School of Engineering and Applied Science,

Ahmedabad University

ANDROID PHONE CONTROLLED ROBOT

Group - 7 The Sparks

TEAM MEMBERS :	**ROLL NO:**
Yesha Shastri	1741035
Devshree Patel	1741075
Muskan Matwani	1741027
Naishi Shah	1741033

1. Introduction and Motivation:

Robots are gadgets which can be programmed by humans in order to give them human-like abilities. Since humans cannot always be accurate in performing certain tasks, robots can be useful to provide us with unbiased and accurate results. In the areas where human intervention is not possible like, in military surveillance, use of robots becomes a necessity. Robots in the agricultural sector will be of great help to the nation. Looking at the magnificence of the applications of the robot and the ease of availability of the Android technology motivated us to think of developing a robot which will be useful in the real-life scenario.

Description:

The system will work on the control commands sent by the app. The commands could either be in the form of touch on the app screen or by the speech. These commands will be received by the Bluetooth module attached to the robot. The received data would then be passed on to Arduino which will further send instructions to the motor driver. The motor driver will amplify the low-current control signal to high-current control signal for the purpose of driving the motors. Ultimately, the motors will make the robot move in desired directions. In return, the robot also sends back information regarding the distance from the closest obstacle to the phone and if it reaches very close to a certain obstacle it will automatically stop.

Final Outcome:

The robot will be able to move in forward, backward, left and right directions. The user will also be able to control the speed of the robot through the app interface. Speech commands could be from (forward, backward, left, right and stop). The spoken commands will be visible on the app screen. Hence, it will result in a full-fledged Android phone controlled robot ready for the addition of features in future to work for real-life applications.

2. Market Survey:

Several products alike our product exist in market. Here, three of them will be discussed by comparing them based on their working and certain functionalities.

1. ***DTMF Controlled Robot*** **:** This product utilises a special technology called DTMF - Dual Tone Multiple Frequency. DTMF is a decoder module with a decoder IC which decodes DTMF tone signal to digital signal for arduino to accept it digitally.

   The “mobile controlled robot” is controlled by a mobile phone that makes a call to the another mobile phone which is attached to the robot. In the course of a call, if any button is pressed, a tone corresponding to the button pressed is heard at the other end of the call. This tone is called “Dual Tone Multiple-Frequency” (DTMF) tone. The robot perceives this DTMF tone with the help of the phone stacked on the robot. The received tone is processed by the microcontroller with the help of a decoder IC. The microcontroller then transmits the signal to the motor driver ICs to operate the motors.

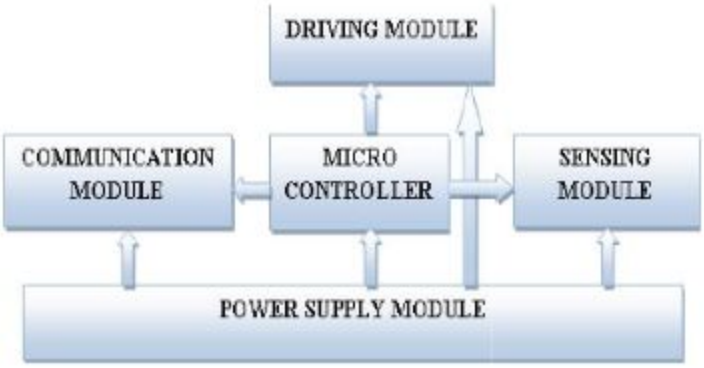
   Advantage:
   - This robot is used for military surveillance.
   - DTMF robot with slight modifications can be used in industrial applications.
   - DTMF robot with human detector sensor can be used at the time of disasters like earth quake to detect the human under buildings.

Comparison with our product: DTMF controlled Robot runs over mobile DTMF technology that exists in Dial tone. There are some frequencies that we use to create DTMF tone by adding two or more frequencies. For eg, tone 1 is created by adding 1209 Hz and 697 Hz frequency. Whereas in our case, Bluetooth module HC05 is controlled by an android app on the phone. It sends commands from the app which is used to control the directions, speed and other functionalities of the robot car.

2. Wireless multifunctional robot for military application: The below given are some of the advantages which have become motivation for the development of following robot.

- Use of 3G Technology will provide wide range of operation and manual control.
- Cost effective by using cell phone as video camera
- Energy efficient by using renewable resource for power supply.
- Used to explore hazardous areas and used for espionage purpose.

Following are the major modules and functionality of the robot:



Power supply module - This module uses 10 watts as renewable source of power supply. A rechargeable battery used to provide continuous power supply which is indeed connected to solar panel via charge controller.

Sensor module - Comprises of PIR sensor, metal detector, gas sensor, temperature sensor and ultrasonic sensor.

Driving module - L293D motor driver governs the rotation and speed of the motors.

Communication module - This module enables both data and video transmission by 3G video calling. GSM module is used to send alert messages to user whenever any hazardous situation occurs or any sensor becomes active. DTMF Decoder MT8870 IC is used to control the robot in manual mode. The decoder converts the dual tone frequencies into their binary equivalent and
controller perform operation corresponding to these binary equivalents. In order to change the path of robot during manual mode, the user initiates the 3G video call to the mobile phone equipped to robot. The smart phone is equipped on the robot to monitor the surrounding and kept on auto receive mode to provide live view of surrounding of robot in order to change its path.

Hence, the overall functionality of the robot is to be used as a surveillance robot for security purposes and emergency rescue where humans cannot be present at that time instant.

Comparison with our product:
Sensor module - Our product uses ultrasonic and IR sensors to detect obstacles upto a certain range.
Driving module - For driving our robot the functionality is similar to that of given robot. Our robot uses 2 L293D motor drivers to govern 4 geared motors.
Communication module - Communication to our product is done using an android app and a bluetooth module which is attached to the robot.
Power supply module - 12V battery is required to power our vehicle. As a part of our future work, we can also install a solar panel along with a rechargeable battery to make the power source renewable.

3. DAKSH - A remotely operated vehicle (ROV)

Daksh is an electrically powered Remotely Operated Vehicle (ROV) designer and developed by the Indian state-owned Defence Research and

Development Organisation (DRDO) at the Research and Development Establishment (Engineers), Pune, India. Primarily designed to recover improvised explosive devices (IEDs), ROV Daksh can safely locate, handle and destroy all types of hazardous objects. The DRDO transferred its ROV technology for production at Dynalog, Theta Controls and Bharat Electronics. In 2010, the Indian Army placed a limited-series-production order for 20 Daksh vehicles after extensive trials, testing and acceptance. Daksh will serve bomb disposal units (BDU) of the army, police and paramilitary forces in handling IEDs and other hazardous materials. The Indian Army has the requirement for 100 ROVs.

Major functionalities:

The ROV is based on a motorised pan-tilt platform.

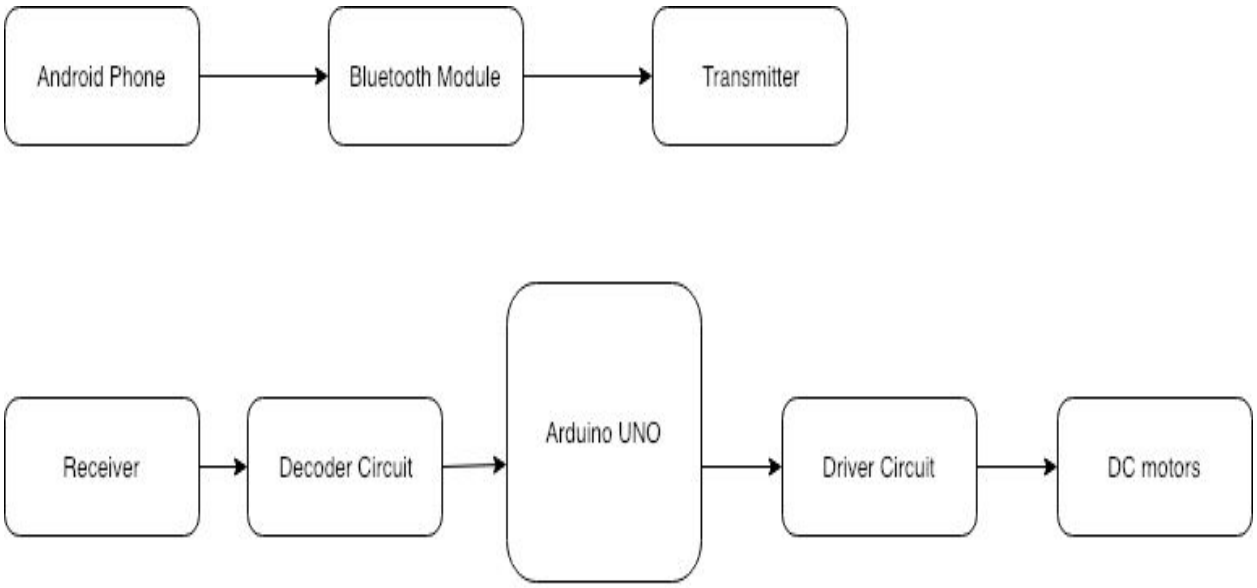
- Range - 500m
- The vehicle's manipulator arm can handle hazardous objects of up to 20kg from 2.5m and 9kg from a 4m distance.
- Daksh can climb stairs and negotiate steep slopes.
- The solid rubber wheels of Daksh can withstand blast impacts.
- The vehicle can tow suspected platforms and operate continuously for three hours once fully recharged.

ROV Daksh is equipped with multiple cameras, IED (Improvised Explosive Device) handling equipment, nuclear biological chemical (NBC) reconnaissance systems, a master control station (MCS) and a shotgun. The ROV Daksh and MCS are transported by a specially designed carrier vehicle.

Comparison with our product:

Our product is feasible to use in the terrains with many obstacles.The ultrasonic sensor in our product is reliable upto 4m whereas the range of daksh is 500m.In future, we can improve our product by implementing caterpillar track or tank tread to make it work on slopes and stairs. We can also improve the quality of material of rubber used in making tyres, such that it can withstand blast impacts like DAKSH.

