**Voice Controlled Obstacle Avoiding Robot**

A Project Report

*Submitted by*

**Dhanush Sanapala (CB.****EN.U4AIE21109)**

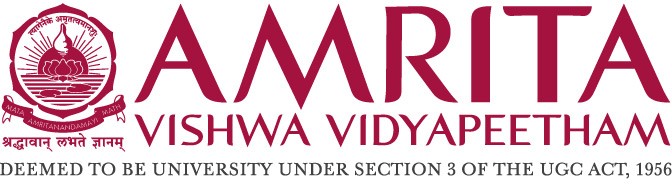
**Chinta Subha Sirkar (CB.****EN.U4AIE21107)**

**Gali Anand Raj (CB.EN.U4AIE21111)**

**Samiksha Kumari (CB.****EN.U4AIE21168)**

*As part of course completion for*

### 21AIE114 - PRINCIPLES OF MEASUREMENTS AND SENSORS



**Centre for Computational Engineering and Networking**

**AMRITA SCHOOL OF ENGINEERING**

## AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641 112 (INDIA)

**July - 2022**

Contents

[Acknowledgement 2](#_Toc108986298)

[List of Figures 2](#_Toc108986299)

[Abstract: 6](#_Toc108986300)

[Introduction: 6](#_Toc108986301)

[Objective 6](#_Toc108986302)

[Hardware and Software Requirements 7](#_Toc108986303)

[Implementation - Model (Optional) or -TINKERCAD circuit 7](#_Toc108986304)

[Theory -Working logic of the project 8](#_Toc108986305)

[Outputs 14](#_Toc108986313)

[Code 14](#_Toc108986314)

[Pros and Cons 18](#_Toc108986315)

[Probable Future Extensions 19](#_Toc108986316)

[Conclusion 19](#_Toc108986317)

[References 20](#_Toc108986318)

## Acknowledgement

We would like to thank our professor Dr. Ambika who gave us her valuable suggestions and ideas when we were in need of them to work on our project “voice-controlled obstacle avoiding robot.”

We are also grateful to our university, Amrita Vishwa Vidyapeetham for giving us the opportunity to work on this project. Last but not the least, we would like to thank our group members, as without their co-operation, we would not have been able to complete the project within the prescribed time.

# List of Figures

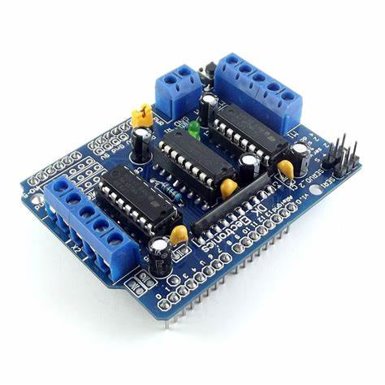
* 1. Arduino UNO



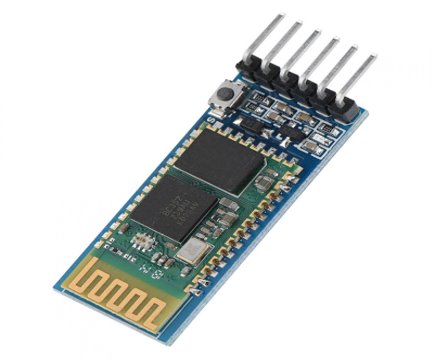
* 1. Ultrasonic Sensor



* 1. L293d Motor Driver Shield



* 1. HC-05 Bluetooth Module



* 1. Servo Motor



* 1. BO motor wheels



**List of Tables**

* 1. **Voice Commands and their de coding:**

|  |  |
| --- | --- |
| Front | ‘^’ |
| Back | ‘-‘ |
| Left | ‘<’ |
| Right | ‘>’ |
| Stop | ‘\*’ |
| Horn | ‘,’ |
| Light | ‘)’ |
| Dim | ‘(‘ |

# Abstract:

The main aim of the robot is to scan the voice input  or joystick input and process it for required operation and to do different tasks over specified area by controlling the robotic vehicle via voice command. The voice control technique consists of an Android app which communicates with the robot via Bluetooth module. Additionally, the robot will also have the proficiency to detect the obstacle, stops and avoid the obstacle by choosing different path with the help of Ultrasonic sensor.

