A Project Report on

GPS Based Driverless Car

Submitted by

Himanshu Alwe	**FS15CO032**
Samruddhi Kedare	**FS15CO042**
Ankita Pandey	**FS15CO046**
Kunjal Shah	**FS15CO048**

In partial fulfillment of the requirements for

Diploma in Computer Engineering

2017-18

Under the guidance of

Smt. V. M. Aswar

Department of Computer Engineering

Government Polytechnic, Mumbai

Mumbai-400051

Department of Computer Engineering

Government Polytechnic, Mumbai

Mumbai-400051

CERTIFICATE

This is to certify that following students have successfully and satisfactorily completed the project on **GPS Based Driverless Car** and presented their report in the partial fulfillment of requirement for Diploma in Computer Engineering from Government Polytechnic, Mumbai under the guidance of Smt. V. M. Aswar in the academic year 2017-2018.

Himanshu Alwe	**FS15CO032**
Samruddhi Kedare	**FS15CO042**
Ankita Pandey	**FS15CO046**
Kunjal Shah	**FS15CO048**

Project Guide

External Examiner

Head Of the Department

Principal

Table of Contents

List of Figures

List of Tables

ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely fortunate to have got this all along the completion of our project work. We respect and thank our project guide Smt. V. M. Aswar, for giving us an opportunity to do the project titled “GPS Based Driverless Car” and providing us all support and guidance which made us complete the project on time.

We are fortunate enough to get constant encouragement, support and ideas from all Teaching staffs of Department of Computer Engineering which helped us in successfully completing our project work .Also, we would like to mention our sincere regards to all the non-teaching staff of department for their extended support for helping us to see our seniors’ documentation books from the Department.

We would genuinely thank Computer Department to organize the Android Workshop for us which helped us to learn Android programming and consequently develop apps for the project.

We also thank Smt. Swati Deshpande, our Principal, for providing us opportunities to carry ahead the project.

Himanshu Alwe	**FS15CO032**
Samruddhi Kedare	**FS15CO042**
Ankita Pandey	**FS15CO046**
Kunjal Shah	**FS15CO048**

ABSTRACT

Our project, GPS Based Driverless Car, is a working model that demonstrates the working of an autonomous driverless car working on the Global Positioning System principle, i.e., it moves automatically from source to destination based on GPS. It also detects obstacles on its way using sensors and adjusts its motion based on those obstacles. We also aim at giving it ability to identify RED SIGNAL LIGHT on traffic signals and stop, just like a normal car stopping at signals. Finally, if GPS does not work properly due to range issues, the car can be operated temporarily by human operator sitting in the car, using Android Application via Bluetooth.

CHAPTER 1

Introduction

Introduction

1.1 Preface

Since its inception of the commercial auto industry in the late 1890s, cars have become increasingly safe and convenient. From simple four wheelers to sports cars, latest car technologies include hydrogen based cars, and other superior innovations to enhance the efficiency and quality of four wheeler locomotion.

A vision to reducing the road accidents, and adding fuel to the fire of technology in the automobile industry is the development of driverless cars. Also, with the world moving towards automation, the GPS based driverless car will enable us to be a part of the automation in locomotion.

Autonomous means self governance. Many historical projects related to vehicle autonomy have been *automated* (made to be *automatic*) due to a heavy reliance on artificial hints in their environment, such as magnetic strips. Autonomous control implies satisfactory performance under significant uncertainties in the environment and the ability to compensate for system failures without external intervention.

In 2017, Audi stated that its latest A8 would be autonomous at up to speeds of 60 km/h using its "Audi AI". In November 2017, Waymo announced that it had begun testing driverless cars without a safety driver at the driver position, however; there is still an employee in the car. In February 2018, Waymo announced that its test vehicles had traveled autonomously for over 5 million miles.

1.2 What is Driverless Car?

A driverless car (sometimes called a self-driving car, an automated car or an autonomous vehicle) is a robotic vehicle that is designed to travel between destinations without a human operator. To qualify as fully autonomous, a vehicle must be able to navigate without human intervention to a predetermined destination over roads that have not been adapted for its use.

1.3 Potential Benefits

Proponents of systems based on driverless cars say they would eliminate accidents caused by driver error, which is currently the cause of almost all traffic accidents. Furthermore, the greater precision of an automatic system could improve traffic flow, dramatically increase highway capacity and reduce or eliminate traffic jams. Finally, the systems would allow commuters to do other things while traveling, such as working, reading or sleeping.

CHAPTER 2

Literature Review

Literature Review

2.1 Background

The current automobile industry, driven by trained men, is highly prone to traffic collisions caused by human errors, such as delayed reaction time, tailgating, rubbernecking or other forms of distracted or aggressive driving; the injuries caused due to these collisions and the subsequent costs.

Experiments have been conducted on automating cars since at least the 1920s; promising trials took place in the 1950s and work has proceeded since then. The first self-sufficient and truly autonomous cars appeared in the 1980s, with Carnegie Mellon University's Navlaband ALV projects in 1984 and Mercedes-Benz and Bundeswehr University Munich's Eureka Prometheus Project in 1987. Since then, numerous major companies and research organizations have developed working prototype autonomous vehicles including Mercedes-Benz, General Motors, Continental Automotive Systems, Autoliv Inc., Bosch, Nissan, Toyota, Audi, Volvo, Vislab from University of Parma, Oxford University and Google. In July 2013, Vislab demonstrated BRAiVE, a vehicle that moved autonomously on a mixed traffic route open to public traffic.

2.2 Referred Papers/ Books/ Concepts

Reference has been taken to various papers like those of Self Driving cars, and other documentation of works by students of other institutes, relevant to the topics. IEEE papers have also been referred to.

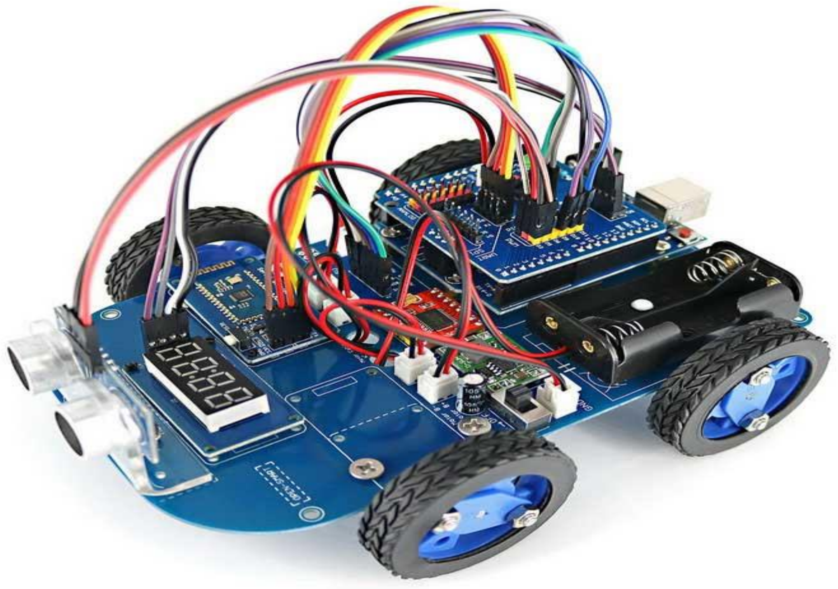


Figure 2-1 Hardware Assembled Car

We have used the concept of providing static coordinates for GPS as of now so as to determine motion of the autonomous car.

2.3 Existing System

The current automobile industry, driven by trained men, is highly prone to traffic collisions caused by human errors, such as delayed reaction time, tailgating, rubbernecking or other forms of distracted or aggressive driving; the injuries caused due to these collisions and the subsequent costs.

To add to this, numerous constraints exist on the driver's ability to drive-namely distracted and texting while driving, intoxicated, prone to seizures, or otherwise impaired. This calls for a rise in the safety requirements of travelling in vehicles.

Moreover, malicious drivers attempt to disobey traffic rules, adding to the threats involved in travelling through vehicles.

As of 2017, most commercial projects focused on autonomous vehicles that did not communicate with other vehicles or an enveloping management regime. Google has already started testing its own autonomous cars on the streets.

CHAPTER 3

Requirement Analysis

Requirement Analysis

3.1 Hardware Requirements

The car needs hardware like chassis, wheels and several 9V DC power supplies to run.

