**IOT BASED ROBOT CAR (4WD)**

# A Project Work

*Submitted in partial fulfilment for the award of the degree of*

**BACHELOR OF ENGINEERING**

**IN**

**Computer Science, IBM: Cloud Computing and Big Data**

### Submitted by:

# GUNEET SIDHU 17BCS4060

**Under the Supervision of:**

# Asst. Prof. Suman S Sarkar



**University Institute of Engineering**

CHANDIGARH UNIVERSITY, GHARUAN

Mohali, Punjab

## October, 2019

**DECLARATION**

We, members of **17AIT-PG20**, student of **‘Bachelor of Engineering in Computer Science [IBM: Cloud and Big Data]**, **2019**, Apex Institute of Technology, Chandigarh University,Punjab, hereby declare that the work presented in this Project Work entitled ‘**IOT Based Robot Car (4WD)’**is the outcome of our own bona fide work and is correct to the best of ourknowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

**17AITPG-20**

# GUNEET SIDHU 17BCS4060

**Date:**

**Place:** AIT, Chandigarh University, Mohali

**CERTIFICATE**

This is to certify that the work embodies in this dissertation entitled ***‘*IOT Based Robot Car (4WD)’** is being submitted by **17AIT-PG20** for partial fulfillment of the requirement for the award of **Bachelor of Engineering** in ***Computer Science [Cloud and Big Data]*** discipline to Apex Institute of Technology, Chandigarh University, Punjabduring the academic year 2019-2020 is a record of bonafide piece of work, undertaken by him/her the supervision of the undersigned.

**Approved and Supervised by**

**(Asst. Prof. Suman S Sarkar)**

**Program Leader, Big Data, AIT**

**Forwarded by**

**Project Coordinator**

**Asst. Prof. Gurjot Singh**

**EXTERNAL EXAMINER**

Signature of External Examiner

**Acknowledgement**

Whenever a module of work is completed successfully, a source of inspiration and guidance is always there for the student. We, hereby take the opportunity to thank those entire people who helped us in many different ways.

First and foremost, we are grateful to our training guide **Suman S Sarkar** at AIT Chandigarh University, for showing faith in our capability and providing able guidance and his generosity and advice extended to me throughout my project of IOT Based Robot Car.

Further, we would like to thank all other faculty and our friends for helping us in all measure of life and for their kind cooperation and moral support while working on this project.

Thanks.

# GUNEET SIDHU 17BCS4060

**ABSTRACT**

Robots remain the focus of researchers and developers, and now they are moving towards IoT based devices and mobile robots to take advantage of the different sensor enables facilities. A robot is a machine capable of carrying out a complex series of actions automatically, especially one programmable by a computer. A robot can be controlled by a human and can be modified by its functionality at runtime by the operator.

The purpose of the project entitled as “**IOT Based Robot Car (4WD)**” is to make a 4-wheel car using IOT components in minimum cost having maximum number of applications or features like:

1. Bluetooth connectivity and controlled by a smartphone.

2. Camera module including live streaming.

3. Face detection.

4. Ultrasonic sensor for obstacle avoiding …etc.

Such robots have a huge scope of adding more and more features and can be improved to a great extent. If connectivity over the internet is made then such robots can be use for security purpose as well.

**Table of Contents:**

|  |  |  |
| --- | --- | --- |
| S.No | Topics |  |
| 1. | Title Page. |  |
| 2. | Declaration. |  |
| 3. | Certificate. |  |
| 4. | Acknowledgement. |  |
| 5. | Abstract |  |
| 6. | **Chapter 1 Introduction.** |  |
| 7. | * 1.1 Problem Introduction. |  |
| 9. | - 1.2 Brief about IOT. |  |
| 10. | - 1.3 Requirement Specification. |  |
| 11. | 1.3.1 Hardware Requirement. |  |
| 12. | 1.3.2 Software Requirement. |  |
| 13. | **Chapter 2 Literature Survey.** |  |
| 14. | - 2.1 Existing System. |  |
| 15. | * 2.2 Proposed System. |  |
| 16. | * 2.3 Feasibility Study. |  |
| 17. | **Chapter 3 System Analysis and Design.** |  |
| 19. | * 3.1 Working Principle. |  |
| 20. | * 3.2 Pin diagrams. |  |
| 21. | - 3.3 Testing process. |  |
| 22. | **Chapter 4 Results/Conclusion and Future Scope.** |  |
| 23. | * 4.1 Conclusion. |  |
| 24. | - 4.2 Scope of future work. |  |
| 25. | **Chapter 5 Bibliography.** |  |
|  |  |  |

