opt/nodejs/bin

Unset NODE\_PATH

Blynk-client 31ec8d95ccc84ee09eeae4d17eae15e7 We have to import blynkapi for the code to get button values. auth\_token = "31ec8d95ccc84ee09eeae4d17eae15e7"

BlynkValue = Blynk(auth\_token, pin = "V3")

### Module 3: Input from the Ultrasonic sensor

If the BlynkValue is 1 then we will start the function ON().First we have to get the status of TRIG and ECHO of sensor. The purpose of the TRIG pin is to cause the module to send out a "Ping". It's what makes the whole thing work. As soon as the TRIG pin goes HIGH then LOW (for a period of not less than 10µs) the internal clocks start ticking. 8 cycles of 40KHz audio are sent out of the transmitter and it starts counting how long it takes for the echo to arrive. ECHO is used to get ping back as input to the sensor. Both values will help to calculate the Pulse start and Pulse end time.

Code: while GPIO.input(ECHO)==0: GPIO.output(led, False)

pulse\_start = time.time()

while GPIO.input(ECHO)==1: GPIO.output(led, False)

pulse\_end = time.time()

pulse\_duration = pulse\_end - pulse\_start

### Module 4: Average Distance Calculation

After getting pulse start and pulse end time, we have to calculate the average distance of the obstacle.The difference between the pulse end and pulse start time will get us pulse duration.Distance is 17150 times of the pulse duration. Average distance is calculated by average values getting from calculating distance for long time.

Code: pulse\_duration = pulse\_end - pulse\_start distance = pulse\_duration \* 17150 avgDistance=avgDistance+distance

### Module 5: Output to the motors

After calculating the average distance, we have to check for obstacle and stop the vehicle at 0.5 meter distance from the obstacle. If no obstacle found, then vehicle

should start moving and again it has to check for obstacles. In this way vehicle will detect and stop at obstacle appearance without any damages.

Code: if avgDistance < 50: GPIO.output(14,GPIO.LOW)

GPIO.output(m11, 0)

GPIO.output(m12, 0)

GPIO.output(m21, 0)

GPIO.output(m22, 0)

else:

GPIO.output(14,GPIO.HIGH)

GPIO.output(m11, 0)

GPIO.output(m12, 1)

GPIO.output(m21, 0)

GPIO.output(m22, 1)

### Module 6: Notification transmission

With the help of UBIDOTS the system will transfer the E-mail and sms to Police station when the opponent vehicle hits the vehicle. First TOKEN ,DEVICE\_LABEL,Content- Type has to be given in header section. We have to specify url of ubidots. With this credentials, we have to make request to send an alert message via email or sms to the police station.

Code: url = "[http://things.ubidots.com](http://things.ubidots.com/)"

url = "{}/api/v1.6/devices/{}".format(url, DEVICE\_LABEL)

headers = {"X-Auth-Token": TOKEN, "Content-Type": "application/json"} req = requests.post(url=url, headers=headers, json=payload)

# CHAPTER 7 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and should require no knowledge of inner design of code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system. The purpose of integration testing is to detect any unpredictability between the software units that are integrated together. System testing is performed on the entire system in the context of a Functional Requirement Specification and a System Requirement Specification.

### INTRODUCTION

Software testing is an examination conducted to provide stakeholders with information about the quality of the product. Software testing can also provide an aim, independent view of the software to allow the business to recognize and understand the risks of software implementation. Test techniques include application with the intent of finding software errors. Software testing involves the execution of a software component to evaluate many properties of interest. The testing phase is performed after the coding to detect all the errors and provide quality assurance and ensure reliability of the software. It is an important stage in the System development life cycle the different testing strategies employed in this project are explained in this chapter.

### UNIT TESTING

Unit testing involves only those characteristics that are vital to the performance of the unit under test. This encourages developers to modify the source code without immediate concerns about how such changes might affect the functioning of other units or the program as a whole. Once all of the units in a program have been found to be working in the efficient-manner and error-free manner possible, larger components of the program can be evaluated by means of integration testing. Unit testing can be time- consuming and tedious. Rigorous documentation must be maintained. Unit testing must be done with an awareness that it may not be possible to test a unit for every input that will occur when the program is run in a real-world environment. The goal of unit testing is to isolate each part of the program first and then testing the sum of its parts, integration testing becomes much easier. In our project, we apply this by testing the various modules of the application and also each feature individually.

### INTEGRATION TESTING

Integration Testing is three phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before system testing. Integration testing takes as its input module that have been tested groups them in larger aggregates, applies tests defined in an integration test plan to those aggregate and delivers as its output the integrated system ready for the system testing.

