FENCING

SRB DEMO FILES

RoboVR - The Olympics of Robots

www.robovr.world

Introduction

The Fencing bot made here is mounted on a four-wheel drive car powered by minimal number of electrical components, featuring instantaneous response to driver's input. The bot is compact and light, enabling it to traverse through any terrain. The driver can modify various features according to his will, the main objective here is to reach the opponents position and strike it with a sword and score points.

Requirements

A list of requirements and goals were developed to guide me in the development of this project. The requirements are as follows.

- The four-wheel drive car and the arm with the sword shall be controlled by a Bluetooth controller.
- The swinging mechanism, the servo should be light weight and should develop enough torque to push the ball of the given weight.

Materials:

Arduino Uno R3
HC-05 Bluetooth Module
12V Lithium-ion Battery
L298N Motor Driver Module
DC Motors (300 RPM)
Car Chassis
Servo Motors

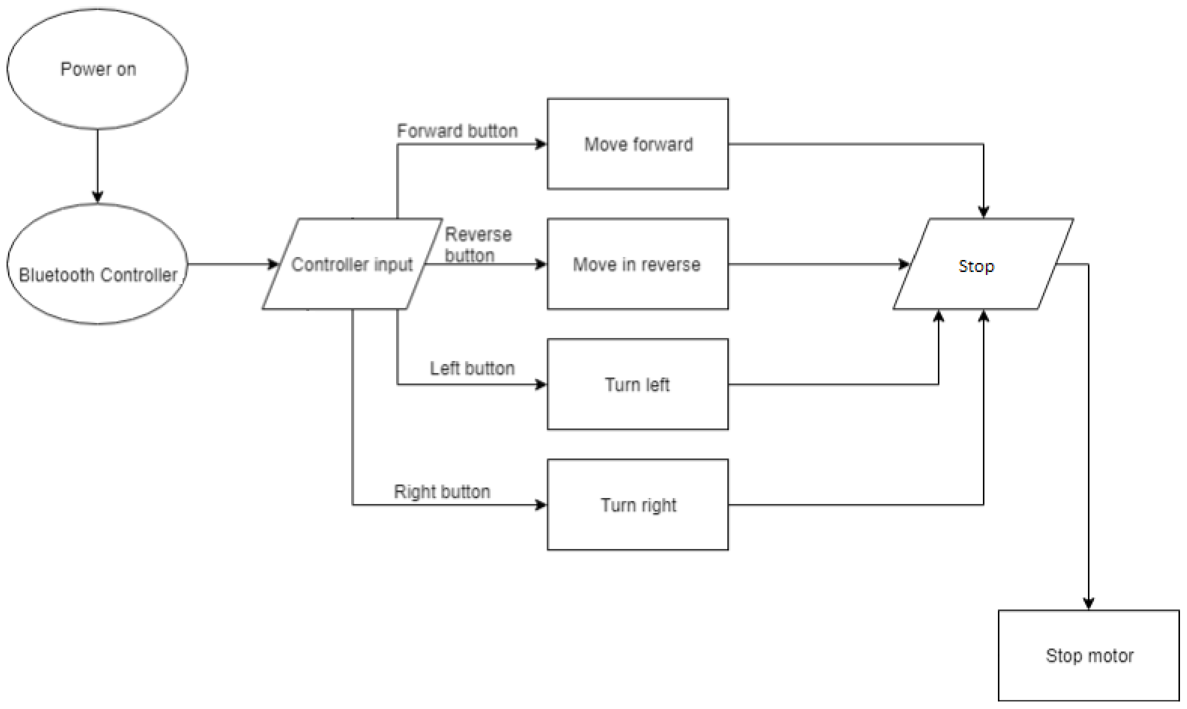
In addition to the hardware used on the arms and the car itself, there was some other hardware necessary to build and the arms of the bot and to run the car. These additional tools included a soldering iron and some solder to assemble the servo motor on the bot and to assemble RC car and solder wires to the pins connected to the motor batteries. In additional an Android phone is required along with the app "**Arduino Bluetooth Control**" which can be found in the Google Play Store.