CHAPTER 1

**INTRODUCTION**

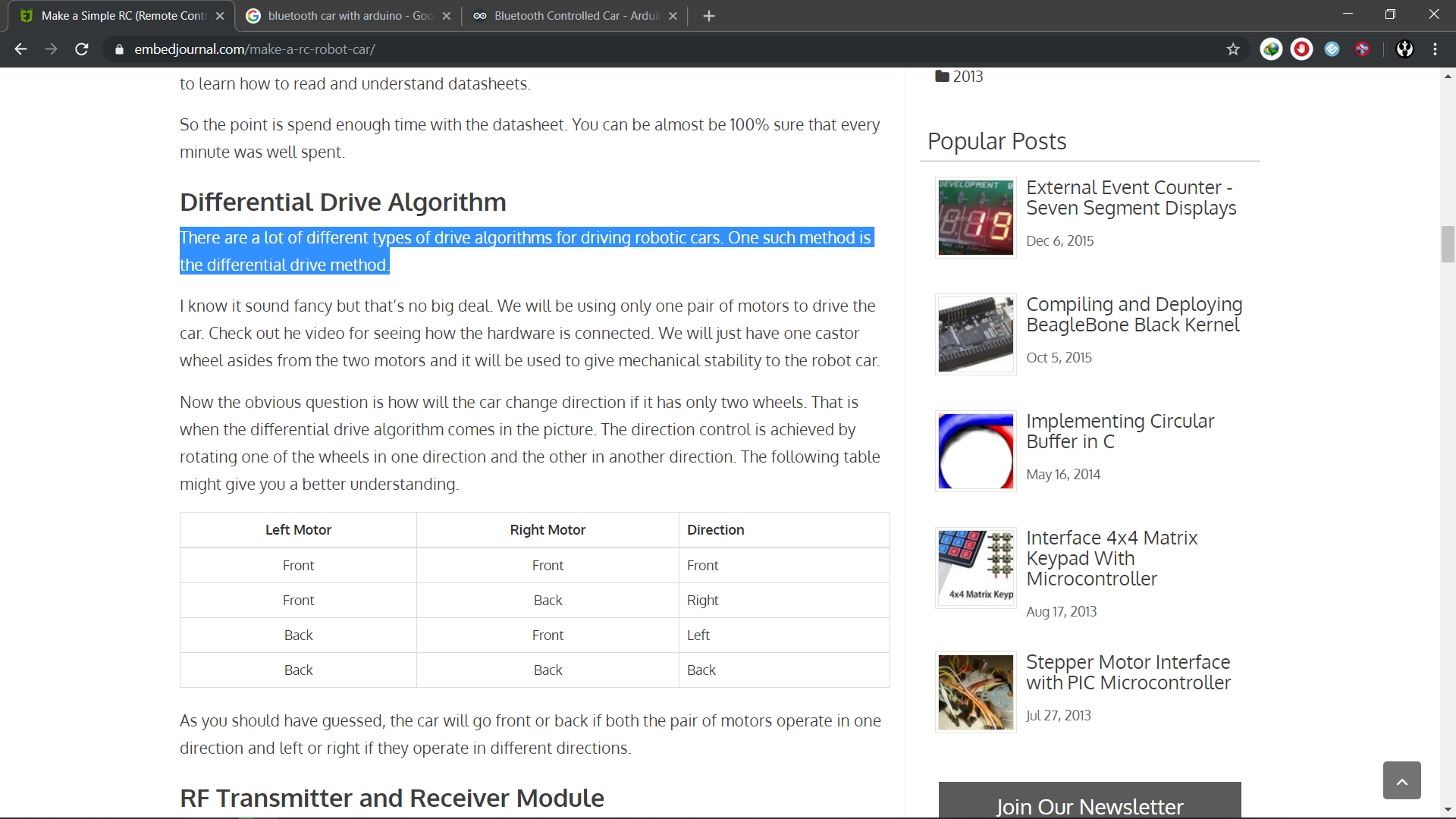
**1.1 Problem Introduction:**

So here the basic aim will be to make a Bluetooth controlled car by using IOT technologies/components by minimizing the cost and maximizing the number of features like live camera streaming, auto drive based on obstacle avoidance etc.

This will include learning about the various components that we’ll be using in our project and deep knowledge about the working of each individual module.

So, we need to understand the working of RC motors for the same.

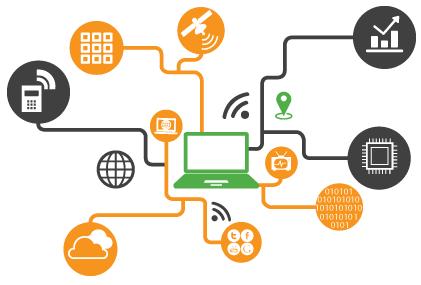
There are a lot of different types of drive algorithms for driving robotic cars. One such method is the differential drive method; the following table shows the movements of the car with 2 moving wheels and in our project, we’ll be implementing the same idea on 4 moving wheels with the help of a motor driver in association with Arduino UNO.



So, on basis of this we’ll start constructing the required 4WD robot car.

**1.2 Brief about IOT**

The entire project is based on the concepts of IOT so let’s see what IOT is and why do we need it?



**What is IOT?**

Internet of Things (IoT) is an ecosystem of connected physical objects that are accessible through the internet. The ‘thing’ in IoT could be a person with a heart monitor or an automobile with built-in-sensors, i.e. objects that have been assigned an IP address and have the ability to collect and transfer data over a network without manual assistance or intervention. The embedded technology in the objects helps them to interact with internal states or the external environment, which in turn affects the decisions taken.

# Why IoT?

An article by Ashton published in the RFID Journal in 1999 said, “If we had computers that knew everything there was to know about things - using data they gathered without any help from us - we would be able to track and count everything, and greatly reduce waste, loss and cost. We would know when things needed replacing, repairing or recalling, and whether they were fresh or past their best. We need to empower computers with their own means of gathering information, so they can see, hear and smell the world for themselves, in all its random glory.” This is precisely what IoT platforms does for us. It enables devices/objects to observe, identify and understand a situation or the surroundings without being dependent on human help.

# What is the scope of IoT?

Internet of Things can connect devices embedded in various systems to the internet. When devices/objects can represent themselves digitally, they can be controlled from anywhere. The connectivity then helps us capture more data from more places, ensuring more ways of increasing efficiency and improving safety and IoT security.

IoT is a transformational force that can help companies improve performance through IoT analytics and **IoT Security** to deliver better results. Businesses in the utilities, oil & gas, insurance, manufacturing, transportation, infrastructure and retail sectors can reap the benefits of IoT by making more informed decisions, aided by the torrent of interactional and transactional data at their disposal.

# How can IoT help?

IoT platforms can help organizations reduce cost through improved process efficiency, asset utilization and productivity. With improved tracking of devices/objects using sensors and connectivity, they can benefit from real-time insights and analytics, which would help them make smarter decisions. The growth and convergence of data, processes and things on the internet would make such connections more relevant and important, creating more opportunities for people, businesses and industries.

**1.3 REQUIREMENT SPECIFICATION**

**INTRODUCTION:**

To be used efficiently, all computer software needs certain hardware components or the other software resources to be present on a computer.

These pre-requisites are known as(computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time.

Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

**1.3.1 HARDWARE REQUIREMENTS:**

1) Arduino Uno

2) Motor Driver Shield

3) Wheels (4x)

4) TT Gear Motor (4x)

5) HC-05 Bluetooth Module

6) 18650 Li-on Battery (2x)

7) 18650 Battery Holder

8) Male and Female Jumper wire

9) Acrylic Sheet - (Offline Store)

10) ESP 32 CAM module

11) FDTI programmer.

12) Ultrasonic Sensor.

13) Servo motor.

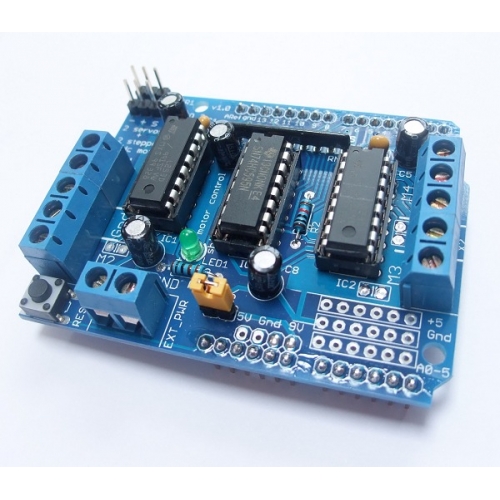
**DETAILS OF HARDWARE REQUIREMENTS**:

**1) Arduino Uno**



Arduinois an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

**2) Motor Driver Shield**



Motor drives are circuits used to run a motor. In other words, they are commonly used for motor interfacing. These drive circuits can be easily interfaced with the motor and their selection depends upon the type of motor being used and their ratings (current, voltage).

**3) HC-05 Bluetooth Module**



HC-05 is a Bluetooth module which is designed for wireless communication. This module can be used in a master or slave configuration. Bluetooth serial modules allow all serial enabled devices to communicate with each other using Bluetooth. It has 6 pins:

1.  **Key/EN:** It is used to bring Bluetooth module in AT commands mode.

2.  **VCC:**Connect 5 V or 3.3 V to this Pin.

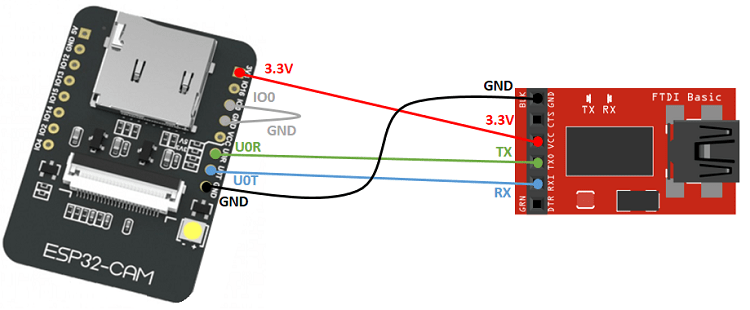
3.  **GND:**Ground Pin of module.