* 1. **TEST CASES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CASES | CASE DESCRIPTION | PROCEDURE | EXPECTED  RESULTS | COMMENT |
| 1. | Determining an  Obstacle | Place an object in front of vehicle. | The vehicle will stop 50 cm ahead  of the object. | SUCCESS |
| 2. | Move forward | Remove the object. | The vehicle will  automatically start moving. | SUCCESS |
| 3. | Email | Vehicle gets hit by an obstacle. | EBS will send an Email to nearby  Police Station. | SUCCESS |
| 4. | Stop vehicle | If any obstacle comes in between  the vehicle's route. | Vehicle will stop. | SUCCESS |
| 5. | Sensor Detection | Placing an object in front of vehicle.  Sensors will send signal and receive  it back. | Sensors will send the signal to EBS and stop the motor. | SUCCESS |
| 6. | LED | Start the vehicle | LED will glow. | SUCCESS |
| 7. | LED | Stop the vehicle | LED will not glow. | SUCCESS |
| 8. | Message | Vehicle gets hit by an obstacle. | EBS will send an Message to nearby  Police Station. | SUCCESS |

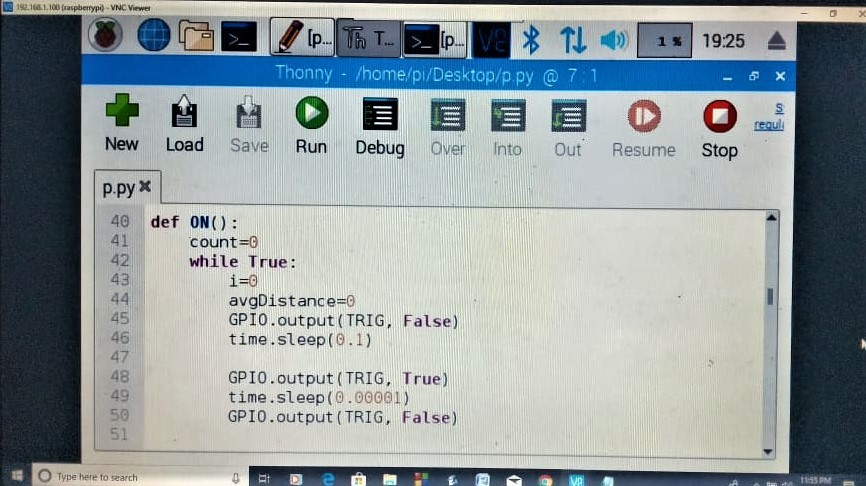
# CHAPTER 8 RESULTS AND DISCUSSIONS

## 8.1: Screenshots:

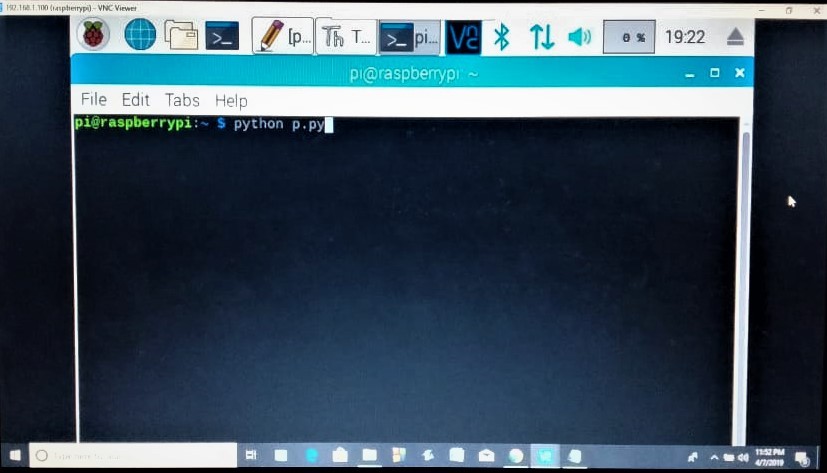
### Figure 8.1.1: VNC Viewer



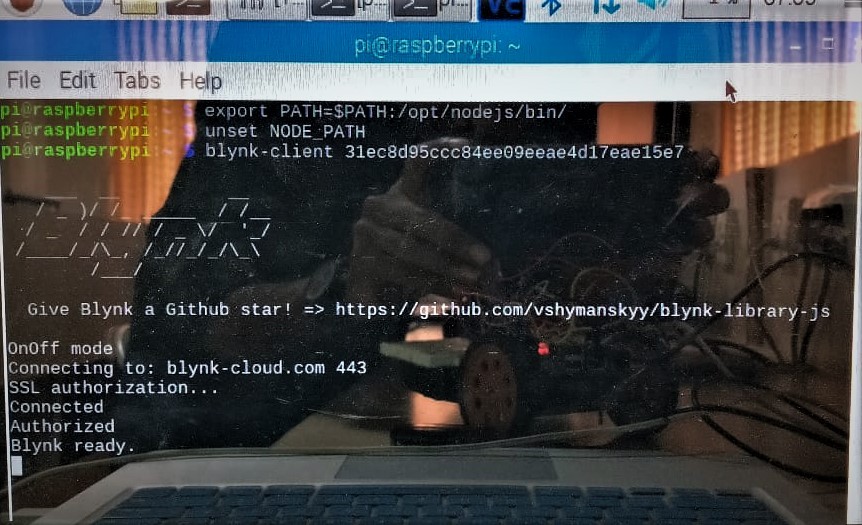
**Figure 8.1.2: System Code stored in Raspberry Pi**



