الجـامعـة المـصرية للتعليم الإلكتروني الاهلية

National Egyptian E-Learning University

Faculty of Computer & Information Technology

Project Title

Vehicle Accident Prevention System

By:

Asmaa Abdelkareem Mohamed (17-00208)

Rana Khaled Mohamed Wakad (17-00122)

Kariem Adel Abdou Hussein (17-00006)

Abanoub Karm Alyon Yssa (16-00047)

Mina Mohsen Maken Zaky (16-00146)

Under Supervision of:

Associate professor. Sanaa Taha,

Manger of

Information Technology Program

Egyptian E-Learning University

Eng. Mohamed Hassan Omar,

Teaching Assistant at faculty of Computer and Information Technology

Egyptian E-Learning University

This thesis is submitted as a fulfillment of the requirements for the degree of Bachelor of Science in Computer & Information Technology

Assiut 2021

Acknowledgement

First, we thank ALLAH for helping us get through our college years and our graduation project.

It has been a great opportunity to gain lots of experience in real time projects, followed by the knowledge of how to actually design and analyze real projects. For that, we want to thank all the people who made it possible for students like us.

The success and outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to get this all along the completion of our project. All that we have done is only due to such supervision and assistance and I would not forget to thank them.

We wish to express our sincere gratitude to our supervisor, **Associate professor Dr.Sanaa Taha**, for her enthusiasm, patience, insightful comments, helpful information, practical advice and unceasing ideas that have helped us tremendously at all times in our project and writing of this paper. We are extremely thankful to her for providing such a nice support and guidance, although she had busy schedule managing the IT department affairs.—

We owe our deep gratitude to our project mentor and **Teaching Assistant Eng. Mohamed Hassan Omar** who took keen interest on our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good project.

Abstract

Now a day's vehicle accident is becoming critical issue all over the world and in Ethiopia in particular that demands engineer's endeavor to find ways to minimize the incidence.

The application of electronic systems and Artificial Intelligence embedded in automobile are expected to contribute a lot on the race to minimize vehicle accident disaster. Car accidents in Egypt rose by 17.8 percent in 2019, the Central Agency for Public Mobilization and Statistics (CAPMAS) announced; with 9,992 car accidents compared to 8,480 in 2018, and the number of accidents that occur on high ways is about 5220.[1]

These accidents have many causes such as Distracted Driving, Speeding, Driving on alcohol, reckless driving, etc. The common factor for all these reasons is the loss of both optical and auditory effect so we aim to design a system that can warn and trigger the driver optically and acoustically to gain attention and try avoid a collision with the other vehicle.

The system are divided into hardware part that consist of Arduino Uno, Piezo buzzer, led lights, raspberry pi, and Laser Reader and software part which is a machine learning model that classify and express the level of danger to make best action to prevent close call. Collision warning system has a great role to enhance the driving safety. In this system, some measures are used to estimate the dangers and the system warns drivers to be more cautious [2]. The real-time processes should be executed in such system, to remain enough time and distance to avoid collision with the other vehicle.

Contents

List of Acronyms or Abbreviations

CAPMAS: Central Agency for Public Mobilization and Statistics

GIDAS: German In-Depth Accident Study

ADAS: Advanced Driving - Assistance system

LGPL: Lesser General Public License

GPL: General Public License

DIY: do-it-yourself

USB: Universal Serial Bus

IDE: integrated development environment

IOREF: input/output reference

PWM: Pulse Width Modulation

SPI: Serial Peripheral Interface

TWI: Two Wire Interface

SDA: Serial Data Line

SCL: Serial clock line

AREF: Analog Reference

UART: Universal Asynchronous Reception and Transmission

SPI: Serial Peripheral Interface

I2C: Inter-Integrated-Circuit

KNN: K-Nearest Neighbors

SVM: Support Vector Machines

RPM: Rotation per minute

DIP: dual-inline-package

AVR: Advanced Virtual RISC

LIDAR: Light Detection and Ranging

ITS: Intelligent Transporting Systems

Chapter 1

Introduction

1.1 Introduction

Since the end of the twentieth century, traffic accidents began to be analyzed by professionals in order to reveal the real underlying causes of traffic and road accidents. Projects like German In-Depth Accident Study (GIDAS) [3] are done to analyze and reconstruct traffic and road accidents. These projects were made for the development of legislation, automotive engineering arrangements, and public relations. Similar studies are made all over the world; however, using scientific methods are far better than classical accident reporting methods.

Vehicle collisions can be classified by mechanism. Common mechanisms include:

- Head-on collisions,
- Run-off-road collisions,
- Rear-end collisions,
- Side collision, and
- Rollovers are the most common once.

Accidents have increased in large numbers, especially on highways. These accidents have many causes:

- Distracted Driving
- Speeding
- Drunk Driving
- Reckless Driving
- Night Driving
- Fog
- Driving Under the Influence of Drugs.

1.2 Problem Statement

The automotive industry of modern days are far from been defined by expensive cars, flashy cars, sports cars and big muscle cars. Rather, features such as safety systems and warning systems but on the other side dominate most of the cars seen on the roads today when we talk about middle and low range car they lake to such systems beside having a reckless driver or losing both optical and auditory effect while driving which may lead to a collision.

1.3 Motivation

We seek to reduce the number of human casualties or material losses caused by accidents by alerting the driver in a way that attracts his attention before the collision occurs due to lack of focus and to be affordable for low-level and mid-range level cars that don't contain ADAS.

Project Help with:

- Reduce number of collisions.
- Save driver and passengers lives.
- Alarm driver about obstacles.
- Measure and divide distance into different zones to calculate danger.
- Consume budget of warning system so to be affordable to a large segment.

1.4 Project Phases

- **Phase 1: Study**

  We seek to transform an abstract idea into a meaningful goal. At this point, we need to develop a feasibility study and define the project at a broad level. To do this, identify the need for the project and Determines details such as project constraints, objectives, budget, projected schedule, etc.

- **Phase 2: Analysis**

  Analysis can be as thorough or simple as you like and the information gathered can be helpful in current or future decision-making. It is helpful to conduct this information regularly, as the information can change quickly during volatile times.

- **Phase 3: Design**

  The requirement gathered in document that is used as an input, hardware and software architecture that is used for implementing system development is derived.

- **Phase 4: Development**

  The Development Phase features a key step in the system construction. The previous phases lay the foundation for system development; the following phases ensure that the system functions as required.

- **Phase 5: Implementation**

  Includes the testing, inspection, adjustment, correction and certification of facilities and systems to ensure that the project performs as specified. The implementation phase begins with the notice to proceed for the construction contract and ends with final acceptance of the project, unless otherwise specified in grant or regulatory requirements.

- **Phase 6: Validation**

  This stage evaluates/qualifies the process designed earlier to ensure it can reproduce consistent and reliable levels of quality.

- **Phase 7: follow up**

  The follow-up phase serves to ensure that all recommendations given after the basic audit are implemented by the deadline. Different areas of responsibility are distinguished in the overall process and it breaks down into four sub-phases: status check I, follow-up I, status check II, and follow-up II.