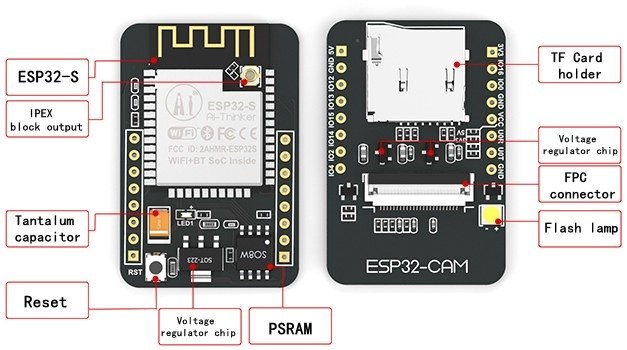
4.  **TXD:**Transmit Serial data.

5.  **RXD:** Receive data.

6.  **State:**It tells whether module is connected or not.

**4) ESP 32 CAM module and FDTI Programmer:**





The ESP32-CAM is a very small camera module with the ESP32-S chip. Besides the OV2640 camera, and several GPIOs to connect peripherals, it also features a microSD card slot that can be useful to store images taken with the camera or to store files to serve to clients. The ESP32-CAM doesn’t come with a USB connector, so you need an FTDI programmer to upload code through the U0R and U0T pins (serial pins).

**5) Ultrasonic Sensor and servo motor.**



The ultrasonic sensor is used for obstacle avoidance and in order to provide it free movement it has to be mounted on servo motor that can be coded as per requirement.

**6) TT Gear Motor (4x) and 18650 Li-on Battery (2x)**



TT gear motor is used for rotation of wheels of the car and the Li-ion battery (3.7v each) to give power to the entire system/car.

**7) Wheels and jumper wires**

Wheels for the movement of car and jumper wire of all types for various connections in the 4 wheel drive.

**1.3.2 SOFTWARE REQUIREMENTS:**

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer in order to code various IOT components as per need to make them work.

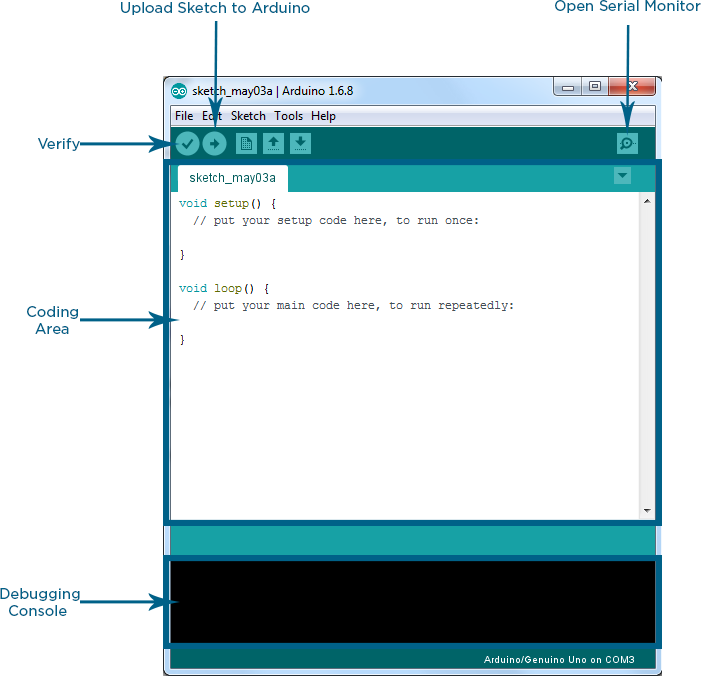
**SOFTWARE REQUIREMENTS FOR PRESENT PROJECT:**

* **Arduino IDE**



Arduino IDE is used for coding the various components

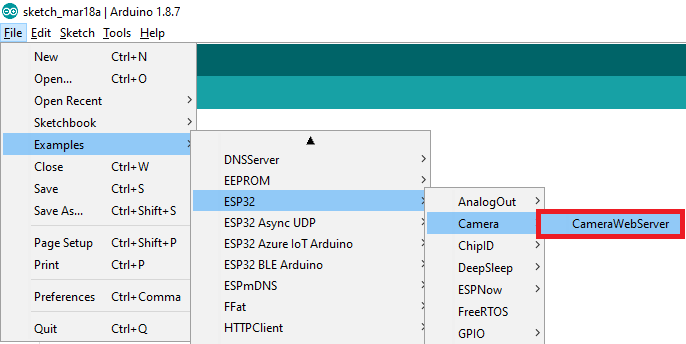
* ESP 32 CAM
* Ultrasonic Sensor
* Servo motor
* Motor driver
* Arduino UNO



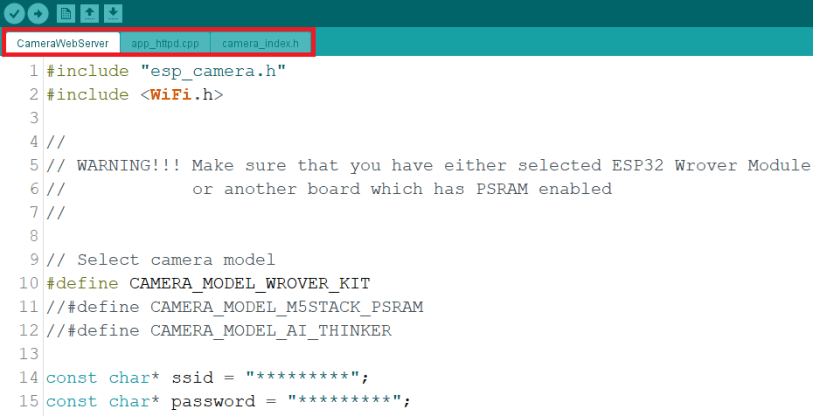
**Code for ESP32 CAM:**

1. Install the esp32 add on in board manager.

2. In your Arduino IDE, go to File > Examples > ESP32 > Camera and open the Camera Webserver example.



3. change ssid and password as per need and upload.



**In this case:**

#include "esp\_camera.h"

#include <WiFi.h>

//

// WARNING!!! Make sure that you have either selected ESP32 Wrover Module,

// or another board which has PSRAM enabled

//

// Select camera model

//#define CAMERA\_MODEL\_WROVER\_KIT

//#define CAMERA\_MODEL\_ESP\_EYE

//#define CAMERA\_MODEL\_M5STACK\_PSRAM

//#define CAMERA\_MODEL\_M5STACK\_WIDE

#define CAMERA\_MODEL\_AI\_THINKER

#include "camera\_pins.h"

const char\* ssid = "realme X";

const char\* password = "asdfghjkl";

void startCameraServer();

void setup() {

Serial.begin(115200);

Serial.setDebugOutput(true);

Serial.println();

camera\_config\_t config;

config.ledc\_channel = LEDC\_CHANNEL\_0;

config.ledc\_timer = LEDC\_TIMER\_0;

config.pin\_d0 = Y2\_GPIO\_NUM;

config.pin\_d1 = Y3\_GPIO\_NUM;

config.pin\_d2 = Y4\_GPIO\_NUM;

config.pin\_d3 = Y5\_GPIO\_NUM;

config.pin\_d4 = Y6\_GPIO\_NUM;

config.pin\_d5 = Y7\_GPIO\_NUM;

config.pin\_d6 = Y8\_GPIO\_NUM;

config.pin\_d7 = Y9\_GPIO\_NUM;

config.pin\_xclk = XCLK\_GPIO\_NUM;

config.pin\_pclk = PCLK\_GPIO\_NUM;

config.pin\_vsync = VSYNC\_GPIO\_NUM;

config.pin\_href = HREF\_GPIO\_NUM;

config.pin\_sscb\_sda = SIOD\_GPIO\_NUM;

config.pin\_sscb\_scl = SIOC\_GPIO\_NUM;

config.pin\_pwdn = PWDN\_GPIO\_NUM;

config.pin\_reset = RESET\_GPIO\_NUM;

config.xclk\_freq\_hz = 20000000;

config.pixel\_format = PIXFORMAT\_JPEG;

