WALTER - A VOICE-CONTROLLED ROBOTIC BLUETOOTH CAR

Alabhya Sharma
20BRS1150
alabhyasharma1273@gmail.com

Pulak Jain
20BRS1126
pulak.jain2020@vit.student.ac.in

CSE 1022 - J component
Faculty : Dr Ganala Santoshi
Slot : G1 & G2

Index

Abstract

Walter, a voice-activated robot that can receive orders over wireless channels such as Bluetooth. Walter can read and execute those orders while recognizing and avoiding impediments. It uses an Arduino microcontroller in conjunction with a motor driver shield used to govern its movement. Walter features an integrated sensor system that gives the microcontroller feedback.

Introduction

Walter is a mobile robot that can identify a path that is clear of obstacles. It is a multi-purpose robot that can do many tasks at the same time by coordinating amongst them based on the situation. It operates using vocal human input that is translated by an application, converted to text, and delivered over a Bluetooth channel to be received by the robot with the help of a mix of advanced pathfinding algorithms and sensor data to interact with their surroundings and travel safely. Because of its compact size, it can go through locations that are physically impossible for humans to reach. Many features can be added or deleted depending on the necessity due to its flexible nature in terms of hardware integration and easily updatable memory.

Existing System

Various pathfinding and obstacle detection algorithms have previously been developed as a result of several breakthroughs in the disciplines of artificial intelligence, neural networks, and augmented reality, amongst others, to make such systems highly sophisticated. Integration of Wi-Fi technology into such systems has even allowed them to have an internet connection, allowing a user to operate them from anywhere in the globe as long as they are connected to the internet.

Problem Statement

Some of the more advanced versions now available have a significant maintenance cost, are extremely vulnerable to physical harm, and rely on high-end architecture. Having a global connection is of little utility for such a system, as it only adds to the initial cost without offering anything in return. One big issue is that pre-made models like this are frequently developed for marketing purposes and cannot be altered to meet our needs. These copyrighted items aren't usually open-sourced, so we don't know what's going on and can't make changes to meet our needs. Understanding the principles allows us to not just interpret how they operate, but also to build our own.

Introduction to proposed Model

A versatile compact robot based on Arduino that can perform smart functions such as pathfinding, obstacle detection, obstacle avoidance, edge mapping, signal following, communication, and so on efficiently and at a low cost by combining sensor applications such as ultrasonic sensors, IR sensors, and signalling chips such as Bluetooth modules and RFID chips with advanced algorithms and optimized codes that do not require a lot of memory.

Block Diagram

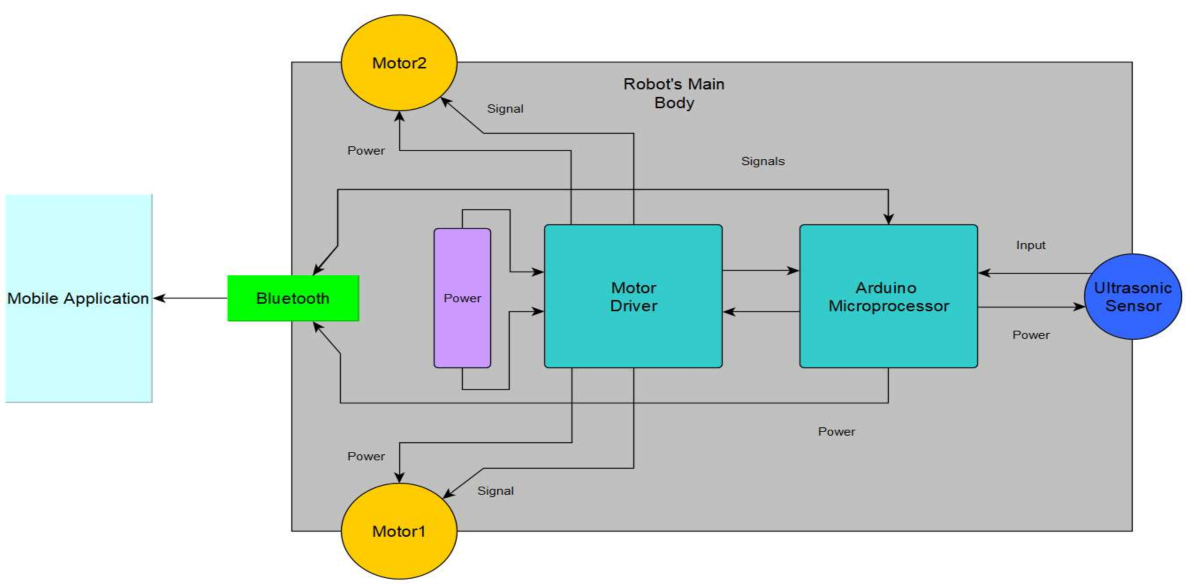

Figure 1

When the Arduino microprocessor detects a command from the mobile application via Bluetooth, the command is forwarded to the Arduino microprocessor's memory for processing. Its internal system translates the instruction to an appropriate set of digits and variables, whilst detecting the signal using an ultrasonic sensor at the same time. This signal is translated to a distance and verified on a regular basis to ensure that no obstacles are approaching. If no barrier is identified, the motion signal is transmitted to the motor driver, who delivers the required power to the associated motor to move in the intended motion; otherwise, it alters its course and goes in the opposite direction.
The battery provides electricity to the entire system.

