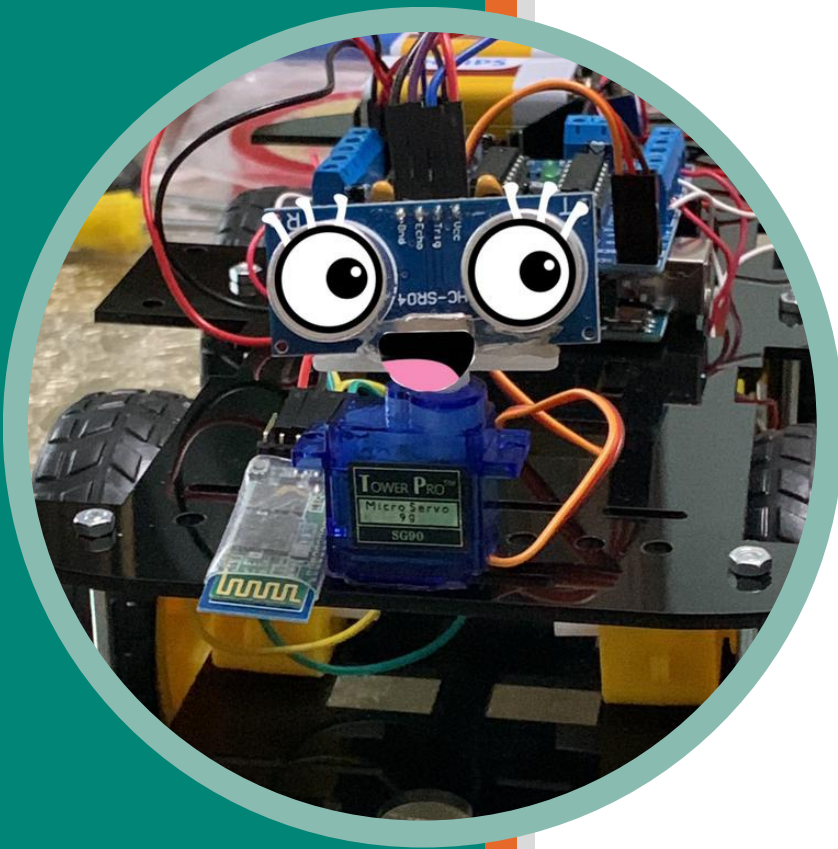


# BLUETOOTH CONTROLLED OBSTACLE AVOIDER

*Computer and Control Engineering*



Prepared for :

Dr. Emad El Sayed

Prepared by :

Aya Mohamed Farouk

Banseh Mohamed Salman

Hla Essam Dawoud

Nour Sherif Aboelenin

Rawan Elsayed Zangir

Rovan Wael Gowed

## COMPUTER ARCHITECTURE PROJECT REPORT





# Table of Content

<b>Table of Content</b>	<b>0</b>
<b>Abstract</b>	<b>1</b>
<b>Introduction</b>	<b>1</b>
Arduino	1
<b>Project Description</b>	<b>1</b>
<b>Hardware Components</b>	<b>2</b>
Robot Smart Car 4WD	2
Motor Driver Shield L293D	2
Arduino Uno Board	3
Ultrasonic Sensor & Distance Measurement Module HC-SR04	5
Servo Motor Micro (180)	6
Bluetooth Module HC-05	7
<b>Circuit Diagram</b>	<b>8</b>
<b>Circuit Operation</b>	<b>9</b>
Circuit Connection	9
<b>Arduino Code</b>	<b>9</b>
<b>Mobile Application</b>	<b>12</b>
Bluetooth Control	12
Voice Control	12
<b>References</b>	<b>13</b>



## Abstract

Bluetooth Controlled Obstacle Avoider is our first try at Arduino and it was such a fun way to learn what hardware components and code can do together.

## Introduction

The Arduino board can be programmed to do anything by simply programming the microcontroller onboard using a set of instructions for which, the Arduino board consists of a USB plug to communicate with your computer and a bunch of connection sockets that can be wired to external devices like motors, LEDs, etc.

### Arduino

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor - and turn it into an output - activating a motor, turning on an LED, or publishing something online.

Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

## Project Description

Bluetooth Controlled Obstacle Avoider is a robot car integrated with an Arduino Board, Bluetooth Module, and Ultrasonic Sensors to get the information from the surrounding area and avoid any obstacle collisions.

## Hardware Components

Components	
Robot Smart Car 4WD (1)	Servo Motor Micro (1)
Motor Driver Shield L293D (1)	Bluetooth Module HC-05 (1)
Arduino Uno Board (1)	Rechargeable 9V Battery (2)
Ultrasonic Sensor & Distance Measurement HC-SR04 (1)	Mini Toggle Switch & Arduino Uno Cable (1)

### 1. Robot Smart Car 4WD

We used a 4WD smart car chassis kits Arduino robot chassis in this project. Our 4WD smart car chassis kits Arduino robot chassis has the following advantages:

- A simple mechanical structure, very convenient to install.
- Four deceleration DC motor turning flexible, good direction. The four-drive horsepower chronological. The Arduino robot 4WD smart car chassis large stability is very easy to extend.
- Arduino robot 4WD smart car chassis Kits can equipped L293D four motor drive module, four tracking module, as well as the 51 seamless connections of the control unit, left red obstacle avoidance expansion hole, the composition of the system is simple and nice.

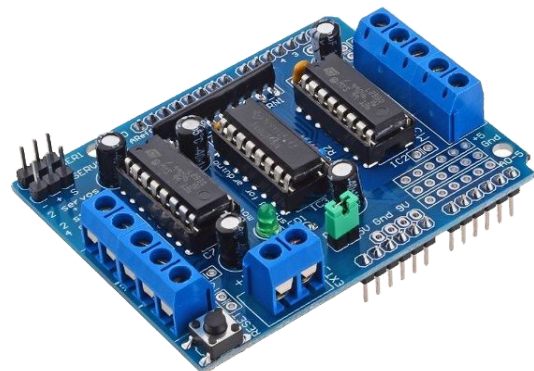
### 2. Motor Driver Shield L293D

Motor Driver Shield L293D is a full-featured motor shield – perfect for many robots and CNC projects. It can drive:

- 4 bi-directional DC motors with 8-bit speed selection(0-255).
- 2 stepper motors with single coil, double coil, interleaved, or micro-stepping.
- 2 servo motors.

The L293D is a dual-channel H-Bridge motor driver capable of driving a pair of DC motors or a single stepper motor.

As the shield comes with two L293D motor driver chipsets, that means it can individually drive up to four DC motors making it ideal for building four-wheel robot platforms. The



shield offers a total of 4 H-Bridges and each H-bridge can deliver up to 0.6A to the motor.

The shield also comes with a 74HC595 shift register that extends 4 digital pins of the Arduino to the 8 direction control pins of two L293D chips.

#### → Output Terminals

The output channels of both the L293D chips are broken out to the edge of the shield with two 5-pin screw terminals viz. M1, M2, M3 & M4. You can connect four DC motors having voltages between 4.5 to 25V to these terminals.

Each channel on the module can deliver up to 600mA to the DC motor. However, the amount of current supplied to the motor depends on the system's power supply.

You can also connect two stepper motors to output terminals. One stepper motor to motor port M1-M2 and the other to M3-M4.

The GND terminal is also provided if you happen to have a unipolar stepper motor. You can connect the center taps of both stepper motors to this terminal.

The shield brings out the 16bit PWM output lines to two 3-pin headers to which you can connect two servo motors.

#### → AFMotor Library

In order to communicate with the shield, we need to install AFMotor.h library so that we can issue simple commands to control DC, stepper & servo motors.

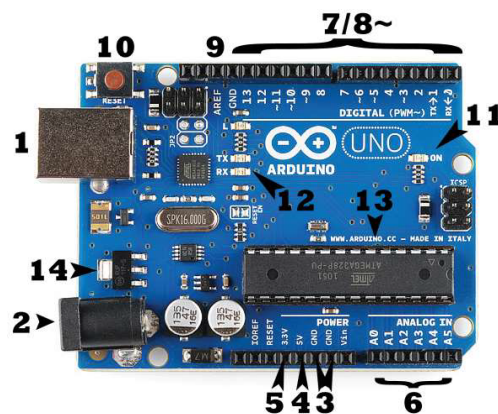
What are the functions to control the speed and spinning direction of a motor?