//init with high specs to pre-allocate larger buffers

if(psramFound()){

config.frame\_size = FRAMESIZE\_UXGA;

config.jpeg\_quality = 10;

config.fb\_count = 2;

} else {

config.frame\_size = FRAMESIZE\_SVGA;

config.jpeg\_quality = 12;

config.fb\_count = 1;

}

#if defined(CAMERA\_MODEL\_ESP\_EYE)

pinMode(13, INPUT\_PULLUP);

pinMode(14, INPUT\_PULLUP);

#endif

// camera init

esp\_err\_t err = esp\_camera\_init(&config);

if (err != ESP\_OK) {

Serial.printf("Camera init failed with error 0x%x", err);

return;

}

sensor\_t \* s = esp\_camera\_sensor\_get();

//initial sensors are flipped vertically and colors are a bit saturated

if (s->id.PID == OV3660\_PID) {

s->set\_vflip(s, 1);//flip it back

s->set\_brightness(s, 1);//up the blightness just a bit

s->set\_saturation(s, -2);//lower the saturation

}

//drop down frame size for higher initial frame rate

s->set\_framesize(s, FRAMESIZE\_QVGA);

#if defined(CAMERA\_MODEL\_M5STACK\_WIDE)

s->set\_vflip(s, 1);

s->set\_hmirror(s, 1);

#endif

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

startCameraServer();

Serial.print("Camera Ready! Use 'http://");

Serial.print(WiFi.localIP());

Serial.println("' to connect");

}

void loop() {

// put your main code here, to run repeatedly:

delay(10000);

}

**Code for rest components in Arduino:**

A) For obstacle avoiding:

#include <AFMotor.h>

#include <Servo.h>

#include <NewPing.h>

#define TRIG\_PIN A0

#define ECHO\_PIN A1

#define MAX\_DISTANCE 300

#define MAX\_SPEED 160

#define MAX\_SPEED\_OFFSET 40

#define COLL\_DIST 30

#define TURN\_DIST COLL\_DIST+20

NewPing sonar(TRIG\_PIN, ECHO\_PIN, MAX\_DISTANCE);

AF\_DCMotor leftMotor1(1, MOTOR12\_1KHZ);

AF\_DCMotor leftMotor2(2, MOTOR12\_1KHZ);

AF\_DCMotor rightMotor1(3, MOTOR34\_1KHZ);

AF\_DCMotor rightMotor2(4, MOTOR34\_1KHZ);

Servo myservo ;

int leftDistance, rightDistance;

int curDist = 0;

String motorSet = "";

int speedSet = 0;

//-------------------------------------------- SETUP LOOP ----------------------------------------------

void setup() {

myservo.attach(10); // attaches the servo on pin 10 (SERVO\_1 on the Motor Drive Shield to the servo object

myservo.write(90); // tells the servo to position at 90-degrees ie. facing forward.

delay(1000); // delay for one seconds

}

//-----------------------------------------------------------------------------------------------------------

//---------------------------------------------MAIN LOOP -------------------------------------

void loop() {

myservo.write(90); // move eyes forward

delay(90);

curDist = readPing(); // read distance

if (curDist < COLL\_DIST) {changePath();} // if forward is blocked change direction

moveBackward(); // move forward

delay(500);

}

//---------------------------------------------------------------------------------------------------

void changePath() {

moveStop(); // stop forward movement

myservo.write(36); // check distance to the right

delay(500);

rightDistance = readPing(); //set right distance

delay(500);

myservo.write(144); // check distace to the left

delay(700);

leftDistance = readPing(); //set left distance

delay(500);

myservo.write(90); //return to center

delay(100);

compareDistance();

}

void compareDistance() // find the longest distance

{

if (leftDistance>rightDistance) //if left is less obstructed

{

turnLeft();

}

else if (rightDistance>leftDistance) //if right is less obstructed

{

turnRight();

}

else //if they are equally obstructed

{

turnAround();

}

}

//-------------------------------------------------------------------------------------------------------

int readPing() { // read the ultrasonic sensor distance

delay(70);

unsigned int uS = sonar.ping();

int cm = uS/US\_ROUNDTRIP\_CM;

return cm;

}

//-----------------------------------------------------------------------------------------------------------------

void moveStop() {leftMotor1.run(RELEASE); leftMotor2.run(RELEASE); rightMotor1.run(RELEASE); rightMotor2.run(RELEASE);} // stop the motors.

//--------------------------------------------------------------------------------------------------------------

void moveForward() {

motorSet = "FORWARD";

leftMotor1.run(FORWARD);

leftMotor2.run(FORWARD);

rightMotor1.run(FORWARD);

rightMotor2.run(FORWARD);

for (speedSet = 0; speedSet < MAX\_SPEED; speedSet +=2) // slowly bring the speed up to avoid loading down the batteries too quickly

{

leftMotor1.setSpeed(speedSet);

leftMotor2.setSpeed(speedSet);

rightMotor1.setSpeed(speedSet);

rightMotor2.setSpeed(speedSet);

delay(5);

}

}

//----------------------------------------------------------------------------------------------------------

void moveBackward() {

motorSet = "BACKWARD";

leftMotor1.run(BACKWARD);

leftMotor2.run(BACKWARD);

rightMotor1.run(BACKWARD);

rightMotor2.run(BACKWARD);

for (speedSet = 0; speedSet < MAX\_SPEED; speedSet +=2) // slowly bring the speed up to avoid loading down the batteries too quickly

{

leftMotor1.setSpeed(speedSet);

leftMotor2.setSpeed(speedSet);

rightMotor1.setSpeed(speedSet);

rightMotor2.setSpeed(speedSet);

delay(5);

}

}

//------------------------------------------------------------------------------------------------------------

void turnRight() {

motorSet = "RIGHT";

leftMotor1.run(FORWARD); // turn motor 1 forward

leftMotor2.run(FORWARD); // turn motor 2 forward

rightMotor1.run(BACKWARD); // turn motor 3 backward

rightMotor2.run(BACKWARD); // turn motor 4 backward

rightMotor1.setSpeed(speedSet+MAX\_SPEED\_OFFSET);

rightMotor2.setSpeed(speedSet+MAX\_SPEED\_OFFSET);

delay(1500); // run motors this way for 1500

motorSet = "FORWARD";

leftMotor1.run(FORWARD); // set both motors back to forward

leftMotor2.run(FORWARD);

rightMotor1.run(FORWARD);

rightMotor2.run(FORWARD);