1.5 Project Organization (Project Plan)

Time Plan

Month	Activity	Notes
first term		
Nov	Study-Analysis	• explored-proposal is written • feasibility study is performed
Dec	Study-Analysis	• requirments are specified • resources budget and sechedule are calculated
Jan	Design-Development	• design choices - choose the most effective • arranging what done in implementation
Feb	Implementation	• develop on the project • controlling and monitoring the project
Mar	Validation-Feedback	
Second term		
Apr	Study-Analysis	• requirments are specified • resources budget and sechedule are calculated
May	Design-Development	• design choices - choose the most effective • arranging what done in implementation
Jun	Implementation	• develop on the project • controlling and monitoring the project
Jul	Validation-Feedback	

Chapter 2

Background

2.1 Arduino

Arduino is an open source computer hardware and software company, project, and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical and digital world. The project's products are distributed as open-source hardware and software, which are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form, or as do-it-yourself (DIY) kits.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards or Breadboards (shields) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler tool chains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.[4]

Most Arduino boards consist of an Atmel 8-bit AVR microcontroller (ATmega8, ATmega168, ATmega328, ATmega1280, ATmega2560) with varying amounts of flash memory, pins, and features. The 32-bit Arduino Due, based on the Atmel SAM3X8E was introduced in 2012.The boards use single or double-row pins or female headers that facilitate connections for programming and incorporation into other circuits. These may connect with add-on modules termed shields. Multiple and possibly stacked shields may be individually addressable via an I^2C serial bus. Most boards include a 5 V linear regulator and a 16 MHz crystal oscillator or ceramic resonator. Some designs, such as the LilyPad, run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions.

2.1.1 Arduino Uno

The Arduino UNO is a widely used open-source microcontroller board based on the ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board features 14 Digital pins and 6 Analog pins. It is programmable with the Arduino IDE (Integrated Development Environment) via a type B USB cable. It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts. It is also similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available. "Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform. The ATmega328 on the Arduino Uno comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol. The Uno also differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip [5].

2.2 Pins

2.2.1 General Pin functions

- **LED**: There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **VIN**: The input voltage to the Arduino/Genuine board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V**: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
- **3V3**: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND**: Ground pins.
- **IOREF**: This pin on the Arduino/Genuine board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.
- **Reset**: Typically used to add a reset button to shields which block the one on the board [5].

2.2.2 Special Pin Functions

Each of the 14 digital pins and 6 Analog pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

In addition, some pins have specialized functions:

- **Serial**: pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts**: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- **PWM** (**P**ulse **W**idth **M**odulation) 3, 5, 6, 9, 10, and 11 Can provide 8-bit PWM output with the analog Write() function.
- **SPI** (**S**erial **P**eripheral **I**nterface): 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- **TWI** (**T**wo **W**ire **I**nterface): A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
- **AREF** (**A**nalog **Ref**erence): Reference voltage for the analog inputs.

2.3 Raspberry Pi

The Raspberry Pi is a family of credit card-sized single board computers that have become the best-selling UK computers of all time.

Key to the Pi's success has been its price. It's not the most powerful machine in the world, but for less than $60 it offers a computer that can be used to build homebrew electronics and put together a vast range of devices on a budget.

The charitable foundation behind the Pi hasn't rested on its laurels, upgrading the Pi's specs three times since launch, while keeping the price at $35 - $55. In that time, the Pi's processing power has grown more than tenfold, putting the Pi into the category of a machine that could be used as an everyday PC. In hard specs, the top-end model, the Raspberry Pi 4 Model B , has a 1.5 GHz quad-core, 64-bit Arm Cortex A72-based, quad-core processor, up to 8GB RAM, a Video Core VI capable of 4K video playback, 802.11ac Wi-Fi and Gigabit Ethernet, and two USB 2.0 and two USB 3.0 ports. [6]

2.3.1 Bi Directional Communication

Serial communication is simply a way to transfer data. The data will be sent sequentially, one bit at a time (1 byte = 8 bits), contrary to parallel communication, where many bits are sent at the same time. More specifically, when you use Serial with Arduino and Raspberry Pi, you're using the UART protocol. UART means "Universal Asynchronous Reception and Transmission". [7]

It's an asynchronous multi-master protocol based on the Serial communication, which will allow you to communicate between the two boards. Be reassured, there are libraries that will handle all the low layers for you.

Multi-master means that all connected devices will be free to send data when they want. This is one of the main difference with master-slaves protocols, where only the master device can initiate a communication. Usually you'll use other protocols such as I2C and SPI when you need master-slaves configurations: for example when you have one Arduino board and multiple sensors or actuators.

2.4 Machine Learning

Machine Learning is the science (and art) of programming computers so they can learn from data. [Machine Learning is the] field of study that gives computers the ability to learn without being explicitly programmed. Arthur Samuel, 1959.[8]

A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.Tom Mitchell, 1997.

Why is machine learning important?

Machine learning is important because it gives enterprises a view of trends in customer behavior and business operational patterns, as well as supports the development of new products. Many of today's leading companies, such as Facebook, Google and Uber, make machine learning a central part of their operations. Machine learning has become a significant competitive differentiator for many companies.[9]

Why we use Machine Learning?

“Just as electricity transformed almost everything 100 years ago, today I actually have a hard time thinking of an industry that I don’t think AI will transform in the next several years.”

It is quite hard to think of any industrial activity which can be done without the use of Machine learning or Artificial Intelligence. Machine learning is important because of its wide range of applications and its incredible ability to adapt and provide solutions to complex problems efficiently, effectively and quickly.

2.4.1 Types of Machine Learning Systems

- **Supervised learning:** In this type of machine learning, data scientists supply algorithms with labeled training data and define the variables they want the algorithm to assess for correlations. Both the input and the output of the algorithm is specified.
- **Unsupervised learning:** This type of machine learning involves algorithms that train on unlabeled data. The algorithm scans through data sets looking for any meaningful connection. The data that algorithms train on as well as the predictions or recommendations they output are predetermined.

- **Reinforcement Learning:** Is a very different beast. The learning system, called an agent in this context, can observe the environment, select and perform actions, and get rewards in return (or penalties in the form of negative rewards. It must then learn by itself what is the best strategy, called a policy, to get the most reward over time. A policy defines what action the agent should choose when it is in a given situation. [10]

When we built the model, we used Supervised learning that has different types of algorithms:

- **K-Nearest Neighbors**

  K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection. (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.[11]

- **Logistic Regression**

  It is a classification model, which is used to predict the odds in favor of a particular event. The odds ratio represents the positive event, which we want to predict, it used the sigmoid function to convert an input value between zero and one. [12]

- **Support Vector Machines (SVMs)**

  Is a supervised machine learning algorithm, which can be, used for both classification and regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well. [13]

- **Decision Trees**

Belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data). [14]

- **Naïve Bayes**

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods. [15]

Chapter 3

Overall Description

3.1 System overview

This system consists of two main parts, the first is the software, which represents a machine-learning model that is implemented on the hardware and takes the appropriate action based on the data sent to it from the hardware. Second, the hardware, which embedded in the prototype (Car) to send status, and location of the obstacle to the model to take the (alarm using led & Piezo buzzer).