- `setSpeed(speed)` function sets the speed of the motor. The speed ranges from 0 to 255 with 0 being off and 255 as a full throttle. You can set the speed whenever you want in a program.
- `run(cmd)` function sets the run-mode of the motor. Valid values for cmd are
  1. FORWARD – run forward (actual direction of rotation will depend on motor wiring).
  2. BACKWARD – run backward (rotation will be in the opposite direction from FORWARD)
  3. RELEASE – Stop the motor. This removes power from the motor and is equivalent to `setSpeed(0)`. The motor shield does not implement dynamic braking, so the motor may take some time to spin down.

### 3. Arduino Uno Board

What's on the Arduino Uno Board?

1. Power (USB / Barrel Jack): Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply that is terminated in a barrel jack. In the picture





above the USB connection is labeled (1) and the barrel jack is labeled (2). The recommended voltage for most Arduino models is between 6 and 12 Volts.

2. Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF): The pins on your Arduino are the places where you connect wires to construct a circuit. They usually have black plastic 'headers' that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions
  - GND (3): Short for 'Ground'. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
  - 5V (4) & 3.3V (5): the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
  - Analog (6): The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
  - Digital (7): Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
  - PWM (8): the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). These pins are being able to simulate analog output (like fading an LED in and out).
  - AREF (9): Stands for Analog Reference. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.
3. Reset Button: the Arduino has a reset button (10). Pushing it will temporarily connect the reset pin to the ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times.
4. Power LED Indicator: Just beneath and to the right of the word "UNO" on the circuit board, there's a tiny LED next to the word 'ON' (11). This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong.
5. TX RX LEDs: TX is short for transmitting, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. There are two places on the Arduino UNO where TX and RX appear, once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs (12). These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).
6. Main IC: The black thing with all the metal legs is an IC, or Integrated Circuit (13). Think of it as the brains of our Arduino. The main IC on the Arduino is slightly

different from board type to board type but is usually from the ATmega line of ICs from the ATMEL company.

7. Voltage Regulator: The voltage regulator (14). The voltage regulator controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts.

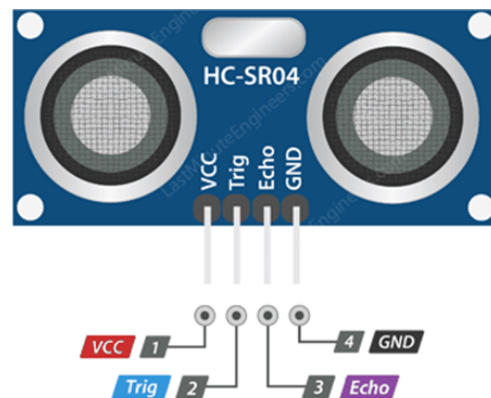
#### 4. Ultrasonic Sensor & Distance Measurement Module HC-SR04

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 waves travel faster than the speed of audible sound (i.e. the sound that humans can hear which its maximum frequency is 20KHz). Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound).

→ HC-SR04 Ultrasonic Sensor Pinout

The HC-SR04 Ultrasonic Sensor has four main pins.

1. VCC: the power supply for HC-SR04 Ultrasonic distance sensor which we connect the 5V pin on the Arduino.
2. Trig (Trigger) pin: used to trigger the ultrasonic sound pulses.
3. Echo pin: produces a pulse when the reflected signal is received. The length of the pulse is proportional to the time it took for the transmitted signal to be detected.
4. GND: should be connected to the ground of Arduino.



→ Features and complete specifications

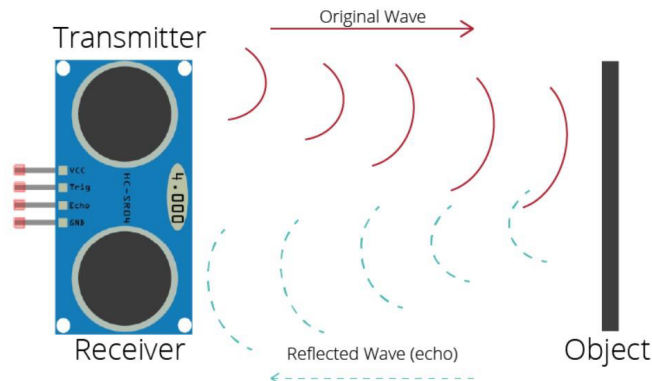
Operating Voltage	DC 5V
Operating Current	15mA
Operating Frequency	40KHz
Max Range	4m
Min Range	2cm
Measuring Angle	15 degree



→ How does it work?

