Engineering Clinics

Smart Robot Car Using Arduino

PROJECT GUIDE

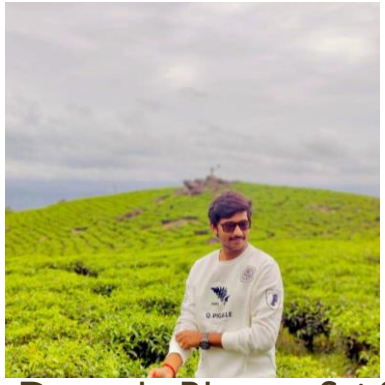
PROF. NEERAJ KUMAR MISHRA

Ph.D under TEQIP-II, PGD (ML & AI from NIT Warangal), Associate Professor at VIT-AP University, ECE, Editorial Board Member of ACTA SCIENTIFIC , Quantum Computing | | Reversible Logic | | Low Power VLSI | | AI & ML

Meet our team

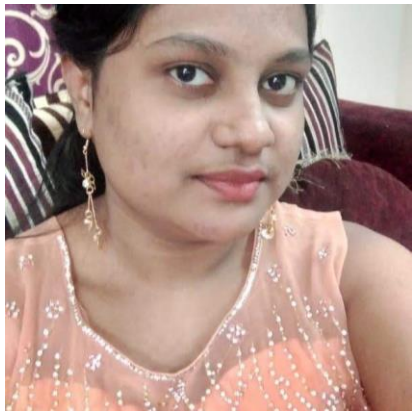
Arumulla Yaswanth Reddy
20BES7010

Kakumanu Srimani
20BCD7133



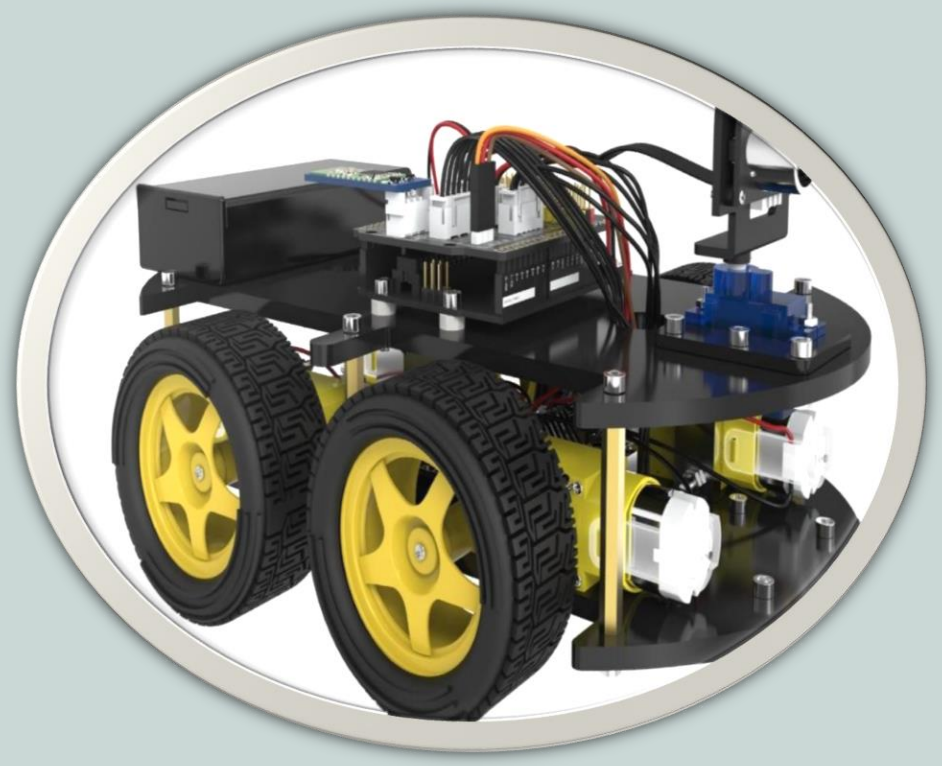
Damarla Bhuvan Sri Sai
20BCD7096

Nuthalapati Jashvika
20BCD7115



Kandukuri Pranavi
20BCR7104

Chandini Kollati
20BCE7379



Agenda

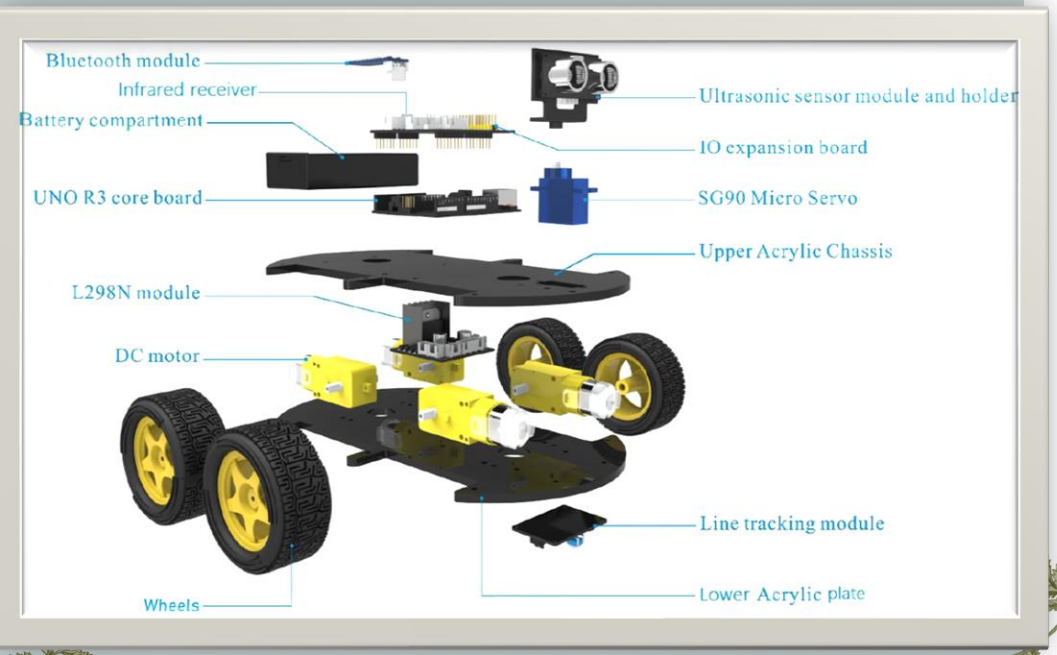
- Introduction
- Parts
- Problem Definition
- Circuit Diagram
- Codes in Appendix
- 3D-Video
- Reference

Introduction

Smart robot car using arduino

An impressive work combining different solutions to the real world problems! The Smart Car has features like Ultrasonic Range finder for collision avoidance and semi-autonomus navigation; Optical sensors for line following navigation; Bluetooth connection for remote control operation; and an Infrared Remote Control.

Parts



- DC motors
- L298N motor driver
- Arduino UNO
- Sensor shield
- SG90 Micro Servo
- Ultrasonic sensor & holder
- Line tracking module
- Bluetooth module
- Infrared receiver
- Rechargeable Battery
- Wheels