}

//-----------------------------------------------------------------------------------------------------------

void turnLeft() {

motorSet = "LEFT";

leftMotor1.run(BACKWARD); // turn motor 1 backward

leftMotor2.run(BACKWARD); // turn motor 2 backward

leftMotor1.setSpeed(speedSet+MAX\_SPEED\_OFFSET);

leftMotor2.setSpeed(speedSet+MAX\_SPEED\_OFFSET);

rightMotor1.run(FORWARD); // turn motor 3 forward

rightMotor2.run(FORWARD); // turn motor 4 forward

delay(1500); // run motors this way for 1500

motorSet = "FORWARD";

leftMotor1.run(FORWARD); // turn it on going forward

leftMotor2.run(FORWARD); // turn it on going forward

rightMotor1.run(FORWARD); // turn it on going forward

rightMotor2.run(FORWARD); // turn it on going forward

}

//------------------------------------------------------------------------------------------------------------

void turnAround() {

motorSet = "RIGHT";

leftMotor1.run(FORWARD); // turn motor 1 forward

leftMotor2.run(FORWARD); // turn motor 2 forward

rightMotor1.run(BACKWARD); // turn motor 3 backward

rightMotor2.run(BACKWARD); // turn motor 4 backward

rightMotor1.setSpeed(speedSet+MAX\_SPEED\_OFFSET);

rightMotor2.setSpeed(speedSet+MAX\_SPEED\_OFFSET);

delay(1700); // run motors this way for 1700

motorSet = "FORWARD";

leftMotor1.run(FORWARD); // set both motors back to forward

leftMotor2.run(FORWARD);

rightMotor1.run(FORWARD);

rightMotor2.run(FORWARD);

}

B) for Bluetooth:

//Arduino Bluetooth Controlled Car//

//// Before uploading the code you have to install the necessary library//

//AFMotor Library https://learn.adafruit.com/adafruit-motor-shield/library-install //

#include <AFMotor.h>

//initial motors pin

AF\_DCMotor motor1(1);

AF\_DCMotor motor2(2);

AF\_DCMotor motor3(3);

AF\_DCMotor motor4(4);

char command;

void setup()

{

Serial.begin(9600); //Set the baud rate to your Bluetooth module.

}

void loop(){

if(Serial.available() > 0){

command = Serial.read();

Stop(); //initialize with motors stoped

//Change pin mode only if new command is different from previous.

//Serial.println(command);

switch(command){

case 'B':

forward();

break;

case 'F':

back();

break;

case 'L':

left();

break;

case 'R':

right();

break;

}

}

}

void forward()

{

motor1.setSpeed(255); //Define maximum velocity

motor1.run(FORWARD); //rotate the motor clockwise

motor2.setSpeed(255); //Define maximum velocity

motor2.run(FORWARD); //rotate the motor clockwise

motor3.setSpeed(255);//Define maximum velocity

motor3.run(FORWARD); //rotate the motor clockwise

motor4.setSpeed(255);//Define maximum velocity

motor4.run(FORWARD); //rotate the motor clockwise

}

void back()

{

motor1.setSpeed(255); //Define maximum velocity

motor1.run(BACKWARD); //rotate the motor anti-clockwise

motor2.setSpeed(255); //Define maximum velocity

motor2.run(BACKWARD); //rotate the motor anti-clockwise

motor3.setSpeed(255); //Define maximum velocity

motor3.run(BACKWARD); //rotate the motor anti-clockwise

motor4.setSpeed(255); //Define maximum velocity

motor4.run(BACKWARD); //rotate the motor anti-clockwise

}

void left()

{

motor1.setSpeed(255); //Define maximum velocity

motor1.run(BACKWARD); //rotate the motor anti-clockwise

motor2.setSpeed(255); //Define maximum velocity

motor2.run(BACKWARD); //rotate the motor anti-clockwise

motor3.setSpeed(255); //Define maximum velocity

motor3.run(FORWARD); //rotate the motor clockwise

motor4.setSpeed(255); //Define maximum velocity

motor4.run(FORWARD); //rotate the motor clockwise

}

void right()

{

motor1.setSpeed(255); //Define maximum velocity

motor1.run(FORWARD); //rotate the motor clockwise

motor2.setSpeed(255); //Define maximum velocity

motor2.run(FORWARD); //rotate the motor clockwise

motor3.setSpeed(255); //Define maximum velocity

motor3.run(BACKWARD); //rotate the motor anti-clockwise

motor4.setSpeed(255); //Define maximum velocity

motor4.run(BACKWARD); //rotate the motor anti-clockwise

}

void Stop()

{

motor1.setSpeed(0); //Define minimum velocity

motor1.run(RELEASE); //stop the motor when release the button

motor2.setSpeed(0); //Define minimum velocity

motor2.run(RELEASE); //rotate the motor clockwise

motor3.setSpeed(0); //Define minimum velocity

motor3.run(RELEASE); //stop the motor when release the button

motor4.setSpeed(0); //Define minimum velocity

motor4.run(RELEASE); //stop the motor when release the button

}

CHAPTER 2

**LITERATURE SURVEY**

**2.1 Existing System/researches.**

Many researches on the design of robotic vehicle using microcontroller have been reported in the literature.

* **Esra Yılmaz, Sibel T. Özyer** worked on Remote and Autonomous Controlled Robotic Carbased on Arduino with Real Time ObstacleDetection and Avoidance

**Abstract:**In robotic car, real time obstacle detection and obstacle avoidance are significant issues. In this study, design and implementation of a robotic car have been presented with regards to hardware, software and communication environments with real time obstacle detection and obstacle avoidance. Arduino platform, android application and Bluetooth technology have been used to implementation of the system. In this paper, robotic car design and application with using sensor programming on a platform has been presented. This robotic device has been developed with the interaction of Android-based device.

Arduino Uno has been used as the robot's brain. The robot has many hardware components such as Bluetooth module, PIR sensor, ultrasonic sensor, and buzzers. It also consists of the software component that uses a mobile application. The desired direction or mode by mobile application can be selected by the user of the robotic car to control the movement of the car. The user can control the robot movements from his/her own intelligent device or take the robot in automatic mode and let the car drive its own way.

Thus, the robot can flee from the obstacle and also detect live objects. The purpose of this article is to alert the civilian and military personnel to potential terrorist attacks especially in military areas with live detectable sensors.

* **S. S. Pujari** et al. [1] designed a Robot for the working families that could monitor children remotely and communicate with the camera. Raspberry Pi 3, camera module, Wifi and Bluetooth technology were used by the robot. For Raspberry Pi, the heart was defined as the Robot and used the Python language to code it.
* **Esra Yılmaz of Cankaya University** worked on Remote and Autonomous Controlled Robotic Car based on Arduino with Real Time Obstacle Detection and Avoidance

**Abstract:**

In robotic car, real time obstacle detection and obstacle avoidance are significant issues. In this study, design and implementation of a robotic car have been presented with regards to hardware, software and communication environments with real time obstacle detection and obstacle avoidance. Arduino platform, android application and Bluetooth technology have been used to implementation of the system. In this paper, robotic car design and application with using sensor programming on a platform has been presented. This robotic device has been developed with the interaction of Android-based device. Arduino Uno has been used as the robot's brain. The robot has many hardware components such as Bluetooth module, PIR sensor, ultrasonic sensor, and buzzers. It also consists of the software component that uses a mobile application. The desired direction or mode by mobile application can be selected by the user of the robotic car to control the movement of the car. The user can control the robot movements from his/her own intelligent device or take the robot in automatic mode and let the car drive its own way. Thus, the robot can flee from the obstacle and also detect live objects. The purpose of this article is to alert the civilian and military personnel to potential terrorist attacks especially in military areas with live detectable sensors.

