ROBOTICS PROJECT - VOICE CONTROLLED ROBOT

NAME – SOORYA RAM SHIMGEKAR

This project describes the implementation of a voice controlled robot using Arduino. With the help of an android app, we give the command(the direction and the distance). At the receiving side, a Bluetooth transceiver module receives the commands and forwards them to the Arduino on the robot.

Based on our voice command, the Arduino moves forward, backwards, either to left or right and moves by the distance given by our voice command.

Required Components:

1. Arduino UNO with cable
2. Bluetooth HC-05
3. L293D motor driver
4. Two-wheel robot chassis
5. Two dc motors
6. jumper wires
7. mini breadboard
8. 9v battery (power bank)
9. 2 Battery clip connectors (1 connector must be suitable with dc jack on Arduino)
10. Speed sensor with encoder wheel

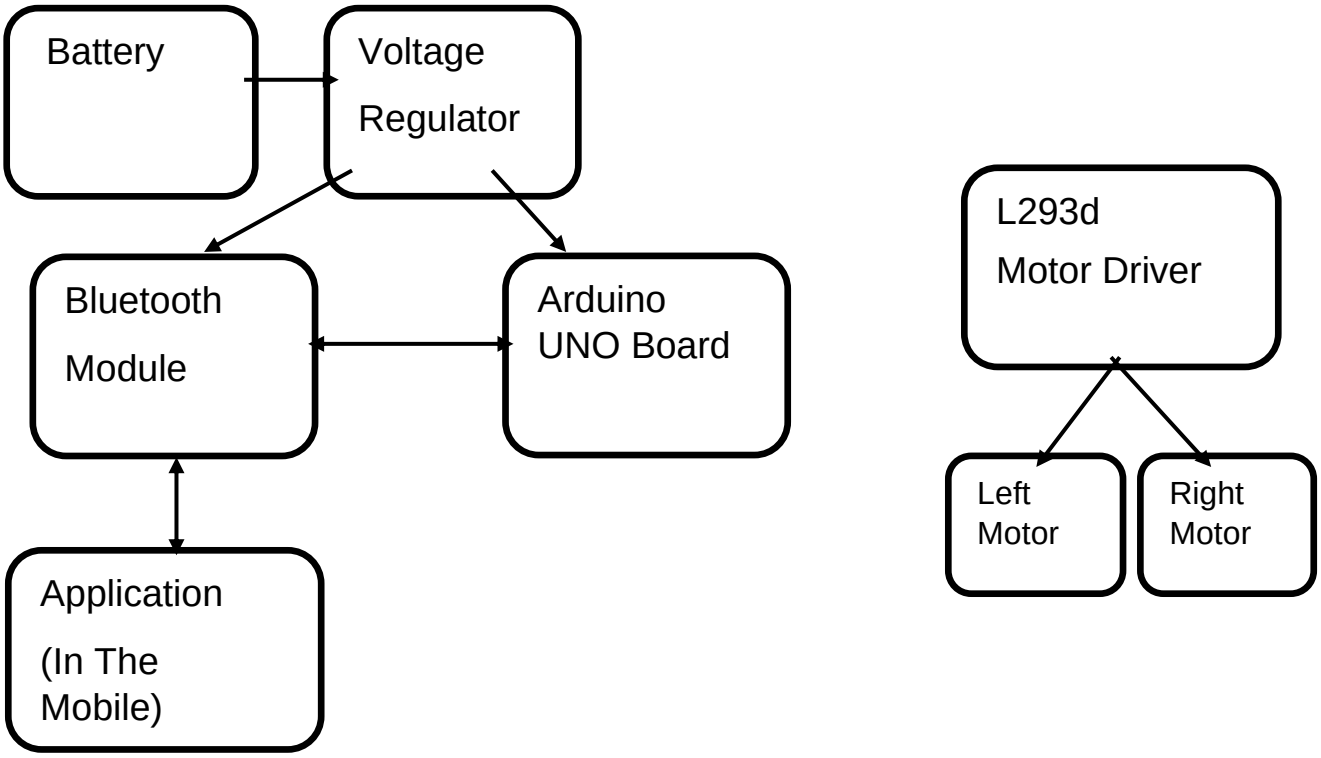
Arduino Uno is a microcontroller board based on the ATmega328P.It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Motor Driver is a module for motors that allows you to control the working speed and direction of two motors simultaneously .