# Introduction:

In this project, we will learn how to make Voice Controlled Robot Car Using Arduino. The robotic car can be controlled wirelessly via voice commands directly from the user. The robot can move forward, backward, left, and right and can also be stopped and can-do commands like light on and off, horn.

The Arduino voice-controlled robot car is interfaced with a Bluetooth module HC-05. We can give specific voice commands to the robot through an Android app installed on the phone. At the receiving side, a Bluetooth transceiver module receives the commands and forwards them to the Arduino and thus the robotic car is controlled.

# Objective

Making the right hardware selection for your project depending on its functionalities.

· Prepare the circuit diagram to connect all the chosen components.

· Assemble all the project parts (mechanical and electronic assembly).

· Finally designing your own ARDUINO-based Voice-controlled Robot.

# Hardware and Software Requirements

Arduino UNO ​

Ultrasonic Sensor ​

L293d Motor Driver Shield ​

HC-05 Bluetooth module ​

Servo motor ​

Foam board ​

IR sensor ​

BO motor and wheels, etc... ​

SOFTWARE REQUIREMENTS:

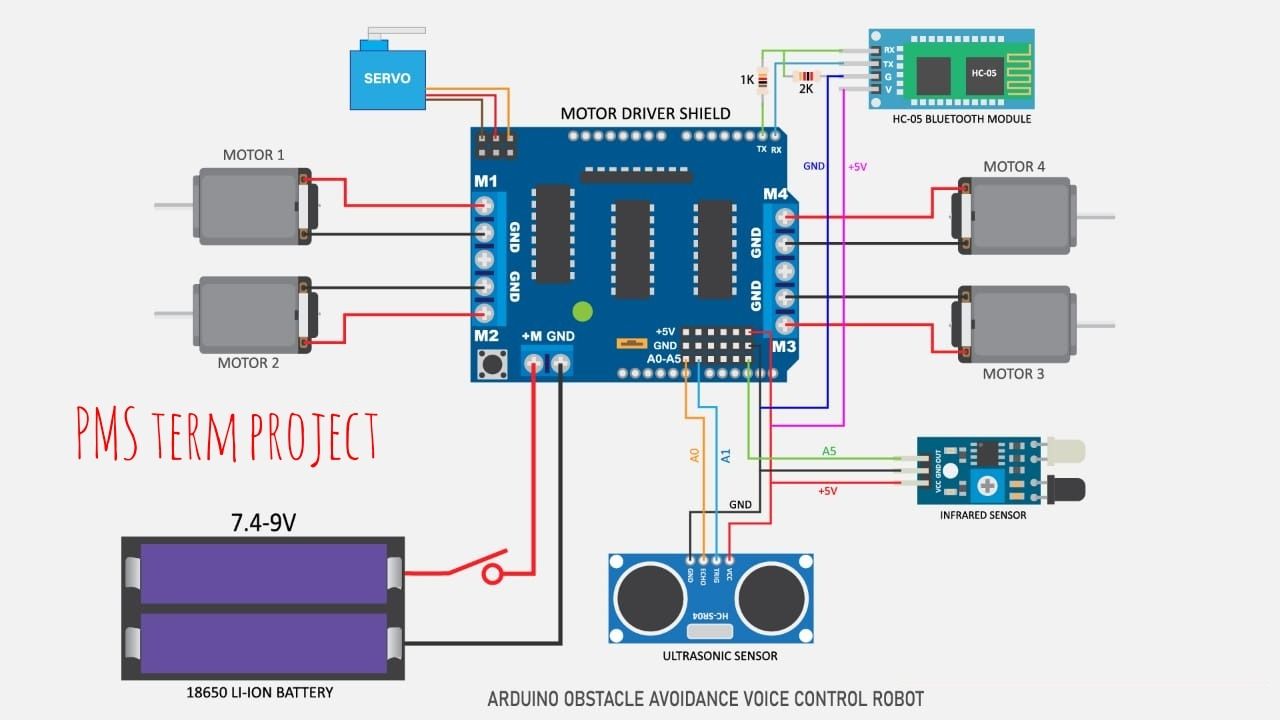
Arduino IDE software

Libraries to install in that:

Adafruit Motor driver shield V2 & V3(for working of motor driver shield)

Servo library is preinstalled in the software.