Components of the model

- ❖ Arduino UNO Microcontroller
- ❖ Ultrasonic sensors
- ❖ Bluetooth Module
- ❖ DC Motors
- ❖ Battery
- ❖ Motor Driver shield
- ❖ Tyres
- ❖ Jumper Cables
- ❖ Arduino Cable

Description of the components (HW / SW / Tech. Specifications)

❖ Arduino UNO 3 Microcontroller :

Arduino UNO is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analogue inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

OPERATING VOLTAGE	5V
INPUT VOLTAGE (RECOMMENDED)	7-12V
INPUT VOLTAGE (LIMIT)	6-20V
DIGITAL I/O PINS	14 (of which 6 provide PWM output)
PWM DIGITAL I/O PINS	6
ANALOG INPUT PINS	6
DC CURRENT PER I/O PIN	20 mA
DC CURRENT FOR 3.3V PIN	50 mA
FLASH MEMORY	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)

Figure 2

❖ Ultrasonic sensor :

An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity. High-frequency sound waves reflect from boundaries to produce distinct echo patterns.

	Grove – Ultrasonic Distance Sensor	HC-SR04
Working Voltage	3.3V / 5V compatible Wide voltage level: 3.2V – 5.2V	5V
Measurement Range	3cm – 350cm	2cm – 400cm
I/O Pins needed	3	4
Operating Current	8mA	15mA

Figure 3

❖ Bluetooth module

Wireless Bluetooth RF Transceiver Module HC-05 RS232 Master Slave for Arduino Feature Bluetooth Serial Transceiver Module with backboard. Works with any USB Bluetooth adapters. This module includes a key interface and state interface compared with Smart Bluetooth Module Baseboard. The Bluetooth Module Baseboard can be compatible with master mode, slave mode and both master-slave mode. Works for Bluetooth TTL transceiver module, which allows your target device to both send or receive the TTL data. The key interface on the baseboard is the master mode button and can be controlled by a high level from external MCU, then this module will search again automatically. This module power supply input is 4.5~6V Use the CSR mainstream Bluetooth chip, Bluetooth V2.0 protocol standards. Module working voltage 3.3 V. Default rate of 9600, the user can be set up. Working current: matching for 30 MA, matching the communication for 8 MA. Dormancy current: no dormancy.

Data Link Protocol	USB , Bluetooth
Included Components	1 xcluma Hc-05 Wireless Bluetooth Rf Transceiver Module Serial/Ttl/Rs232 Arduino
Item Weight	4.00 grams
Manufacturer Series Number	BE-000068
Model Number	BE-000068

Figure 4

❖ Motor Driver Shield

L293D shield is a driver board based on L293 IC, which can drive 4 DC motors and 2 stepper or Servo motors at the same time. Each channel of this module has a maximum current of 1.2A and doesn't work if the voltage is more than 25v or less than 4.5v. So be careful with choosing the proper motor according to its nominal voltage and current. For more features of this shield, let's mention compatibility with Arduino UNO and MEGA, electromagnetic and thermal protection of motor and disconnecting circuit in case of unconventional voltage raise.

Wide Supply-Voltage Range: 4.5 V to 36 V
Separate Input-Logic Supply
Internal ESD Protection
High-Noise-Immunity Inputs
Output Current 1 A Per Channel (600 mA for L293D)
Peak Output Current 2 A Per Channel (1.2 A for L293D)
Output Clamp Diodes for Inductive Transient Suppression (L293D)

Figure 5

❖ DC Motor

A DC motor is the most common type of engine that can be used for many applications. We can see it in remote control cars, robots, etc. This motor has a simple structure. It will start rolling by applying proper voltage to its ends and change its direction by switching voltage polarity. The DC motor's speed is directly controlled by the applied voltage. When The voltage level is less than the maximum tolerable voltage, the speed would decrease.

❖ Battery

A nine-volt battery, either disposable or rechargeable, is usually used in smoke alarms, smoke detectors, walkie-talkies, transistor radios, test and instrumentation devices, medical batteries, LCDs, and other small portable appliances.

Model	9V
Model Name	9V
Product Dimensions	1 x 3.5 x 8.9 cm; 100 Grams
Batteries	1 Lithium ion batteries required. (included)
Item model number	9V
Compatible Devices	Remote, Smoke Alarm, Clock, Toy, Torch
Mounting Hardware	Battery
Number Of Items	2
Voltage	9 Volts
Batteries Included	Yes
Batteries Required	Yes
Battery Cell Composition	Alkaline
Has Self Timer	No

Figure 6

Working Mechanism

The AMR_Voice (Android Meets Robot Voice) application on the smartphone is first connected through Bluetooth. The Arduino is then given an input instruction, such as "Forward," through the mobile application. This application translates oral speech into text, which is specifically tailored to the Arduino commands that have been pre-programmed. If the orders are all correct, the car will continue ahead, scanning the path for any obstacles with the obstacle detecting system. If an impediment is encountered on the path, the obstruction is avoided using the obstacle avoidance method, and the journey proceeds normally.