Smart Robot Car – Initial Test

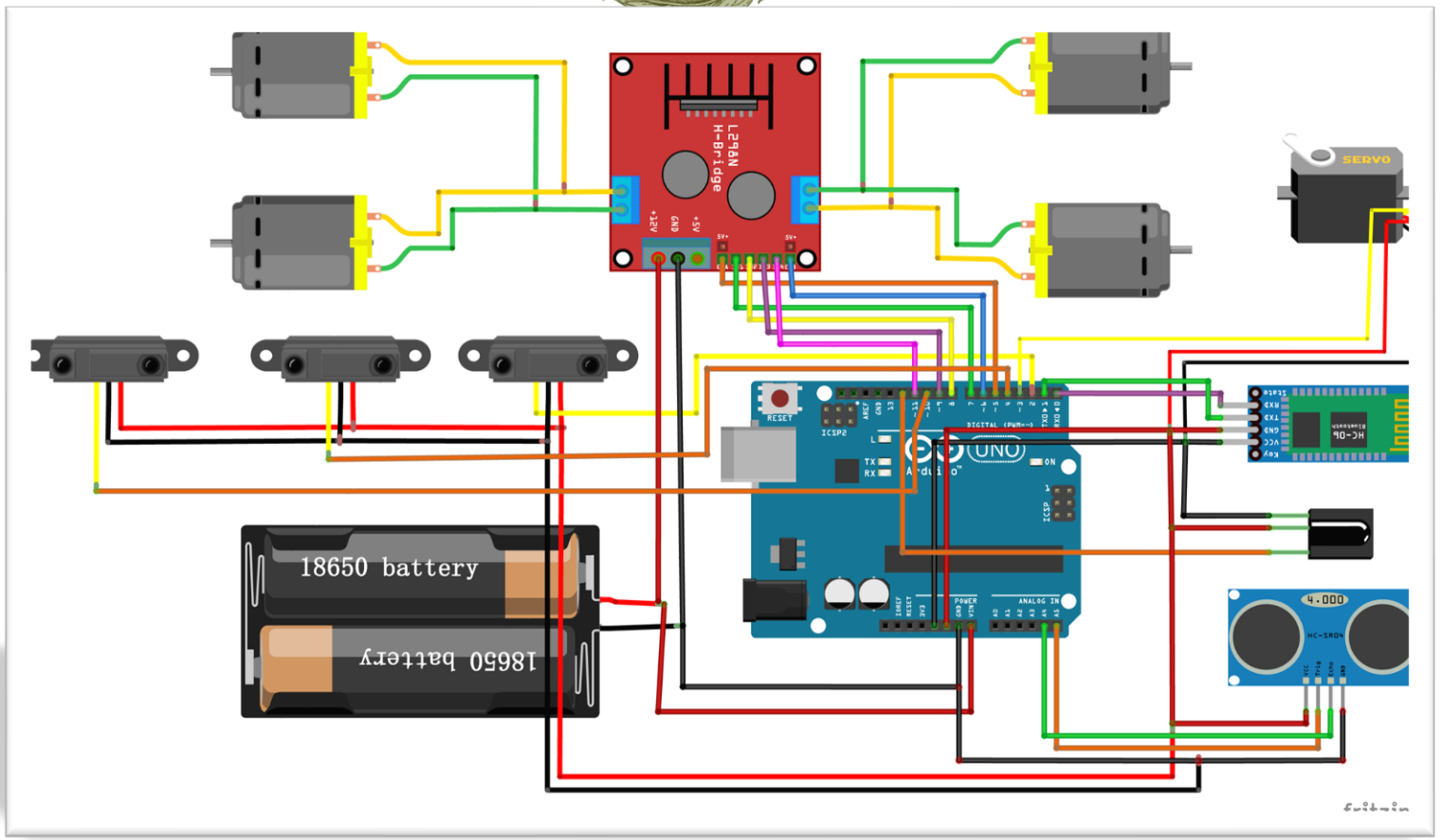
The Smart Robot Car has the following functions:

1. Bluetooth Mode
2. Line Tracking Mode
3. Obstacle-avoidance Mode
4. IR-Remote Control Mode

Problem Definition

- This project is aims at developing Arduino based smart robotic control to provide a better solution to society.
- In this modern digital world, everyone is moving towards automation Robotics allows automation where machines perform a well-defined step safely and productively, in autonomous or partial autonomous manners.
- A number of robot car bases are available for just such a project. These inexpensive bases are generally made of acrylic and come complete with a set of small DC motors.