3.2 System functions

3.2.1 Functional Requirement:

- Measure distance between two cars or obstacles and car speed.
- Send the measured (speed , average speed ,distance and moved) to the model.
- The model classifies the data entered into it and determines the level of danger, according to level divided into four classes

Class A	zone A
Class B	zone B
Class C	zone C
Class D	zone D

To which the data belongs

- The Arduino receives the class (zone) to which the data belongs, and accordingly implements the alarm (using led and piezo buzzer) and controls the car.

3.2.2 Non-Functional Requirement:

Any Requirement That Specifies How the System Performs a Certain Function, In other words a non-functional requirement is a specification that describes the system's operation capabilities and constraints that enhance its functionality.

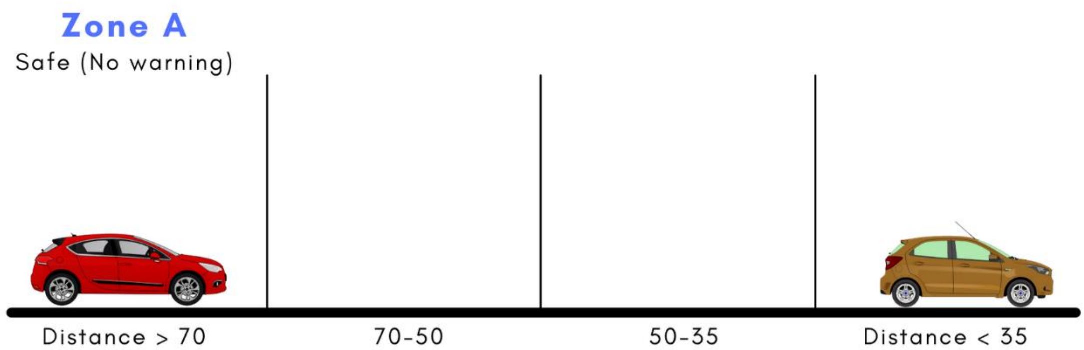
- **Performance** :LiDAR is a sampling tool. What meant by that is that it sends over 160,000 pulses per second. For every second, each 1-meter pixel gets about 15 pulses. so in system performance is very high in addition to high response time. LiDAR systems are very accurate because it's being controlled in a platform.

- **Scalability**: The system will continue to meet performance requirements even if sizes or tools are changed for example If we replace car with bigger one (truck) so we replace with big size big weight big wheels and higher power so the system can handle scaling based on requirement.

3.3 System Scenario

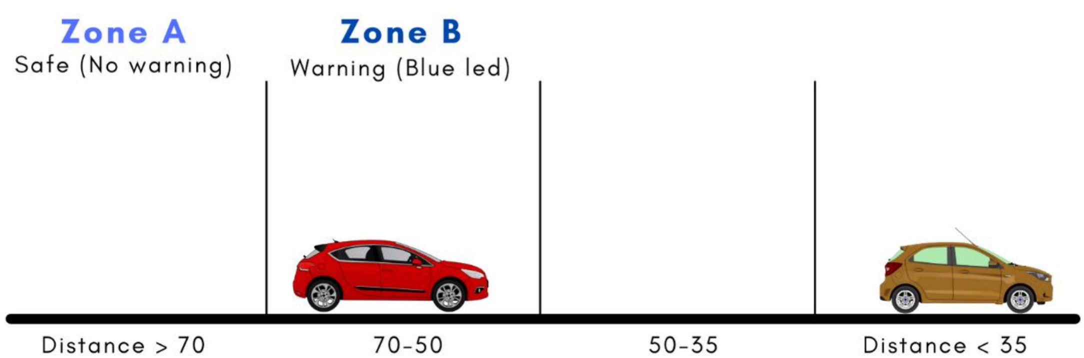
Distance between vehicles segmented into four different zones:

- **Zone A**: if distance is greater than 70 units then No warning.

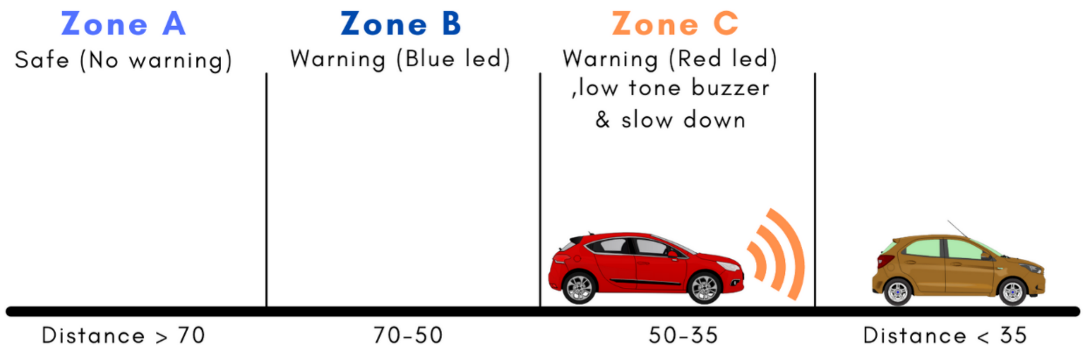


- **Zone B:** if distance confined between 50-70 units then only visual warning

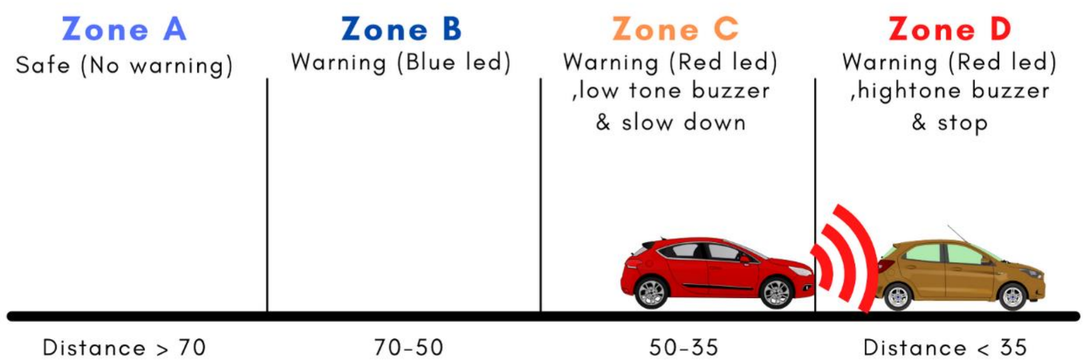
  The blue LED will glow to alert.



- **Zone C**: if distance confined between 35-50 units then visual warning red LED will glow to alert and Buzzer will start at low frequency and slow down (In this zone driver has to pay attention).