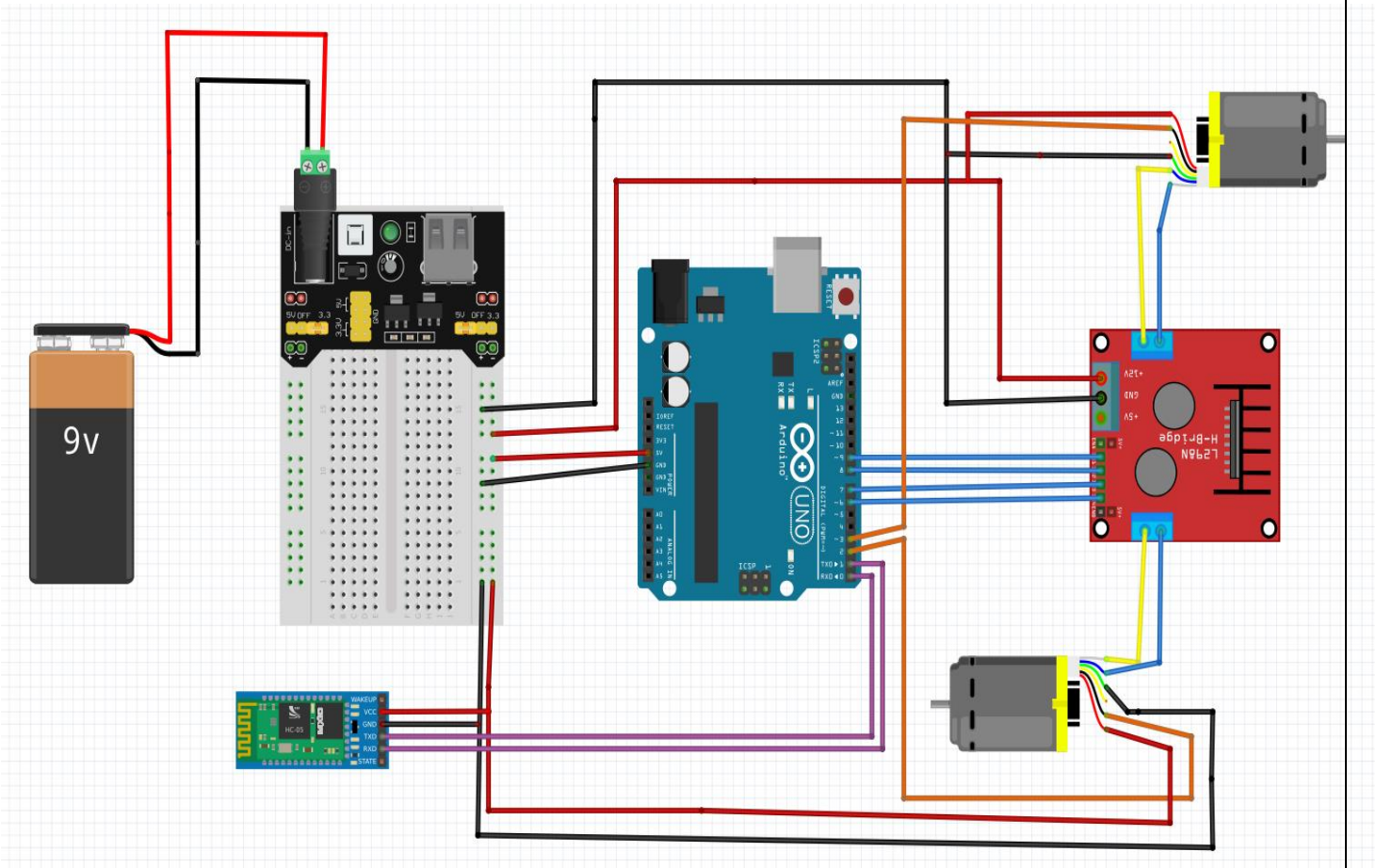
HC-05 Bluetooth Module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. Its communication is via serial communication which makes an easy way to interface with controller or PC.

The Photoelectric Speed Sensor Encoder Coded Disc code wheel is a Slotted Opto isolator module, with an IR transmitter & a photodiode mounted on it. Performs Non-Contact Object Sensing. This is normally used as positional sensor switch (limit switch) or as Position Encoder sensors used to find the position of the wheel.

BLOCK DIAGRAM:



Circuit:-



Code:-

```
#include <SoftwareSerial.h>
#include "TimerOne.h"
#include <Wire.h>

SoftwareSerial blueT(0, 1);

 const byte MOTOR_A = 2;  // Encoder Motor Left Channel A
 const byte MOTOR_B = 3;  // Encoder Motor Right Channel A

  const float stepcount = 520;  // No of pulses per complete rotation of motor shaft
  const float wheeldiameter = 70; // Wheel diameter in mi  llimeters, change if different

  volatile int counter_A = 0;
  volatile int counter_B = 0;

String data = "";
String Direction="";
int Distance = "";
String words[3];

 int in1 = 6; //Motor A and B
 int in2 = 7;
 int in3 = 8;
 int in4 = 9;

void setup() {
  // put your setup code here, to run once:
  blueT.begin(9600);
  Serial.begin(9600);


  attachInterrupt(digitalPinToInterrupt (MOTOR_A), ISR_countA, RISING);  // Increase counter A when speed sensor pin goes
High
```

```
  attachInterrupt(digitalPinToInterrupt (MOTOR_B), ISR_countB, RISING);  // Increase counter B when speed sensor pin goes High
}

void loop()
{
  if(blueT.available())
  {
    data = blueT.readString();
  }
Serial.println(data);
data = data.substring(0, data.length() - 1);



int c=0;
int d=0

for (int i = 0; i < data.length(); i++)
{
  if (data.substring(i, i+1) == " ")
  {
    words[d] = data.substring(c, i);
    c = i+1;
    d=d+1;
  }
}
words[2] = data.substring(c)

for (int i = 0; i < 3; i++)
{
  Serial.println(words[i]);
}

if ((words[0] == "*forward") || (words[0] == "*reverse") || (words[0] == "*left") || (words[0] == "*right") ||
(words[0]=="*auto") || (words[0]=="*line") || (words[0]=="*stop`"))
 {
   Direction = words[0];
   if (words[1] == NULL)
```

```
  {
    Distance = 0;
  }
  if  (words[1]== "distance")
  {
    Distance = words[2].toInt();
  }
 }

if (Direction == "*forward")
 {
 if (Distance == 0)
  {
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  }
 else
  {
   MoveForward(Steps(Distance));
  }
 }

else if (Direction == "*reverse")
 {
 if (Distance == 0)
  {
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  }
  else
  {
    MoveReverse(Steps(Distance));
```

```
   }
  }

else if (Direction == "*stop")
  {
   digitalWrite(in1, LOW);
   digitalWrite(in2, LOW);
   digitalWrite(in3, LOW);
   digitalWrite(in4, LOW);
  }

else if (Direction == "*left")
  {
  if (Distance == 0)
   {
    SpinLeft(Steps(19));
   }
  else
   {
    SpinLeft(Steps(Distance));
   }
  }

else if (Direction == "*right")
 {
  if (Distance == 0)
  {
   SpinRight(Steps(19));
  }
   else
  {
   SpinRight(Steps(Distance));
  }
}

data="";
```

```
Distance = 0;
Direction = "";
words[0] = "";
words[1] = "";
words[2] = "";



 while(!blueT.available());
}



 void ISR_countA()
  {
    counter_A++;  // increment Motor A counter value
  }

  // Motor B pulse count ISR
  void ISR_countB()
  {
    counter_B++;  // increment Motor B counter value
  }



int Steps(float cm)
{
    int result;  // Final calculation result
    float circumference = (wheeldiameter * 3.14) / 10; // Calculate wheel circumference in cm
    float cm_step = circumference / stepcount;  // CM per Step
    float f_result = cm / cm_step;  // Calculate result as a float
    result = (int) f_result; // Convert to an integer (note this is NOT rounded)
    return result;  // End and return result
}

 void MoveForward(int steps)
  {
     counter_A = 0;  //  reset counter A to zero
```

```
  counter_B = 0;  //  reset counter B to zero

  while (steps > counter_A || steps > counter_B)
  {
  // Set Motor A forward
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);

  // Set Motor B forward
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  }
 // Stop when done
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);

  // Set Motor B forward
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);

  counter_A = 0;  //  reset counter A to zero
  counter_B = 0;  //  reset counter B to zero
}

void MoveReverse(int steps)
 {
  counter_A = 0;  //  reset counter A to zero
  counter_B = 0;  //  reset counter B to zero
  while (steps > counter_A || steps > counter_B) {
  // Set Motor A reverse
 digitalWrite(in1, LOW);
 digitalWrite(in2, HIGH);
 // Set Motor B reverse
 digitalWrite(in3, LOW);
 digitalWrite(in4, HIGH);
  }
```

```
  // Stop when done
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);

  // Set Motor B reverse
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  counter_A = 0;  //  reset counter A to zero
  counter_B = 0;  //  reset counter B to zero

}

 void SpinRight(int steps)
{
  Serial.println("Entering Right Function");
   counter_A = 0;  //  reset counter A to zero
   counter_B = 0;  //  reset counter B to zero
   while (steps > counter_A || steps > counter_B){
   // Set Motor A reverse
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);

  // Set Motor B forward
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
   }

  // Stop when done
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);

  // Set Motor B reverse
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
```

```
  counter_A = 0;  //  reset counter A to zero
  counter_B = 0;  //  reset counter B to zero

}

void SpinLeft(int steps)
{
   counter_A = 0;  //  reset counter A to zero
   counter_B = 0;  //  reset counter B to zero
   while (steps > counter_A || steps > counter_B) {
   // Set Motor A forward
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);

  // Set Motor B reverse
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
   }

  // Stop when done
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);

  // Set Motor B reverse
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  counter_A = 0;  //  reset counter A to zero
  counter_B = 0;  //  reset counter B to zero
}
```