Circuit Diagram



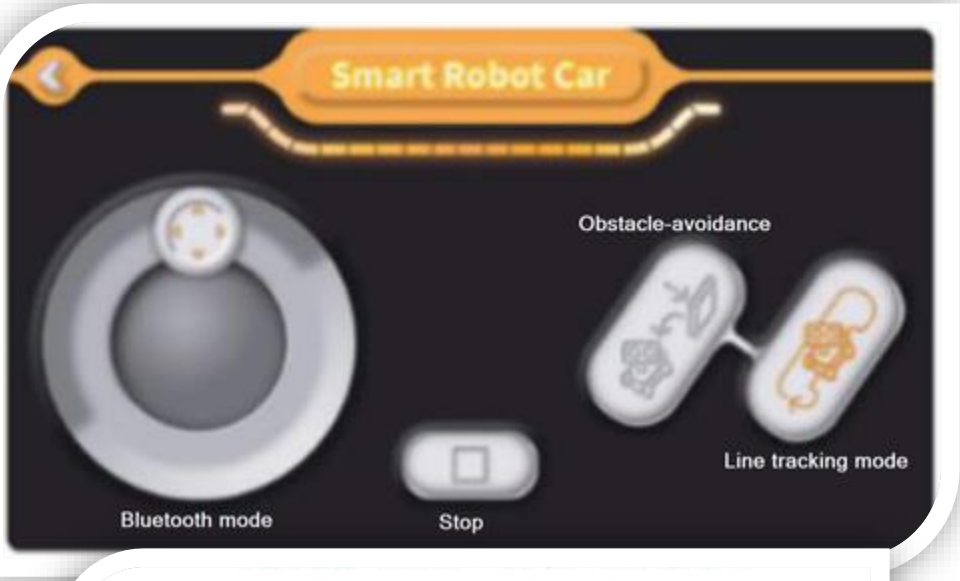
Codes in Appendix

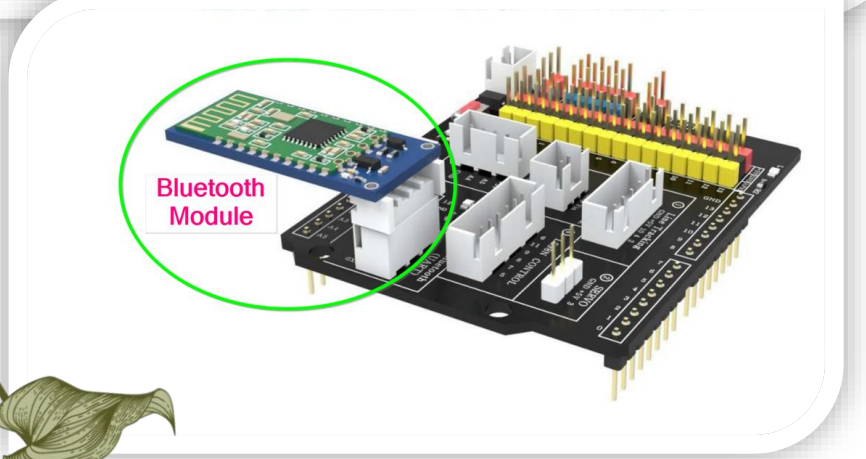
AUTO_GO.ino

```
//define L298n module IO Pin
#define ENA 5
#define ENB 6
#define IN1 7
#define IN2 8
#define IN3 9
#define IN4 11

void forward(){
  digitalWrite(ENA,HIGH); //enable L298n A channel
  digitalWrite(ENB,HIGH); //enable L298n B channel
  digitalWrite(IN1,HIGH); //set IN1 hight level
  digitalWrite(IN2,LOW);  //set IN2 low level
  digitalWrite(IN3,LOW);  //set IN3 low level
  digitalWrite(IN4,HIGH); //set IN4 hight level
  Serial.println("Forward");//send message to serial monitor
}

void back(){
  digitalWrite(ENA,HIGH);
  digitalWrite(ENB,HIGH);
  digitalWrite(IN1,LOW);
  digitalWrite(IN2,HIGH);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  Serial.println("Back");
}

void left(){
  digitalWrite(ENA,HIGH);
  digitalWrite(ENB,HIGH);
  digitalWrite(IN1,LOW);
  digitalWrite(IN2,HIGH);
  digitalWrite(IN3,LOW);
  digitalWrite(IN4,HIGH);
  Serial.println("Left");
}

void right(){
  digitalWrite(ENA,HIGH);
  digitalWrite(ENB,HIGH);
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  Serial.println("Right");
}

//before execute loop() function,
//setup() function will execute first and only execute once
void setup() {
  Serial.begin(9600);//open serial and set the baudrate
  pinMode(IN1,OUTPUT);//before useing io pin, pin mode must be set first
  pinMode(IN2,OUTPUT);
  pinMode(IN3,OUTPUT);
  pinMode(IN4,OUTPUT);
  pinMode(ENA,OUTPUT);
  pinMode(ENB,OUTPUT);
}

//Repeat execution
void loop() {
  forward();  //go forward
  delay(1000);//delay 1000 ms
  back();     //go back
  delay(1000);
  left();     //turning left
  delay(1000);
  right();    //turning right
  delay(1000);
}
```