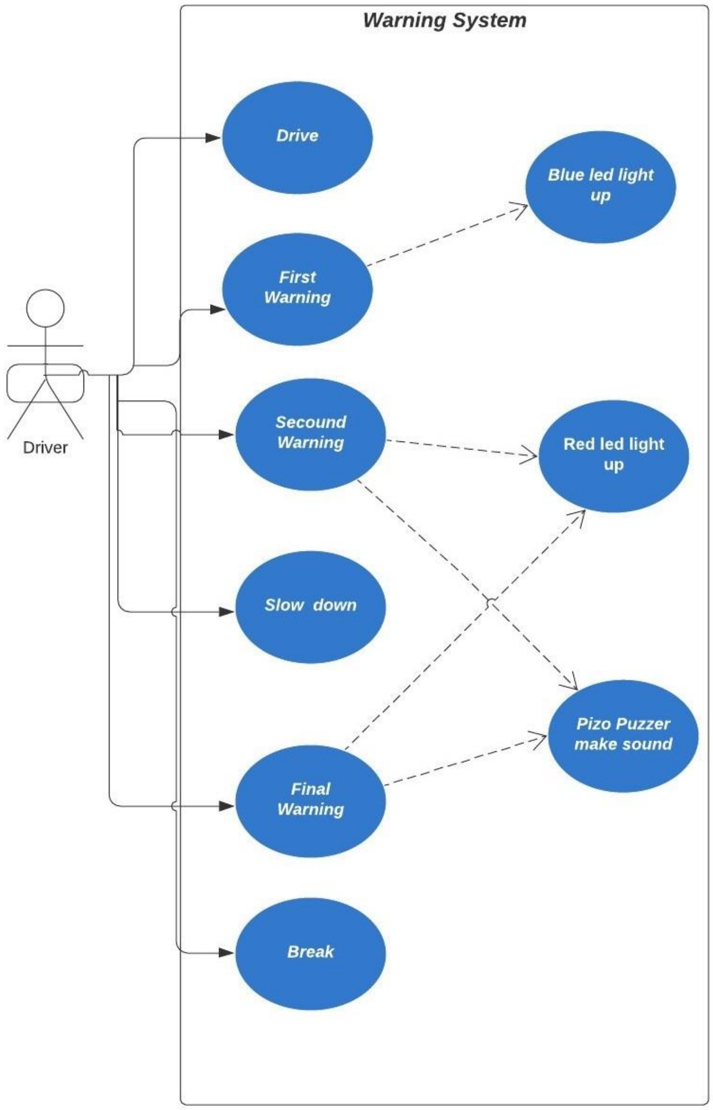


- **Zone D**: if distance is less than 35 units then visual warning red LED will glow to alert and Buzzer will high frequency and stop the car.



3.4 User characteristic:

User of this System consider the one who have a (old, low rang, mid-range) car.



Chapter 4

Implementation

4.1 Hardware Implementation

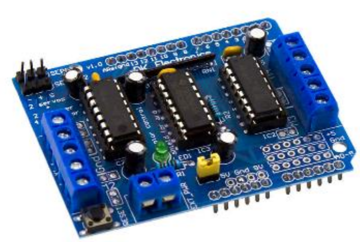
In the beginning, we assembled the Arduino Car, by using the following:

- **Wheels (4x)**

- **TT Gear Motor:**
  The 200-RPM (Rotation per minute) Dual Shaft BO Motor. Motor gives good torque and rpm at lower operating voltages [16]

- **Microcontroller (Arduino Uno)**
  The Arduino Uno is a microcontroller board based on a removable, dual-inline-package (DIP) ATmega328 AVR microcontroller. It has 20 digital input/output pins (of which 6 can be used as PWM outputs and 6 can be used as analog inputs). Programs can be loaded on to it from the easy-to-use Arduino computer program. [17]

- **Motor Driver Shield**
  L293D use 16-pin DIP package, its internal integration is bipolar H - bridge circuit. This kind of bipolar pulse width method has many advantages, such as the current continuous, or micro current vibration when the motor stops, which as a lubrication effect. It can eliminate the dead zone of static friction when positive and negative. [18]



- **18650 Li - on Battery**

  The 18650 battery is a lithium-ion cell classified by its 18mm x 65mm size, which is slightly larger than an AA battery.

- **TF-mini sensor (LIDAR)**

  The TF Mini LIDAR is a fixed position LIDAR source-sensor array. This product is based on ToF

  (Time of Flight) principle and integrated with unique optical and electrical designs, to achieve stable, precise, high sensitivity and high-speed distance detection.

  Its operating principle is as follows: a modulated near-infrared light is sent from the sensor and reflected by an object; the distance to the object to be shot can be converted with the sensor by calculating the time difference or phase difference between the light sending and the light reflection, so as to produce the depth information. [19]

  Despite its minimal cost, it has some impressive specifications:

  - Range of 30 cm to 12m (1 foot to 40 feet)
  - Accuracy to 0.5 cm within the range
  - Up to 100 scans per second
  - Low power consumption
  - Built in processor with serial I/O

- **LED**

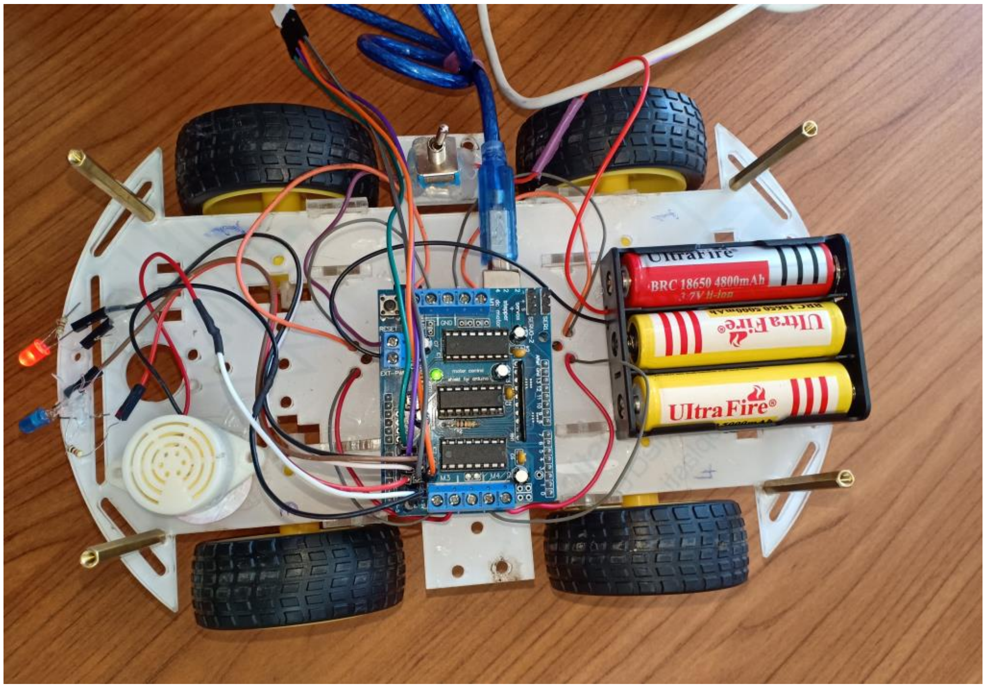
  Is a semiconductor device that emits light when an electric current is passed through it.