3. Block Diagram:

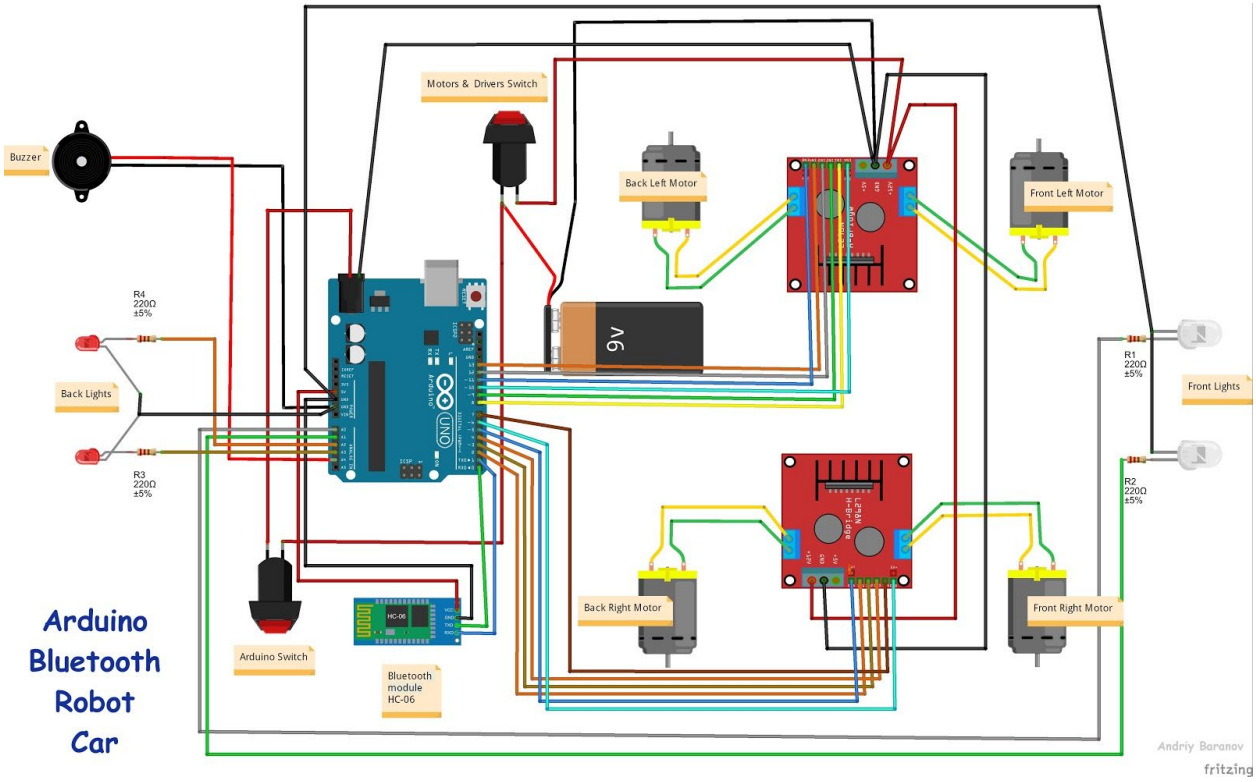


Block diagram of Proposed System

The system comprises of 2 parts namely, the software and the hardware(circuit).

- The software: The command from the Android phone is sent to the transmitter of the Bluetooth module which in turn sends it to the hardware of the system.
- The hardware: The circuit consists of the microcontroller, Arduino UNO, and the other components for processing and execution of the commands received by the hardware.

4. Circuit Diagram:



The circuit comprises:

- Actuators:
  - DC Motor: The DC motor was chosen taking into consideration the small amount of voltage on which it runs with great efficiency
  - Buzzer: An audio-signalling device that produces sound when given electrical energy. The buzzer has been chosen to produce sound for the horn or as a proximity signal.
  - LED: LEDs are used as front and rear lights of the car.

- Sensors:
  - Ultrasonic sensor: Used as a proximity sensor at the rear of the car.

- IR sensor: Used as proximity sensor at the front end of the car.

SELECTION CRITERIA OF COMPONENTS:

Arduino Uno:

It is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It is intended for artists, designers, hobbyists and anyone interested in creating interactive objects or environments.
The board is very easy to use as the user simply needs to connect it to a computer with a USB cable, or power it with an AC-to-DC adaptor or battery to get started. The microcontroller on the board is programmed using Arduino programming language using the Arduino development environment.

DC motor:

A DC motor is a type of motor with a magnetic coil inside. When an electric current passes through the coil, the magnetic force generated proceeds to turn the motor. The current is passed through a commutator before entering the coil, which switches the direction of the current at the apex point, so the spinning continues. The speed of the motor can be controlled by limiting the current, and the direction is affected by the direction of the current.

HC 06 Bluetooth Module:

The HC-06 is a class 2 slave Bluetooth module designed for transparent wireless serial communication. Once it is paired to a master Bluetooth device such as PC, and tablet, its operation becomes transparent to the user. All data received through the serial input is immediately transmitted over the air. When the module receives wireless data, it is sent out through the serial interface exactly at it is received. No user code specific to the Bluetooth module is needed at all in the user microcontroller program. The HC-06 will work with a supply voltage of 3.6VDC to 6VDC, however, the

logic level of RXD pin is 3.3V and is not 5V tolerant. It is responsible for enabling Bluetooth Communication between Arduino and Android Phone.

L293D:

L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motor with a single L293D IC. Here, 2 L293D drivers control 4 DC Motors which are connected to wheels of the robot. Also, it is one of the most popular drivers in the market. There are several reasons which make L293D the preferred driver to the users, such as, cheap price (compared to other drivers), proper shape and size, easy control, no need for protective circuit and diodes, no need for heat sinks and good resistance to temperature and high-speed variations. This IC can set up motors with a voltage between 5V to 36V and a current of up to 600 mA. However, it can withstand a current up to 1200 mA in 100 microsecond and non-repetitive. The frequency of this IC is 5 kHz.

5. Arduino and other Microcontroller features:

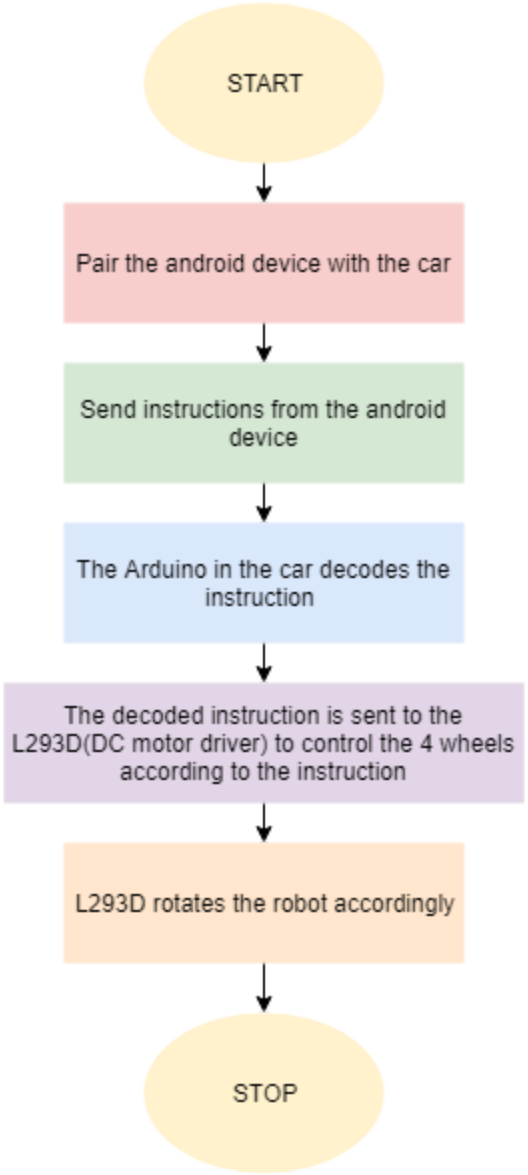
Arduino is an open-source, prototyping platform. It has many features which include:

- *14 digital input/output pins, out of which 6 can be used as output PWM pins:* We have used these pins as input, output as well as PWM output pins in our car to read from sensors, switch on or switch off LEDs and to control speed of the car respectively
- *6 analog inputs:* We have used analog input pins to take inputs from the bluetooth module HC-05
- *16 MHz quartz crystal:* we have used this for specific time delays
- *USB connection*
- *Power jack:* To power the Arduino which can in turn give power of upto 5V to an actuator
- *Reset button*

Point of comparison	**Arduino**	**Raspberry Pi**	**Intel Galileo**	**BeagleBone**
Description	A microcontroller i.e. it can run only one program at a time	A mini computer i.e. it can run multiple programs at a time	An open source single board computer	Mini computer
Power	Can be easily powered using a battery pack	Difficult to power using a battery pack	Difficult to power using a battery pack	Difficult to power using a battery pack

Clock speed	Has a frequency of 16MHz	Frequency of 700MHz	Frequency of 400 MHz	Frequency of 700MHz
Storage	Has onboard storage of 32kB	No onboard storage and thus, uses an SD card	Using an SD card, 32GB of data can be stored	4GB of data can be stored using a micro SD card
Digital input/ output pins	14	8	14	66
Analog input	6 10 bit	N/A	6 10 bit	7 12 bit
PWM outputs	6		6	8
Interfacing with sensors	very simple to interface sensors and other electronic components to Arduino	requires complex tasks like installing libraries and software for interfacing sensors and other components	Same as Arduino	Works better with sensors if the project is on a larger scale as compared to Arduino

6. Flowchart:


START
Pair the android device with the car
Send instructions from the android device
The Arduino in the car decodes the instruction
The decoded instruction is sent to the L293D(DC motor driver) to control the 4 wheels according to the instruction
L293D rotates the robot accordingly
STOP

7. Details of sensors:

Sensors used:

1.Ultrasonic Sensor HC-SR04

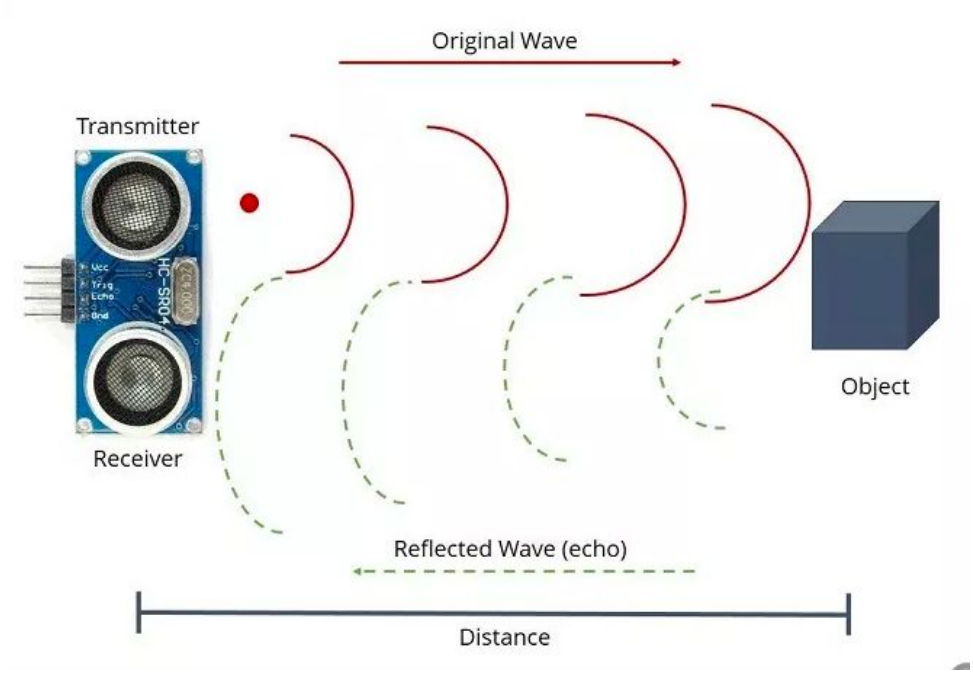
<u>Operating principle of Ultrasonic sensor:-</u>

Ultrasonic sensors emit short, high-frequency sound pulses at regular intervals. These propagate in the air at the velocity of sound. If they strike an object, then they are reflected back as echo signals to the sensor, which itself computes the distance to the target based on the time-span between emitting the signal and receiving the echo.