* **M. R. Mishi** et al. [2] designed a robotic car. Arduino Uno and Raspberry Pi were used together to control robot in this project. GPS was also used to trace the car and the distances between the obstacle and the path are measured. The data in the cloud was used without having to be online. Thus, the multi-motion system was controlled.
* **D. Chakraborty** et al. [3] designed and developed a robotic car using sensors and Bluetooth technology. They had established communication between smart device and the robot. Thanks to the phone camera, they had observed the living beings. The obstacles in the opposite direction were prevented from colliding with the ultrasonic ranging sensor. Images recorded with the camera were recorded in the database and analyzed.
* **E. Amareswar** et al. [5] designed a robot used for the military area. Thanks to the metal detector, the robot played an important role in the detection of explosives, and the surroundings could be viewed thanks to the camera of the used Android device. This robot system consisted of Android device, Bluetooth module, a microcontroller (Arduino Uno), DC motors, motor driver, wireless camera and metal detector.

**Russell and Norvig** [14] proposed cooperative Robotics. This has to do with the use of multiple robotics agents assisting each other to perform a task that is either too difficult or impossible for a single robot to perform alone. The study presents a search-and-rescue algorithm, referred to as SARA-1 that is designed to enable a team of cooperative autonomous robots to search an area for a stationary target.

The robots use wireless communication to build and share collective maps of the environment. They attempt to spread out their cooperative search and being careful not to explore the same area again. This algorithm is pertinent to both indoor and outdoor applications. The range of the applications is limited only by the user's imagination and might include such tasks as hazardous waste location and removal, planetary exploration, warehouse organization, and human search-and-rescue.

**2.2 Proposed System.**

Our system (robot car) aims to implement the following features in minimum cost:

1. Bluetooth control for giving commands to the car.

2. Auto-drive with help of ultrasonic sensor for obstacle avoidance and providing movement using a servomotor.

3. Live camera streaming and face detection with ESP 32 Cam module.

4. Connectivity over the internet from all over the world.

5. Synchronizing 4 wheels with Arduino and motor driver.

With all these features it’ll still have an open scope for adding features in upcoming time and upgrading the system as per the need of the user.

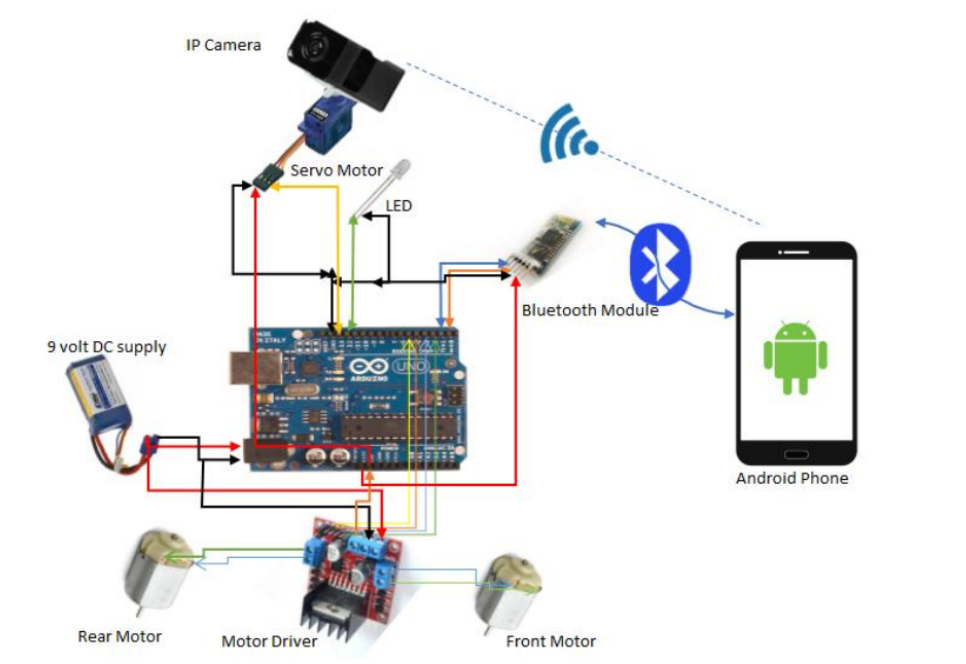
Along with the implementation of the project, learning is involved to a large extent where deep knowledge about each component used and connections should be there to properly understand the concepts and logics behind the working model.

**2.3 Feasibility Study**

CHAPTER 3

**SYSTEM DESIGN AND ANALYSIS**

**3.1 Working Principle**



1. Once all the connections are made the user need to open the app on the smartphone to connect to the robot car.

2. Once power is supplied to the car the HC-05 will start and get connected to the smartphone.

3. HC-05 will receive the command and Arduino will invoke the function as per the command received.

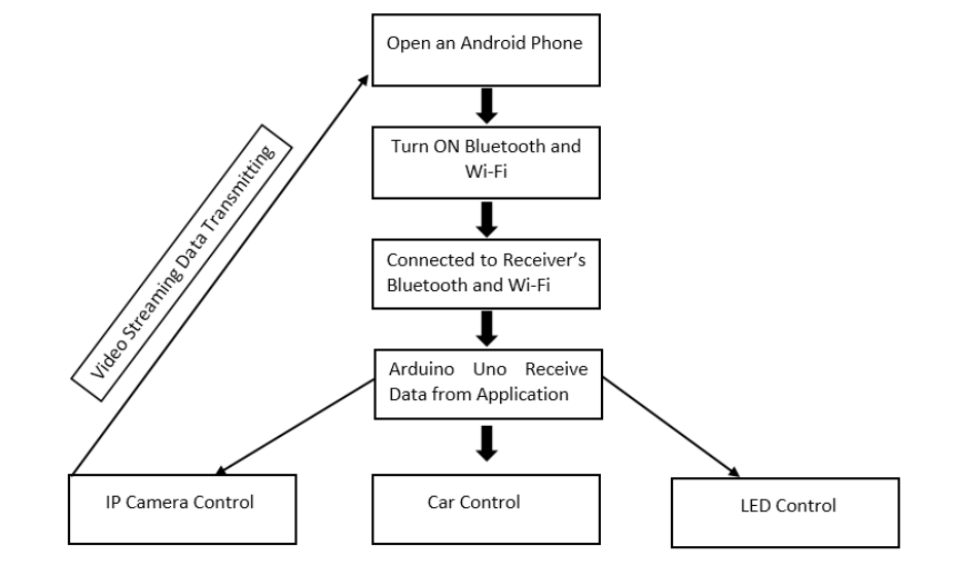
4. Now as per the code the Arduino will give command to motor driver shield and make the movement possible.

5. For obstacle avoidance, ultrasonic sensor will be enabled that will be mounted on servo motor that will provide the mobility to the ultrasonic sensor. When another code is uploaded only servo motor will automatically detect the path to move. [work for stable obstacles only]

6. ESP 32 CAM module will also be enabled that will have its code stored on itself and will provide a live streaming of video and face detection that’s an inbuilt feature of this module.

**3.2 Pin diagram**

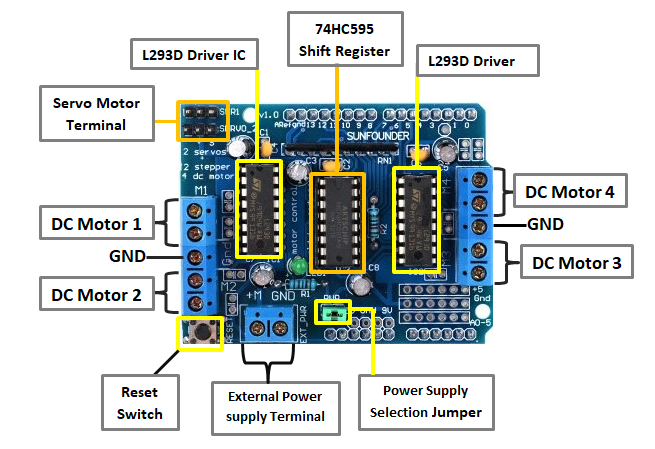
1. Flow chart.