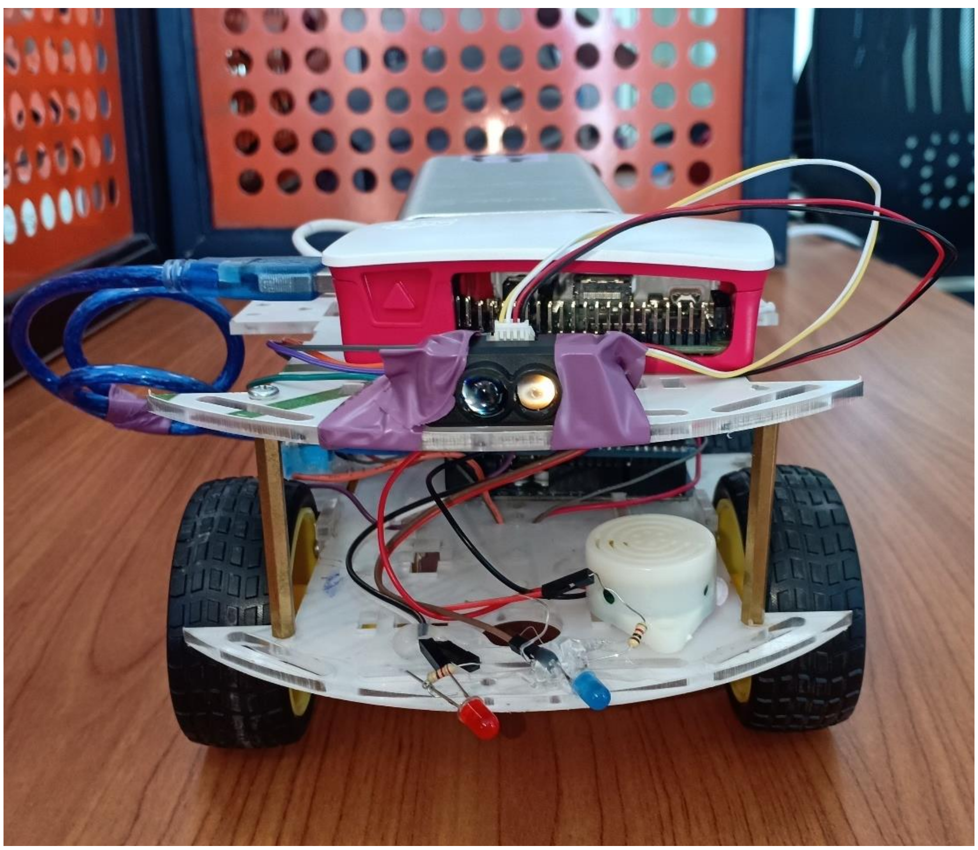
- **Piezo Buzzer**

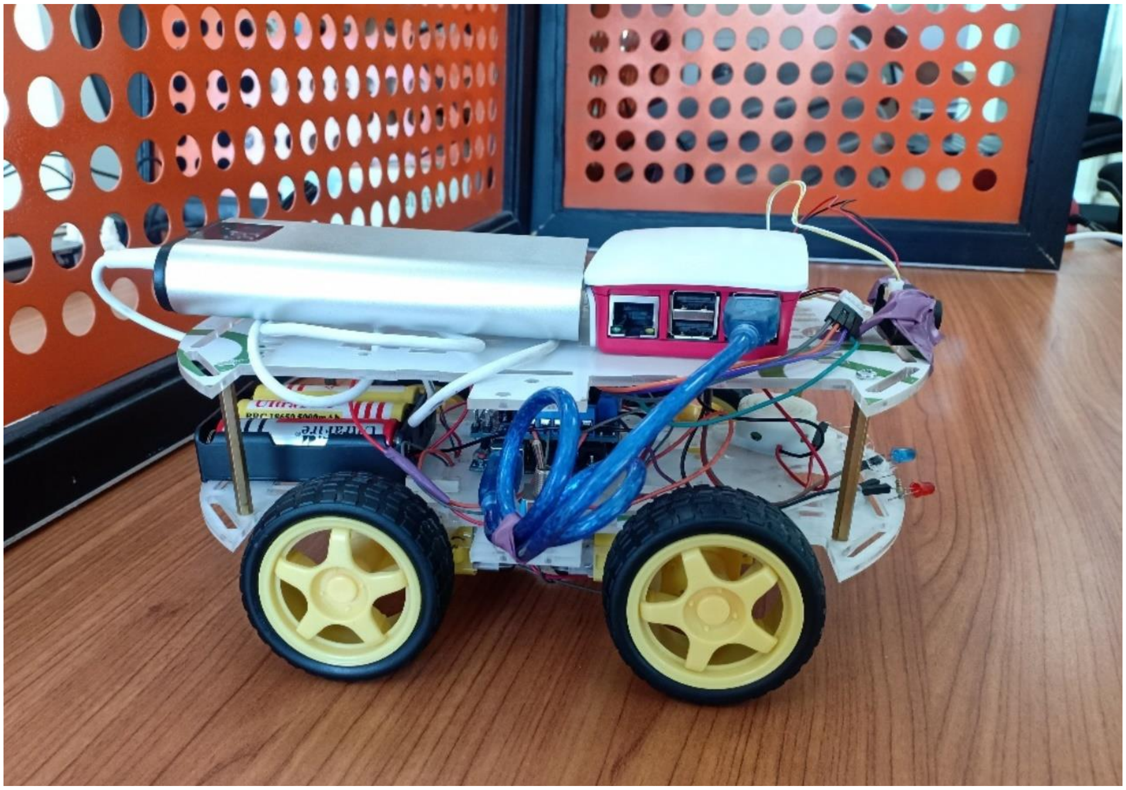
  In simplest terms, a piezo buzzer is a type of electronic device that's used to produce a tone, alarm or sound. It's lightweight with a simple construction, and it's typically a low-cost product. Yet at the same time, depending on the piezo ceramic buzzer specifications, it's also reliable and can be constructed in a wide range of sizes that work across varying frequencies to produce different sound outputs, it can generate a pressure wave which the human ear picks up as sound. [20]