The ultrasonic sensor uses sonar to determine the distance to an object. Here's what happens:

1. The transmitter (trig pin) sends a signal: a high-frequency sound.

2. When the signal finds an object, it is reflected and the transmitter (echo pin) receives it.



Measurement Range:-

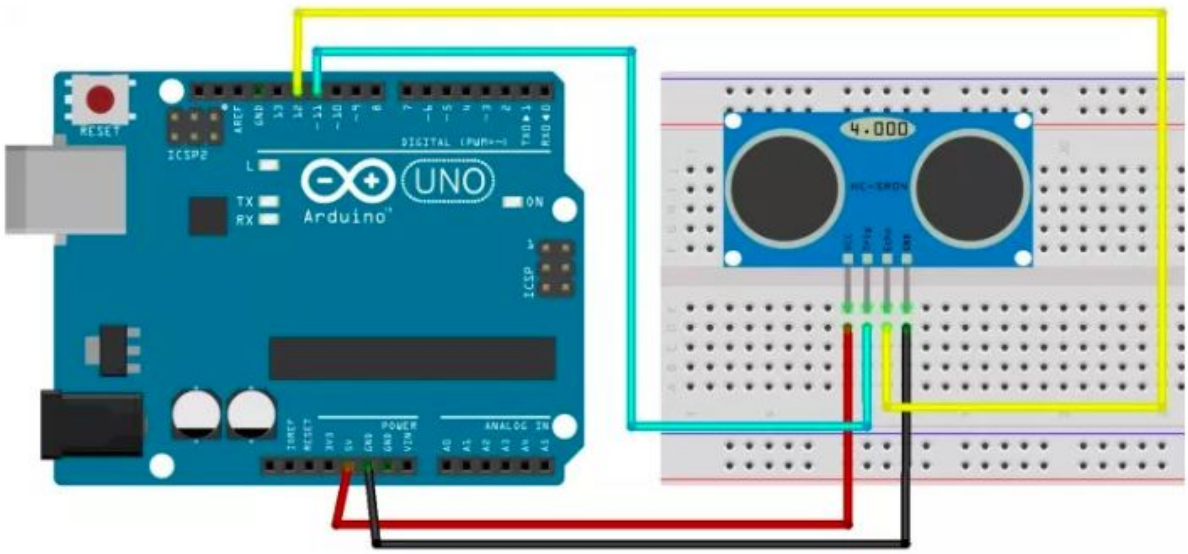
1.Distance:- 2 cm to 4 m
2.Angle:-15 degrees

Physical dimensions:- 45mm x 20mm x 15mm

Power ratings :- +5V DC voltage , 15mA current

Pin diagram:-

Ultrasonic Sensor HC-SR04	Arduino
VCC	5V
Trig	Pin 11
Echo	Pin 12
GND	GND



The sensors consist of two main components:

An Ultrasonic Transmitter – This transmits the ultrasonic sound pulses, it operates at 40 KHz

An Ultrasonic Receiver – The receiver listens for the transmitted pulses. If it receives them it produces an output pulse whose width can be used to determine the distance the pulse travelled.

The HC-SR04 Ultrasonic Module has 4 pins, Ground, VCC, Trig and Echo. The Ground and the VCC pins of the module needs to be connected to the Ground and the 5 volts pins on the Arduino board respectively and the trig and echo pins to any digital I/O pins.

In order to generate the ultrasound, you need to set the Trig on a High State for 10 µs. That will send out an 8 cycle sonic burst which will travel at the speed sound and it will be received in the Echo pin. The Echo pin will output the time in microseconds the sound wave travelled.

CODE:-

```
const int trigPin = 11;
const int echoPin = 12;
long duration;
int distance;
void setup()
{
        pinMode(trigPin, OUTPUT);
        pinMode(echoPin, INPUT);
        Serial.begin(9600);
}
void loop()
{
        digitalWrite(trigPin, LOW);
        delayMicroseconds(2);
        digitalWrite(trigPin, HIGH);
        delayMicroseconds(10);
        digitalWrite(trigPin, LOW);
        duration = pulseIn(echoPin, HIGH);
        distance= duration*0.034/2;
        Serial.print("Distance: ");
        Serial.println(distance);
}
```

SIZE OF CODE:- 855 bytes

2.Bluetooth module HC05

Working principle of HC05 :-

HC-05 Bluetooth module provides switching mode between master and slave mode which means it able to use neither receiving nor transmitting data.
The Android app is built to send serial data to the Bluetooth Module HC-05 by pressing the ON button. As Bluetooth Module HC-05 works on serial communication. It receives the data from the app and sends it through the TX pin of Bluetooth module to RX pin of Arduino. The uploaded code inside Arduino checks the data received. If the received data is 1, the LED turns ON, and if the received data is 0 the LED turns OFF.

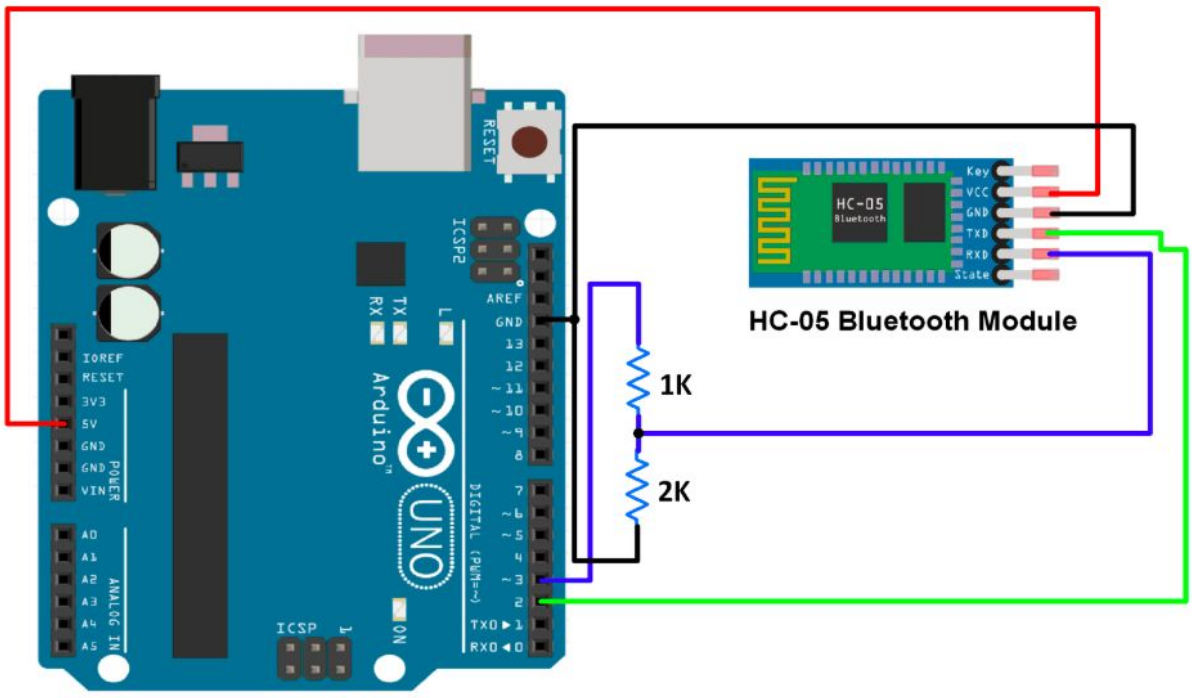
Measurement Range: 10 meters

Physical dimensions:- 26.9mm x 13mm x 2.2 mm

Power ratings :- +3.3V DC ,50mA

Pin Description and Diagram:-

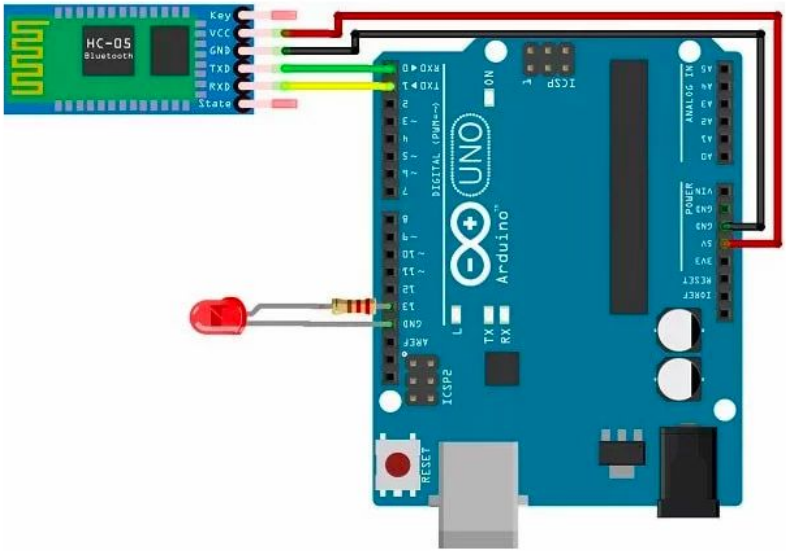
Pin	Description	Function
VCC	+5V	Connect to +5V
GND	Ground	Connect to Ground
TXD	UART_TXD, Bluetooth serial signal sending PIN	Connect with the MCU's (Microcontroller and etc) RXD PIN.
RXD	UART_RXD, Bluetooth serial signal receiving PIN	Connect with the MCU's (Microcontroller and etc) TXD PIN.
KEY	Mode switch input	If it is input low level or connect to the air, the module is at paired or communication mode. If it's input high level, the module will enter to AT mode.



We need to connect the RXD and TXD pins of Bluetooth module with RX and TX pins of Arduino respectively. VCC and GND pins are connected to their respective pins of Arduino. Key enable and state pins are left untouched. For checking the working of Bluetooth module we need to make

use LED. If LED glows after Bluetooth connection then the module works fine.

LED interfacing diagram with HC-05 and arduino:-



if received data is equal to 1, LED turns on and if data is equal to 0, LED turns OFF.

CODE:-

```
char data = 0; //Variable for storing received data
void setup()
{
   Serial.begin(9600); //Sets the baud for serial data transmission
   pinMode(13, OUTPUT); //Sets digital pin 13 as output pin
}
void loop()
{
  if(Serial.available() > 0)  // Send data only when you receive data:
  {
```

```
    data = Serial.read();       //Read the  incoming  data and store it into
variable data
    Serial.print(data);         //Print Value inside data in Serial monitor
    if(data == '1')             // Checks whether value of data is equal to 1
      digitalWrite(13, HIGH);   //If value is 1 then LED turns ON
    else if(data == '0')        //  Checks  whether value of data is equal to 0
      digitalWrite(13, LOW);    //If value is 0 then LED turns OFF
  }
}
```

SIZE OF CODE:- 840 bytes

3. IR SENSOR:

WORKING PRINCIPLE: An infrared sensor is an electronic device, that emits in order to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion.These types of sensors measures only infrared radiation, rather than emitting it that is called as a passive IR sensor. Usually in the infrared spectrum, all the objects radiate some form of thermal radiations. These types of radiations are invisible to our eyes, that can be detected by an infrared sensor.The emitter is simply an IR LED (Light Emitting Diode) and the detector is simply an IR photodiode which is sensitive to IR light of the same wavelength as that emitted by the IR LED. When IR light falls on the photodiode, The resistances and these output voltages, change in proportion to the magnitude of the IR light received.

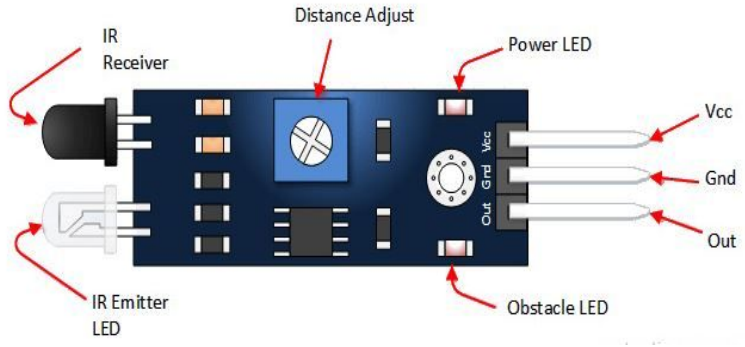
MEASUREMENT RANGE: 2 - 30 cm

MEASUREMENT UNIT: cm - a metric unit of length, equal to one hundredth of a metre.