The app has prebuilt symbols, the user just needs to assign various characters(A-Z or a-z) to these symbols and press enter after assigning characters to each symbols.

Once all the required characters are assigned the user should select "switch to controller mode" for connecting the Bluetooth with the Arduino microcontroller.

If asked for a password, the password may be 0000 or 1234 and the name of the Bluetooth module will be HC-05.

Working

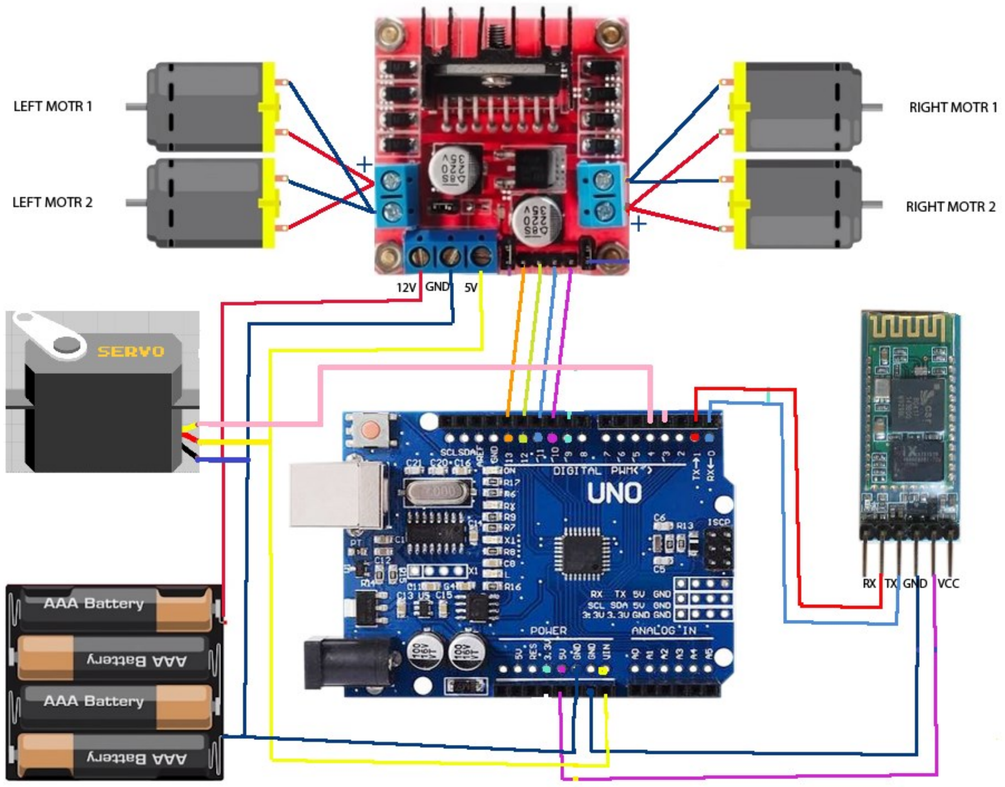


The working of this fencing bot is based on Arduino UNO microcontroller. The block diagram above explains the movement that the car will take in order to reach a specific location and there are buttons on the app for moving the sword in the predefined manner. The controller is used for communication between various electrical components. Motor driver is connected to the controller and is used to provide sufficient power to the motors connected to it and acts as a medium for communication between Arduino and the motors. The Bluetooth module helps in communicating the desired motion via a mobile application. The predefined code helps to achieve forward, backward, right turn and left turn motion as well as it gives us the control of the arm of the bot.

NOTE:

When uploading the program to the Arduino the connection to the GPIO pin D0 and D1 had to be temporarily removed until uploading. This became somewhat of a hassle after having to upload new code repeatedly. This must be done since the D0 and D1 pin is used for serial communication between the Arduino and computer.