# Implementation - Model (Optional) or -TINKERCAD circuit



* + 1 extra led and one extra buzzer+led at pin’s 12 and 13 respectively

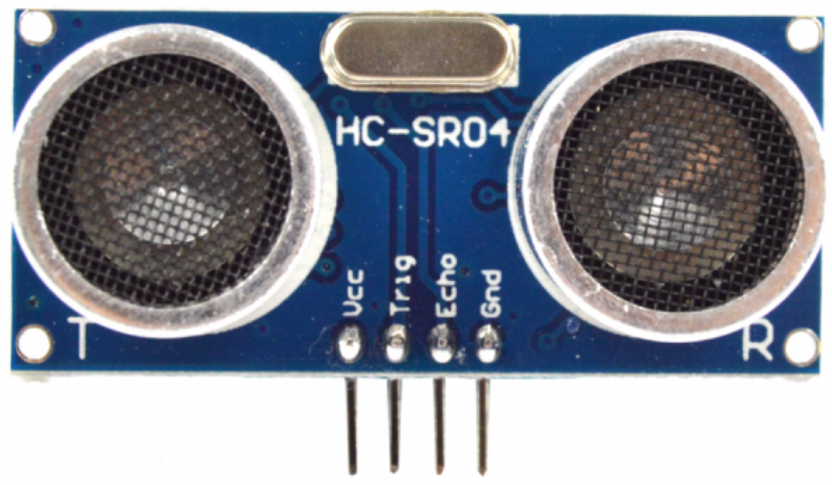
# Theory -Working logic of the project

**ROLE OF ULTRASONIC SENSOR IN OBSTACLE AVOIDING ROBOT**

**WHAT IS ULTRASONIC SENSOR?**

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal.

**ULTRASONIC SENSOR**



**CONNECTIONS OF ULTRASONIC SENSOR:**

1. Vcc - (connected to 5V pin in Arduino)
2. Trig - (Transmitter which is connected to Analog pins in Arduino)
3. Echo - (Receiver which is connected to Analog pins in Arduino)
4. Gnd - (Gnd is ground which is connected to ground pin in Arduino)

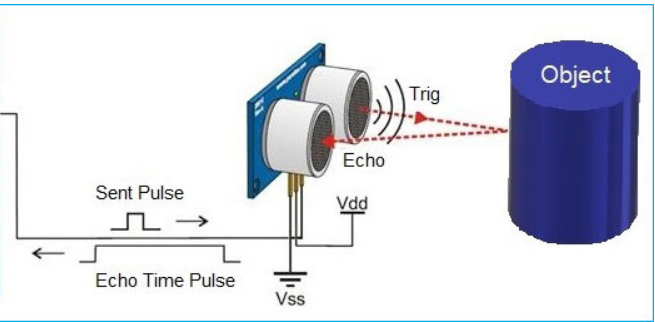
### Working of ultrasonic sensor in Obstacle Avoidance:

### Ultrasonic sensor is used to sense the obstacles in the path by calculating the distance between the robot and the obstacle. If a robot finds any obstacle it changes its direction and continues moving.

### How to build obstacle avoiding robot using Ultrasonic Sensor?

### Before going to build the robot, it is important to understand how the ultrasonic sensor works because this sensor will have an important role in detecting obstacles. The basic principle behind the working of ultrasonic sensor is to note down the time taken by sensor to transmit ultrasonic beams and receive the ultrasonic beams after hitting the surface.

The signal then hits the surface and returns back and captured by the receiver Echo pin of HC-SR04. The Echo pin had already made high at the time sending high.



### 

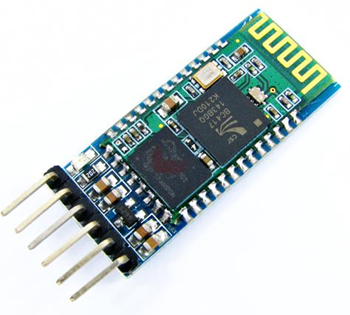
### The distance is calculated using the formula below:

### The time taken by beam to return back is saved in variable and converted to distance using appropriate calculations

### Distance= (Time x Speed of Sound in Air (343 m/s))/2

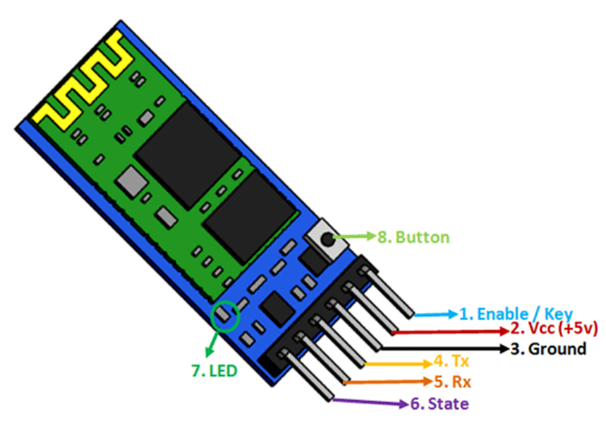
**HC-05 Bluetooth module**

HC-05 is a Bluetooth module which is designed for wireless communication. This module can be used in a master or slave configuration. It has range up to <100m which depends upon transmitter and receiver, atmosphere, geographic & urban conditions. It uses serial communication to communicate with devices. It communicates with microcontroller using serial port (USART).



Bluetooth serial modules allow all serial enabled devices to communicate with each other using Bluetooth.

Pin configuration:



**Enable/key:** This pin is used to toggle between Data Mode (set low) and AT command mode (set high). By default, it is in Data mode.

The two modes:

**Data mode:** Exchange of data between devices.

Command mode: It uses AT commands which are used to change setting of HC-05. To send these commands to module serial (USART) port is used.

**Vcc:** Powers the module. Connect to +5V or +3.3V Supply voltage.

**GND:** Ground Pin of module.

**TXD:** Transmit Serial data. Everything received via Bluetooth will be given out by this pin as serial data.

**RXD:** Receive data serially. Every serial data given to this pin will be broadcasted via Bluetooth.

**State:** The state pin is connected to on board LED, it can be used as feedback to check if Bluetooth is working properly.

**LED:** Indicates the status of Module

**Blink once in 2 sec:** Module has entered Command Mode

**Repeated Blinking:** Waiting for connection in Data Mode

**Blink twice in 1 sec:** Connection successful in Data Mode

**Button:** The command and data mode states are changeable through a button present on the module.

Here, we are not using the state pin and the enable/key pin.

**HC-05 Default Settings:**

Default Bluetooth Name: “HC-05”

Default Password: 1234 or 0000

Default Communication: Slave

Default Mode: Data Mode

Command Mode Baud Rate: 38400, 8, N, 1

**Working of HC-05 Bluetooth module:**

The HC-05 has two operating modes, one is the Data mode in which it can send and receive data from other Bluetooth devices and the other is the AT Command mode where the default device settings can be changed. We can operate the device in either of these two modes by using the key pin as explained in the pin description.

It is very easy to pair the HC-05 module with microcontrollers because it operates using the Serial Port Protocol (SPP). Simply power the module with +5V and connect the Rx pin of the module to the Txd of MCU and Txd pin of module to Rxd of MCU.

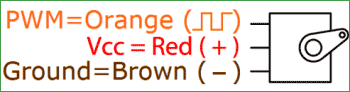
During power up the key pin can be grounded to enter into Command mode, if left free it will by default enter into the data mode. As soon as the module is powered you should be able to discover the Bluetooth device as “HC-05” then connect with it using the default password 1234 and start communicating with it. The name password and other default parameters can be changed.

**Servo Motor**

A servo motor is a rotary actuator that allows for precise control of angular position. It consists of a motor coupled to a sensor for position feedback. It also requires a servo drive to complete the system. The drive uses the feedback sensor to precisely control the rotary position of the motor.



**Wire configuration:**



Brown: Ground wire connected to the ground of system.

Red: Powers the motor typically +5V is used.

Orange: PWM signal is given in through this wire to drive the motor.

**Working of servo motor:**

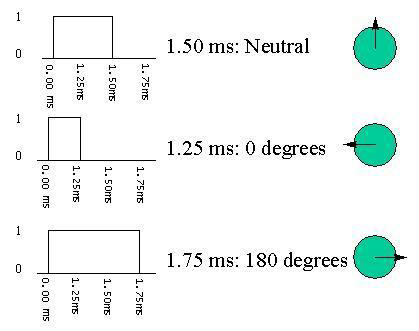
The servo motor has some control circuits and a potentiometer connected to the output shaft. The pot can be seen on the right side of the circuit board allows the control circuitry to monitor the current angle of the servo motor.