The ultrasonic sensor uses sonar to determine the distance to an object.

- The ultrasound transmitter (trig pin) emits a high-frequency sound (40 kHz).
- The sound travels through the air. If it finds an object, it bounces back to the module.
- The ultrasound receiver (echo pin) receives the reflected sound (echo).



## 5. Servo Motor Micro (180)

A servo motor is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision. If you want to rotate an object at some specific angles or distance, then you use a servo motor. It is just made up of a simple motor which runs through a servo mechanism.

→ Controlling Servo Motor

All servo motors have three wires coming out of them. Out of which two will be used for Supply (positive and negative) and one will be used for the signal that is to be sent from the MCU.

Servo motor is controlled by PWM (Pulse with Modulation) which is provided by the control wires. There is a minimum pulse, a maximum pulse and a repetition rate. Servo motor can turn 90 degree from either direction from its neutral position. The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns.





## 6. Bluetooth Module HC-05

The HC-05 is a popular module that can add two-way (full-duplex) wireless functionality to your projects. You can use this module to communicate between two microcontrollers like Arduino or communicate with any device with Bluetooth functionality like a Phone or Laptop. There are many android applications that are already available which makes this process a lot easier.

The HC-05 has a range of up to 100m which depends upon transmitter and receiver, atmosphere, and geographic and urban conditions.

HC-05 has a red LED that indicates connection status, whether the Bluetooth is connected or not. Before connecting to the HC-05 module this red LED blinks continuously in a periodic manner. When it gets connected to any other Bluetooth device, its blinking slows down to two seconds.

This module works on 3.3 V. We can connect 5V supply voltage as well since the module has on board 5 to 3.3 V regulator.

### → Pin Description

Bluetooth serial modules allow all serial-enabled devices to communicate with each other using Bluetooth. It has 6 pins.



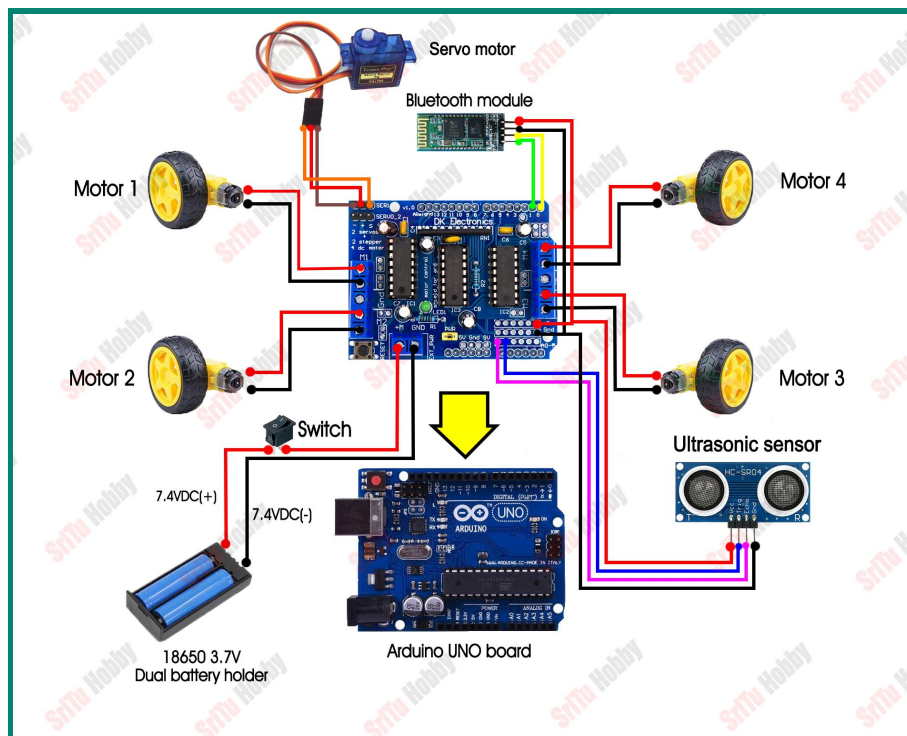
1. Key/EN: It is used to bring Bluetooth module into AT commands mode. If the Key/EN pin is set to high, then this module will work in command mode. Otherwise, by default, it is in data mode. The default baud rate of HC-05 in command mode is 38400bps and 9600 in data mode. HC-05 module has two modes:
  - Data mode: Exchange of data between devices.
  - Command mode: It uses AT commands which are used to change the setting of HC-05. To send these commands to module serial (USART) port is used.
2. VCC: Connect 5 V or 3.3 V to this Pin.
3. GND: Ground Pin of module.
4. 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.