Flow Diagram / Flow Chart

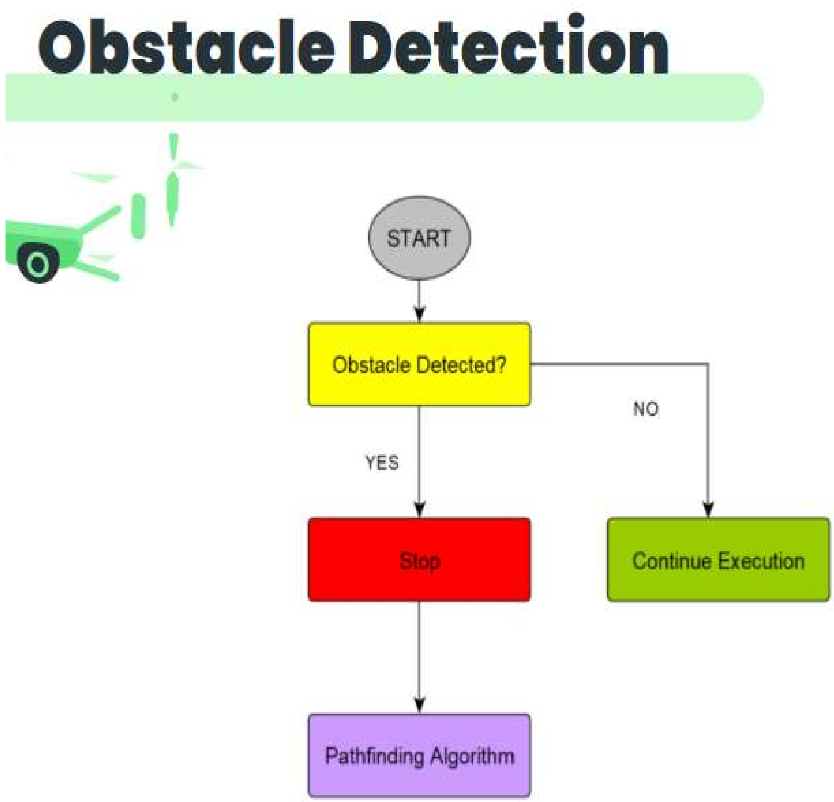

Figure 7

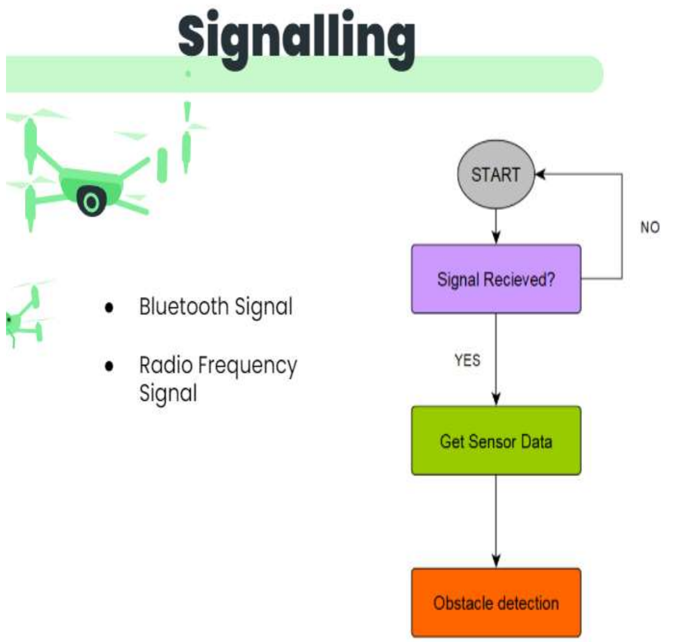

Figure 8

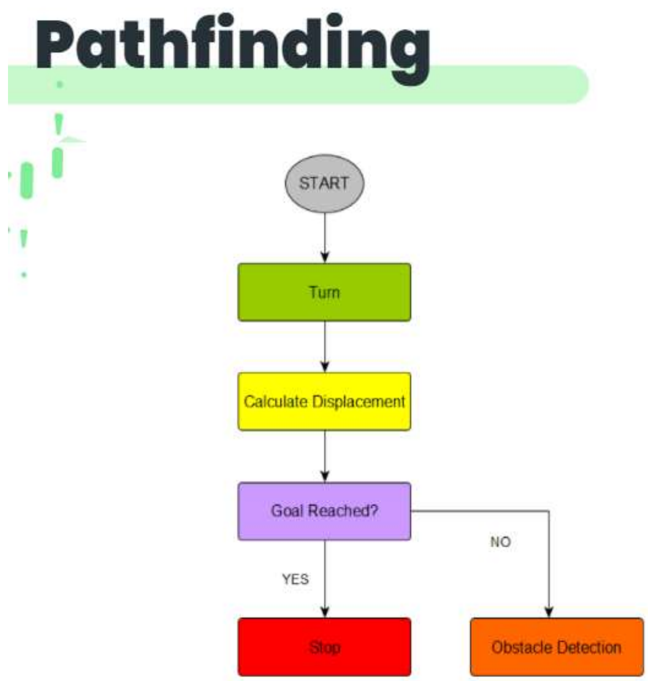

Figure 8

Code

```
#include <AFMotor.h>
#include <SoftwareSerial.h>
#define echoPin 7
#define trigPin 6

AF_DCMotor motor1(1, MOTOR12_1KHZ);
AF_DCMotor motor2(2, MOTOR12_1KHZ);

SoftwareSerial bt(0, 1);
long duration;
int distance;
String command;

void setup()
{
  pinMode(echoPin,INPUT);
  pinMode(trigPin,OUTPUT);
  bt.begin(9600);
  Serial.begin(9600);
}

void Stop()
{
  motor1.run(RELEASE);
  motor2.run(RELEASE);
}
int dist()                              //Figure 7: Obstacle Detection
{
  digitalWrite(trigPin,LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin,LOW);
  duration = pulseIn(echoPin,HIGH);
  distance = (duration*0.0343)/2;
  Serial.println(distance);
  delay(250);
  return distance;
}
```

```
void avoid()              //Figure 9: Pathfinding
{
  dist();
  while(dist() < 10)
  {
  Serial.println("Obstacle detected\n");
  Stop();
  motor1.setSpeed(255);
  motor1.run(BACKWARD);
  motor2.setSpeed(255);
  motor2.run(FORWARD);
  delay(500);
  }
  Stop();
}

void forward()
{
  void avoid();
  motor1.setSpeed(255);
  motor1.run(FORWARD);
  motor2.setSpeed(255);
  motor2.run(BACKWARD);
  delay(800);
  Stop();