Bluetooth Mode

Once you verified the assembly and motor operation of the Smart Car using the "Auto Go" sketch, you can use the "Bluetooth Car" Sketch form the downloaded files to perform the Bluetooth operational test. First, install the Bluetooth App, BLE Tool", on your smartphone from Google Play Store for Android or from App Store for an iPhone.





- **Then, upload the "Bluetooth Car" Sketch to your Smart Car Arduino board. A copy of the "Bluetooth Car" Sketch is also provided here.**
  Connect the Smart Car to the computer.
- Disconnect the Bluetooth module
- Upload the downloaded Arduino sketch, “bluetooth_car.ino” to your Smart Car.
- Disconnect the programming cable.
- Reinstall the Bluetooth module.
- Position the Smart Car on a flat surface and turn the power switch to ON
- Open the “BLE Tool” App on your smartphone.
- Pair the Smart Car Bluetooth with the App.
- Inside the “Rocker Control Panel” of the App, use the Bluetooth mode joystick control to move the car forward, back, left, right, and Stop to test the Bluetooth operation.

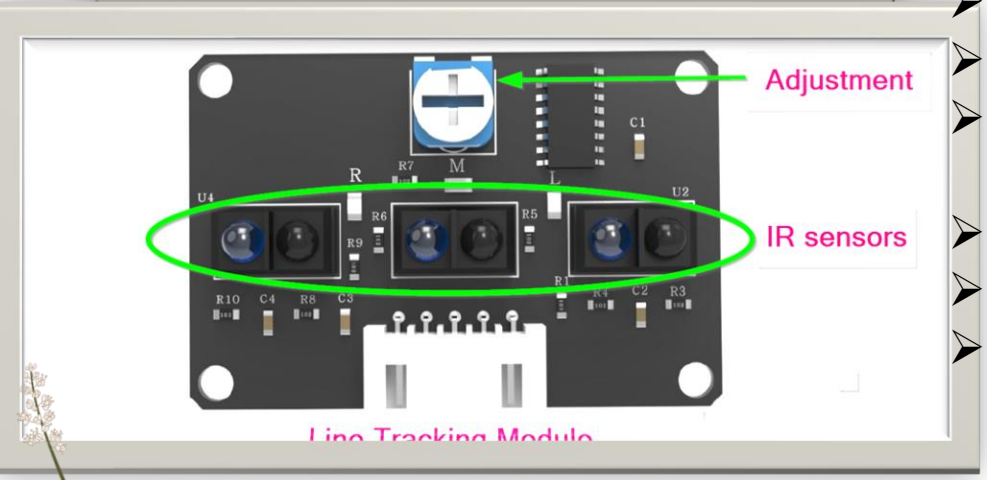
bluetooth_car.ino

```
#define ENA 5
#define ENB 6
#define IN1 7
#define IN2 8
#define IN3 9
#define IN4 11
#define LED 13

unsigned char carSpeed = 250;
bool state = LOW;

void forward(){
  digitalWrite(ENA,HIGH);
  digitalWrite(ENB,HIGH);
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  digitalWrite(IN3,LOW);
  digitalWrite(IN4,HIGH);
  Serial.println("Forward");
}

void back(){
  digitalWrite(ENA,HIGH);
  digitalWrite(ENB,HIGH);
  digitalWrite(IN1,LOW);
  digitalWrite(IN2,HIGH);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  Serial.println("Back");
}

void left(){
  analogWrite(ENA,carSpeed);
  analogWrite(ENB,carSpeed);
  digitalWrite(IN1,LOW);
  digitalWrite(IN2,HIGH);
  digitalWrite(IN3,LOW);
  digitalWrite(IN4,HIGH);
  Serial.println("Left");
}
```

```
}

void right(){
  analogWrite(ENA,carSpeed);
  analogWrite(ENB,carSpeed);
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  Serial.println("Right");
}

void stop(){
  digitalWrite(ENA,LOW);
  digitalWrite(ENB,LOW);
  Serial.println("Stop!");
}

void stateChange(){
  state = !state;
  digitalWrite(LED, state);
  Serial.println("Light");
}

void setup() {
  Serial.begin(9600);
  pinMode(LED, OUTPUT);
  pinMode(IN1,OUTPUT);
  pinMode(IN2,OUTPUT);
  pinMode(IN3,OUTPUT);
  pinMode(IN4,OUTPUT);
  pinMode(ENA,OUTPUT);
  pinMode(ENB,OUTPUT);
  stop();
}
```

```
void loop() {
  if(Serial.available())