5. TXD: Transmits Serial Data. Everything received via Bluetooth will be given out by this pin as serial data.
6. RXD: Receive Serial Data. Every serial data given to this pin will be broadcasted via Bluetooth.
7. State: The state pin is connected to on board LED, it can be used as a feedback to check if Bluetooth is working properly.

→ Bluetooth communication between Devices

To communicate smartphone with HC-05 Bluetooth module, smartphone requires Bluetooth terminal application for transmitting and receiving data. You can find Bluetooth terminal applications for android and windows in respective app stores.

## Circuit Diagram



## Circuit Operation

### Circuit Connection

Firstly, we connect the 4 DC motors to the Motor driver shield L293D, to do this we use the circuit diagram above.

We connect two motors (M1,M2) to the channels for DC motors (1&2) and the other two (M3,M4) to the channels for DC motors (3&4).

Then, we attach the Servo motor and Ultrasonic sensor to the channels for servo motors where GND → GND, echo → AO, Trig → A1, Vcc → 5v.

Afterward, we connect the Bluetooth module to the Motor driver shield L293D by welding the RXD → TX(D1), TXD → RX(D0) and connecting the GND → GND, Vcc → 5v.

Next, we connect the Arduino Uno board to our Arduino IDE and upload our Arduino code. After uploading the code, we connect the Arduino Uno board to the motor driver shield L293D by pressing them together.

Finally we connect the battery socket to the power port negative to the GND and positive to +M.

## Arduino Code

```
#include <Servo.h>
#include <AFMotor.h>
#define Echo A0
#define Trig A1
#define motor 10
#define Speed 170
#define spoint 103
char value;
int distance;
int Left;
int Right;
int L = 0;
int R = 0;
int L1 = 0;
int R1 = 0;
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(Echo, INPUT);
```

```

servo.attach(motor);
M1.setSpeed(Speed);
M2.setSpeed(Speed);
M3.setSpeed(Speed);
M4.setSpeed(Speed); }
void loop() {
//Obstacle();
//Bluetoothcontrol();
//voicecontrol(); }
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(); } }
void Obstacle() {
distance = ultrasonic();
if (distance <= 12) {
Stop(); backward(); delay(100); Stop();
L = leftsee();
servo.write(spoint);
delay(800);
R = rightsee();
servo.write(spoint);
if (L < R) {
right();
delay(500);
Stop();
delay(200); }
else if (L > R) {
left();
delay(500);
Stop();
delay(200); } }
else { forward(); } }
void voicecontrol() {
if (Serial.available() > 0) {
value = Serial.read();
Serial.println(value);
if (value == '^') { forward(); }
else if (value == '-') { backward(); }
else if (value == '<') {
L = leftsee();
servo.write(spoint);
if (L >= 10 ) {
left();
delay(500);
Stop(); }
else if (L < 10) { Stop(); } }
else if (value == '>') {
R = rightsee();
servo.write(spoint);