If the shaft is at the correct angle, then the motor shuts off. If the circuit finds that the angle is not correct, it will turn the motor until it is at a desired angle. The output shaft of the servo is capable of traveling somewhere around 180 degrees. Usually, it is somewhere in the 210-degree range, however, it varies depending on the manufacturer. A normal servo is used to control an angular motion of 0 to 180 degrees. It is mechanically not capable of turning any farther due to a mechanical stop built on to the main output gear.

The power applied to the motor is proportional to the distance it needs to travel. So, if the shaft needs to turn a large distance, the motor will run at full speed. If it needs to turn only a small amount, the motor will run at a slower speed. This is called proportional control.

**Communication of angle at which it should turn:**

The control wire is used to communicate the angle. The angle is determined by the duration of a pulse that is applied to the control wire. This is called Pulse Coded Modulation. The servo expects to see a pulse every 20 milliseconds (.02 seconds). The length of the pulse will determine how far the motor turns. A 1.5 millisecond pulse, for example, will make the motor turn to the 90-degree position (often called the neutral position). If the pulse is shorter than 1.5 milliseconds, then the motor will turn the shaft closer to 0 degrees. If the pulse is longer than 1.5 milliseconds, the shaft turns closer to 180 degrees.



**Arduino code:**

For this you must have the library servo.h.

Servo - write()

Writes a value to the servo, controlling the shaft accordingly. On a standard servo, this will set the angle of the shaft (in degrees), moving the shaft to that orientation. On a continuous rotation servo, this will set the speed of the servo (with 0 being full-speed in one direction, 180 being full speed in the other, and a value near 90 being no movement).

**Syntax:**

servo.write(angle)

Servo - attach()

Attach the Servo variable to a pin.

Syntax:

servo.attach(pin)   
servo.attach(pin, min, max)

# Outputs





# Code

#include <Servo.h>

#include <AFMotor.h>

#define Echo A0

#define Trig A1

#define motor 10

#define Speed 255

#define spoint 103

#define IR A2

**char** value;

**int** distance;

**int** Left;

**int** Right;

**int** Straight;

**int** L = 0;

**int** R = 0;

**int** F = 0;

**int** L1 = 0;

**int** R1 = 0;

**int** buzzer=13;

**int** led=12;

Servo servo;

AF\_DCMotor M1(1);

AF\_DCMotor M2(2);

AF\_DCMotor M3(3);

AF\_DCMotor M4(4);

**void** setup() {

Serial.begin(9600);

pinMode(Trig, OUTPUT);

pinMode(buzzer, OUTPUT);

pinMode(led, OUTPUT);

pinMode(Echo, INPUT);

pinMode(IR, INPUT);

servo.attach(motor);

M1.setSpeed(Speed);

M2.setSpeed(Speed);

M3.setSpeed(Speed);

M4.setSpeed(Speed);

}

**void** loop() {

Bluetoothcontrol();

//voicecontrol();

**int** irval = digitalRead(IR);

Serial.println(irval);

**if** (irval == 0){

Stop();

digitalWrite(buzzer, HIGH);

}

**if** (irval == 1){

digitalWrite(buzzer, LOW);

}

F = frontsee();

servo.write(spoint);

**if** (F < 10) {

Stop();

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

}

}

**void** Bluetoothcontrol() {

**if** (Serial.available() > 0) {

value = Serial.read();

// Serial.println(value);

**if** (value == 'F') {

forward();

} **else** **if** (value == 'B') {

backward();

} **else** **if** (value == 'L') {

left();

} **else** **if** (value == 'R') {

right();

} **else** **if** (value == 'S') {

Stop();

}

**else** **if** (value == 'W') {

digitalWrite(led, HIGH);

delay(1000);

digitalWrite(led, LOW);

}

**else** **if** (value == 'V') {

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

}

}

}

**void** voicecontrol() {

**if** (Serial.available() > 0) {

value = Serial.read();

// Serial.println(value);

**if** (value == '^') {

F = frontsee();

servo.write(spoint);

**if** (F >= 10 ) {

forward();

} **else** **if** (F < 10) {

Stop();

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

}

} **else** **if** (value == '-') {

**int** irval = digitalRead(IR);

**if** (irval == 0){

Stop();

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

} **else** {

backward();

}

} **else** **if** (value == '<') {

L = leftsee();

servo.write(spoint);

**if** (L >= 10 ) {

left();

delay(2500);

Stop();

} **else** **if** (L < 10) {

Stop();