Table 3-1 Major Hardware Specifications

Name of component	Purpose
Arduino Uno	Programming
Ultrasonic Sensor	Front side obstacle detection
Infrared Sensors	Side obstacle detection
AC Motors	Motion of the car
Driver Controller IC L293D	To control motors based on Arduino
Bluetooth module HC-05	To establish communication between mobile and car

3.2 Software Requirements

An Android based mobile phone with mobile data, Bluetooth and GPS location features is required. Arduino IDE with Laptop is also useful to watch data on serial monitor and track status of the implementation of project.

Table 3-2 Major Software Specifications

Name of component	Purpose
Arduino IDE	To provide connectivity between car and Android application
Android Studio	To develop Android app

3.3 Other Requirements

Strong GPS range, using HIGH ACCURACY MODE must be available in the environment in that area.

3.4 Feasibility study

Feasibility study is an assessment of the practicality of a proposed project or system. It has three parts:

1. Economic Feasibility: The driverless car is cost effective as it saves human operator wages.
2. Technical Feasibility: Current cars run on manual operator knowledge. Driverless cars will use advanced GPS technology that will allow users to travel even if no one knows the path to the destination.
3. Behavioural Feasibility: The car, however, will affect drivers as they become jobless. However, the travelers are availed of more flexibility because of automation.

CHAPTER 4

System Design

System Design

4.1 Data Flow Diagram

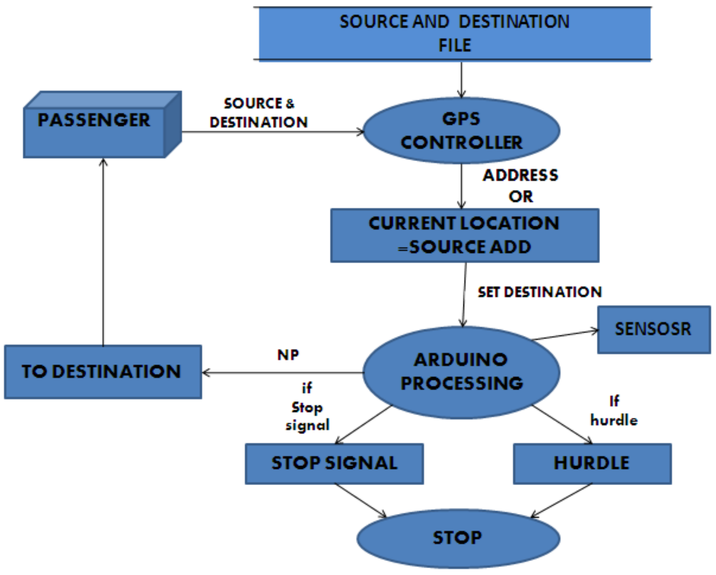


Figure 4-1 Data Flow Diagram for GPS

This diagram is a DFD showing how data flows through the system while functions like GPS and obstacle detection are performed. Data like strategic coordinates, motion flags and others are analysed how they move and are processed through the system.

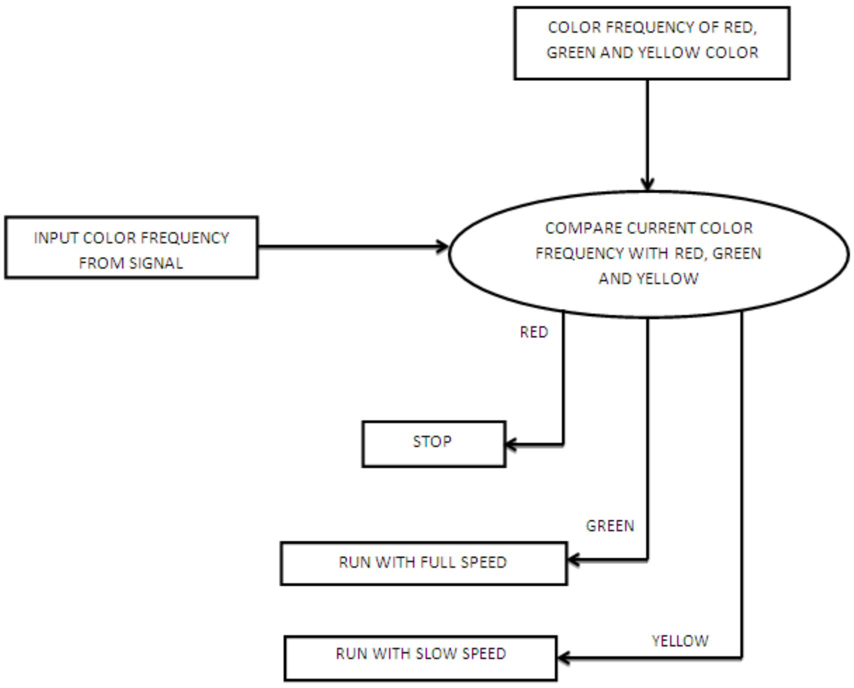

COLOR FREQUENCY OF RED, GREEN AND YELLOW COLOR
INPUT COLOR FREQUENCY FROM SIGNAL
COMPARE CURRENT COLOR FREQUENCY WITH RED, GREEN AND YELLOW
RED
STOP
GREEN
RUN WITH FULL SPEED
YELLOW
RUN WITH SLOW SPEED

Figure 4-7 Data Flow Diagram for colour sensor

This diagram is a DFD showing how data flows through the system while colour sensor module is implemented.

4.2 Algorithm

The algorithm of operation of our car on GPS is:

1. Start
2. Enter strategic coordinates on Android app and respective motion flags.
3. Car receives coordinates via Bluetooth.
4. As per GPS, motors move comparing current location with strategic coordinates and reading motion flags.
5. If *current_cord==destination_cord* then
    a. Stop.
6. Else
    Go to step 4.
7. Stop.

The algorithm of operation of obstacle detection is:

1. Start
2. If obstacle detected then
    a. Motor stops or appropriately turns.
3. If obstacle removed then
    a. Motor resumes.
4. If *current_cord==destination_cord* then
    a. Stop.
5. Else
    Go to step 2.
6. Stop.

The algorithm of operation of colour sensor is:

1. Start
2. If *red colour detected* then
    a. Motor stops.
3. If *another colour detected* then
    a. Motor resumes.

4. If *current_cord==destination_cord* then
    a. Stop.
5. Else

    Go to step 2.
6. Stop.