  {
    char getstr = Serial.read();
    switch(getstr){
      case 'f': forward(); break;
      case 'b': back();    break;
      case 'l': left();    break;
      case 'r': right();   break;
      case 's': stop();    break;
      case 'a': stateChange(); break;
      default:  break;
    }
  }
}
```

Line Tracking Mode



- In order to test the Line Tracking Module, download the Arduino sketch, "Line_tracking_car.ino" to your Smart Car and position the Smart Car on the closed path. Turn on the power switch and then the Smart Car will start to navigate along the black line until you stop the car by turning the power switch off. The Arduino sketch is also provided here for convenience. Disconnect the Bluetooth module
- Upload the downloaded Arduino sketch, “Line_tracking_car.ino” to your Smart Car.
- Disconnect the programming cable.
- Reinstall the Bluetooth module.
- Position the Smart Car on the closed path (black tape), turn the power switch to ON
- Open the “ BLE Tool” App on your smartphone, if you haven’t done it so.
- Pair the Smart Car Bluetooth with the APP.
- Inside the “Rocker Control Panel” of the App, select the "Line Tracking" button and observe the Smart Car navigating along the closed path. Enjoy.

Line_tracking_car.ino

```
//Line Tracking IO define
#define LT_R !digitalRead(10)
#define LT_M !digitalRead(4)
#define LT_L !digitalRead(2)

#define ENA 5
#define ENB 6
#define IN1 7
#define IN2 8
#define IN3 9
#define IN4 11

#define carSpeed 250

void forward(){
  analogWrite(ENA, carSpeed);
  analogWrite(ENB, carSpeed);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  Serial.println("go forward!");
}

void back(){
  analogWrite(ENA, carSpeed);
  analogWrite(ENB, carSpeed);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  Serial.println("go back!");
}
```

```
void left(){
  analogWrite(ENA, carSpeed);
  analogWrite(ENB, carSpeed);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  Serial.println("go left!");
}

void right(){
  analogWrite(ENA, carSpeed);
  analogWrite(ENB, carSpeed);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  Serial.println("go right!");
}

void stop(){
   digitalWrite(ENA, LOW);
   digitalWrite(ENB, LOW);
   Serial.println("Stop!");
}

void setup(){
  Serial.begin(9600);
  pinMode(10,INPUT);
  pinMode(4,INPUT);
  pinMode(2,INPUT);
}
```

```
void loop() {
  if(LT_M){
    forward();
  }
  else if(LT_R) {
    right();
    while(LT_R);
  }
  else if(LT_L) {
    left();
    while(LT_L);
  }
}
```