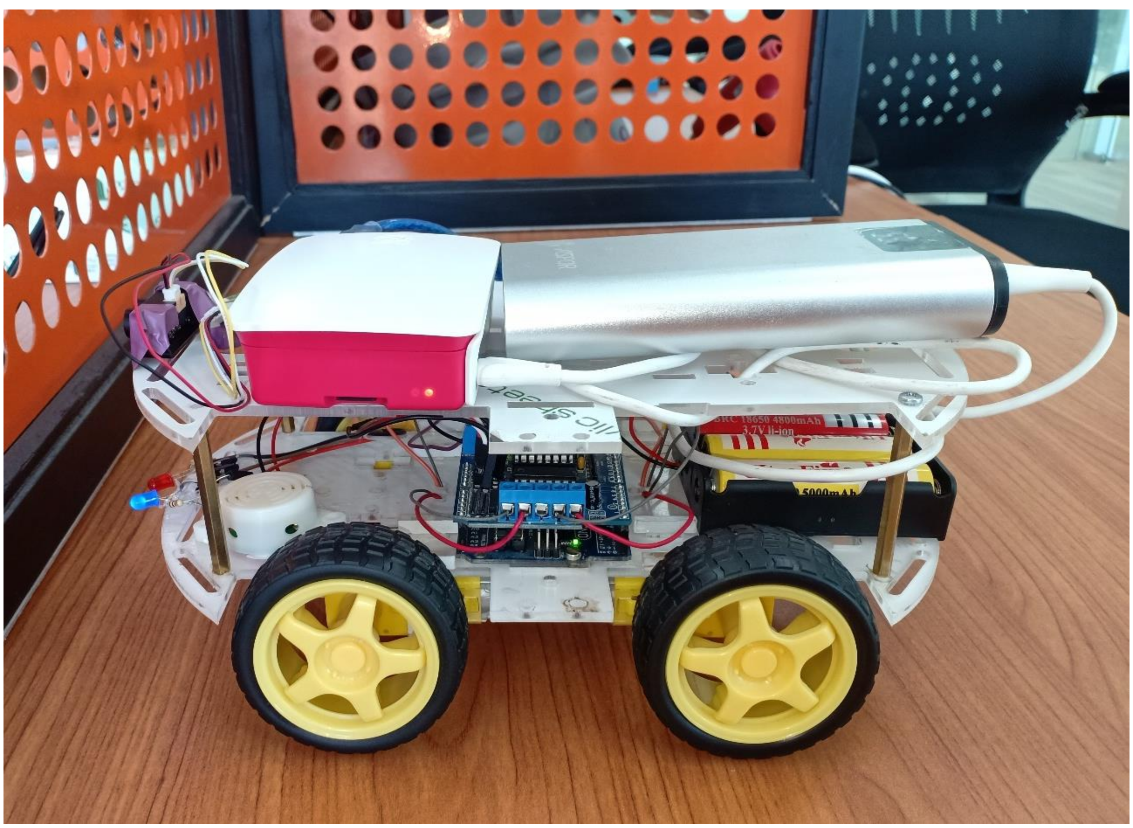
- **Raspberry pi**

The Raspberry Pi is a low cost, credit card sized computer. A capable little device enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It is capable of doing everything you would expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games. We will loaded our machine learning model which written by python programing language to classification inputs then implements it to hardware by Arduino. [21]


UltraFire
BRC 18650 4800mAh
UltraFire
UltraFire







4.2 Software Implementation

Software implementation consist of two phases:

- **First phase** is the Arduino code for your information method of transferring result is used trough Bi directional serial communication between Arduino and raspberry pi using the UART protocol. Moreover, the Arduino take the proper action by initiate an alarm (Visual and auditory). Next part will explain two phases in details.
- **Second phase** is machine learning model that are loaded in Raspberry pi that can predict and classify danger based in inputs that is given to inputs are(speed , average speed ,distance and moved).After model classify level of danger which can be level A or B or C or D it transfers result to the Arduino.

Following steps demonstrate previous phases:

- Step 1: This code is implemented on Arduino IDE which measure distance by TF-mini

```
if (mySerial.available())//check whether the serial port has data input
{
  if (mySerial.read() == HEADER) // determine data package frame header 0x59
  {
    uart[0] = HEADER;
    if (mySerial.read() == HEADER) //determine data package frame header 0x59
    {
      uart[1] = HEADER;
      for (i = 2; i < 9; i++) // store data to array
      {
        uart[i] = mySerial.read();
      }
      check = uart[0] + uart[1] + uart[2] + uart[3] + uart[4] + uart[5] + uart[6] + uart[7];
      if (uart[8] == (check & 0xff)) // check the received data as per protocols
      {
        dist = uart[2] + uart[3] * 256; // calculate distance value
        strength = uart[4] + uart[5] * 256; // calculate signal strength value
        Serial.print("dist = ");
        Serial.print(dist);// output LiDAR tests distance value

        Serial.print('\t');
        Serial.print("strength = ");
        Serial.print(strength);// output signal strength value
        Serial.print('\n');
      }
    }
  }
```

- Step 2: Check Bi directional serial communication between Arduino and Raspberry pi using the UART protocol.

  Arduino communication code:

```
void setup() {
  Serial.begin(9600);
}
void loop()
{
  //bi direction communication
  if (Serial.available() > 0) {
    data = Serial.readStringUntil('\n');
    Serial.println(data);
  }
```

Raspberry pi communication code:

```
import serial
import time
```

```
if __name__ == '__main__':
    ser = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
    ser.flush()
    while True:
        distans = ser.readline().decode('utf-8').rstrip()
        print(distans)
```

- **Step 3:** Machine learning model that are loaded in Raspberry pi that can predict and classify danger based in inputs that is given.

In this model, we used proper libraries to be imported in model

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Then we used dataset that were provided by (Zenodo) and contain variables and records of data

```
# Importing the dataset
dataset= pd.read_csv('points_engine_uc1_simulator.csv',sep=';')

dataset.sample(frac=1)
```

This is a samples of dataset records:

	speed_kmh	speed_avg_kmh	distance_delta_m	moved	zone
0	50	49.86670	68.93	1	B
1	50	51.20000	59.51	1	B
2	50	49.93330	69.27	1	B
3	59	54.33330	82.35	1	A
4	50	47.73330	76.74	1	A
5	50	53.86670	69.36	1	B
6	50	49.73330	69.18	1	B
7	5	6.86667	7.17	1	D
8	60	56.93330	80.52	1	A
9	30	74.00000	34.97	1	D
10	60	56.33330	83.18	1	A
11	10	17.80000	11.10	1	D
12	50	58.00000	68.30	1	B
13	50	49.86670	68.71	1	B
14	10	34.00000	13.90	1	D
15	130	13.33330	122.59	1	A
16	129	99.53330	185.51	1	A
17	5	40.93330	6.71	1	D
18	50	49.93330	68.17	1	B
19	10	28.60000	5.54	1	D
20	50	50.40000	69.90	1	B
21	5	5.00000	6.93	1	D
22	80	116.00000	104.96	1	A
23	4	13.60000	5.54	1	D
24	10	5.33333	12.51	1	D
25	49	50.26670	67.98	1	B
26	10	10.00000	13.86	1	D
27	50	34.00000	65.80	1	B
28	50	50.00000	69.54	1	B
29	5	7.06667	6.93	1	D
...	...	...	...	...	...