### Figure 8.1.3: Console of Raspberry Pi



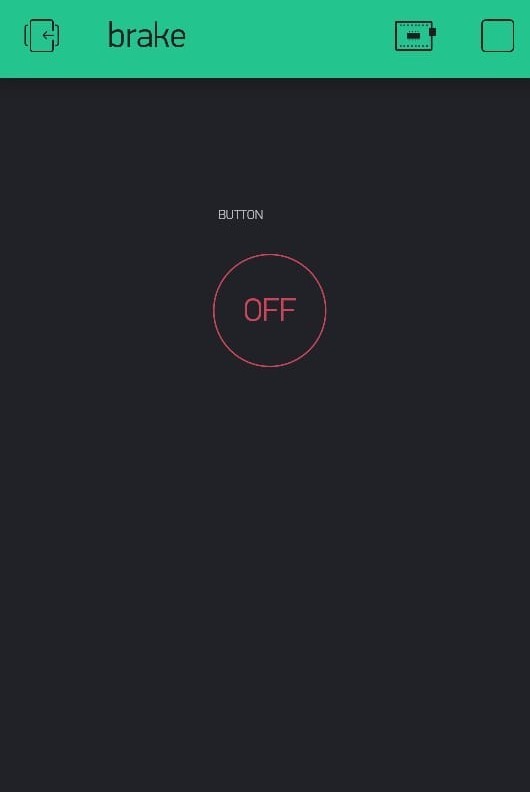
**Figure 8.1.4: Initialization of Blynk Application**



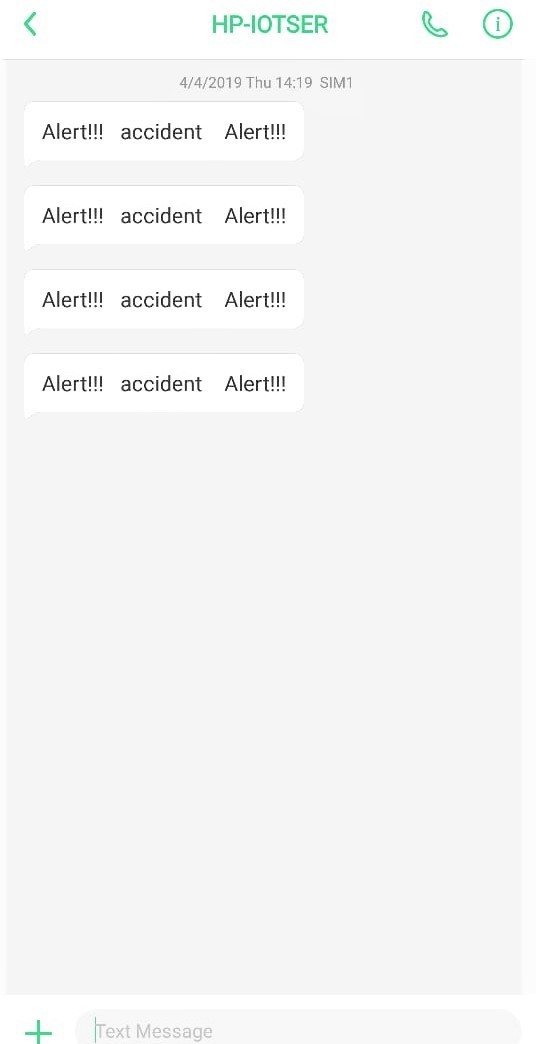
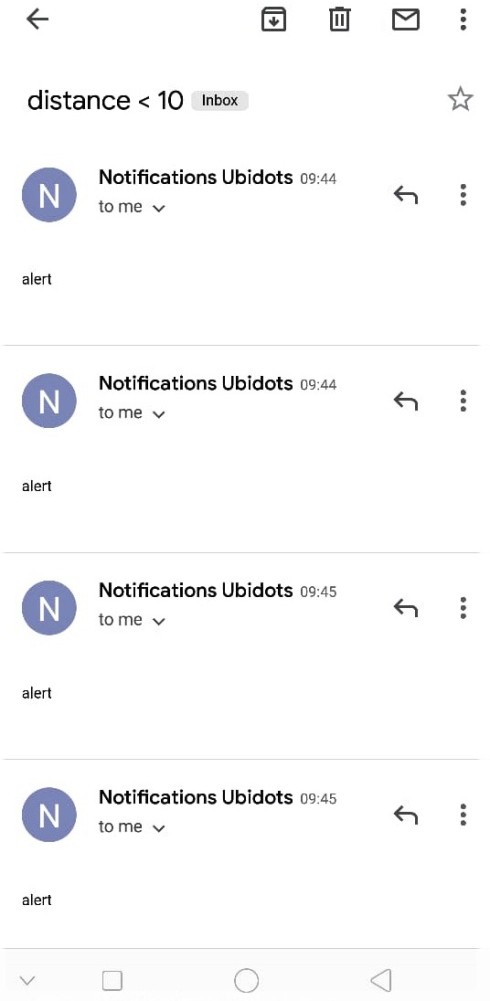
### Figure 8.1.5: Vehicle with Emergency Braking System