}

void back()
{
  void avoid();
  motor1.setSpeed(255);
  motor1.run(BACKWARD);
  motor2.setSpeed(255);
  motor2.run(FORWARD);
  delay(800);
  Stop();
}

void right()
```

```
{
  void avoid();
  motor1.setSpeed(255);
  motor1.run(BACKWARD);
  motor2.setSpeed(255);
  motor2.run(BACKWARD);
  delay(400);
  Stop();
}

void left()
{
  void avoid();
  motor1.setSpeed(255);
  motor1.run(FORWARD);
  motor2.setSpeed(255);
  motor2.run(FORWARD);
  delay(400);
  Stop();
}

void loop()
{
  dist();
  delay(10);
  while(bt.available())                    //Figure 8: Signalling
  {
  command = "";
  command = bt.readString();
  avoid();
  if (command.length() > 0)
  {
    Serial.print(command);
    if(command == "*move forward#" || command == "*forward#")
    {
      forward();
    }
    else if(command == "*move backward#" || command == "*backward#" || command ==
"*move back#" || command == "*back#")
    {
```

```
      back();
    }
    else if(command == "*turn left#" || command == "*move left#" || command == "*left#")
    {
      left();
    }
    else if(command == "*turn right#" || command == "*move right#" || command ==
"*right#")
    {
      right();
    }
    else if(command == "*stop#" || command == "*halt#" || command == "*pause#")
    {
      Stop();
    }
    command = "";
  }
  }
}
```