```

```

if (R >= 10 ) {
right();
delay(500);
Stop(); }
else if (R < 10) { Stop(); } }
else if (value == '**') { Stop(); } } }
// Ultrasonic sensor distance reading function
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(BACKWARD);
M2.run(BACKWARD);
M3.run(FORWARD);
M4.run(FORWARD); }
void left() {
M1.run(FORWARD);
M2.run(FORWARD);
M3.run(BACKWARD);
M4.run(BACKWARD); }
void Stop() {
M1.run(RELEASE);
M2.run(RELEASE);
M3.run(RELEASE);
M4.run(RELEASE); }
int rightsee() {
servo.write(20);
delay(800);
Left = ultrasonic();
return Left; }
int leftsee() {
servo.write(180);
delay(800);
Right = ultrasonic();
return Right; }

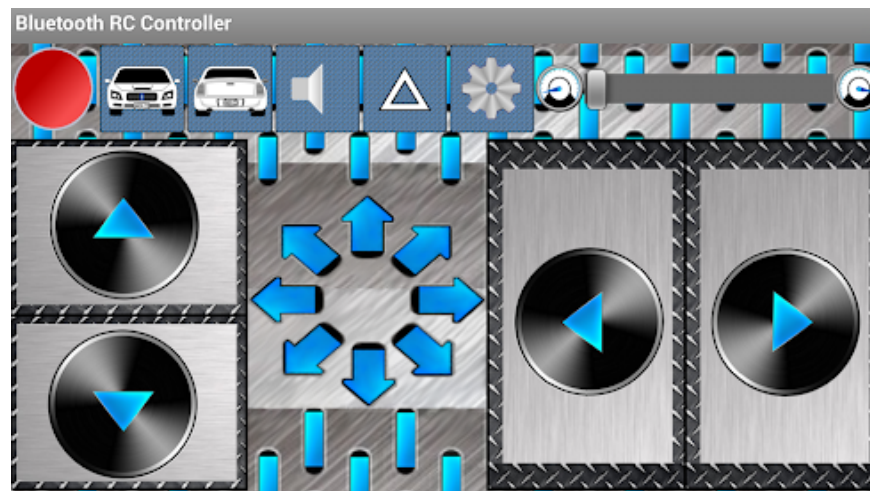
```

## Mobile Application

### Bluetooth Control

We downloaded and installed the android app **Bluetooth RC Car** for this part of the project.

The application allowed us to control our micro controller and Bluetooth fitted RC car with our Android smart phone. The app lets us control the car with either virtual buttons or the phone's accelerometer. There are also two buttons for front and back lights. A flashing light lets us know when the phone is connected to the car, and arrows light up letting us know the car's driving direction.



### Voice Control

We downloaded and installed the android app **Arduino Bluetooth Control** for this part of the project.

Arduino Bluetooth Control is an application that allowed us to control our Arduino Uno board via Bluetooth. The application also smartly remembers our bluetooth module and tries to connect automatically to the latest one we have used, so we won't have to select it every time you use it.

Using Arduino Bluetooth Control, we customized our own vocal commands and used them to control our microcontroller-based boards.





## References

1. "Arduino Introduction." 2016. Electronics Hub. [Arduino Introduction](#).
2. "Arduino L293D Motor Driver Shield Tutorial - Arduino Project Hub." 2019. Arduino Cloud. [Arduino L293D Motor Driver Shield Tutorial](#).
3. "Control DC, Stepper & Servo with L293D Motor Driver Shield & Arduino." n.d. Last Minute Engineers -. [Control DC, Stepper & Servo with L293D Motor Driver Shield & Arduino](#).
4. guide, step. n.d. "What is an Arduino?" Sparkfun Learn. [What is an Arduino?](#).
5. "How HC-SR04 Ultrasonic Sensor Works & How to Interface It With Arduino." n.d. Last Minute Engineers -. [How HC-SR04 Ultrasonic Sensor Works & How to Interface It With Arduino](#).
6. "How to make a multi-function Arduino robot." n.d. SriTu Hobby. [How to make a multi-function Arduino robot - SriTu Hobby](#).
7. Raj, Aswinth. 2015. "Servo Motor Basics, Working Principle & Theory." Circuit Digest. [Servo Motor Basics, Working Principle & Theory](#).
8. Santos, Rui. n.d. "Complete Guide for Ultrasonic Sensor HC-SR04 with Arduino." Random Nerd Tutorials. [Complete Guide for Ultrasonic Sensor HC-SR04 with Arduino | Random Nerd Tutorials](#).
9. "Sensors Modules Bluetooth Module Hc 05 | Sensors Modules." n.d. ElectronicWings. [HC-05 Bluetooth module](#).
10. "UNO R3 | Arduino Documentation." n.d. Arduino Documentation. [UNO R3 | Arduino Documentation](#).
11. "What is Arduino?" n.d. Arduino. [What is Arduino? | Arduino](#).