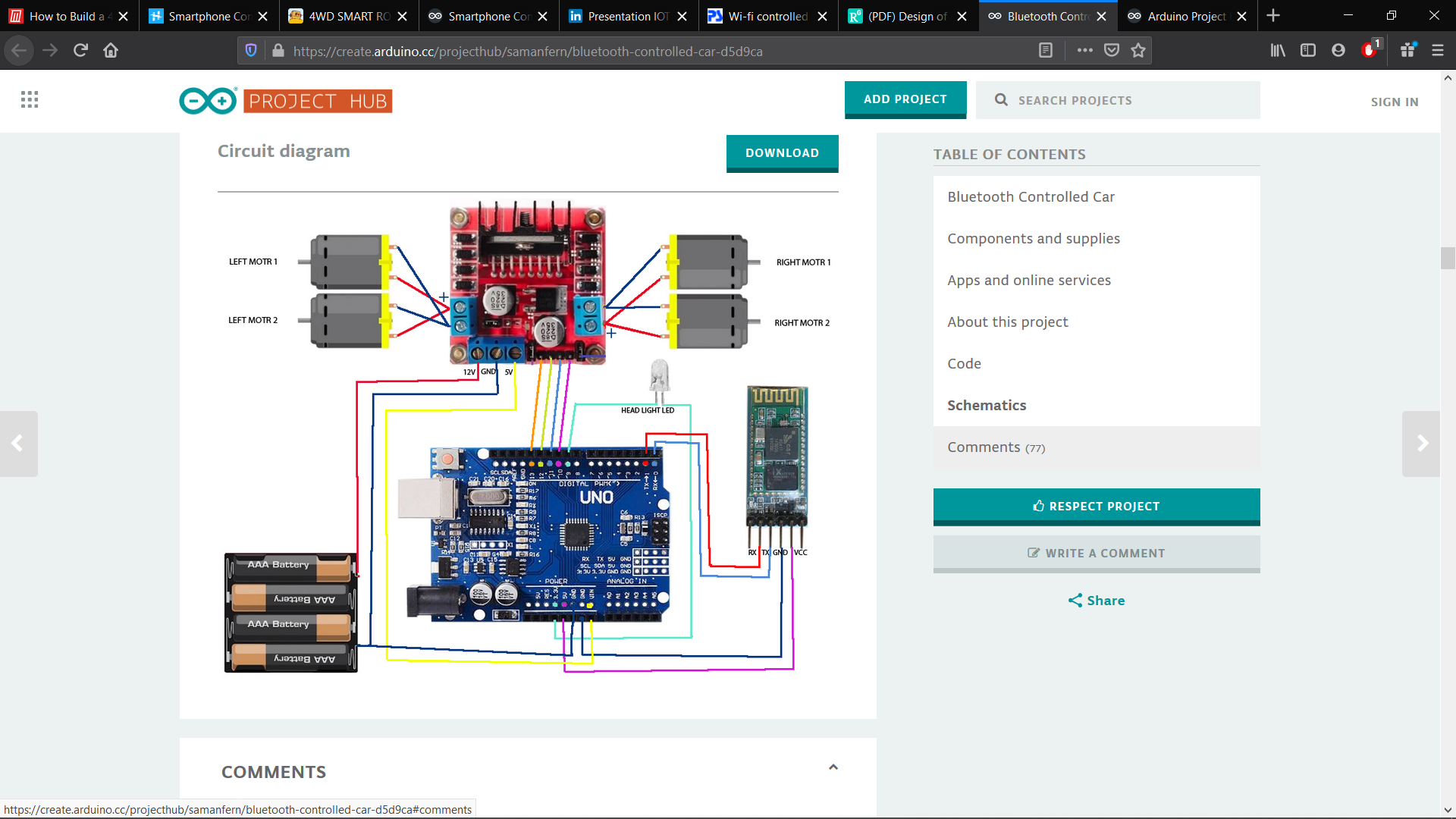
2. Arduino Pin Diagram.



3. Motor Driver Pin Diagram.

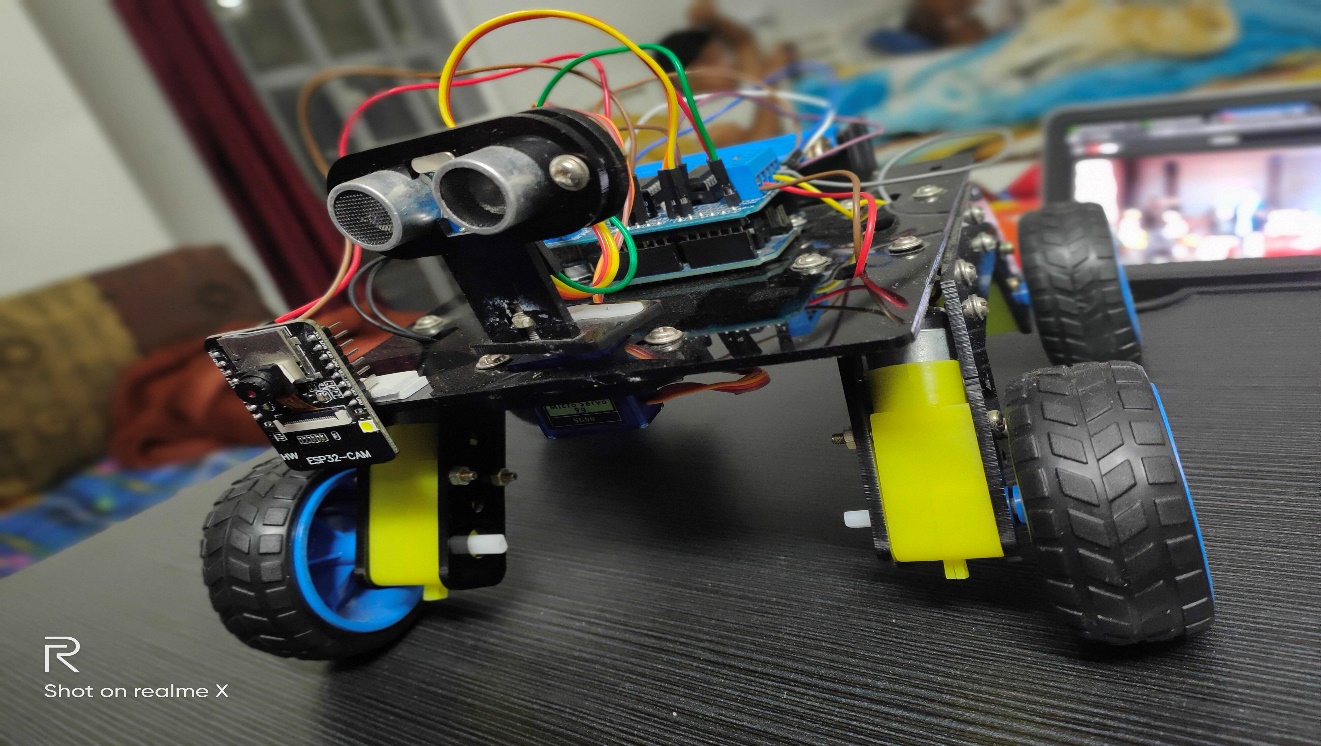


4. Connections of various components in robot car.



**3.3 Testing**

Snapshots of complete working model of the robotcar.