4.3 Flowcharts

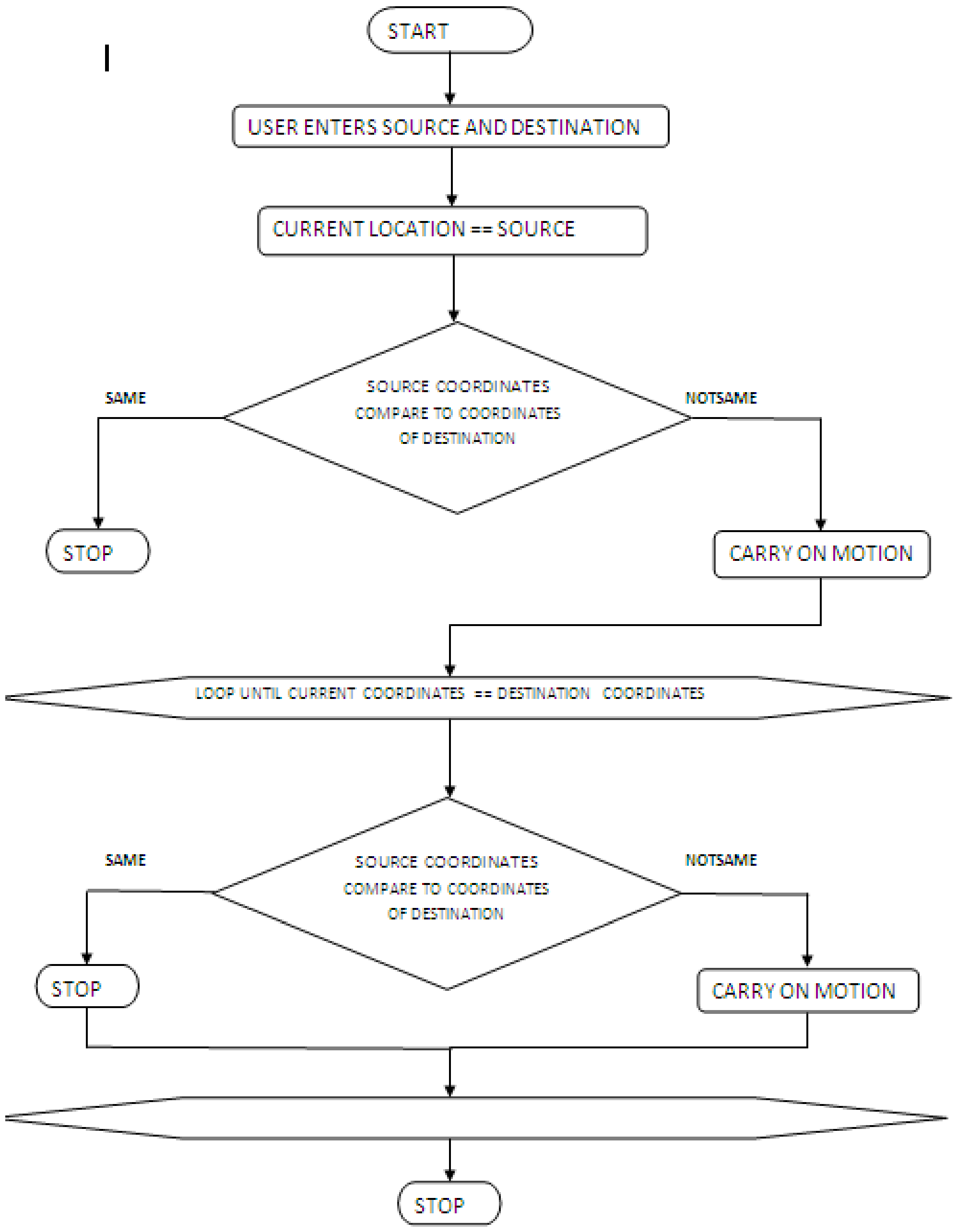

START
USER ENTERS SOURCE AND DESTINATION
CURRENT LOCATION == SOURCE
SOURCE COORDINATES COMPARE TO COORDINATES OF DESTINATION
SAME
NOTSAME
STOP
CARRY ON MOTION
LOOP UNTIL CURRENT COORDINATES == DESTINATION COORDINATES
SOURCE COORDINATES COMPARE TO COORDINATES OF DESTINATION
SAME
NOTSAME
STOP
CARRY ON MOTION
STOP

Figure 4-2 Flowchart for GPS

This diagram is a flowchart that shows various program flow, branches and conditional coverage through the GPS flow of the program. It also highlights looping while the GPS module is implemented.

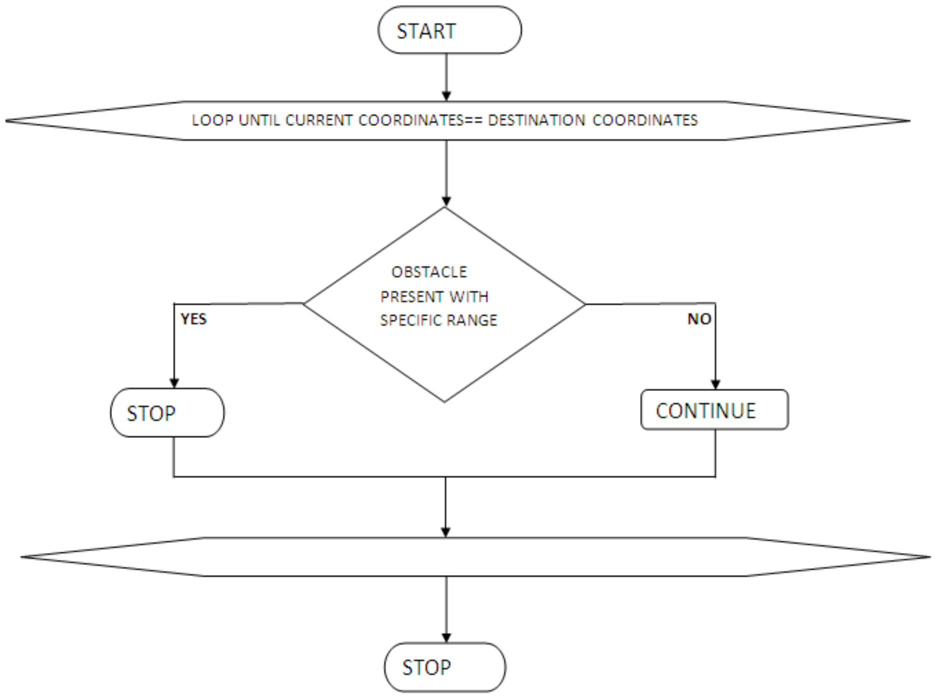

START
LOOP UNTIL CURRENT COORDINATES== DESTINATION COORDINATES
OBSTACLE PRESENT WITH SPECIFIC RANGE
YES
NO
STOP
CONTINUE
STOP

Figure 4-3 Flowchart for Obstacle detection

This diagram is a flowchart that shows various program flow, branches and conditional coverage through the obstacle detection of the program. It also highlights looping while the Obstacle module is implemented.

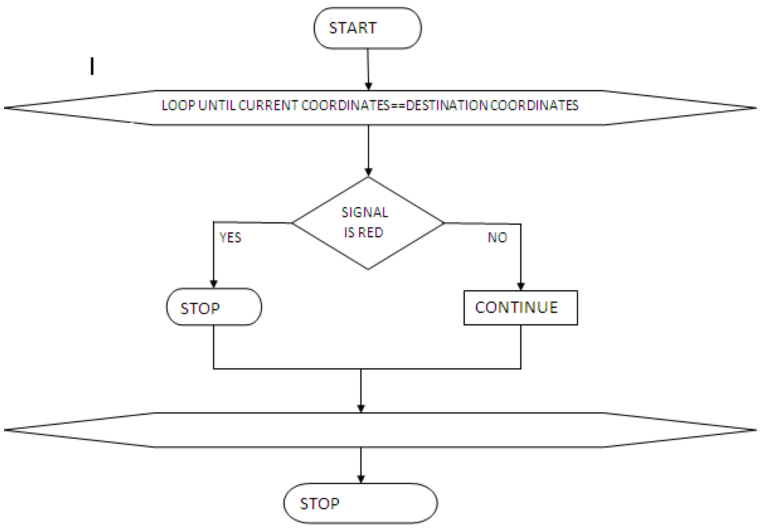


Figure 4-4 Flowchart for colour sensor

This diagram is a flowchart that shows various program flow, branches and conditional coverage through the colour sensor module of the program. It also highlights looping while the GPS module is implemented.

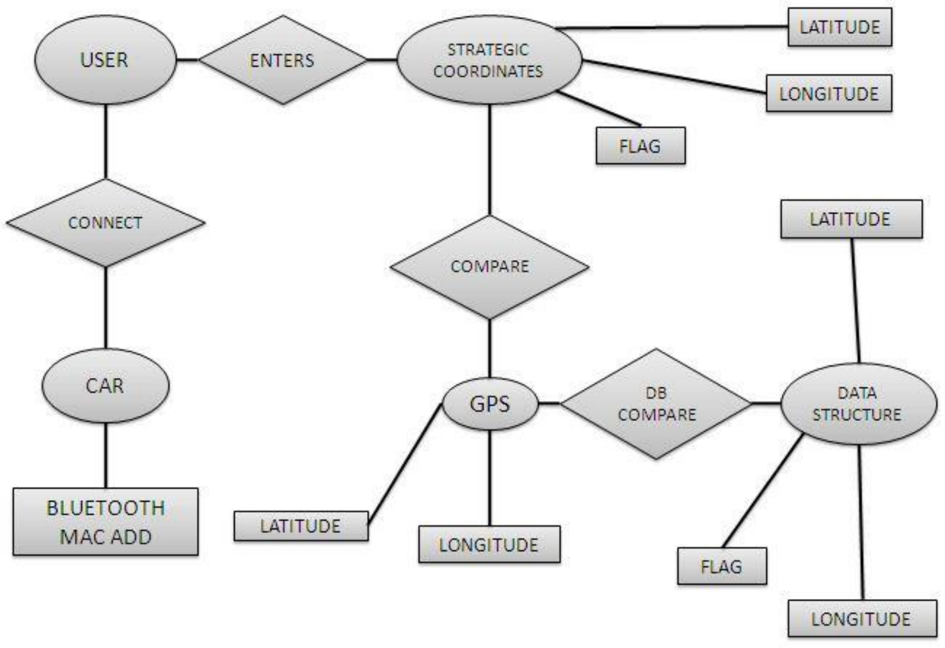


Figure 4-5 ER diagram for GPS

This diagram is an entity relationship diagram for the GPS function of the car. It highlights various entities, and the relationships existent between those entities. It also highlights attributes between those entities.