Obstacle-Avoidance Mode

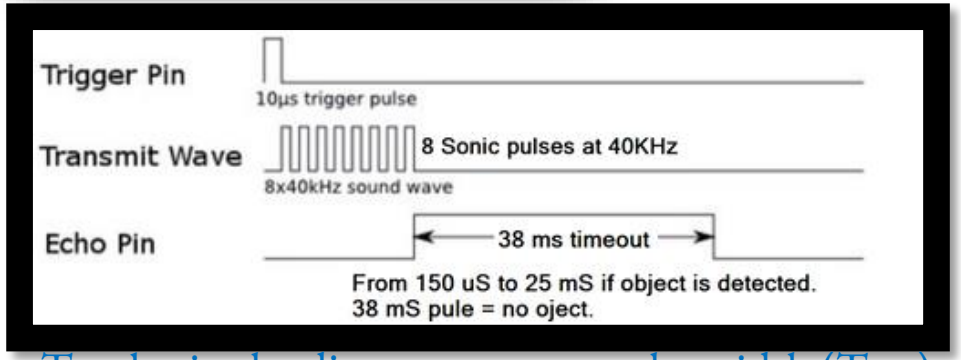
The ultrasonic sensor has four pins:

VCC – This is the 5-volt positive power connection.

TRIG – This is the "Trigger", an input for the pulse we will be sending from the ultrasonic transmitter.

ECHO – This is an output that sends back the received pulse.

GND – The Ground connection.



To obtain the distance, measure the width (Ton) of Echo pin.

Time = Width of Echo pulse, in uS (micro second)

- Distance in centimeters = Time / 58
- Distance in inches = Time / 148
- Or you can utilize the speed of sound, which is 340m/s

The ultrasonic module can be tested using Arduino microcontroller and the following sketch function.

```
int ultrasonic_test(){
  digitalWrite(Trig, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);
  float distance = pulseIn(Echo,
HIGH);
  distance = distance / 58;
}
```

•Obstacle Avoidance – Operation

The principle of obstacle or collision avoidance is as simple as “if – else if – else” statement in C++ or any other programing languages. The ultrasonic sensor module will detect the distance between the car and an obstacle in front of it and sending the data to the microcontroller. Then, the microcontroller sends a corrective action to the smart car and this process continues repeatedly.

The algorithm follows the following sequence.Ultrasonic sensor measures the distance to the nearest object until obstacle detected.

•Stop the car.
•Measure the distance to the right and left of the Smart Car.
•Turn the car in the direction that you measure the longest distance.
• Move forward.
•Upload the downloaded Arduino sketch, “Obstacle_Avoidance_Car.ino” to your Smart Car.
•Disconnect the programming cable.
•Reinstall the Bluetooth module.
•Position the Smart Car on a flat surface, turn the power switch to ON
•Open the “BLE Tool” App to your smartphone, if you haven’t done it on the previous section.
•Pair the Smart Car Bluetooth with the App.
•Inside the “Rocker Control Panel” of the App, select obstacle avoidance and enjoy the autonomous navigation in action.