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

}

} **else** **if** (value == '>') {

R = rightsee();

servo.write(spoint);

**if** (R >= 10 ) {

right();

delay(2500);

Stop();

} **else** **if** (R < 10) {

Stop();

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

}

} **else** **if** (value == '\*') {

Stop();

} **else** **if** (value == ',') {

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

}**else** **if** (value == ')') {

digitalWrite(led, HIGH);

}**else** **if** (value == '(') {

digitalWrite(led, LOW);

}

}

}

**int** ultrasonic() {

digitalWrite(Trig, LOW);

delayMicroseconds(4);

digitalWrite(Trig, HIGH);

delayMicroseconds(10);

digitalWrite(Trig, LOW);

**long** t = pulseIn(Echo, HIGH);

**long** cm = t / 29 / 2; //time convert distance

**return** cm;

}

**void** forward() {

M1.run(FORWARD);

M2.run(FORWARD);

M3.run(FORWARD);

M4.run(FORWARD);

}

**void** backward() {

M1.run(BACKWARD);

M2.run(BACKWARD);

M3.run(BACKWARD);

M4.run(BACKWARD);

}

**void** right() {

M1.run(FORWARD);

M2.run(FORWARD);

M3.run(BACKWARD);

M4.run(BACKWARD);

}

**void** left() {

M1.run(BACKWARD);

M2.run(BACKWARD);

M3.run(FORWARD);

M4.run(FORWARD);

}

**void** Stop() {

M1.run(RELEASE);

M2.run(RELEASE);

M3.run(RELEASE);

M4.run(RELEASE);

}

**int** rightsee() {

servo.write(20);

delay(200);

Left = ultrasonic();

**return** Left;

}

**int** leftsee() {

servo.write(180);

delay(200);

Right = ultrasonic();

**return** Right;

}

**int** frontsee() {

Straight = ultrasonic();

**return** Straight;

}

# Pros and Cons

Over the years as technology advanced and grew at a tremendous pace, so did the horizon of ideas and creation in our mind. If we look a few years back, a teenager like me would dare think about creating a robot that would operate listening to their voice and follow the given commands. All they could do was dream about making such a fancy robot and let the idea pass from their mind. But with the help of today's technology, those great ideas are no more left being dreams but are coming true. Having not to deal anymore with complex circuits, Arduino gives us a very simple approach into building projects.

By our project we can accomplish a carrier transport system which works on the voice control and informs us if there is any obstacle and then we can change the direction according to the output.

Any industry needs this kind of system for less worker presence and it is very much important in these covid conditions by which we can-do hands-free transport securely. Which can be implemented in restaurants.

# Probable Future Extensions

* By attaching a bomb detector to the robot, we can use our personal computer to send it anywhere—to a battle field, a forest, a coal mine. The sensor can detect the bomb and send information.
* Instead of sending a human to a risky area and having them deal with difficulties there, we can send a robot there instead, and the sensor will identify the temperature, due to a temperature sensor connected to the robot.
* We can obtain information about the concentration of gases or smoke by connecting a smoke sensor to the robot in coal mines, dangerous zones, etc.
* It can be used for home security and military purposes where commands can be given to robots without involving human risk by increasing the range and by installing cameras.

# Conclusion

Our project voice-controlled robot can be used in various applications such as **Applications:**

* Basic applications of robots utilizing voice recognition are to support people with disability, executing present commands etc.
* In agriculture for farming purposes by connecting the appropriate instruments to it. The robot can move right based on our commands, allowing us to perform some agricultural tasks entirely from a computer.
* Indoor assistive robots which will navigate around to pick up objects from one place and place them at another using speech commands technique.
* Surveillance applications to send live feed from camera and track down an object.
* Important application in Industrial Robots where user can control the robotic vehicle via voice commands.
* On board digital assistants for automobiles.

Hence, we can conclude that there are vast applications of voice-controlled robot.

# References

1. Arduino project Hub
2. Arduino cook Book

# Contributions

*DHANUSH SANAPALA [CB.EN.U4AIE21109]*

&

*CHINTA SUBHA SRIKAR [CB.EN. U4AIE21107]* - Circuit connections , mobile synchronization and coding

*ANAND RAJ GALI [CB.EN.U4AIE21111]* - ultrasonic sensor analysis and servo controlling

*SUDIKSHA KUMARI [CB.EN.U4AIE21168]* - Bluetooth module analysis and connection via app.