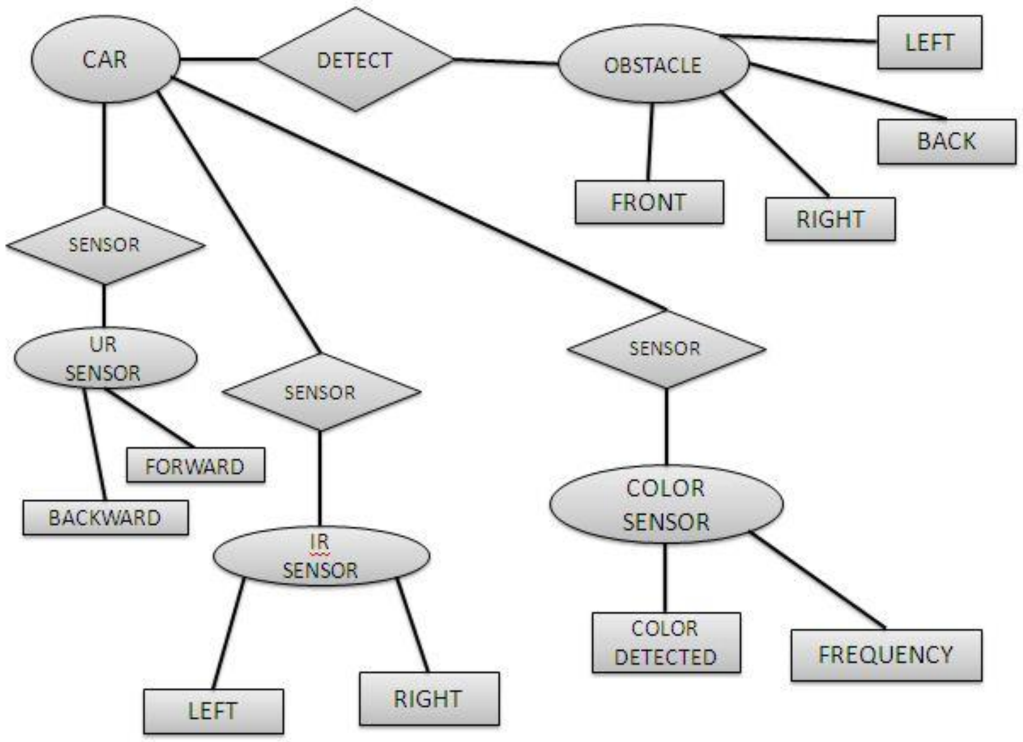


Figure 4-6 ER diagram for obstacle detection

This diagram is an entity relationship diagram for the GPS function of the car. It highlights various entities, and the relationships existent between those entities. It also highlights attributes between those entities.

CHAPTER 5

Implementation

Implementation

5. 1 GPS coding

Car receives strategic coordinates from user via android app. Its default motion is forward. While comparing its current location with the coordinates received, if it meets any of the latter, the respective motion flag is checked and motion is changed accordingly. This continues till we reach destination.

Location.java

```
package com.example.newcastel.carcontroller;


import android.Manifest;
import android.annotation.SuppressLint;
import android.app.AlertDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Build;
import android.provider.Settings;
import android.support.annotation.RequiresApi;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.math.BigDecimal;
import java.util.Set;
import java.util.UUID;
public class Location extends AppCompatActivity{
int count, count2, flag=0;
private Button b, b2;
private TextView t;
private LocationManagerlocationManager;
private LocationListener listener;
private final String DEVICE_ADDRESS = "00:21:13:01:FB:20"; //MAC Address of
Bluetooth Module
// private final String DEVICE_ADDRESS = "00:21:13:01:F8:54"; //MAC Address
of Bluetooth Module
private final UUID PORT_UUID = UUID.fromString("00001101-0000-1000-8000-
00805f9b34fb");

private BluetoothDevice device;
private BluetoothSocket socket;
private OutputStreamoutputStream;
private InputStreaminputStream;
EditText pt1, pt2, pt3;
Button go;
String command;
long lat[]=new long[20];
long lon[]=new long[20];
String cords[]=new String[20];
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
```

```
setContentView(R.layout.location);
count2=0;
pt1=(EditText)findViewById(R.id.editText);
pt2=(EditText)findViewById(R.id.editText2);
pt3=(EditText)findViewById(R.id.editText3);
go=(Button)findViewById(R.id.button2);
t = (TextView) findViewById(R.id.text_location);
b = (Button) findViewById(R.id.button_location);
b2=(Button) findViewById(R.id.button);

locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
count=0;

go.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View view) {
lat[count2]= Long.parseLong((pt1.getText().toString()));
lon[count2]=Long.parseLong(pt2.getText().toString());
cords[count2]=pt3.getText().toString();
count2++;
}
});

listener = new LocationListener() {
@Override
public void onLocationChanged(android.location.Location location) {
if (flag == 0)
{       try {
byte b[] = "0".getBytes();
outputStream.write(b);
} catch (IOException e) {
e.printStackTrace();
}
flag=1;
}

double latitude=location.getLatitude();
double longitude=location.getLongitude();
long lat2= (long) (latitude*10000);
long lon2=(long) (longitude*10000);
t.setText(lat2+ " "+lon2);

if(lat2==lat[count] && lon2==lon[count])
{
try {
outputStream.write(cords[count].getBytes());
count++;
} catch (IOException e) {
e.printStackTrace();
}
}
}

@Override
public void onStatusChanged(String s, int i, Bundle bundle) {
}

@Override
public void onProviderEnabled(String s) {
}

@Override
public void onProviderDisabled(String s) {
Intent i = new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
startActivity(i);
}
};
```

```
configure_button();
b2.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
if(BTinit())
{
BTconnect();
}
}
});
}

@Override
public void onRequestPermissionsResult(intrequestCode, @NonNull String[] permissions, @NonNullint[] grantResults) {
switch (requestCode){
case 10:
configure_button();
break;
default:
break;
}
}
voidconfigure_button(){
// first check for permissions
if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED&&ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
if (Build.VERSION.SDK_INT>= Build.VERSION_CODES.M) {
requestPermissions(new String[]{Manifest.permission.ACCESS_COARSE_LOCATION,Manifest.permission.ACCESS_FINE_LOCATION,Manifest.permission.INTERNET}
,10);
}
}
// this code won't execute IF permissions are not allowed, because in the line above
there is return statement.
b.setOnClickListener(new View.OnClickListener() {
@SuppressLint("MissingPermission")
@Override
public void onClick(View view) {
//noinspectionMissingPermission
locationManager.requestLocationUpdates("gps", 100, 0, listener);
t.setText("Fetching location...");
try {
outputStream.write("8".getBytes());
} catch (IOException e) {
e.printStackTrace();
}
}
});
}

publicbooleanBTinit()
{
boolean found = false;
BluetoothAdapterbluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
if(bluetoothAdapter == null) //Checks if the device supports bluetooth
{
Toast.makeText(getApplicationContext(), "Device doesn't support bluetooth",
Toast.LENGTH_SHORT).show();
}
if(!bluetoothAdapter.isEnabled()) //Checks if bluetooth is enabled. If not, the program
will ask permission from the user to enable it
{
Intent enableAdapter = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
startActivityForResult(enableAdapter,0);
try
{
Thread.sleep(1000);
}
catch(InterruptedException e)
{
e.printStackTrace();
}
}
Set<BluetoothDevice>bondedDevices = bluetoothAdapter.getBondedDevices();
if(bondedDevices.isEmpty()) //Checks for paired bluetooth devices
{
Toast.makeText(getApplicationContext(), "Please pair the device first",
Toast.LENGTH_SHORT).show();
}
else
{
for(BluetoothDevice iterator : bondedDevices)
{
if(iterator.getAddress().equals(DEVICE_ADDRESS))
{
device = iterator;
found = true;
break;
}
}
}
return found;
}

publicbooleanBTconnect()
```

```
{
boolean connected = true;

try
{
socket = device.createRfcommSocketToServiceRecord(PORT_UUID); //Creates a socket to handle the outgoing connection
socket.connect();

Toast.makeText(getApplicationContext(),
"Connection to bluetooth device successful", Toast.LENGTH_SHORT).show();
}
catch(IOException e)
{
e.printStackTrace();
connected = false;
}
if(connected)
{
try
{
outputStream = socket.getOutputStream(); //gets the output stream of the socket
}
catch(IOException e)
{
e.printStackTrace();
}
}
return connected;
}
}
```

Location.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent">

<GridLayout
android:layout_width="match_parent"
android:layout_height="match_parent">

<LinearLayout
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical">

<Button
android:id="@+id/button"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:layout_marginTop="40dp"
android:text="Connect to car" />

<Button
android:id="@+id/button_location"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:layout_marginTop="16dp"
android:text="Start with location" />