Obstacle_Avoidance_Car.ino

```
#include <Servo.h>  //servo library
Servo myservo;      // create servo object to control servo

int Echo = A4;
int Trig = A5;

#define ENA 5
#define ENB 6
#define IN1 7
#define IN2 8
#define IN3 9
#define IN4 11
#define carSpeed 250
int rightDistance = 0, leftDistance = 0, middleDistance = 0;

void forward(){
  analogWrite(ENA, carSpeed);
  analogWrite(ENB, carSpeed);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  Serial.println("Forward");
}

void back() {
  analogWrite(ENA, carSpeed);
  analogWrite(ENB, carSpeed);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  Serial.println("Back");
}
```

```
void left() {
  analogWrite(ENA, carSpeed);
  analogWrite(ENB, carSpeed);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  Serial.println("Left");
}

void right() {
  analogWrite(ENA, carSpeed);
  analogWrite(ENB, carSpeed);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  Serial.println("Right");
}

void stop() {
  digitalWrite(ENA, LOW);
  digitalWrite(ENB, LOW);
  Serial.println("Stop!");
}

//Ultrasonic distance measurement Sub function
int Distance_test() {
  digitalWrite(Trig, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(20);
  digitalWrite(Trig, LOW);
  float Fdistance = pulseIn(Echo, HIGH);
  Fdistance= Fdistance / 58;
  return (int)Fdistance;
}
```

```
void setup() {
  myservo.attach(3,700,2400);  // attach servo on pin 3 to servo object
  Serial.begin(9600);
  pinMode(Echo, INPUT);
  pinMode(Trig, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(ENA, OUTPUT);
  pinMode(ENB, OUTPUT);
  stop();
}

void loop() {
    myservo.write(90);  //setservo position according to scaled value
    delay(500);
    middleDistance = Distance_test();

    if(middleDistance <= 40) {
      stop();
      delay(500);
      myservo.write(10);
      delay(1000);
      rightDistance = Distance_test();

      delay(500);
      myservo.write(90);
      delay(1000);
      myservo.write(180);
      delay(1000);
      leftDistance = Distance_test();

      delay(500);
      myservo.write(90);
      delay(1000);
      if(rightDistance > leftDistance) {
        right();
        delay(360);
      }
      else if(rightDistance < leftDistance) {
        left();
        delay(360);
      }
      else if((rightDistance <= 40) || (leftDistance <= 40)) {
        back();
        delay(180);
      }
      else {
        forward();
      }
    }
    else {
        forward();
    }
}
```

3D-Video

- 3D VIDEO

https://drive.google.com/file/d/1A6Up0yYtof8j6NiwP9hy5LY1pZkhlrB2/view?usp=share_link

- Bluetooth smart Robot car

https://drive.google.com/file/d/15xlaTEJ0iitTXJI593jNsaKOYIkEk3l-/view?usp=share_link

- Line Tracking Smart Robot Car

https://drive.google.com/file/d/19TptkaigApa1uHSMcX7IRNyZWdMNC8RA/view?usp=share_link

- Obstacle Avoidance Car

https://drive.google.com/file/d/1J58Tdl3vVIYg1RK4mMYCK9bem89Majs6/view?usp=share_link

- Infrared Remote Control Car

https://drive.google.com/file/d/1OfQempyMAheJR_98Z7nkb7P0pc1B7pXx/view?usp=share_link

REFERENCE

- https://create.arduino.cc/projecthub/samanfern/bluetooth-controlled-car-d5d9ca
- https://create.arduino.cc/projecthub/comptek4/ir-remote-control-car-fcb8a5?ref=search&ref_id=Infrared%20Remote%20Control%20Car&offset=1
- https://create.arduino.cc/projecthub/chandran0303cn/obstacle-avoidance-bot-using-ir-sensors-08f8e9?ref=search&ref_id=Obstacle%20Avoidance%20Car&offset=1
- https://create.arduino.cc/projecthub/SXHXC/smart-face-tracking-robot-car-f23d76?ref=search&ref_id=Line%20Tracking%20Car&offset=5

Thank you

- ARUMULLA YASWANTH REDDY

  20BES7010

- KAKUMANU SRIMANI

  20BCD7133

- DAMARLA BHUVAN SRI SAI

  20BCD7096

- NUTHALAPATI JASHVIKA

  20BCD7115

- KANDUKURI PRANAVI

  20BCR7104

- CHANDINI KOLLATI

  20BCE7379