**Figure 8.1.6: Blynk UI for the driver**



**Figure 8.1.7: Email Notifications Figure 8.1.8: Message Notifications**



# CHAPTER 9 CONCLUSION AND FUTURE WORK

### CONCLUSION

We have successfully completed the fabrication of emergency braking system model prototype and this project presents the implementation of an Automatic Braking System for Forward Collision Avoidance, intended to use in vehicles where the drivers may not brake manually, but the speed of the vehicle can be reduced automatically due to the sensing of the obstacles. It reduces the accident levels and tends to save the lives of so many people. By doing this project practically we gained the knowledge about working of automatic braking system and with this future study and research, we hope to develop the system into an even more advanced speed control system for automobile safety, while realizing that this certainly requires tons of work and learning, like the programming and operation of microcontrollers and the automobile structure. Hence we believe that the incorporation of all components in Emergency Braking System will maximize safety and also give such system a bigger market space and a competitive edge in the market.

### FUTURE WORK

As we have demonstrated this system as prototype, in future days we were trying to implement in real world vehicles and helping to stop collision. We are further working more on real-time location sharing exactly where the collision happened if the opponent has made the collision. In India, as the citizens, we are trying to implement in all kinds of automobiles to reduce the death rates. Indian Government should make the strict rules to inculcate the proposed system in their vehicles and also the automobiles manufacturing companies to add advanced braking system in all kinds of vehicles.

# REFERENCES

1. **Bommes Michael,Fazekas Adrian,Volkenhoff Tobias,Oeser Markus (2016)** ,” Video based Intelligent Transportation Systems – state of the art and future development”, Published by Elsevier B.V.pp. 4495-4504.
2. **Mohamed A, Issam A, Mohamed B, Abdellatif B (2015)** ,” Real-time detection of vehicles using the haar-like features and artificial neuron networks”, Published by Elsevier B.V.pp. 24-31.
3. **Sravan M.S., Natarajan S,Krishna E.S., Kailath B.J.(2018),**” Fast and accurate on- road vehicle detection based on color intensity segregation “,Published by Elsevier Ltd,.pp. 594-603.
4. **Nemade Bhushan(2016)**,” Automatic Traffic Surveillance using Video Tracking”, Published by Elsevier B.V.pp. 402-409.
5. https://[www.raspberrypi.org/app/uploads/2017/05/Raspberry-Pi-3-1-1619x1080.jpg](http://www.raspberrypi.org/app/uploads/2017/05/Raspberry-Pi-3-1-1619x1080.jpg)
6. https://[www.electronicshub.org/wp-content/uploads/2018/02/Raspberry-Pi-Ultrasonic-](http://www.electronicshub.org/wp-content/uploads/2018/02/Raspberry-Pi-Ultrasonic-) Sensor-Interface-HC-SR04.jpg
7. https://[www.robotshop.com/media/catalog/product/cache/image/900x900/9df78eab33](http://www.robotshop.com/media/catalog/product/cache/image/900x900/9df78eab33) 525d08d6e5fb8d27136e95/s/o/solarbotics-regular-motor-2.jpg
8. https://[www.engineersgarage.com/sites/default/files/xL293D\_1.jpg.pagespeed.ic.lcK8](http://www.engineersgarage.com/sites/default/files/xL293D_1.jpg.pagespeed.ic.lcK8) sB0It7.webp
9. https://circuitdigest.com/sites/default/files/circuitdiagram\_mic/Raspberry-pi-obstacle- avoiding-robot-circuit-diagram.gif
10. https://raspberrypihq.com/wp-content/uploads/2018/01/01\_Blinking-LED\_bb.jpg