<TextView
android:id="@+id/text_location"
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:layout_centerVertical="true"
android:textSize="16dp" />

<LinearLayout
android:layout_width="match_parent"
android:layout_height="100dp"
android:orientation="horizontal"
android:layout_marginTop="32dp"
>

<EditText
android:id="@+id/editText"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:ems="10"
android:inputType="number" />

<EditText
android:id="@+id/editText2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:ems="10"
android:inputType="number" />

<EditText
android:id="@+id/editText3"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
```

```
android:ems="10"
android:inputType="number" />
</LinearLayout>

<Button
android:id="@+id/button2"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="go........" />

</LinearLayout>
</GridLayout>

</RelativeLayout>
```

gps.ino

```
char a;
const int motorPin1=5;
const int motorPin2=6;
const int motorPin4=11;
const int motorPin3=10;

int flag=0;
void setup() {
  // put your setup code here, to run once:
pinMode(motorPin1, OUTPUT);
pinMode(motorPin2, OUTPUT);
pinMode(motorPin3, OUTPUT);
pinMode(motorPin4, OUTPUT);

Serial.begin(38400);
}
```

```
void loop() {

  // put your main code here, to run repeatedly:
a=Serial.read();
switch(a)
{
case '2': Serial.println(a);
left();
delay(3000);
forward();
break;

case '1': Serial.println(a);
right();
delay(3000);
forward();
break;
case '0': Serial.println(a);
forward();

break;

case '9' : Serial.println(a);
stopp();
break;



case '8' : Serial.println(a);
break;


}
}
void right()
{
```

```
digitalWrite(motorPin1, HIGH); // right
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
}

void left()
{
digitalWrite(motorPin1, HIGH); // left
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, LOW);
}
void forward()
{
digitalWrite(motorPin1, HIGH); // forward
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
}
voidstopp()
{
digitalWrite(motorPin1,LOW);
digitalWrite(motorPin2,LOW);
digitalWrite(motorPin3,LOW);
digitalWrite(motorPin4,LOW);
flag=0;
}
```

5. 2 Obstacle detection coding

Car moves naturally. If obstacle is detected then it stops or adjusts motion, else it keeps moving.

ur_sensor.ino

```
const int motorPin1 = 5;
const int motorPin2 = 6;
const int motorPin3 = 10;
const int motorPin4 = 11;
const int echoPin1 = 3;
const int trigPin1 = 2;
const int echoPin2 = 7;
const int trigPin2 = 8;
intr_IR = 13;
intl_IR = 12;

int duration1, distance1, duration2, distance2;

void setup(){
Serial.begin(9600);

pinMode(l_IR,INPUT);
pinMode(r_IR,INPUT);
pinMode(motorPin1, OUTPUT);
pinMode(motorPin2, OUTPUT);
pinMode(motorPin3, OUTPUT);
pinMode(motorPin4, OUTPUT);
pinMode(echoPin1, INPUT);
pinMode(trigPin1, OUTPUT);
pinMode(echoPin2, INPUT);
pinMode(trigPin2, OUTPUT);

}

void loop(){
```

```
digitalWrite(trigPin1, LOW);
delayMicroseconds(500);
digitalWrite(trigPin1, HIGH);
delayMicroseconds(500);
digitalWrite(trigPin1, LOW);
  duration1 = pulseIn(echoPin1, HIGH);
  distance1 = (duration1 / 2) / 29.1;

digitalWrite(trigPin1, LOW);
delayMicroseconds(500);
digitalWrite(trigPin2, HIGH);
delayMicroseconds(500);
digitalWrite(trigPin2, LOW);
  duration2 = pulseIn(echoPin2, HIGH);
  distance2 = (duration2 / 2) / 29.1;

if (distance2>20)
 {
forward();
 }
else if(distance2<20)
  {
stopm();
if(distance1>20)
    {
backward();
  }
else if(distance1<20)
  {
stopm();
  }
  }

  //ir();
```

```
}
voidir(){
int l1=digitalRead(l_IR);
int r1=digitalRead(r_IR);
if((l1==LOW)&&(r1==LOW) )
   {
     //stop
stopm();
   }
else if((l1==LOW)&&(r1==HIGH)) {
       //turns right
right();
    }
else if((l1==HIGH)&&(r1==LOW)) {
     //left
left();
    }
else if((l1==HIGH)&&(r1==HIGH)) {
     //forward
forward();
    }
}
void forward(){
digitalWrite(motorPin1, HIGH); // forward
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
}
void backward(){
digitalWrite(motorPin1, LOW); // backward
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, HIGH);
}
```

```
void right(){

digitalWrite(motorPin1, HIGH); // right
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
}
void left(){
digitalWrite(motorPin1, HIGH); // left
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, LOW);
}
voidstopm(){
digitalWrite(motorPin1, LOW); // stop
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, LOW);
}
```

5. 3 Colour sensor coding

Car moves naturally. If red colour is detected then it stops, else it keeps moving.

final_signal_color.ino

```
//Motor A
const int motorPin1  = 5;  // Pin 15 of L293
const int motorPin2  = 6;  // Pin 10 of L293
//Motor B
const int motorPin4  = 11; // Pin  7 of L293
const int motorPin3  = 10;  // Pin  2 of L293

// color sensor
const int S0= A0;
const int S1=A1;
const int S2 =A2;
const int S3 =A3;
const int sensorOut=4;
const int leftir=13;
const int rightir=12;
int left=0;
int right=0;
// Stores frequency read by the photodiodes
int redFrequency = 0;
int greenFrequency = 0;
int blueFrequency = 0;

// Stores the red. green and blue colors
int redColor = 0;

int greenColor = 0;
int blueColor = 0;



void setup(){
pinMode(S0, OUTPUT);
pinMode(S1, OUTPUT);
pinMode(S2, OUTPUT);
pinMode(S3, OUTPUT);
pinMode(left, INPUT);
pinMode(right,INPUT );
  // Setting the sensorOut as an input
pinMode(sensorOut, INPUT);

  // Setting frequency scaling to 20%
digitalWrite(S0,HIGH);
digitalWrite(S1,LOW);
```

```
Serial.begin(9600);
pinMode(motorPin1, OUTPUT);
pinMode(motorPin2, OUTPUT);
pinMode(motorPin3, OUTPUT);
pinMode(motorPin4, OUTPUT);
}

void loop(){
digitalWrite(motorPin1,HIGH);
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
digitalWrite(S2,LOW);
digitalWrite(S3,LOW);

while(true)
 {
redFrequency = pulseIn(sensorOut, LOW);
left=digitalRead(leftir); // iR left pin number
right=digitalRead(rightir);// IR right pin number
Serial.println(redFrequency);

if((redFrequency>=150 &&redFrequency<=280) /*|| left==LOW || right==LOW*/)
{
Serial.println(" - RED detected!");
digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, LOW);

  }
else
  {
break;
```

```
  }
 }
}
```

5. 4 Manual car control coding

If problem occurs, car can be controlled manually.

MainActivity.java

```
import android.annotation.SuppressLint;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Set;
import java.util.UUID;


public class MainActivity extends AppCompatActivity {
private final String DEVICE_ADDRESS = "00:21:13:01:FB:20"; //MAC Address of Bluetooth Module
private final UUID PORT_UUID = UUID.fromString("00001101-0000-1000-8000-00805f9b34fb");
private BluetoothDevice device;
private BluetoothSocket socket;
public OutputStreamoutputStream;
private InputStreaminputStream;

Button forward_btn, forward_left_btn, forward_right_btn, reverse_btn,
bluetooth_connect_btn, button_map;
TextView textView;
String command; //string variable that will store value to be transmitted to the
bluetooth module

@SuppressLint("ClickableViewAccessibility")
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

//declaration of button variables
forward_btn = (Button) findViewById(R.id.forward_btn);
forward_left_btn = (Button) findViewById(R.id.forward_left_btn);
forward_right_btn = (Button) findViewById(R.id.forward_right_btn);
reverse_btn = (Button) findViewById(R.id.reverse_btn);
bluetooth_connect_btn = (Button) findViewById(R.id.bluetooth_connect_btn);
button_map = (Button) findViewById(R.id.map);
textView = (TextView) findViewById(R.id.textView);

button_map.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View view) {
Intent new_act = new Intent(MainActivity.this,Location.class);
startActivity(new_act);
}
});
```

```
forward_btn.setOnLongClickListener(new View.OnLongClickListener() {
@Override
public Boolean onLongClick(View view) {
command="1";
try {
outputStream.write(command.getBytes());
} catch (IOException e) {
e.printStackTrace();
}
return false;
}
});