****

**Servo motor and Ultrasonic sensor:**

Servo motor provide mobility to ultrasonic sensor and provides a movement of 180 degree and let the ultrasonic sensor scan for obstacles.

Sometimes the detection of obstacle fails but mostly it works accurately 80% of times.

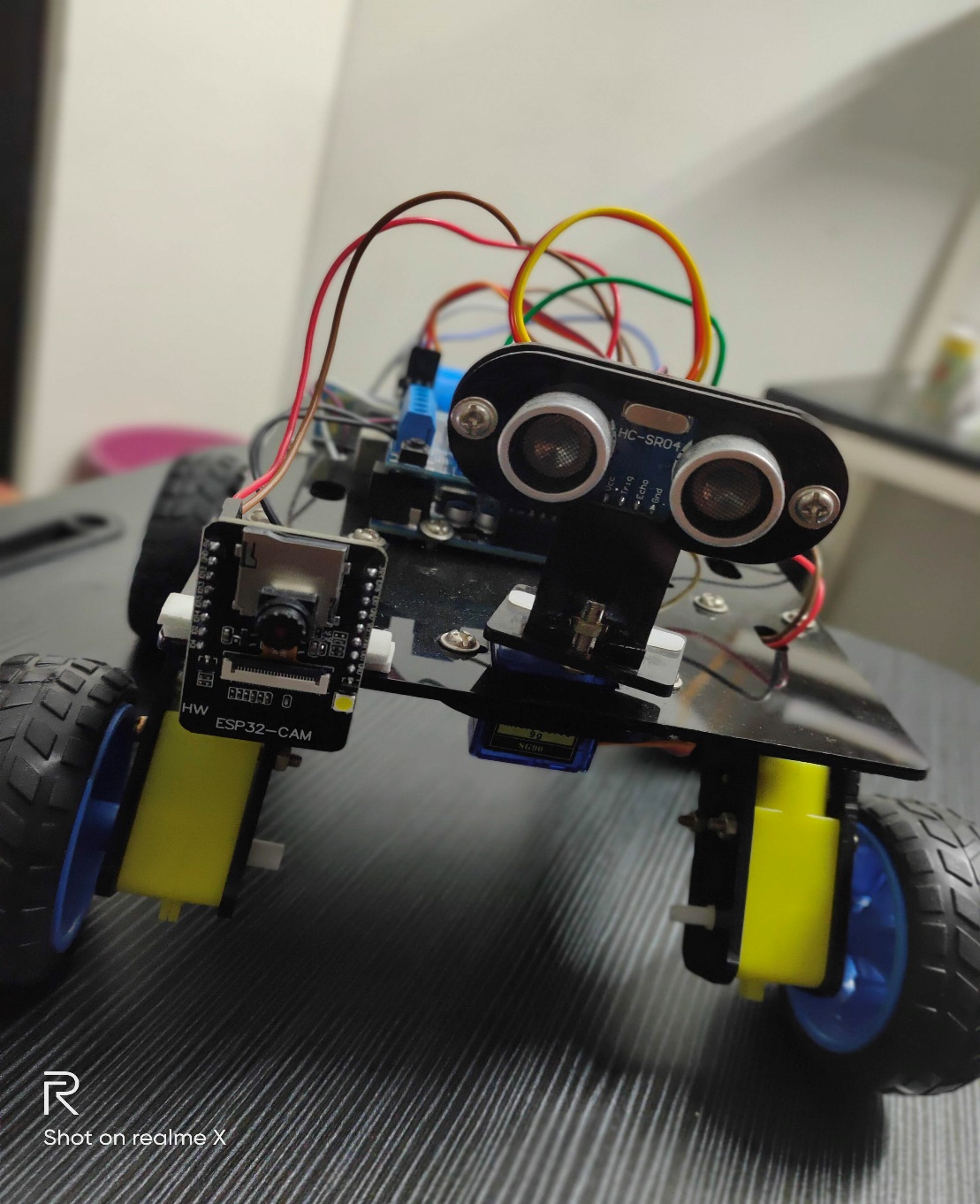
Servo motor is working as expected with no errors.

**HC05:**

Bluetooth module is working 100% accurately. It uses the 2.45GHz frequency band.

It has range of connection up to 10m in open area and its able to transfer commands efficiently.

**Note:** Batteries have to be charged properly to give best output as a consistent voltage between 5V and 12V is needed. Batteries may cause issues.

****

**ESP32 CAM:**

It’s successfully configured to stream live video and show output on given IP.

It has OV2640 camera, and several GPIOs to connect peripherals. It also features a microSD card slot that can be useful to store images taken with the camera or to store files to serve to clients.

Here is a list with the ESP32-CAM features:

* Up to 160MHz clock speed, summary computing power up to 600 DMIPS
* Built-in 520 KB SRAM, external 4MPSRAM
* Support OV2640 and OV7670 cameras, built-in flash lamp
* Support image WiFI upload
* Support TF card
* Supports multiple sleep modes
* Embedded Lwip and FreeRTOS
* Supports STA/AP/STA+AP operation mode

CHAPTER 4

**RESULTS/CONCLUSION AND FUTURE SCOPE**

**4.1 Result/Conclusion**

The components when assembled together and coded correctly gave expected output and each of the part is working accurately and robot car is working properly. All modules/component specified in the project work has been implemented and are successfully running as expected.

There is a large scope for adding components and other features to make it even better.

**4.2 Future Scope**

This project has a large room for improvement as well as for adding features.

It can be modified into a spy drone and can be used for security purpose.

Following feature can be added:

1. Connectivity all over the world with the help of ESP 32 module and sim driver may be used.

2. Advanced Machine Learning algorithm can be associated with various components to make it even better.

3. Use of solar panel may be used for charging purpose of cells.

4. Arduino may be replaced by raspberry pie to improve the functionality.

5. Can be used for security purpose and lights can be added.

6. Multiple ultrasonic sensors may be used for better obstacle avoidance.

CHAPTER 5

**BIBLIOGRAPHY**

1. <https://www.researchgate.net>

2. [https://create.arduino.cc](https://create.arduino.cc/projecthub/samanfern/bluetooth-controlled-car-d5d9ca)

3. [https://howtomechatronics.com](https://howtomechatronics.com/tutorials/arduino/arduino-robot-car-wireless-control-using-hc-05-bluetooth-nrf24l01-and-hc-12-transceiver-modules/)

4. [https://circuitdigest.com](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=18&cad=rja&uact=8&ved=2ahUKEwj00u_OwdPlAhUFWCsKHZ81AGoQFjARegQIBRAB&url=https%3A%2F%2Fcircuitdigest.com%2Fmicrocontroller-projects%2Fbluetooth-controlled-robot-car-using-arduino&usg=AOvVaw2DR2RRN4-QidkueyUJwpYE)

5. <https://create.arduino.cc/projecthub/projects/tags/car>

6. <https://www.youtube.com/watch?v=Q36NbjPMV5k>

7. <https://www.youtube.com/watch?v=1n_KjpMfVT0&t=204s>

8. <https://github.com/espressif/arduino-esp32>

9. <https://www.youtube.com/watch?v=MicAM_A0_lU&t=180s>