PHYSICAL DIMENSIONS: 50*20*10 mm

POWER RATINGS: 3-5 VDC

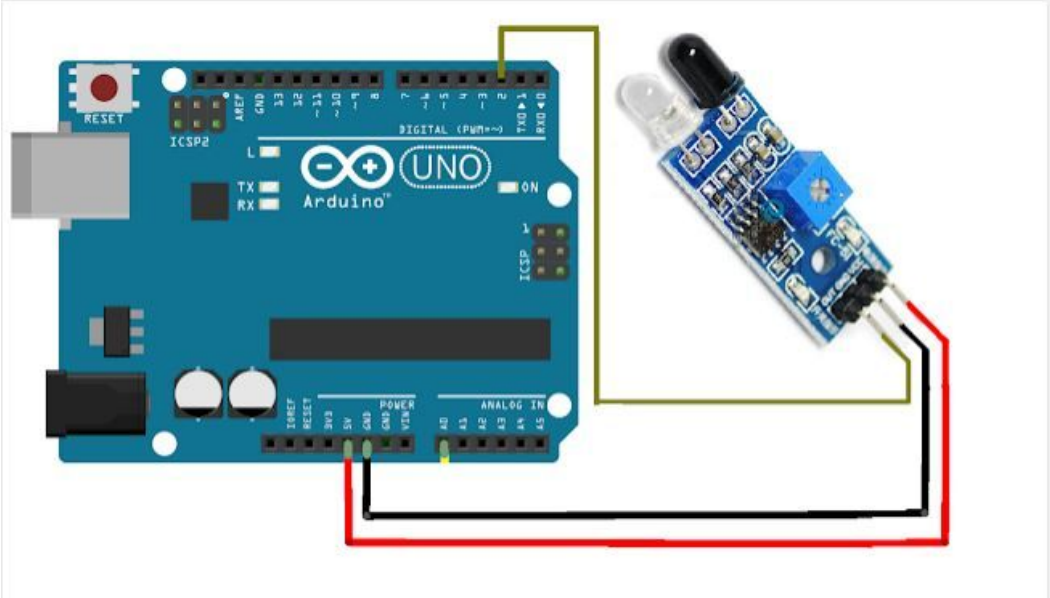
PIN DIAGRAM:



PIN DESCRIPTION:

Pin, Control Indicator	Description
Vcc	3.3 to 5 Vdc Supply Input
Gnd	Ground Input
Out	Output that goes low when obstacle is in range

INTERFACING WITH ARDUINO:



The obstacle sensor gives logic 0 as output to pin2 when there is no obstacle in front of it, and when obstacle is placed in front of it, it will give logic high output i.e. +5V. These logic changes are read on the arduino using digitalRead Command.

CODE:

```
const int ProxSensor=2;

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
  //Pin 2 is connected to the output of proximity sensor
  pinMode(ProxSensor,INPUT);
}

void loop() {
  if(digitalRead(ProxSensor)==HIGH)      //Check the sensor output
  {
    digitalWrite(13, HIGH);   // set the LED on
  }
```

```
  else
  {
    digitalWrite(13, LOW);    // set the LED off
  }
  delay(100);              // wait for a second
}
```

SIZE OF C CODE: 1054 bytes

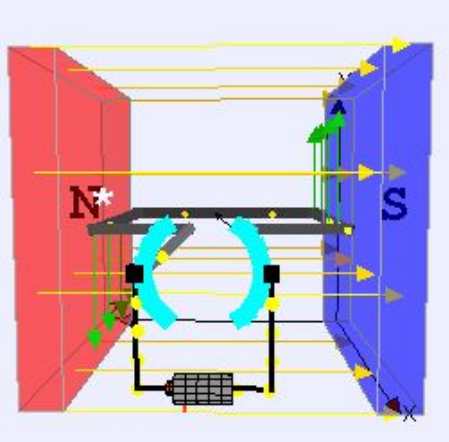
8. Details of Actuators:

ACTUATORS USED:

(A)DC Motor:

Operating principle of DC motor:-

An electric motor is an electrical machine which converts electrical energy into mechanical energy. The basic working principle of a DC motor is: "***whenever a current carrying conductor is placed in a magnetic field, it experiences a mechanical force***". The direction of this force is given by Fleming's left-hand rule and its magnitude is given by F = BIL. Where, B = magnetic flux density, I = current and L = length of the conductor within the magnetic field.



Animation: Working of DC Motor

(credit: Lookang)

When armature windings are connected to a DC supply, an electric current sets up in the winding. Magnetic field may be provided by field winding

(electromagnetism) or by using permanent magnets. In this case, current carrying armature conductors experience a force due to the magnetic field, according to the principle stated above.

Commutator is made segmented to achieve unidirectional torque. Otherwise, the direction of force would have reversed every time when the direction of movement of conductor is reversed in the magnetic field.

Physical dimensions:- 20*65 mm

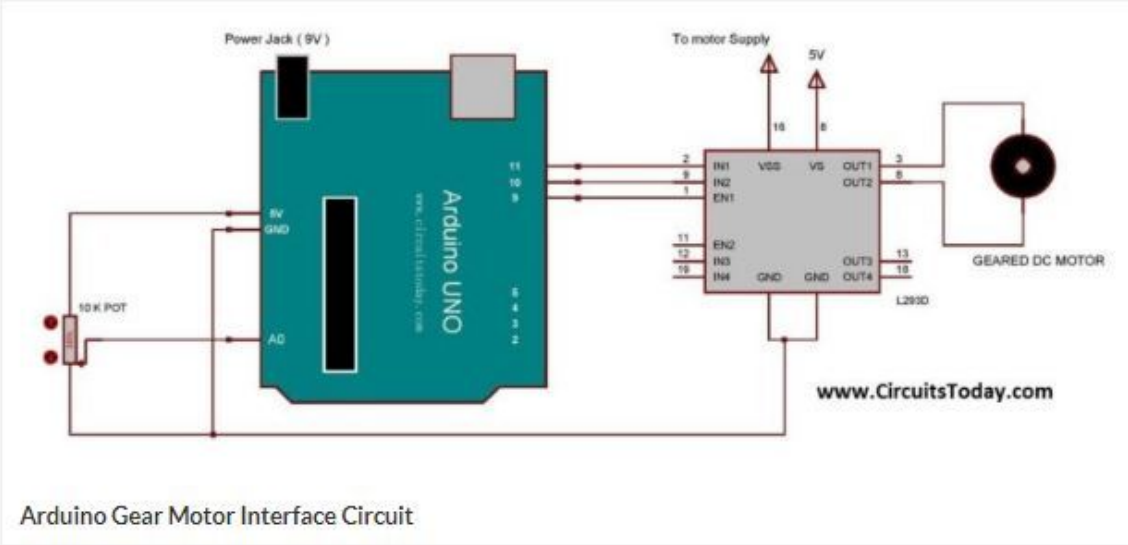
Power ratings:- 3-12 V, without loading 40 - 180 mA, 20 - 100 rpm

MODEL		VOLTAGE		NO LOAD		AT MAXIMUM EFFICIENCY					STALL		
		OPERATING RANGE	NOMINAL	SPEED	CURRENT	SPEED	CURRENT	TORQUE		OUTPUT	TORQUE		CURRENT
			V	r/min	A	r/min	A	N.M	Kg.cm	w	N.M	Kg.cm	A
TGPO1S-A130	14150-120	3.0 -12.0	4.5	100	0.20	84	0.50	0.066	0.675	0.40	0.265	2.700	1.25
	18100-220	3.0 -12.0	3.0	50	0.20	40	0.50	0.120	1.233	0.6	0.393	4.003	2.15

Pin Diagram:-

IN1	IN2	Motor behaviour
0	0	**BRAKE**
1	0	**FORWARD**
0	1	**BACKWARD**
1	1	**BRAKE**

INTERFACING WITH ARDUINO:



Arduino Gear Motor Interface Circuit

Code:-

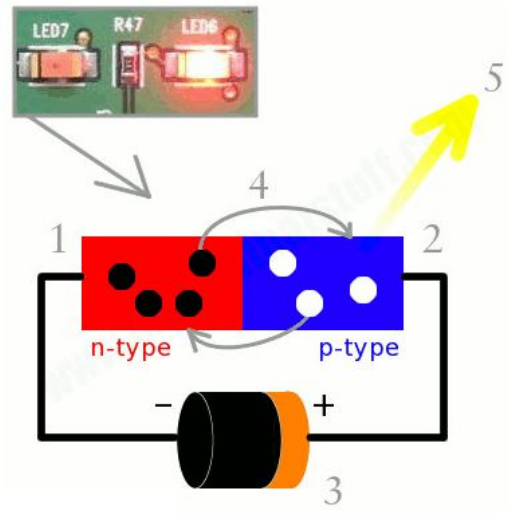
```
const int pwm = 2 ; //initializing pin 2 as pwm
const int in_1 = 10 ;
const int in_2 = 11 ;
//For providing logic to L298 IC to choose the direction of the DC motor

void setup() {
  pinMode(pwm,OUTPUT) ; //we have to set PWM pin as output
  pinMode(in_1,OUTPUT) ; //Logic pins are also set as output
  pinMode(in_2,OUTPUT) ;
}

void loop() {
  //For Clock wise motion , in_1 = High , in_2 = Low
  digitalWrite(in_1,HIGH) ;
  digitalWrite(in_2,LOW) ;
  analogWrite(pwm,255) ;
  /* setting pwm of the motor to 255 we can change the speed of rotation
```

```
   by changing pwm input but we are only using arduino so we are using
highest
   value to driver the motor */
   //Clockwise for 3 secs
   delay(3000) ;
   //For brake
   digitalWrite(in_1,HIGH) ;
   digitalWrite(in_2,HIGH) ;
   delay(1000) ;
   //For Anti Clock-wise motion - IN_1 = LOW , IN_2 = HIGH
   digitalWrite(in_1,LOW) ;
   digitalWrite(in_2,HIGH) ;
   delay(3000) ;
   //For brake
   digitalWrite(in_1,HIGH) ;
   digitalWrite(in_2,HIGH) ;
   delay(1000) ;
}
```

Size of C code:- 1074 bytes

(B) LED:-

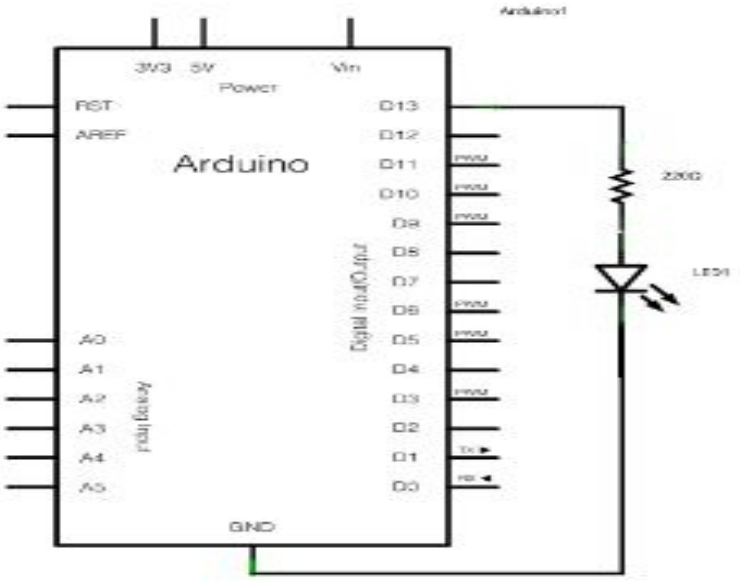
Operating Principle:- A light-emitting diode is a two-lead semiconductor light source. It is a p–n junction diode that emits light when activated. When a suitable voltage is applied to the leads, electrons are able to recombine with electron holes within the device, releasing energy in the form of photons.