//OnTouchListener code for the forward button (button long press)
forward_btn.setOnTouchListener(new View.OnTouchListener() {
@Override
publicbooleanonTouch(View v, MotionEvent event) {

/*if (event.getAction() == MotionEvent.ACTION_DOWN) //MotionEvent.ACTION_DOWN is when you hold a button down
{
command = "1";

try
{
outputStream.write(command.getBytes()); //transmits the value of command to the bluetooth module
//outputStream.write((lattitude+longitude).getBytes());
}
catch (Exception e)
{
e.printStackTrace();
```

```
        Toast.makeText(getApplicationContext(), "No Car Connected to Control..!!!", 
Toast.LENGTH_SHORT).show();
        }
        }
        else */ if(event.getAction() == MotionEvent.ACTION_UP) 
//MotionEvent.ACTION_UP is when you release a button
        {
        command = "0";
        try
        {
        outputStream.write(command.getBytes());
        //outputStream.write(lattitude.getBytes());
        }
        catch(Exception e)
        {
        e.printStackTrace();
        Toast.makeText(getApplicationContext(), "No Car Connected to Control..!!!", 
Toast.LENGTH_SHORT).show();
        }
        }
        return false;
        }
        });

        //OnTouchListener code for the reverse button (button long press)
        reverse_btn.setOnTouchListener(new View.OnTouchListener(){
        @Override
        publicbooleanonTouch(View v, MotionEvent event)
        {
        if(event.getAction() == MotionEvent.ACTION_DOWN)
        {
        command = "2";

        try
```

```
{
outputStream.write(command.getBytes());
}
catch (Exception e)
{
e.printStackTrace();
Toast.makeText(getApplicationContext(), "No Car Connected to Control..!!!",
Toast.LENGTH_SHORT).show();
}
}
else if(event.getAction() == MotionEvent.ACTION_UP)
{
command = "10";
try
{
outputStream.write(command.getBytes());
}
catch(Exception e)
{
e.printStackTrace();
Toast.makeText(getApplicationContext(), "No Car Connected to Control..!!!",
Toast.LENGTH_SHORT).show();
}

}
return false;
}
});



//OnTouchListener code for the forward left button (button long press)
forward_left_btn.setOnTouchListener(new View.OnTouchListener(){
@Override
publicbooleanonTouch(View v, MotionEvent event)
```

```
{
if(event.getAction() == MotionEvent.ACTION_DOWN)
{
command = "3";

try
{
outputStream.write(command.getBytes());
}
catch (Exception e)
{
e.printStackTrace();
Toast.makeText(getApplicationContext(), "No Car Connected to Control..!!!",
Toast.LENGTH_SHORT).show();
}
}
else if(event.getAction() == MotionEvent.ACTION_UP)
{
command = "10";
try
{
outputStream.write(command.getBytes());
}
catch(Exception e)
{
e.printStackTrace();
Toast.makeText(getApplicationContext(), "No Car Connected to Control..!!!",
Toast.LENGTH_SHORT).show();
}


}
return false;
}
});
```

```
//OnTouchListener code for the forward right button (button long press)
forward_right_btn.setOnTouchListener(new View.OnTouchListener(){
@Override
publicbooleanonTouch(View v, MotionEvent event)
{
if(event.getAction() == MotionEvent.ACTION_DOWN)
{
command = "4";

try
{
outputStream.write(command.getBytes());
}
catch (Exception e)
{
e.printStackTrace();
Toast.makeText(getApplicationContext(), "No Car Connected to Control..!!!", Toast.LENGTH_SHORT).show();
}
}
else if(event.getAction() == MotionEvent.ACTION_UP)
{
command = "10";
try
{
outputStream.write(command.getBytes());
}
catch(Exception e)
{
e.printStackTrace();
Toast.makeText(getApplicationContext(), "No Car Connected to Control..!!!", Toast.LENGTH_SHORT).show();
}
}
return false;
}
});

//Button that connects the device to the bluetooth module when pressed
bluetooth_connect_btn.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {

if(BTinit())
{
BTconnect();
}

}
});

}

//Initializes bluetooth module
publicbooleanBTinit()
{
boolean found = false;

BluetoothAdapterbluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

if(bluetoothAdapter == null) //Checks if the device supports bluetooth
{
Toast.makeText(getApplicationContext(), "Device doesn't support bluetooth",
Toast.LENGTH_SHORT).show();
}
```

```
        if(!bluetoothAdapter.isEnabled()) //Checks if bluetooth is enabled. If not, the program
will ask permission from the user to enable it
        {
        Intent enableAdapter = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableAdapter,0);

        try
        {
        Thread.sleep(1000);
        }
        catch(InterruptedException e)
        {
        e.printStackTrace();
        }
        }

        Set<BluetoothDevice>bondedDevices = bluetoothAdapter.getBondedDevices();

        if(bondedDevices.isEmpty()) //Checks for paired bluetooth devices
        {
        Toast.makeText(getApplicationContext(), "Please pair the device first",
Toast.LENGTH_SHORT).show();
        }
        else
        {
        for(BluetoothDevice iterator : bondedDevices)
        {
        if(iterator.getAddress().equals(DEVICE_ADDRESS))
        {
        device = iterator;
        found = true;
        break;
        }
```

```
}
}

return found;
}

publicbooleanBTconnect()
{
boolean connected = true;

try
{
socket = device.createRfcommSocketToServiceRecord(PORT_UUID); //Creates a
socket to handle the outgoing connection
socket.connect();

Toast.makeText(getApplicationContext(),
"Connection to bluetooth device successful", Toast.LENGTH_SHORT).show();
}
catch(IOException e)
{
e.printStackTrace();
connected = false;
}

if(connected)
{
try
{
outputStream = socket.getOutputStream(); //gets the output stream of the socket
}
catch(IOException e)
{
e.printStackTrace();
```

```
}
}

return connected;
}

@Override
protected void onStart()
{

super.onStart();
}

}
```

<u>activity_main.xml</u>

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="@drawable/universe"
android:orientation="vertical"
tools:context="com.example.newcastel.carcontroller.MainActivity">

<LinearLayout
android:id="@+id/la1"
android:layout_width="match_parent"
android:layout_height="60dp"
android:orientation="vertical">

<Button
android:id="@+id/bluetooth_connect_btn"
android:layout_width="wrap_content"
```

```
android:layout_height="50dp"
android:layout_gravity="center"
android:text="Connect to Car" />
</LinearLayout>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="horizontal"
>

<RelativeLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical">

<Button
android:id="@+id/map"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"

android:layout_gravity="center"
android:text="GPS" />
</RelativeLayout>

</LinearLayout>




<RelativeLayout
android:layout_width="match_parent"
android:layout_height="match_parent"
```

```
android:orientation="vertical">
<Button
android:id="@+id/forward_btn"
android:layout_width="90dp"
android:layout_height="80dp"
android:layout_alignParentTop="true"
android:layout_marginTop="30dp"
android:layout_toLeftOf="@+id/forward_right_btn"
android:layout_toStartOf="@+id/forward_right_btn"
android:background="#527a8e"
android:text="Forward" />

<Button
android:id="@+id/forward_left_btn"
android:layout_width="90dp"
android:layout_height="80dp"
android:layout_below="@+id/forward_btn"
android:layout_toLeftOf="@+id/forward_btn"
android:layout_toStartOf="@+id/forward_btn"
android:background="#527a8e"
android:text="LEFT" />

<Button
android:id="@+id/forward_right_btn"
android:layout_width="90dp"
android:layout_height="80dp"
android:layout_above="@+id/reverse_btn"
android:layout_toEndOf="@+id/reverse_btn"
android:layout_toRightOf="@+id/reverse_btn"
android:background="#527a8e"
android:text="RIGHT" />

<Button
android:id="@+id/reverse_btn"
```

```
android:layout_width="90dp"
android:layout_height="80dp"
android:layout_below="@+id/forward_left_btn"
android:layout_centerHorizontal="true"
android:layout_marginTop="0dp"
android:background="#527a8e"
android:text="REVERSE" />

<TextView
android:id="@+id/textView"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_alignParentBottom="true"
android:text="Group Members: Himanshu, Samruddhi, Ankita, Kunjal   "
android:textAlignment="viewEnd"
android:textColor="#bc443c"
android:textStyle="italic" />

</RelativeLayout>
</LinearLayout>
```