Circuit Connection :



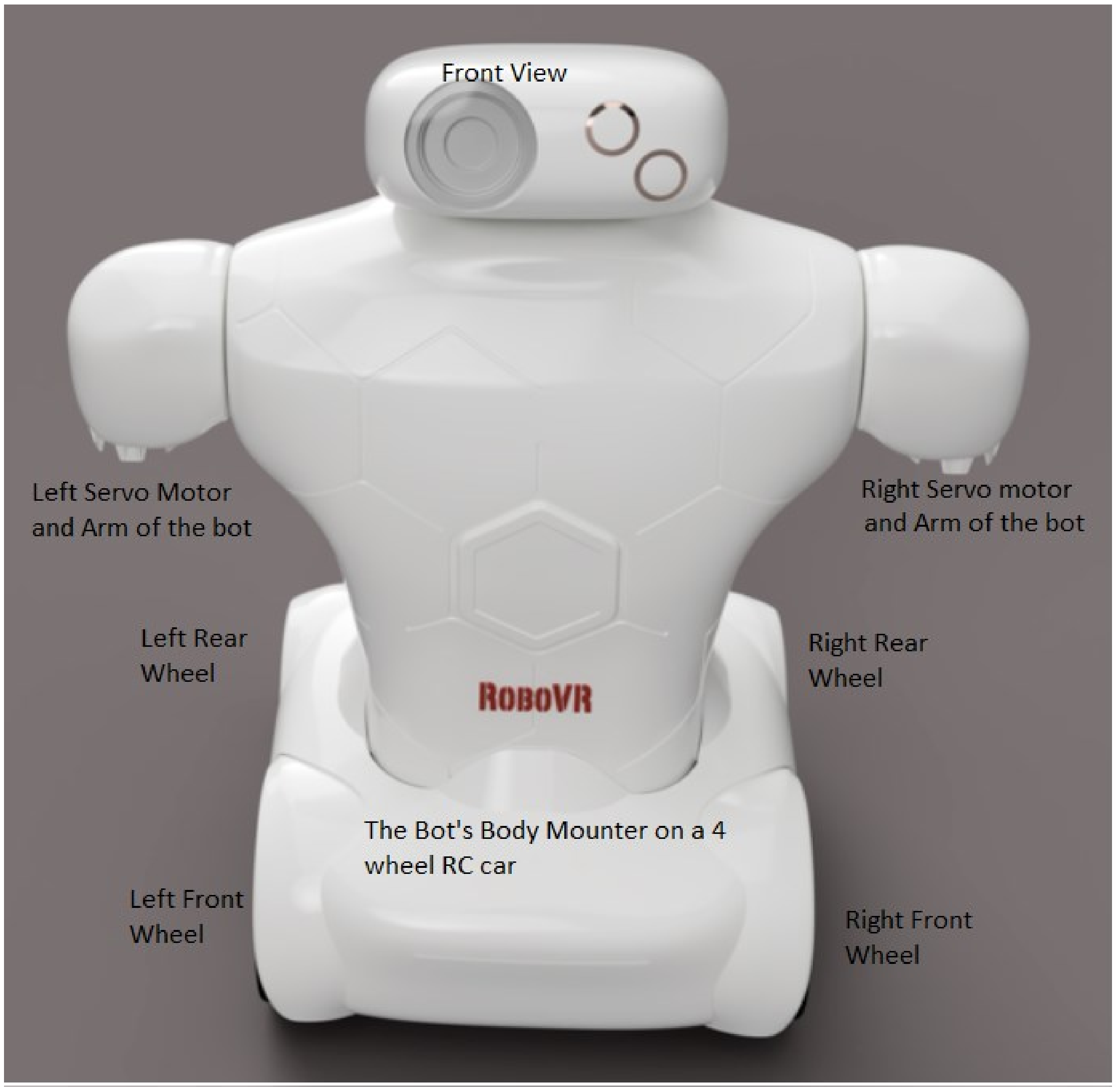
The circuit connections are quite simple, there is one servo motor connected to a pin of the Arduino module that will be used as control inputs for performing fencing related movements.