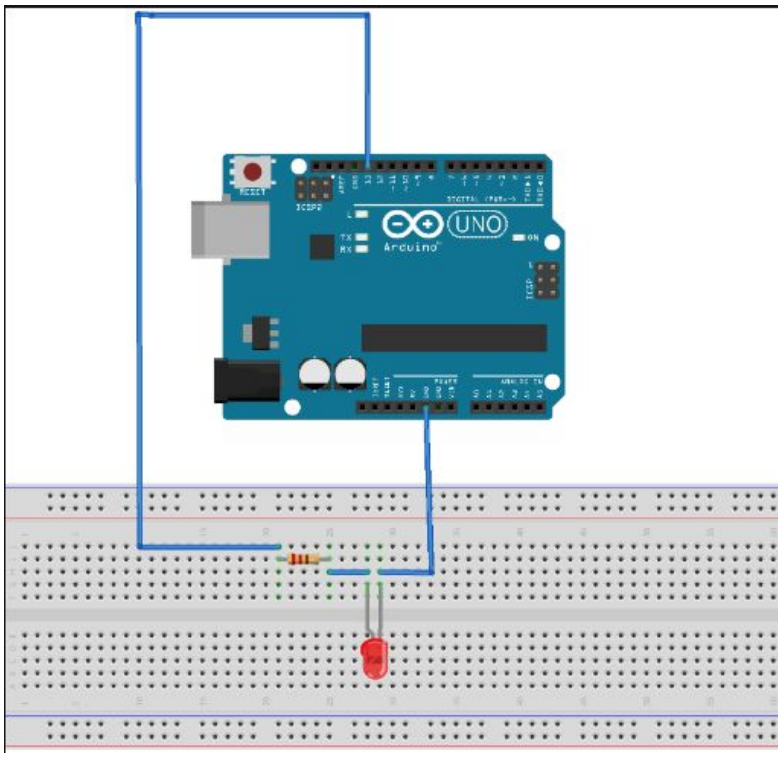
Physical dimensions:- 5mm

Power ratings:- 6V DC , load current <=280mA, operating voltage 1.5-6.5V DC

Pin diagram:-



Interfacing with arduino:-



Code:-

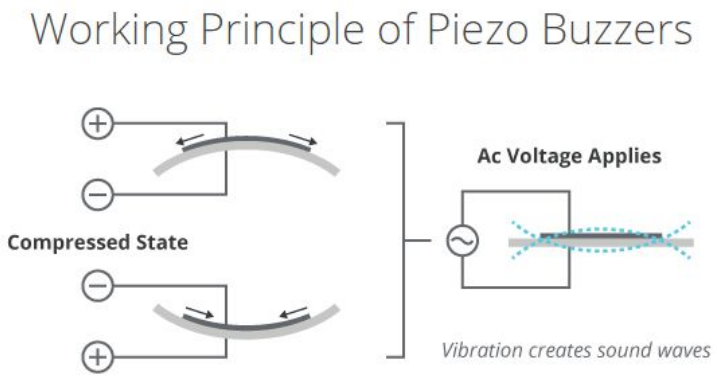
```
int led = 13; // the pin the LED is connected to
void setup() {
  pinMode(led, OUTPUT); // Declare the LED as an output
}

void loop() {
  digitalWrite(led, HIGH); // Turn the LED on
}
```

Size of C code:- 724 bytes

(C) Buzzer

Operating Principle:-

The vibrating disk in a magnetic buzzer is attracted to the pole by the magnetic field. When an oscillating signal is moved through the coil, it produces a fluctuating magnetic field which vibrates the disk at a frequency equal to that of the drive signal.



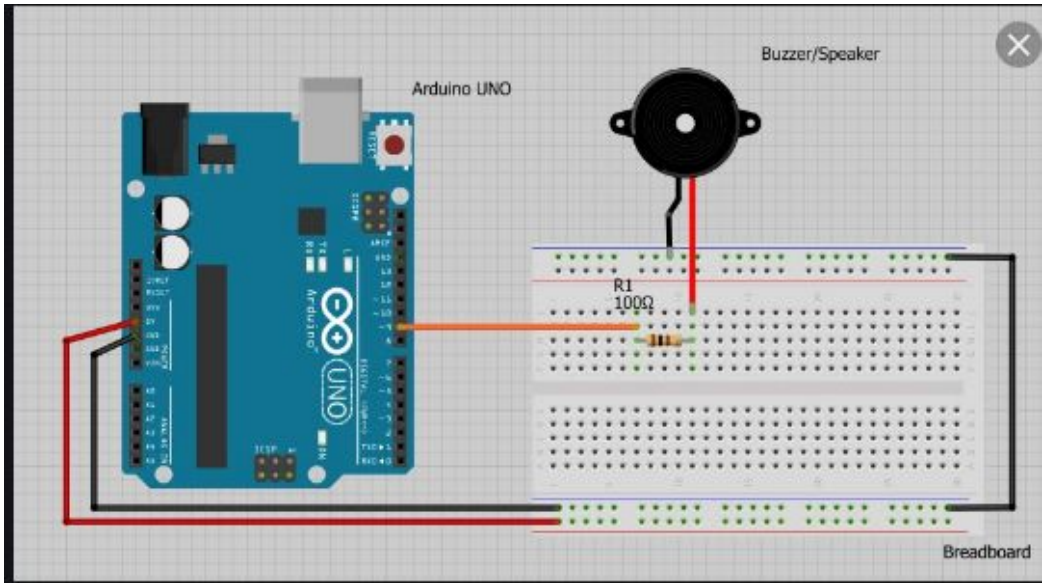
Power ratings:-

Operating voltage:- 4-8V DC
Rated current:- <30mA

Pin diagram:-

Pin Number	Pin Name	Description
1	Positive	Identified by (+) symbol or longer terminal lead. Can be powered by 6V DC
2	Negative	Identified by short terminal lead. Typically connected to the ground of the circuit

Interfacing with arduino:-



Code:-

```
const int buzzer = 9; //buzzer to arduino pin 9
void setup()
{
  pinMode(buzzer, OUTPUT); // Set buzzer - pin 9 as an output
}
void loop()
{
   tone(buzzer, 1000); // Send 1KHz sound signal...
  delay(1000);        // ...for 1 sec
  noTone(buzzer);     // Stop sound...
  delay(1000);        // ...for 1 sec
 }
```

Size of c code:- 1664 bytes

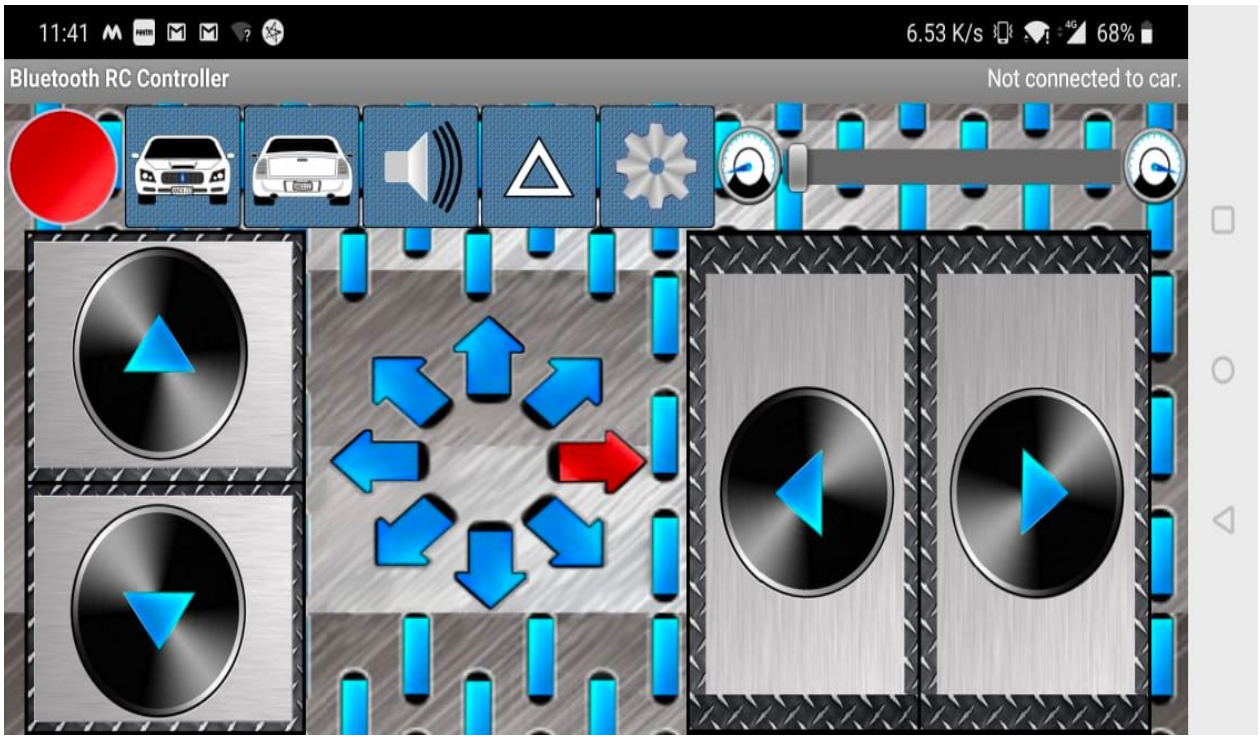
9.DETAILS OF SUPPORTING TOOLS:

1. Android application technical details

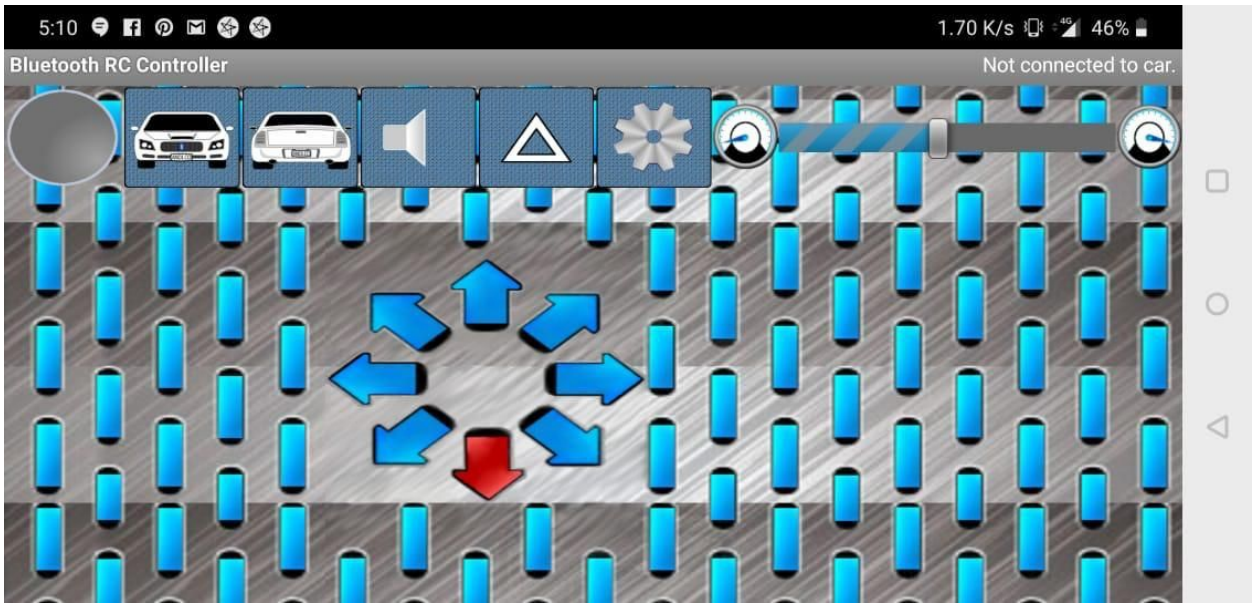
The android application we have used with this project is called bluetooth RC controller. The app is connected to the car via bluetooth. The car won't respond if the bluetooth connection is lost. To connect the device via bluetooth app we need to enter a pin 0000 or 1234. Once connected, it can be used for controlling the car, and if the app is closed then the bluetooth connection goes away. The application is used to control actuators , i.e motors connected to wheels.The app is used to control the directions of motors- left,right,back and forward. It is also used for controlling speed coefficient and horn. It also has the functionality of turning on the front and back head lights. It includes an additional functionality of controlling the car using accelerometer other than gesture control. For instance, when you tilt the phone containing the app in a specific direction, the car moves in that part particular direction.