RC_Car.ino

```
char a;
const int motorPin1=5;
const int motorPin2=6;
const int motorPin4=11;
const int motorPin3=10;

void setup() {
pinMode(motorPin1, OUTPUT);
pinMode(motorPin2, OUTPUT);
pinMode(motorPin3, OUTPUT);
pinMode(motorPin4, OUTPUT);
```

```
Serial.begin(38400);
}

void loop() {
   {
  a=Serial.read();
  // Serial.println(a);
switch(a)
  {
case '1': Serial.println("1");

digitalWrite(motorPin1,HIGH);
digitalWrite(motorPin2,LOW);
digitalWrite(motorPin3,LOW);
digitalWrite(motorPin4,HIGH);
delay(10000);

break;

case '2':Serial.println("2");
digitalWrite(motorPin1, LOW); // backward
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, HIGH);
delay(10000);
break;
case '3': Serial.println("3");
digitalWrite(motorPin1, HIGH); // left
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, LOW);
delay(10000);

break;
case '4': Serial.println("4");
digitalWrite(motorPin1, HIGH); // right
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
delay(10000);

break;
default:
digitalWrite(motorPin1, LOW); //stop
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, LOW);
break;
        }
  }
}
```

CHAPTER 6

Testing

Testing

We performed White Box Testing for the project, wherein we designed test case scenarios based on the coding of Android. We detected bugs like:

1. Car not receiving Bluetooth data. We fixed it by changing rate of communication baud in the Arduino IDE.

We performed Unit Testing for the project, wherein we tested different modules of our project separately. The various modules tested were:

1. GPS feature
2. Random motion of car- motion was too slow. Bug fixed by increasing voltage.
3. UR sensor
4. IR sensor
5. Colour sensor module- red colour was not being detected. Bug fixed by changing frequency.

We performed Integration Testing for the project, wherein we tested integrating UR and IR features. The problem was that they were not working properly. This was resolved by using appropriate baud rate.

For user acceptance testing, we showed our project to our respected guide, who advised us to use more voltage to increase speed of our car.

CHAPTER 7

Output/ Screenshots

Output and Screenshots

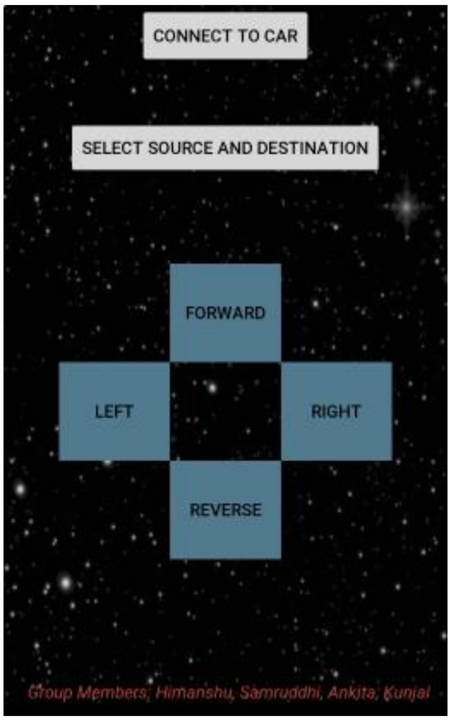

CONNECT TO CAR
SELECT SOURCE AND DESTINATION
FORWARD
LEFT
RIGHT
REVERSE
Group Members, Himanshu, Samruddhi, Ankita, Kunjal