Output Screenshots

Serial Monitor Output

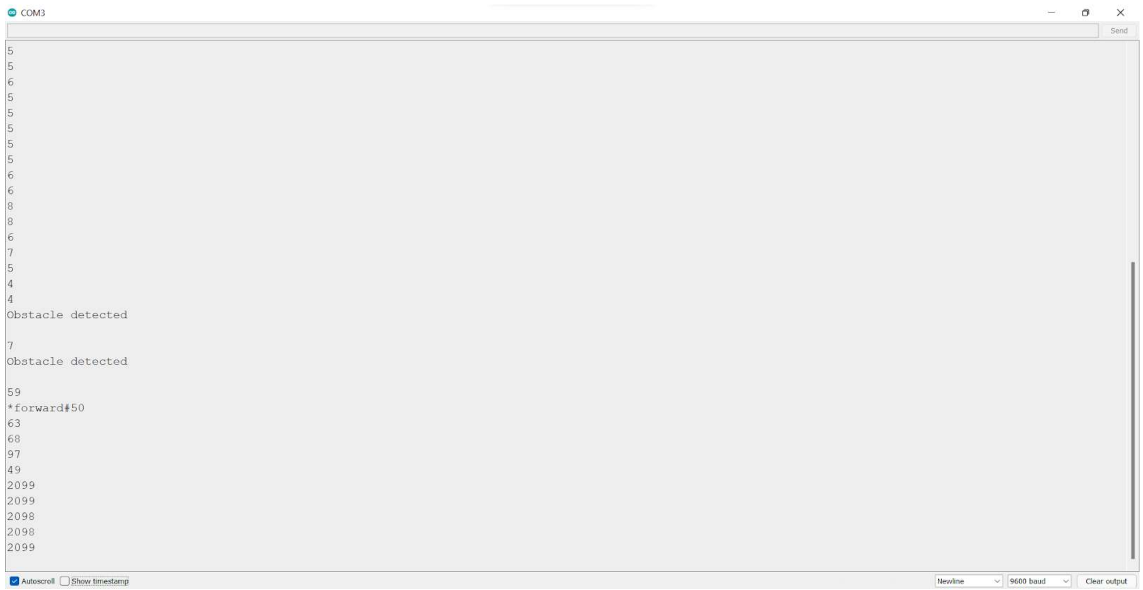

Figure 10

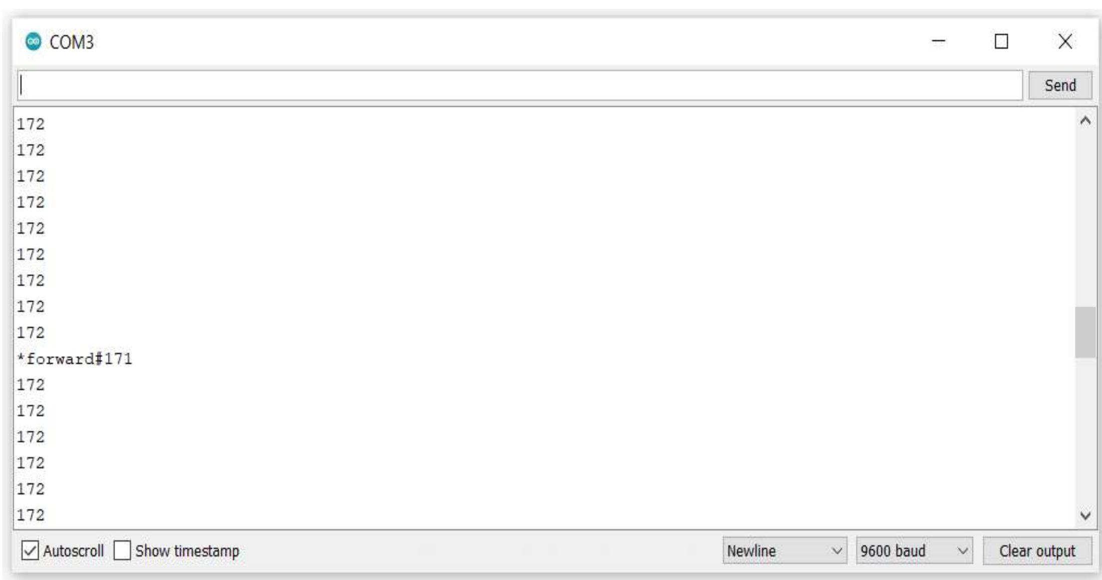

Figure 11

Robot Model

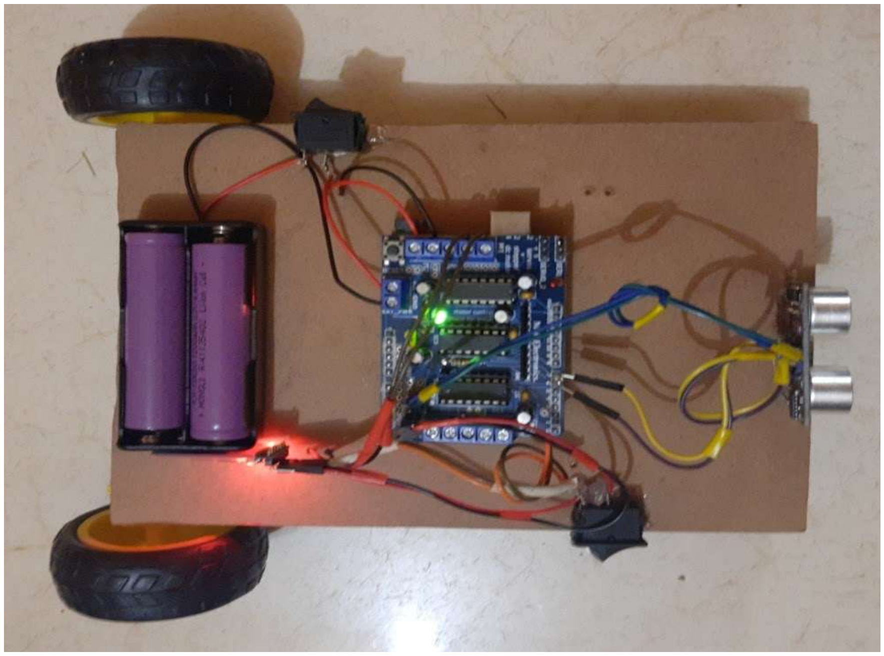

Figure 12

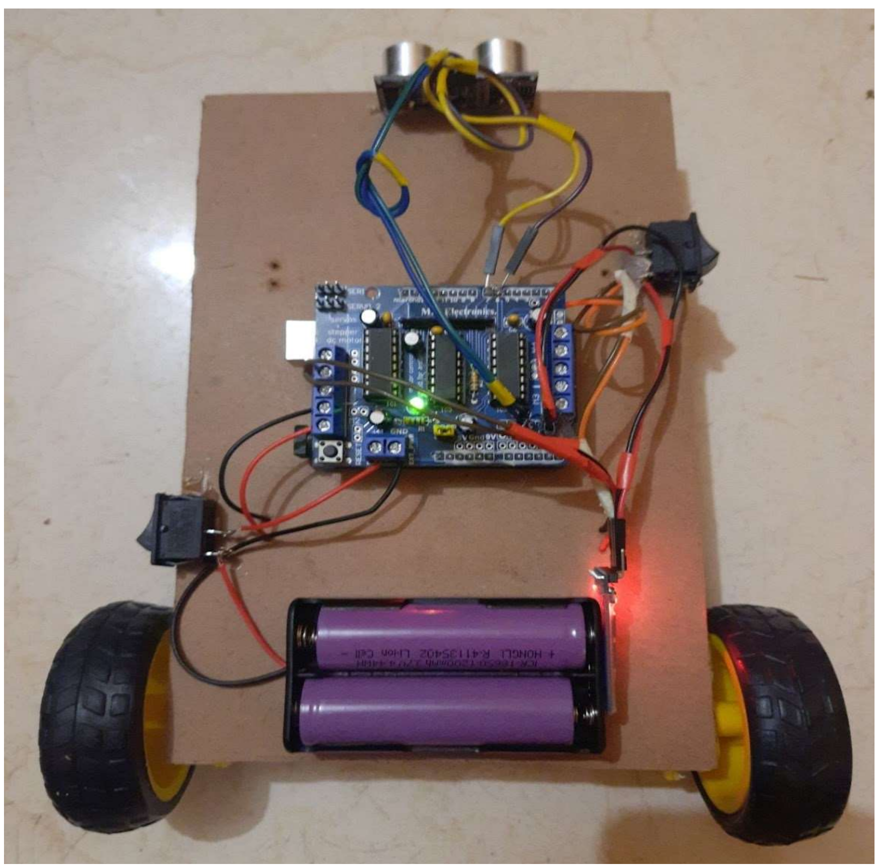

Figure 13

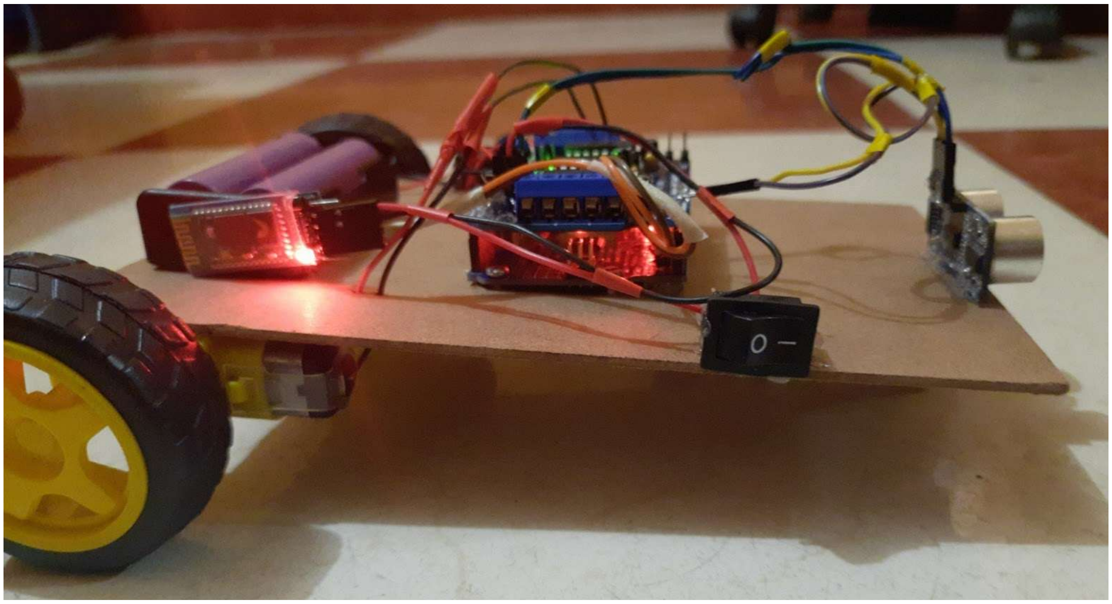

Figure 14

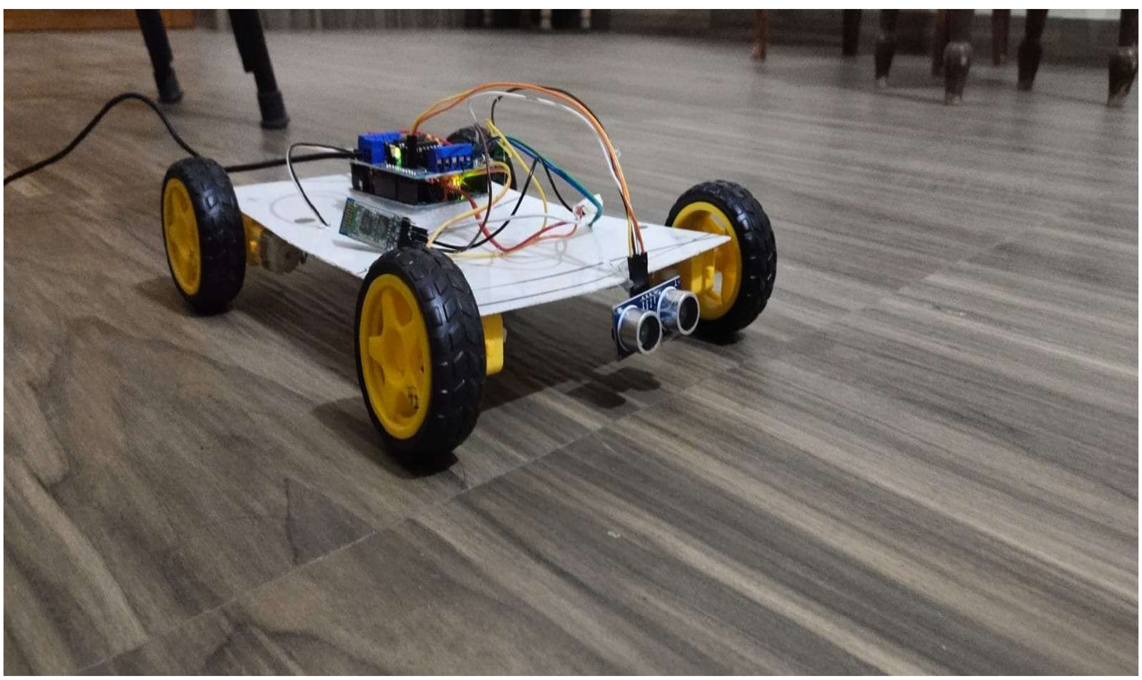