2. Photo of the working android app

(a) Gesture Control



(b) Accelerometer:



10.COMPLETE C PROGRAM:

1.Program for controlling actuators using HC-05 bluetooth module

```
#define horn_Buzz 18    //Horn Buzzer       pin A4 for Arduino Uno
#define ENA_m1 5        // Enable/speed motor Front Right
#define ENB_m1 6        // Enable/speed motor Back Right
#define ENA_m2 10       // Enable/speed motor Front Left
#define ENB_m2 11       // Enable/speed motor Back Left

#define IN_11  2        // L298N #1 in 1 motor Front Right
#define IN_12  3        // L298N #1 in 2 motor Front Right
#define IN_13  4        // L298N #1 in 3 motor Back Right
#define IN_14  7        // L298N #1 in 4 motor Back Right

#define IN_21  8        // L298N #2 in 1 motor Front Left
#define IN_22  9        // L298N #2 in 2 motor Front Left
#define IN_23  12       // L298N #2 in 3 motor Back Left
#define IN_24  13       // L298N #2 in 4 motor Back Left

int command;            //Int to store app command state.
int speedCar = 100;     // 50 - 255.
int speed_Coeff = 4;
boolean lightFront = false;
boolean lightBack = false;
boolean horn = false;

void setup() {
   pinMode(horn_Buzz, OUTPUT);

   pinMode(ENA_m1, OUTPUT);
```

```
    pinMode(ENB_m1, OUTPUT);
    pinMode(ENA_m2, OUTPUT);
    pinMode(ENB_m2, OUTPUT);

    pinMode(IN_11, OUTPUT);
    pinMode(IN_12, OUTPUT);
    pinMode(IN_13, OUTPUT);
    pinMode(IN_14, OUTPUT);

    pinMode(IN_21, OUTPUT);
    pinMode(IN_22, OUTPUT);
    pinMode(IN_23, OUTPUT);
    pinMode(IN_24, OUTPUT);

  Serial.begin(9600);

  }

void goBack(){

      digitalWrite(IN_11, LOW);
      digitalWrite(IN_12, HIGH);
      analogWrite(ENA_m1, speedCar);


      digitalWrite(IN_13, HIGH);
      digitalWrite(IN_14, LOW);
      analogWrite(ENB_m1, speedCar);


      digitalWrite(IN_21, HIGH);
      digitalWrite(IN_22, LOW);
      analogWrite(ENA_m2, speedCar);
```

```
    digitalWrite(IN_23, LOW);
    digitalWrite(IN_24, HIGH);
    analogWrite(ENB_m2, speedCar);

  }

void goAhead(){

    digitalWrite(IN_11, HIGH);
    digitalWrite(IN_12, LOW);
    analogWrite(ENA_m1, speedCar);

    digitalWrite(IN_13, LOW);
    digitalWrite(IN_14, HIGH);
    analogWrite(ENB_m1, speedCar);


    digitalWrite(IN_21, LOW);
    digitalWrite(IN_22, HIGH);
    analogWrite(ENA_m2, speedCar);


    digitalWrite(IN_23, HIGH);
    digitalWrite(IN_24, LOW);
    analogWrite(ENB_m2, speedCar);


  }

void goRight(){
```

```
    digitalWrite(IN_11, LOW);
    digitalWrite(IN_12, HIGH);
    analogWrite(ENA_m1, speedCar);


    digitalWrite(IN_13, HIGH);
    digitalWrite(IN_14, LOW);
    analogWrite(ENB_m1, speedCar);


    digitalWrite(IN_21, LOW);
    digitalWrite(IN_22, HIGH);
    analogWrite(ENA_m2, speedCar);


    digitalWrite(IN_23, HIGH);
    digitalWrite(IN_24, LOW);
    analogWrite(ENB_m2, speedCar);


  }

void goLeft(){

    digitalWrite(IN_11, HIGH);
    digitalWrite(IN_12, LOW);
    analogWrite(ENA_m1, speedCar);


    digitalWrite(IN_13, LOW);
    digitalWrite(IN_14, HIGH);
    analogWrite(ENB_m1, speedCar);
```

```
    digitalWrite(IN_21, HIGH);
    digitalWrite(IN_22, LOW);
    analogWrite(ENA_m2, speedCar);


    digitalWrite(IN_23, LOW);
    digitalWrite(IN_24, HIGH);
    analogWrite(ENB_m2, speedCar);


  }

void goAheadRight(){

    digitalWrite(IN_11, HIGH);
    digitalWrite(IN_12, LOW);
    analogWrite(ENA_m1, speedCar/speed_Coeff);

    digitalWrite(IN_13, LOW);
    digitalWrite(IN_14, HIGH);
    analogWrite(ENB_m1, speedCar/speed_Coeff);


    digitalWrite(IN_21, LOW);
    digitalWrite(IN_22, HIGH);
    analogWrite(ENA_m2, speedCar);


    digitalWrite(IN_23, HIGH);
    digitalWrite(IN_24, LOW);
    analogWrite(ENB_m2, speedCar);
```

```
  }

void goAheadLeft(){

      digitalWrite(IN_11, HIGH);
      digitalWrite(IN_12, LOW);
      analogWrite(ENA_m1, speedCar);

      digitalWrite(IN_13, LOW);
      digitalWrite(IN_14, HIGH);
      analogWrite(ENB_m1, speedCar);


      digitalWrite(IN_21, LOW);
      digitalWrite(IN_22, HIGH);
      analogWrite(ENA_m2, speedCar/speed_Coeff);


      digitalWrite(IN_23, HIGH);
      digitalWrite(IN_24, LOW);
      analogWrite(ENB_m2, speedCar/speed_Coeff);

  }

void goBackRight(){

      digitalWrite(IN_11, LOW);
      digitalWrite(IN_12, HIGH);
      analogWrite(ENA_m1, speedCar/speed_Coeff);


      digitalWrite(IN_13, HIGH);
      digitalWrite(IN_14, LOW);
```

```
    analogWrite(ENB_m1, speedCar/speed_Coeff);

    digitalWrite(IN_21, HIGH);
    digitalWrite(IN_22, LOW);
    analogWrite(ENA_m2, speedCar);

    digitalWrite(IN_23, LOW);
    digitalWrite(IN_24, HIGH);
    analogWrite(ENB_m2, speedCar);

  }

void goBackLeft(){

    digitalWrite(IN_11, LOW);
    digitalWrite(IN_12, HIGH);
    analogWrite(ENA_m1, speedCar);

    digitalWrite(IN_13, HIGH);
    digitalWrite(IN_14, LOW);
    analogWrite(ENB_m1, speedCar);

    digitalWrite(IN_21, HIGH);
    digitalWrite(IN_22, LOW);
    analogWrite(ENA_m2, speedCar/speed_Coeff);


    digitalWrite(IN_23, LOW);
    digitalWrite(IN_24, HIGH);
    analogWrite(ENB_m2, speedCar/speed_Coeff);

  }
```

```
void stopRobot(){

      digitalWrite(IN_11, LOW);
      digitalWrite(IN_12, LOW);
      analogWrite(ENA_m1, speedCar);

      digitalWrite(IN_13, LOW);
      digitalWrite(IN_14, LOW);
      analogWrite(ENB_m1, speedCar);

      digitalWrite(IN_21, LOW);
      digitalWrite(IN_22, LOW);
      analogWrite(ENA_m2, speedCar);

      digitalWrite(IN_23, LOW);
      digitalWrite(IN_24, LOW);
      analogWrite(ENB_m2, speedCar);

  }

void loop(){

    digitalWrite(A1, HIGH);
if (Serial.available() > 0) {
  command = Serial.read();
  stopRobot();

    if (horn) {digitalWrite(horn_Buzz, HIGH);}
    if (!horn) {digitalWrite(horn_Buzz, LOW);}

    switch (command)
    {
        case 'F':goAhead();break;
```

```
        case 'B':goBack();break;
        case 'L':goLeft();break;
        case 'R':goRight();break;
        case 'I':goAheadRight();break;
        case 'G':goAheadLeft();break;
        case 'J':goBackRight();break;
        case 'H':goBackLeft();break;
        case '0':speedCar = 100;break;
        case '1':speedCar = 115;break;
        case '2':speedCar = 130;break;
        case '3':speedCar = 145;break;
        case '4':speedCar = 160;break;
        case '5':speedCar = 175;break;
        case '6':speedCar = 190;break;
        case '7':speedCar = 205;break;
        case '8':speedCar = 220;break;
        case '9':speedCar = 235;break;
        case 'q':speedCar = 255;break;
        case 'V':horn = true;break;
        case 'v':horn = false;break;

        }
    }
}
```

***Size of file :-* 7KB**

2.Program for controlling sensors

```
const int ProxSensor_R=A0;
int inputVal_r = 0;
const int ProxSensor_L=A1;
int inputVal_l = 0;
const int trigPin = 12;
const int echoPin = 11;
// defining variables
long duration;
int distance;

void setup()
{

  pinMode(ProxSensor_R,INPUT);    //Pin 2 is connected to the output of
proximity sensor
  pinMode(ProxSensor_L,INPUT);
  pinMode(5,OUTPUT);
  pinMode(1, OUTPUT);
    pinMode(2, OUTPUT);
  Serial.begin(9600);
   digitalWrite(5, LOW);

  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.begin(9600); // Starts the serial communication
}

void loop()
{
```

```
//---------------ULTRASONIC ------------
  // Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2;
// Prints the distance on the Serial Monitor
Serial.print("Distance measured by Ultrasonic sensor: ");
Serial.println(distance);

//On the front LEDs
digitalWrite(1,HIGH);
digitalWrite(2,HIGH);



//---------------IR  ---------
  inputVal_r = analogRead(ProxSensor_R);
inputVal_l = analogRead(ProxSensor_L);

  Serial.println(inputVal_r);
Serial.println(inputVal_l);

//checking range of distance of both IR and ultrasonic sensors
 if((200<inputVal_r && inputVal_r<300)||(200<inputVal_l &&
inputVal_l<300) || (distance>0 && distance<=15))      //Check the sensor
output
  {
```

```
      digitalWrite(5, HIGH);
    Serial.println("ON");
  }
  else
  {
    digitalWrite(5, LOW);
     Serial.println("OFF");
  }

delay(1000);              // wait for a second

}
```

Size of file :- **2KB**

11.SUMMARY:

We have tried to implement a bluetooth controlled car driven by an application. It has various functionalities facilitated by the application. The motors are connected via L293D motor drivers. Motors are connected to wheels for making the circuit mobile. The ultrasonic sensor is connected on the rear side so as to warn the user when device is in the proximity of obstacle. The IR sensors in the front side ensure the obstacle avoidance when driving forward. There are front LEDS that glow whenever the car is powered. There is a horn that is controlled by the android application, like the direction of the car. There is another buzzer which indicates the proximity of the car to an obstacle i.e. the buzzer will produce sound when the car is within a certain distance near any obstacle. The application has an additional functionality of accelerometer which enables the control of car Other than gesture control. For an instance, when we tilt our mobile in a specific direction, then car moves swiftly in that particular direction.