Figure 7-1 Main screen of app

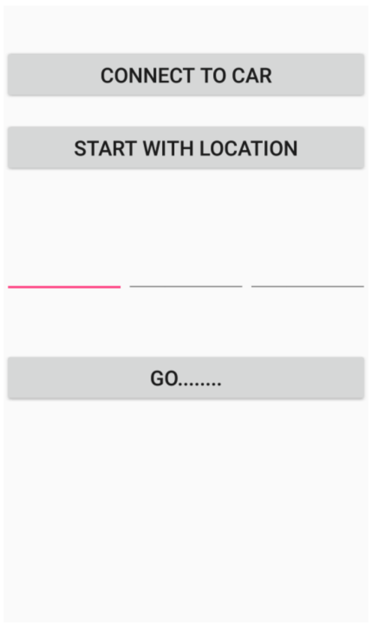


Figure 7-2 Form for GPS

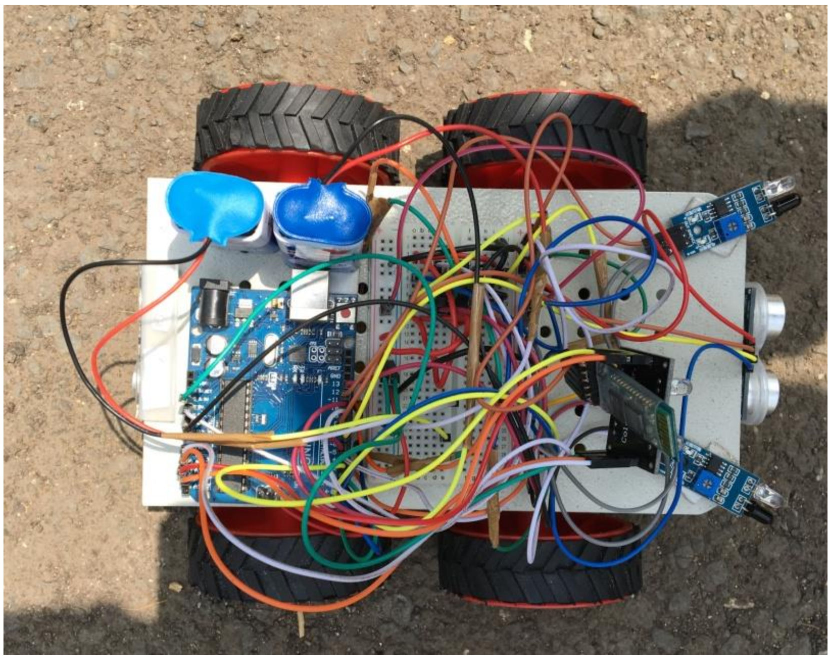


Figure 7-3 Top View of Car

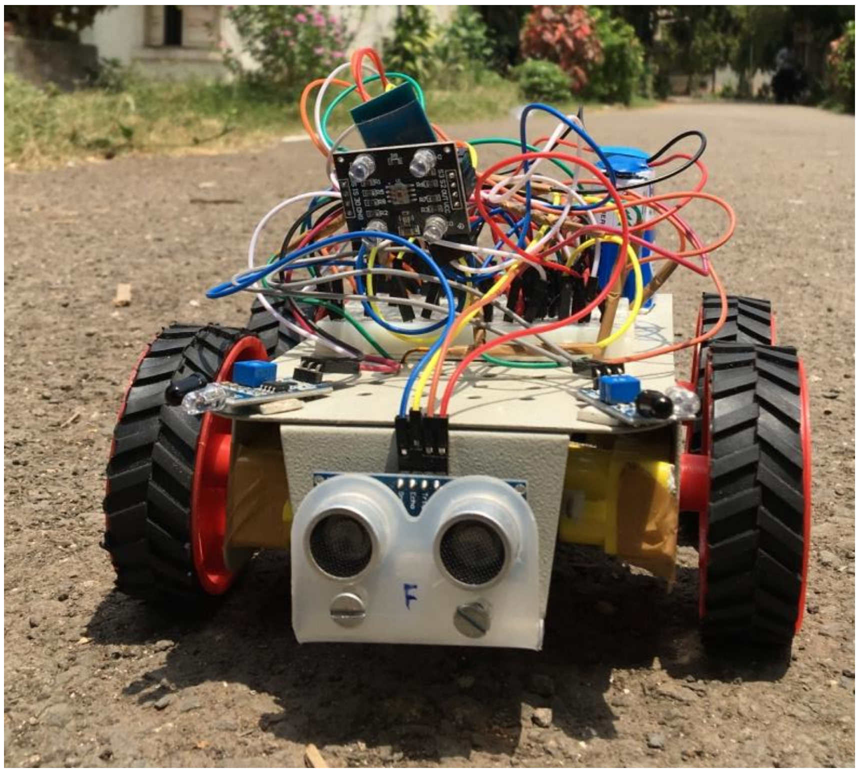


Figure 7-4 Front view of sensors

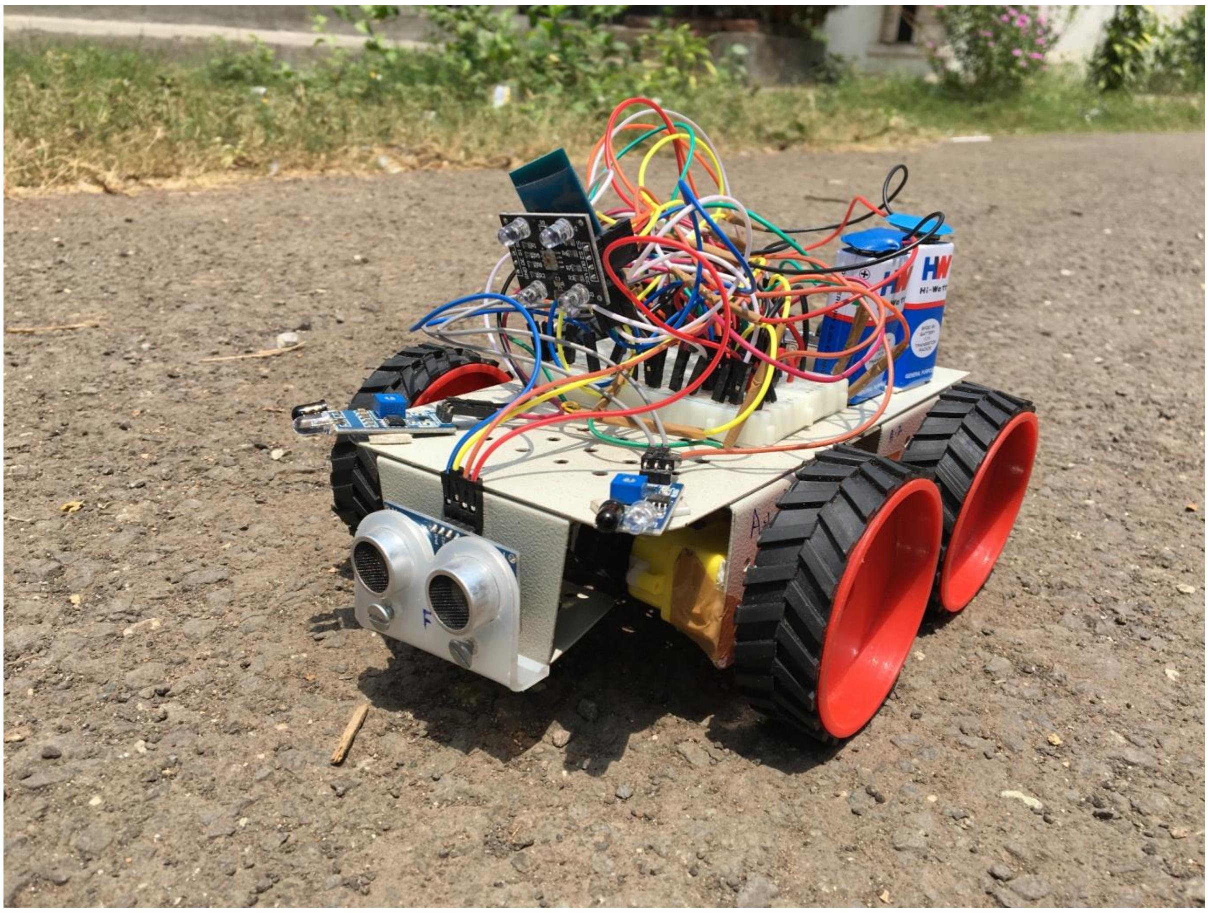


Figure 7-5 Full Car

CHAPTER 8

Conclusion

Conclusion

In conclusion, there are many strong socio-economic motivators for the adoption of autonomous vehicles. Human safety, infrastructure efficiency, quality of life and a ready customer base are just a few of the key factors that will help make self-driving cars a reality. Yet to be solved are the complex issues associated with society. Gradual introduction of these features combined with strong economic motivators are sure to overcome such obstacles.

The prototype of this model was developed firstly by making a Bluetooth controlled car, which could be controlled through an Android application. Then the concept of obstacle detection was developed through ultrasonic and infrared sensors, which detected obstacles at the front and side. Further color sensor detected red colour and the car stopped. Finally, the GPS feature with static coordinates was implemented, which led to successful completion of the project.

Through a prototyping model we aim at boosting the concept of autonomous cars in the society.

CHAPTER 9

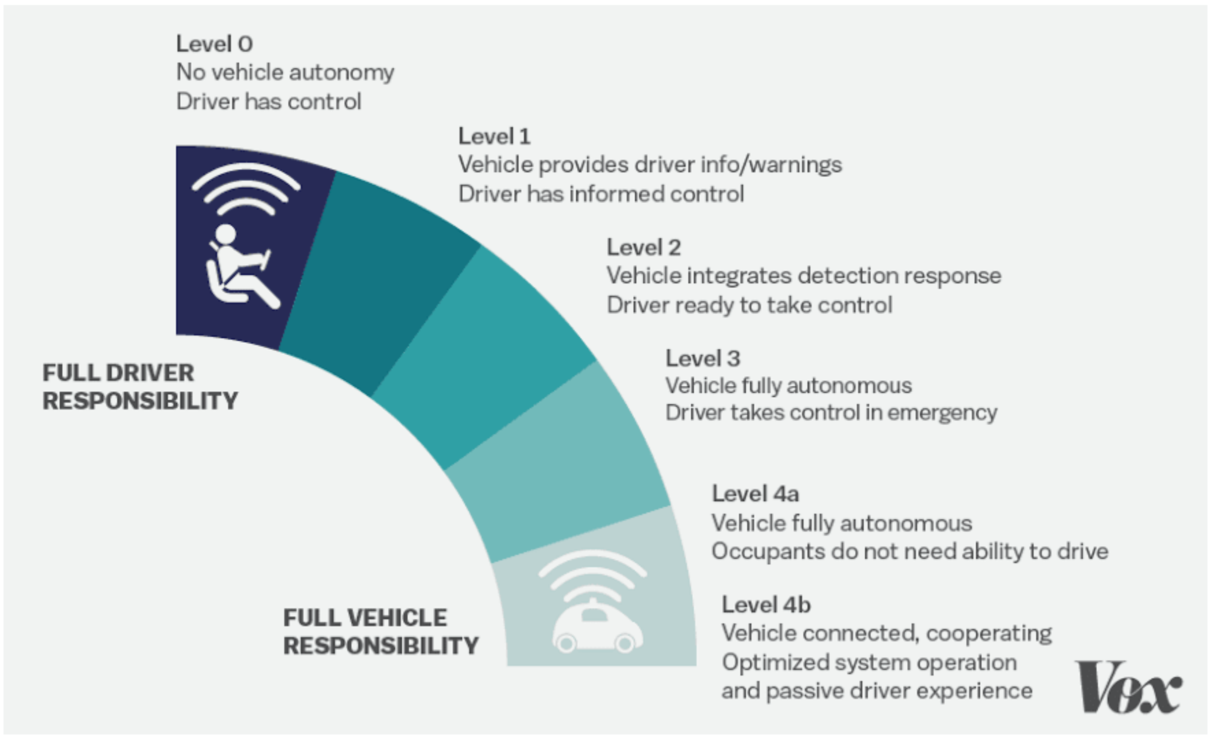
Future Scope

Future Scope

We aim at adding additional safety features and advanced detection of environmental traffic and other controls. Google Maps will guide the car, instead of static coordinates.

In the near future, such autonomous vehicles will be an indispensable part of transport systems.

We also aim at working on enhancing motor speed and provide acceleration.



Level 0
No vehicle autonomy
Driver has control
Level 1
Vehicle provides driver info/warnings
Driver has informed control
Level 2
Vehicle integrates detection response
Driver ready to take control
Level 3
Vehicle fully autonomous
Driver takes control in emergency
Level 4a
Vehicle fully autonomous
Occupants do not need ability to drive
Level 4b
Vehicle connected, cooperating
Optimized system operation
and passive driver experience
FULL DRIVER RESPONSIBILITY
FULL VEHICLE RESPONSIBILITY
Vox

Figure 9-1 Future Scope

Bibliography and References

Bibliography and References

1. http://www.beginnertopro.in
2. http://www.instructables.com
3. https://howtomechatronics.com
4. Autonomous decision making for a driver-less car- IEEE
5. https://create.arduino.cc
6. https://pastebin.com
7. https://www.youtube.com
8. https://www.androidhive.info/
9. Programming Arduino Getting Started with Sketches, Simon Monk
10. Android Programming: The Big Nerd Ranch Guide
11. How Autonomous Cars can be made safer, Financial Express, 01-Apr-2017.
12. Indian Company Eyes Driver's Seat In Driverless Vehicle Technology, NDTVAuto.com, 18-Jun-2017.