Figure 15

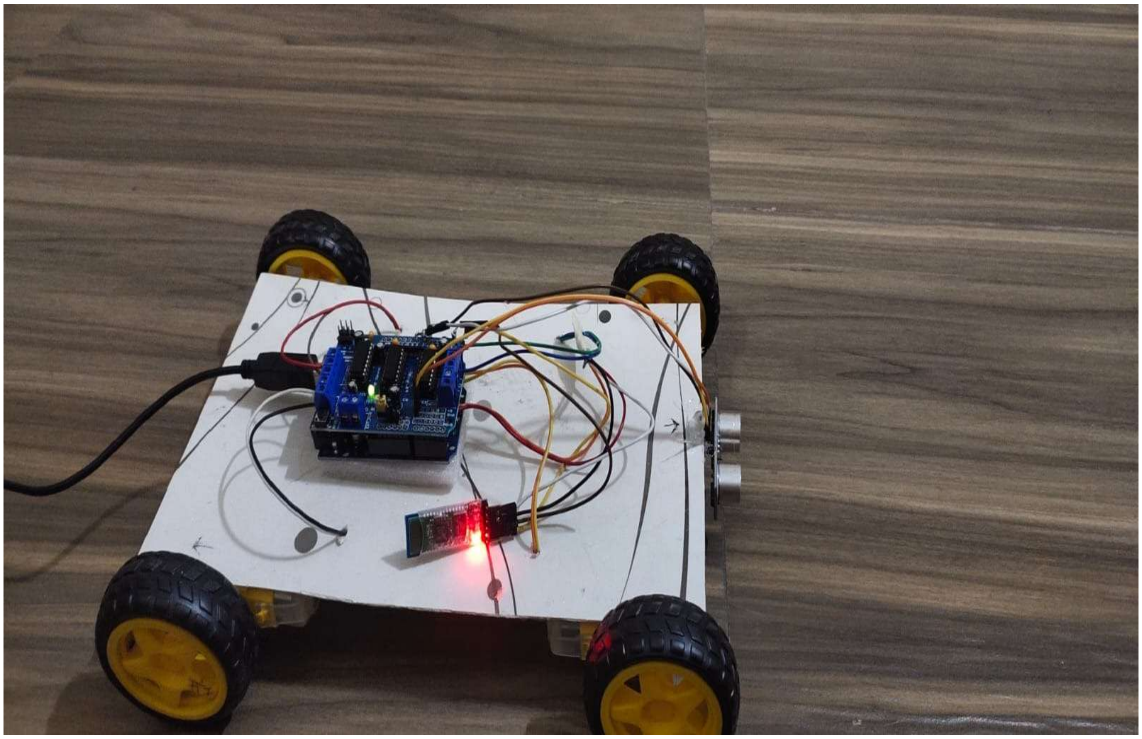

Figure 16

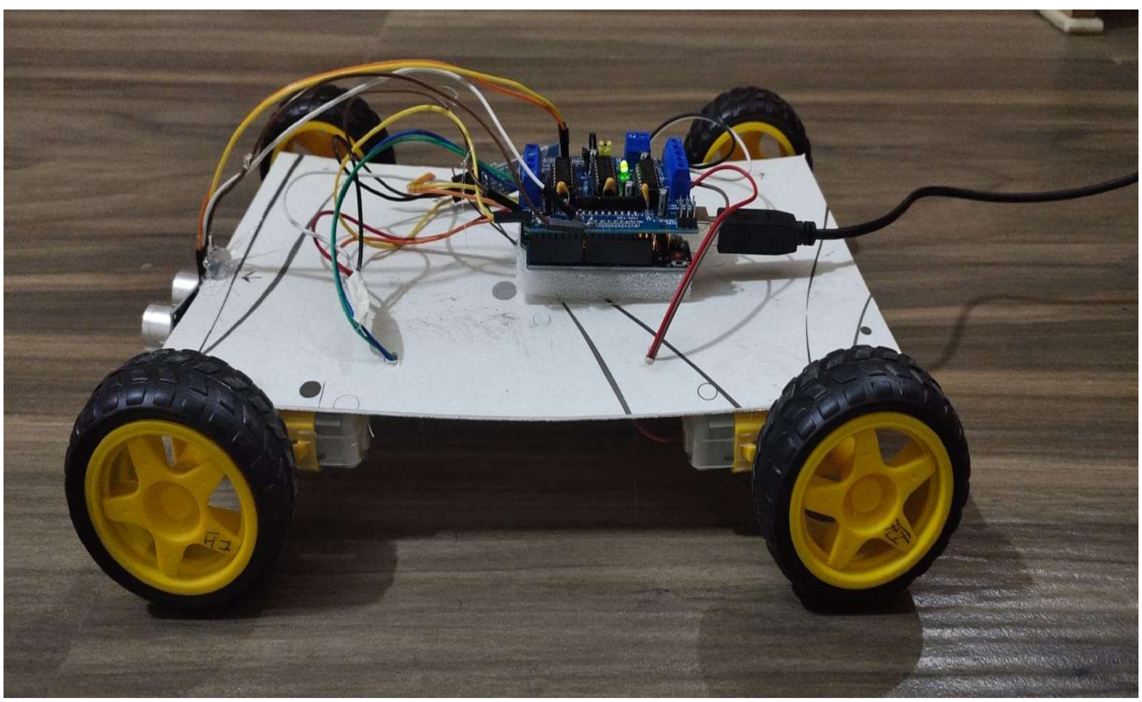

Figure 17

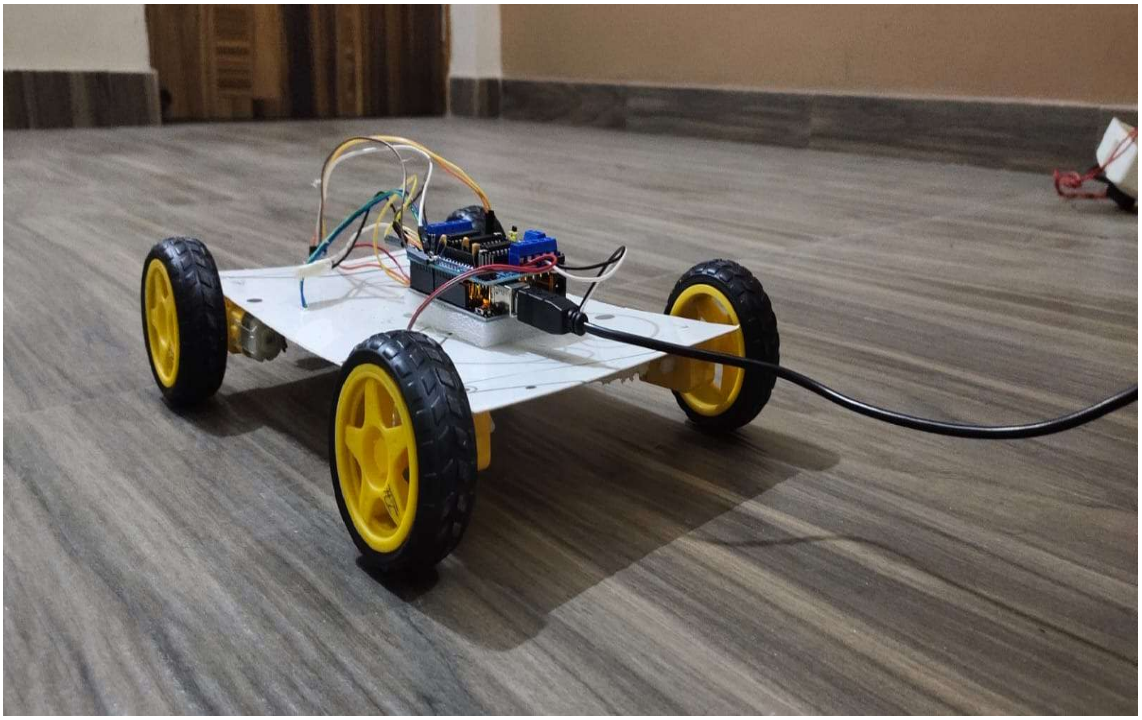

Figure 18

Conclusion

- Using Bluetooth, a fully working voice-controlled robot has been created.
- Capable of movement with the assistance of an Arduino and a motor driver shield.
- Capable of understanding and interpreting vocal inputs.
- Skilled in detecting and avoiding obstacles.
- Able to find a path to the given objectives.
- Navigation to the specified goal.

References

1. http//:ahg:sensors-21-03286.pdf
2. https://www.sciencedirect.com/science/article/pii/S1474667016437535
3. https://create.arduino.cc/projecthub/Alfa0420/obstacle-avoiding-robot-using-arduino-and-ultrasonic-sensor-c179bb
4. https://vitacin-my.sharepoint.com/:p:/g/personal/alabhya_sharma2020_vitstudent_ac_in/EX9wYTAv7OJKscfWEF7kRi8BJP0uMds0RI2KXOORP-Zevg?e=ASAxsn
5. https://vitacin-my.sharepoint.com/:p:/g/personal/alabhya_sharma2020_vitstudent_ac_in/EX9wYTAv7OJKscfWEF7kRi8BJP0uMds0RI2KXOORP-Zevg?e=ASAxsn