TIMELINE OF THE PROJECT:-

TASK	FINISH DATE
Selection of project topic and preparing list for components required	07/02/19
Understanding the working of components	13/02/19
Integrating all components and finalizing the circuit design	25/02/19
Updating report 1 to submit report 2	5/03/19
Writing a primary code to dump on the system	10/03/19

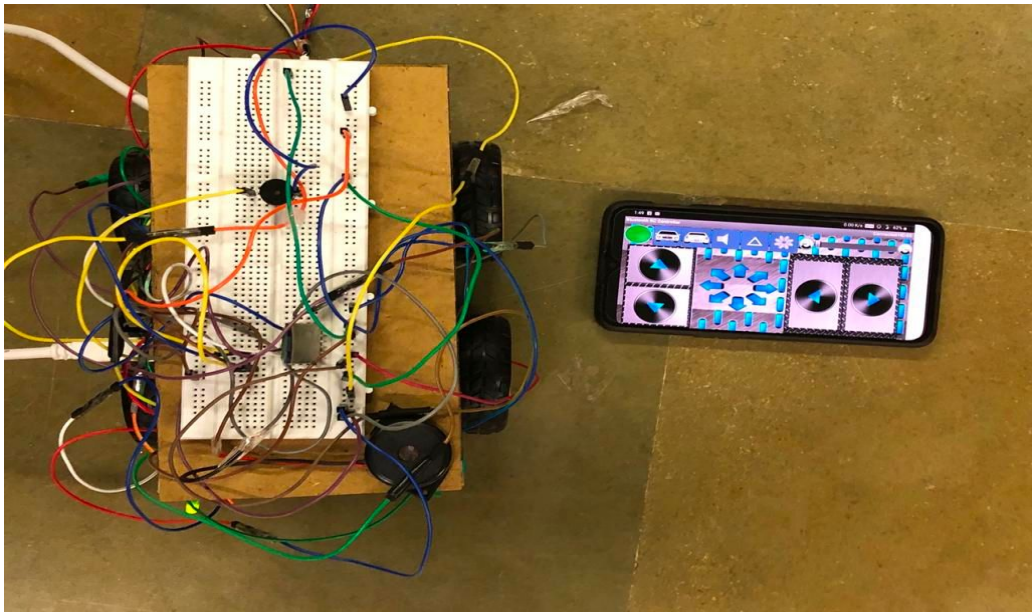
Debugging the code	15/03/19
Submission of report 3	19/03/19
Making android application and integrating the app with Bluetooth module	25/03/19
Integrating the final project	30/03/19
Final presentation	06/04/19

POTENTIAL FUTURE WORK:

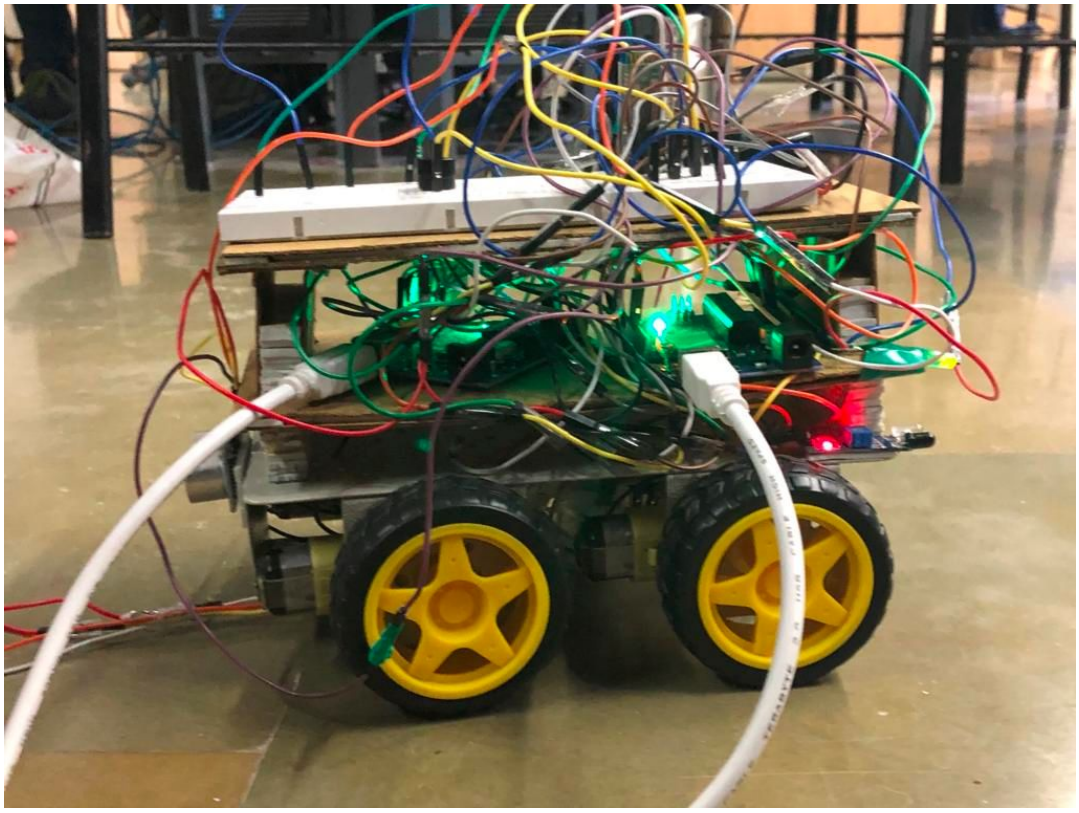
We plan to implement night vision cameras at the rear end and in the front end to avoid obstacles during night. Cameras will help the user in proper parking of car. Headlights can be controlled by using LDR , i.e. we can manage the intensity of light during day and night time. Extra functionality can be of horn. That is to beep the horn when the front distance with obstacle is less than 1m with some running threshold speed. We can also improve our product by implementing caterpillar track or tank tread to make it work on slopes and stairs. We can also improve the quality of material of rubber used in making tyres, such that it can withstand blast impacts like DAKSH (Indian military Robot).

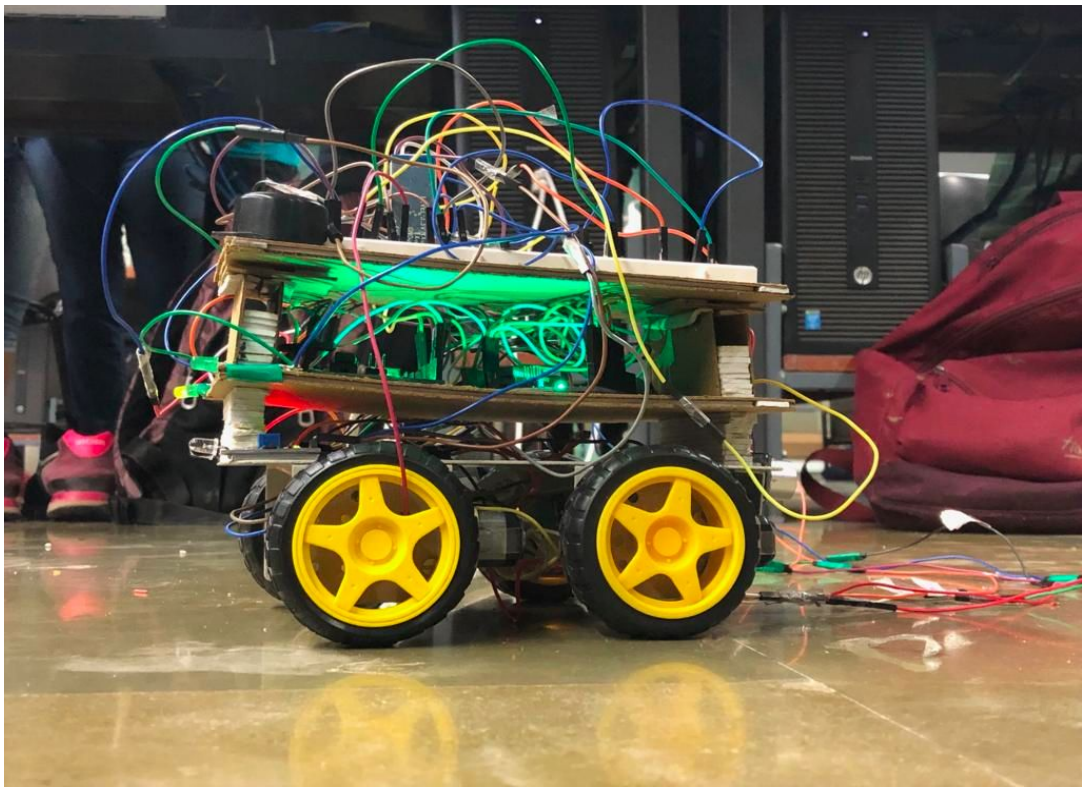
Photos of our working model:-

Top view:

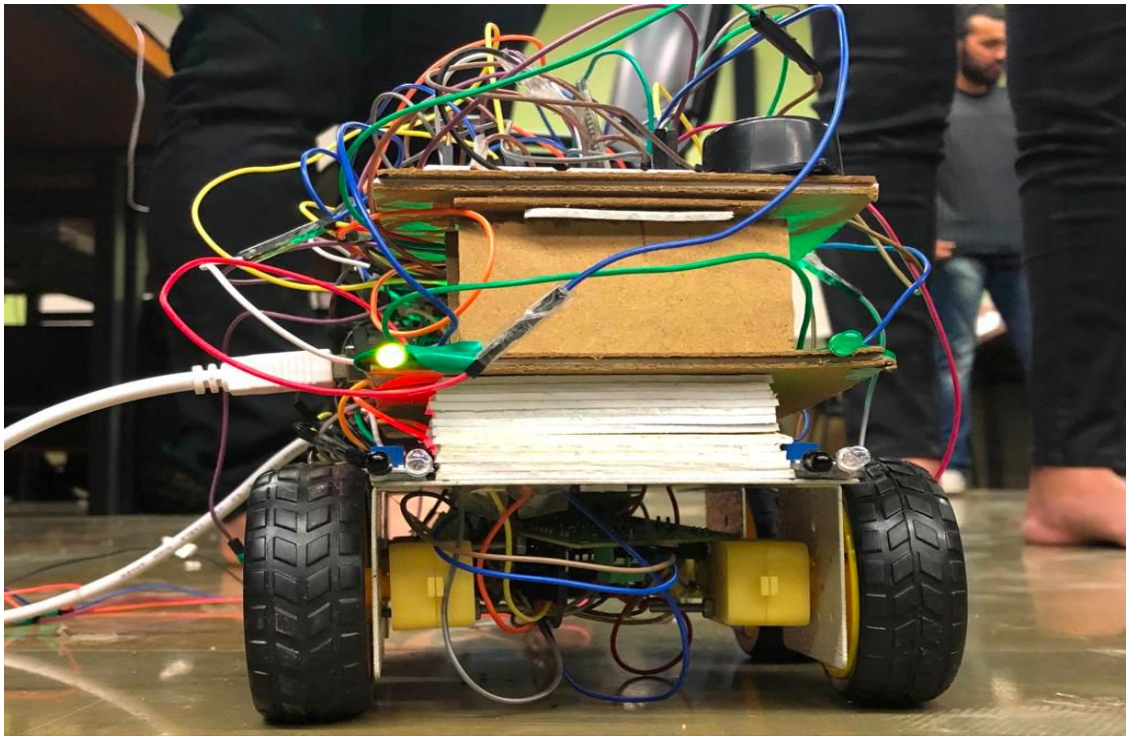


Side view

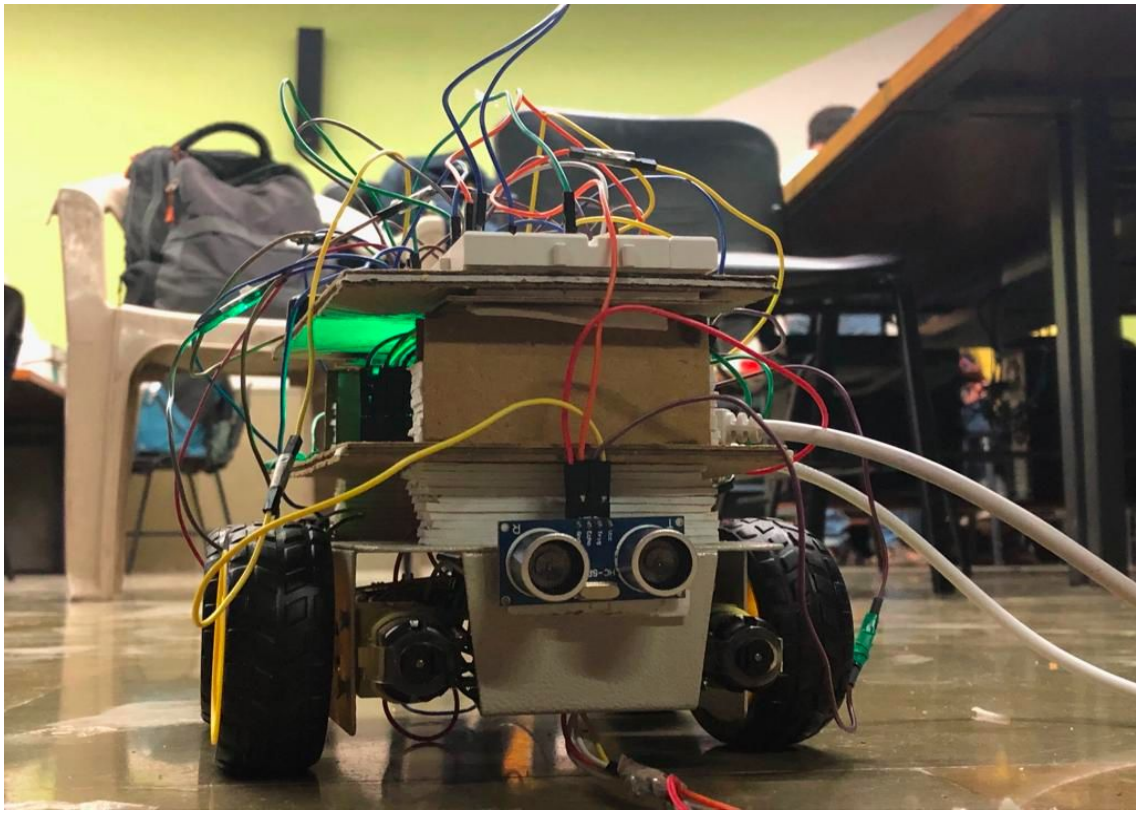




Front



Rear view



Video links of the working model :-

https://drive.google.com/open?id=1mgLRZLt1_rKvtBpzx8KyRiWtEmmRvJjs

12.REFERENCES:

1. **"Android Apps Controlled Arduino Robot Car", *Arduino Project Hub*, 2019. [Online]. Available: https://create.arduino.cc/projecthub/platinum/android-apps-controlled-arduino-robot-car-8c39c4?ref=search&ref_id=android%20controlled%20robot&offset=2. [Accessed: 06- Apr- 2019].**
2. **"Arduino ir proximity sensor interfacing", *Blog.circuits4you.com*, 2019. [Online]. Available: http://blog.circuits4you.com/2016/04/arduino-ir-proximity-sensor-interfacing.html. [Accessed: 06- Apr- 2019].**
3. **"Ultrasonic Sensor HC-SR04 and Arduino Tutorial", *HowToMechatronics*, 2019. [Online]. Available: https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/. [Accessed: 06- Apr- 2019].**
4. **"Bluetooth module hc-05 with Arduino", *YouTube*, 2019. [Online]. Available: https://www.youtube.com/watch?v=DLwzwXC3N2Q. [Accessed: 06- Apr- 2019].**
5. ***Instructables.com*, 2019. [Online]. Available: https://www.instructables.com/id/Bluetooth-Controlled-Robot-Car-Using-Arduino/. [Accessed: 06- Apr- 2019].**
6. **"Phone Controlled Robot Using Arduino | Full Projects with Source Code", *Electronics For You*, 2019. [Online]. Available: https://electronicsforu.com/electronics-projects/hardware-diy/arduino-android-phone-controlled-robot/2. [Accessed: 06- Apr- 2019].**
7. **"Arduino DC Motor Control with L293D Motor Driver IC", *YouTube*, 2019. [Online]. Available: https://www.youtube.com/watch?v=d7oFD-zQpuQ. [Accessed: 06- Apr- 2019].**

APPENDIX A: DATASHEETS

SENSORS:

1) HC-SRO4 Ultrasonic Sensor

Working Voltage	**DC 5 V**
Working Current	**15mA**
Working Frequency	**40Hz**
Max Range	**4m**
Min Range	**2cm**
MeasuringAngle	**15 degree**
Trigger Input Signal	**10uS TTL pulse**
Echo Output Signal	**Input TTL lever signal and the range in proportion**
Dimension	**45*20*15mm**

2) HC-05 Bluetooth Module

Pin	Name	Function
1	Key	The pin state determines whether the module works in AT command mode or normal mode [High=AT commands receiving mode(Commands response mode). Low or NC= Bluetooth module normally working]
2	Vcc	+5V Positive supply needs to be given to this pin for powering the module
3	Gnd	Connect to ground
4	TXD	Serial data is transmitted by module through this pin (at 9600bps by default). 3.3V logic
5	RXD	Serial data is received by module through this pin (at 9600bps by default).3.3V logic
6	State	The pin is connected to the LED on the board to represent the state of the module

3. IR Sensor

1.Descriptions

The Multipurpose Infrared Sensor is an add-on for your line follower robot and obstacle avoiding robot that gives your robot the ability to detect lines or nearby objects. The sensor works by detecting reflected light coming from its own infrared LED. By measuring the amount of reflected infrared light, it can detect light or dark (lines) or even objects directly in front of it. An onboard RED LED is used to indicate the presence of an object or detect line. Sensing range is adjustable with inbuilt variable resistor.

The sensor has a 3-pin header which connects to the microcontroller board or Arduino board via female to female or female to male jumper wires. A mounting hole for easily connect one or more sensor to the front or back of your robot chassis.

2.Features

- 5VDC operating voltage.
- I/O pins are 5V and 3.3V compliant.
- Range: Up to 20cm.
- Adjustable Sensing range.
- Built-in Ambient Light Sensor.
- 20mA supply current.
- Mounting hole.

3.Specifications

- Size: 50 x 20 x 10 mm (L x B x H)
- Hole size: ϕ2.5mm

ACTUATORS:

1) DC Motor

DC motor 6/9V

Item	Specification	Reference
Rated Voltage	6V DC	
No load speed	12000±15%rpm	
No load current	≤280mA	
Operating voltage	1.5-6.5V DC	
Starting Torque	≥250g.cm(according to ourself developed blade)	
starting current	≤5A	
Insulation Resistance	above 10Ω between the case and the terminal	DV 100V
Rotation Direction	CW:[+]terminal connected to the positive power supply,[-]terminal connected to nagative power,clockwise is deemed by the direction of the output shaft	
shaft gap	0.05-0.35mm	

2) Buzzer

Specifications:

Rated Voltage	: 6V DC
Operating Voltage	: 4 to 8V DC
Rated Current*	: ≤30mA
Sound Output at 10cm*	: ≥85dB
Resonant Frequency	: 2300 ±300Hz
Tone	: Continuous
Operating Temperature	: -25°C to +80°C
Storage Temperature	: -30°C to +85°C
Weight	: 2g

*Value applying at rated voltage (DC)

APPENDIX B: PROGRAMMING REVIEW

1) C

One of the most important programming languages in the IoT system is C. This is the lowest layer of software that is close to the hardware. C has been the foundation for many other coding languages over the year. This makes its knowledge of basic necessity for anyone in the IoT projects. The reason behind this is that it doesn't require a lot of processing power. C is available on almost every advanced embedded system platform. C is procedural rather than object-oriented as it does not have built-in capabilities. This programming language is compiled making it great for IoT projects.

2) Java

Java is the well-known programming languages used by the experts. They consider it is the best choice for IoT as it is known for write once, run anywhere. Developers can easily produce and debug code on their computer. It can be transferred it on to any chip using Java Virtual Machine. As a result, it can be run on places where JVMs are used and on any other machine as well.

Java has incorporated coding techniques from the languages such as Mesa, Eiffel, C, and C++. Java has the built-in capabilities making it object-oriented and portable with the least hardware dependency. Along with this, Java has hardware support libraries that can access generic code.

3) Python

Python is mostly used for writing web applications but it has gained popularity in the IoT system. It is an interpreted language that offers readability with syntax without compromising the size. This language has a large number of libraries, it can get more stuff done with fewer codes. Python's clean syntax is suitable for database arrangement. In case your app needs the data to be arranged in a database format or use tables. Python is the right choice available.

APPENDIX C: TROUBLESHOOTING and DEBUGGING

We faced many issues while implementing this project. Some of them are:

1. The number of pins we needed to actualize all the functionalities could not be realized on 1 Arduino board. Thus, we have used 2 boards.
2. Initially, we had connected the front LEDs to the main Arduino with controls to switch it ON using the android app but, the power was insufficient for the LEDs due to the presence of other components which consumed power like the motor drivers, the bluetooth module(HC-05) and the buzzer used for the horn. Thus, we had to shift the LEDs to the other Arduino where we had to compromise with the controls since it was not connected to the bluetooth module(which would be required to implement the controls).
3. A regular DC motor won't be able to provide sufficient torque for the rotation of the wheels, considering the weight it would have to bear and pull. Thus, we used geared motors with wheels.