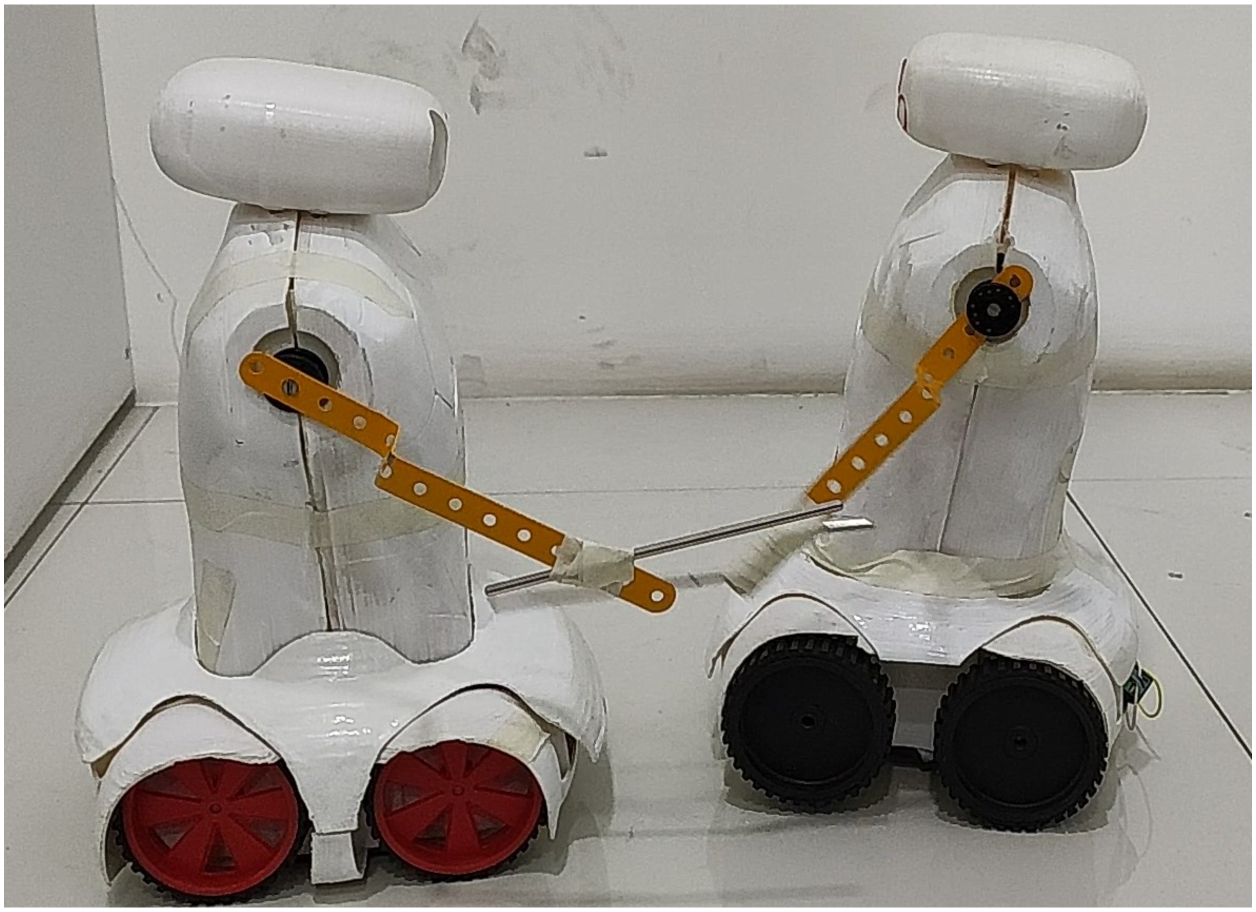
The Vcc and Ground pins of the servo motors should be connected to the Vcc and ground pins of the Arduino.