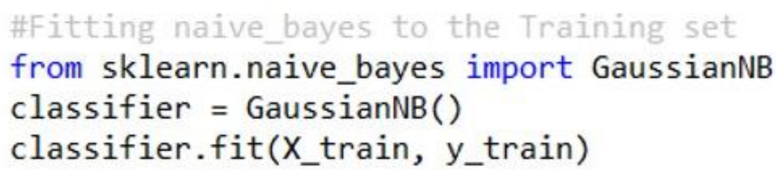
1045137	15	12.50000	21.47	1	D
1045138	80	37.14290	145.40	1	A
1045139	50	56.71430	72.25	1	A
1045140	80	80.00000	110.85	1	A
1045141	50	50.00000	68.25	1	B
1045142	50	52.66670	69.44	1	B
1045143	50	50.00000	59.32	1	B
1045144	60	58.80000	82.32	1	A
1045145	60	60.26670	83.15	1	A
1045146	50	50.00000	69.24	1	B
1045147	4	4.00000	5.02	1	D
1045148	5	5.00000	6.61	1	D
1045149	5	5.00000	6.63	1	D
1045150	7	8.60000	10.77	1	D
1045151	50	50.00000	69.27	1	B
1045152	50	51.33330	69.45	1	B
1045153	114	62.13330	156.38	1	A
1045154	49	49.73330	67.92	1	B
1045155	10	67.33330	19.66	1	D
1045156	80	64.00000	110.76	1	A
1045157	5	5.00000	7.02	1	D
1045158	5	5.00000	6.63	1	D
1045159	50	53.20000	69.95	1	B
1045160	49	49.86670	68.17	1	B
1045161	10	6.00000	13.87	1	D
1045162	50	49.93330	68.57	1	B
1045163	50	48.66670	69.42	1	B
1045164	50	35.13330	68.88	1	B
1045165	50	50.00000	65.60	1	B
1045166	10	10.00000	13.86	1	D

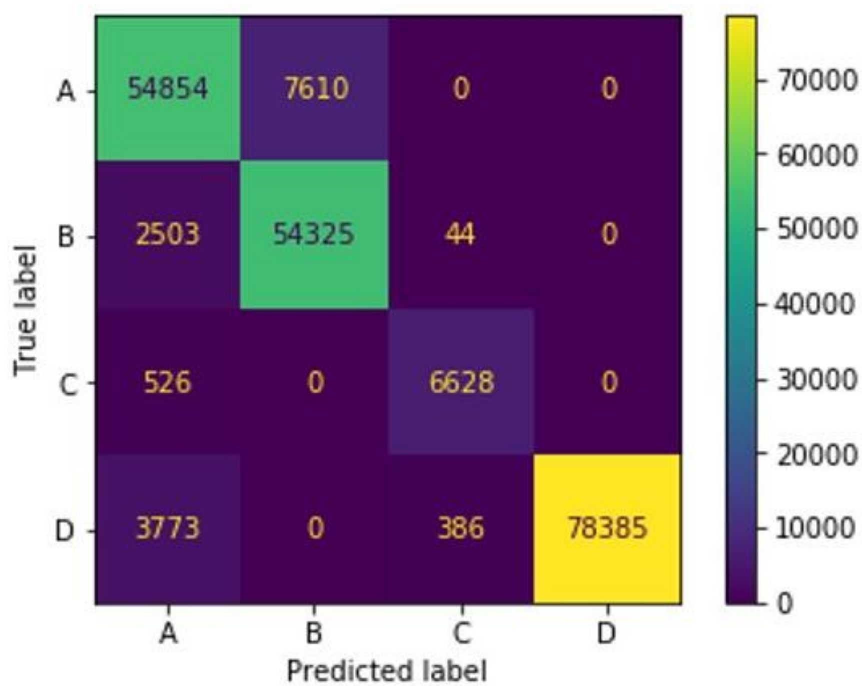
[1045167 rows x 5 columns]>

Next part choosing the most suitable algorithm that achieve high efficiency so we select many algorithms and test them to select the accurate one, the selected as follows:

- **Naive Bayes**

  With accuracy = **92%**

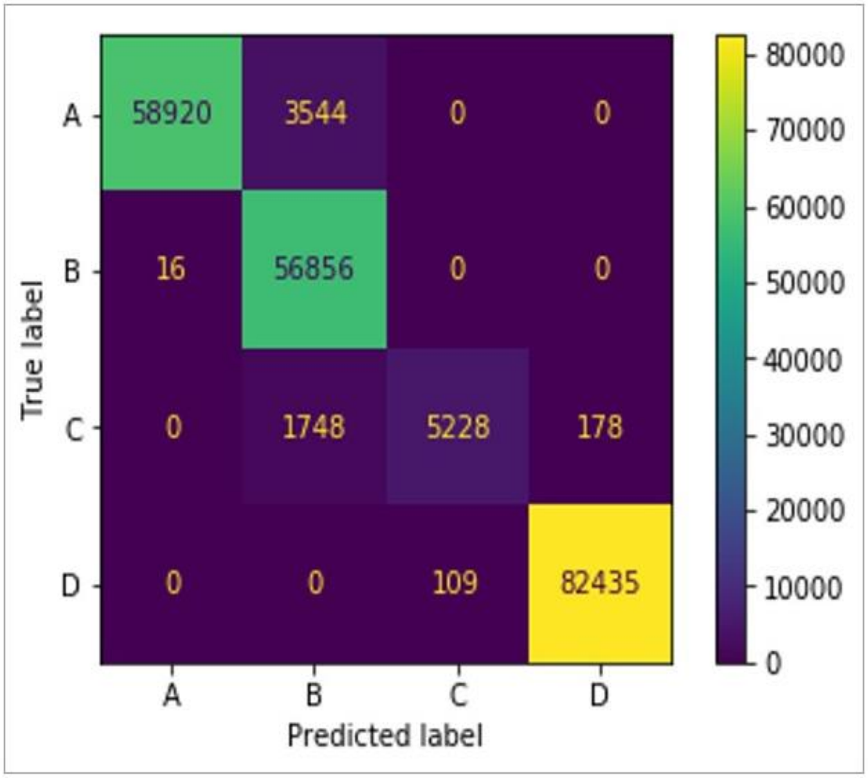




- **Logistic Regression**

  With accuracy = **97%**

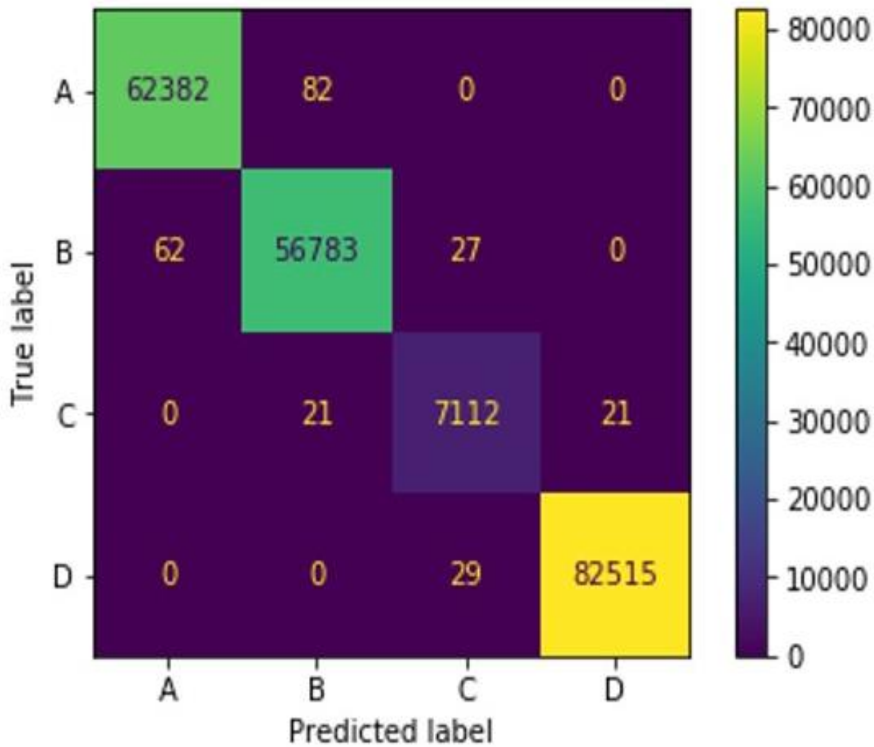
```
#Fitting Logistic Regression to the Training set
from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression()
classifier.fit(X_train,y_train)

#
```



- **K-Nearest Neighbors**

  With accuracy = **99%**

```
#Fitting KNN to the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=7)
classifier.fit(X_train, y_train)
```



Therefore, according to the previous **KNN** achieved the highest accuracy **99%**

Therefore, we use it in implementation. Therefore, this a comparison between advantage and disadvantage of **KNN** algorithm. [22]

Advantage	**Disadvantage**
There is no training period for it. It stores the training dataset and learns from it only at the time of making real time predictions. This makes the KNN algorithm much faster than other algorithms that require training e.g. SVM, Linear Regression etc.	Need feature scaling: We need to do feature scaling (standardization and normalization) before applying KNN algorithm to any dataset. If we do not do so, KNN may generate wrong predictions.
Since the KNN algorithm requires no training before making predictions, new data can be added seamlessly which will not affect the accuracy of the algorithm.	Does not work well with high dimensions: The KNN algorithm does not work well with high dimensional data because with large number of dimensions, it becomes difficult for the algorithm to calculate the distance in each dimension.
Easy to implement. There are only two parameters required to implement KNN i.e. the value of K and the distance function (e.g. Euclidean or Manhattan etc.)	Sensitive to noisy data, missing values and outliers: KNN is sensitive to noise in the dataset. We need to manually impute missing values and remove outliers.

Next part based on accuracy of KNN, we try to test model

- load KNN model from disk

```
modelfilename = 'KNN_99.sav'
loaded_model = pickle.load(open(modelfilename, 'rb'))
```

- predict function to return the right zone

```
def predict(reads):
    predication=loaded_model.predict(reads)
    return predication
```

- call function to test the model and print the result

```
reads=[80,64,30,1]
reads=np.array(reads)
output=predict([reads])
print('result=',output)
```

- samples of testing output

```
In [26]: reads=[80,64,110,1]
    ...: reads=np.array(reads)
    ...: output=predict([reads])
    ...: print('result=',output)
result= ['A']
```

```
In [28]: reads=[80,64,55,1]
    ...: reads=np.array(reads)
    ...: output=predict([reads])
    ...: print('result=',output)
result= ['B']
```

```
In [27]: reads=[80,64,45,1]
    ...: reads=np.array(reads)
    ...: output=predict([reads])
    ...: print('result=',output)
result= ['C']
```

```
In [29]: reads=[80,64,30,1]
    ...: reads=np.array(reads)
    ...: output=predict([reads])
    ...: print('result=',output)
result= ['D']
```

Chapter 5

Conclusion

5.1 Conclusions

The past few years witnessed a high increase in accident and collision numbers according to statistics. Accidents Prevention is a vital process that benefit both the society and authorities and help us save lives, properties and the project aim to add a low cost warning system that can make an obvious effect that can help in preventing prober collision. System consist of two parts hardware components that are responsible for gathering inputs and software part is machine learning model that are responsible for analyzing and processing inputs and predict better and optimize performance we choose KNN algorithm because it came up with highest efficiency rate. This project were made from the point that we believe that we can make an impact even if it is not that big to reduce number of collision incidents and save more life’s. Of course, there are more features we hope to add to our system and we hope to apply this project in reality and next section we will illustrate some of future works that we hope to be added.

5.2 Future Scope

Many different adaptations, tests, and experiments have been left for the future due to lack of time (i.e. the experiments with real data are usually very time consuming).Future work concerns deeper analysis of particular mechanisms, new proposals to try different methods, or simply curiosity. There are some ideas that we would have liked to try during the description and the development of the project functions. The following ideas could be tested and aspire with the level of efficiency of our system as follows:

1. Develop prevention system that can interact with ITS

   (Intelligent Transporting system) to monitor wide range of vehicles.

2. Develop mobile application that can receive signals from ITS and in case of bad call send notifications to authorities and get help as soon as possible.
3. Design and fabricate an automotive chip that can be small and portable to be added in any vehicle.

References

[1] https://www.capmas.gov.eg/Pages/IndicatorsPage.aspx?ind_id=2302

[2] https://ijarcce.com/wp-content/uploads/2018/10/IJARCCE.2018.797.pdf

[3] https://www.gidas.org/en/about-gidas/ueberblick-ueber-gidas/

[4] https://www.tutorialspoint.com/arduino/index.htmUltrasonic

[5] https://www.arduino.cc/en/Main/ArduinoBoardUnoSMD

[6] https://www.liquidlight.co.uk/blog/raspberry-pi-what-is-it-and-why-do-i-need-one/

[7] https://roboticsbackend.com/raspberry-pi-arduino-serial-communication/

[8]https://books.google.com.eg/books/about/Hands_On_Machine_Learning_with_Scikit_Le.html?id=HnetDwAAQBAJ&printsec=frontcover&source=kp_read_button&redir_esc=y#v=onepage&q&f=false

[9] https://quantilus.com/why-is-machine-learning-important-and-how-will-it-impact-business/

[10] https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/

[11]https://www.geeksforgeeks.org/k-nearest-neighbors-with-python-ml/#:~:text=The%20K-Nearest%20Neighbors%20(KNN,are%20near%20to%20each%20other.

[12] https://www.geeksforgeeks.org/differentiate-between-support-vector-machine-and-logistic-regression/

[13] https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code

[14] https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

[15] https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/

[16] https://www.electronicscomp.com/200-rpm-dual-shaft-bo-motor-straight

[17] https://store.arduino.cc/usa/arduino-uno-rev3

[18] https://yaoota.com/en-ke/product/motor-drive-shield-l293d-for-arduino-duemilanove-mega-uno-price-from-jumia-kenya

[19] https://www.dfrobot.com/blog-1016.html?tracking=5b4f5b0b16c23

[20] https://www.americanpiezo.com/standard-products/buzzers.html

[21] https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/

[22] https://www.fromthegenesis.com/pros-and-cons-of-k-nearest-neighbors