The four motors that are used to move the bot are connected to the L298N motor driver module as shown in the figure.

The servo motor is connected to pin 4 of the Arduino and moves in the clockwise direction when user presses the button on the Bluetooth app and thus the sword is moved due to this motion of the servo motor.

RoboVR Design:


Front View
Left Servo Motor and Arm of the bot
Right Servo motor and Arm of the bot
Left Rear Wheel
Right Rear Wheel
ROBOVR
The Bot's Body Mounter on a 4 wheel RC car
Left Front Wheel
Right Front Wheel



Code:

```
#include<Servo.h>
Servo myservo;
char ch='S'; //initially STOP condition
const int IN1=9;// Define RIGHT MOTOR
const int IN2=10;
const int IN3=11;// Define LEFT MOTOR
const int IN4=12;
const int IN5=4;
int pos=0;
```

```
void setup()
{

pinMode(IN1,OUTPUT);
pinMode(IN2,OUTPUT);
pinMode(IN3,OUTPUT);
pinMode(IN4,OUTPUT);
pinMode(IN5,OUTPUT);
myservo.attach(4); // attaches the servo on pin 4 to the servo object
Serial.begin(9600);
}

void loop()
{
if(Serial.available()) // check if data is availabe or not
{
ch=Serial.read(); // read the incoming byte
}
Serial.println(ch);
// say what you got
if (ch=='R')
{
RIGHT();
}
if (ch=='L')
{
LEFT();
}
if (ch=='F')
{
FORWARD();
}
if (ch=='B')
{
BACKWARD();
}
if (ch=='S')
{
STOP();
}
if(ch=='X'){
for (pos = 0; pos <= 130; pos += 1) { // goes from 0 degrees to 180
degrees
// in steps of 1 degree
myservo.write(pos); // tell servo to go to position in variable 'pos'
delay(3);
}
for (pos = 130; pos >= 0; pos -= 1) { // goes from 180 degrees to 0
degrees
myservo.write(pos); // tell servo to go to position in variable 'pos'
delay(3);
}
}
}
void FORWARD(){
Serial.println("forward");
//When we want to move the Motor in forward direction,
digitalWrite(IN1,HIGH);
digitalWrite(IN2,LOW);
digitalWrite(IN3,LOW);
digitalWrite(IN4,HIGH);
}
//--------------------------------------------
void BACKWARD(){
Serial.println("backward");
//When we want to move the Motor in backward direction,

digitalWrite(IN1,LOW);
digitalWrite(IN2,HIGH);
digitalWrite(IN3,HIGH);
digitalWrite(IN4,LOW);
}
//--------------------------------------------
void LEFT(){
Serial.println("LEFT");
//When we want to move the Motor in left direction,

digitalWrite(IN1,LOW);//MOVING RIGHT MOTOR IN CLOCKWISE DIRECTION
digitalWrite(IN2,HIGH);
digitalWrite(IN3,LOW);//MOVING LEFT MOTOR IN ANTICLOCKWISE DIRECTION
digitalWrite(IN4,HIGH);
}
//-------------------------------------------
void RIGHT(){
Serial.println("RIGHT");
//When we want to move the Motor in right direction,

digitalWrite(IN1,HIGH);//MOVING RIGHT MOTOR IN ANTICLOCKWISE
DIRECTION
digitalWrite(IN2,LOW);
digitalWrite(IN3,HIGH);//MOVING LEFT MOTOR IN CLOCKWISE DIRECTION
digitalWrite(IN4,LOW);
}

//-------------------------------------------
void STOP(){
Serial.println("stop");
//When we want to Stop the Motor ,

digitalWrite(IN1,LOW);
digitalWrite(IN2,LOW);
digitalWrite(IN3,LOW);
digitalWrite(IN4,LOW);
digitalWrite(IN5,LOW);
}
```