Mechatronics

Basic operation of Arduino Microcontroller and control of Motor

ADITYA SHAH

#IndianByHeart

#MechatronicsEnthusiast

#NatureLover

#Wanderer

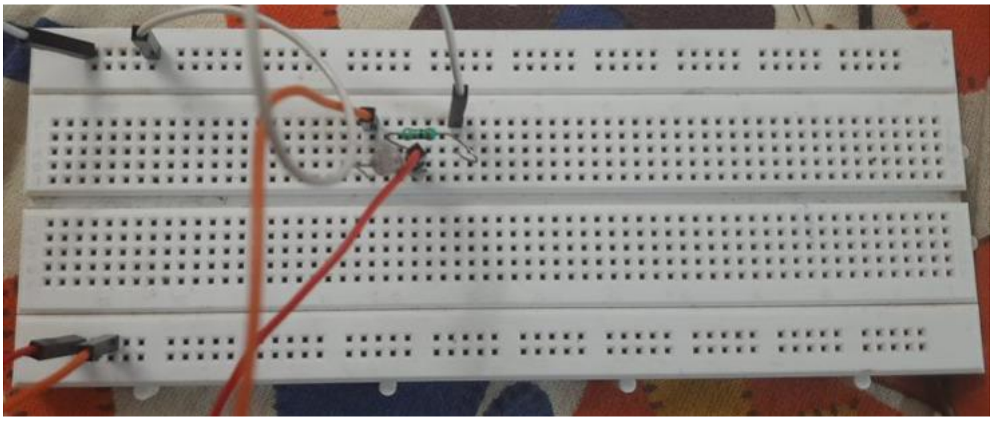
1. Aim of the Experiments.

1. Using LDR as a lux sensor
    a. Signal processing of LDR as a sensor
    b. ADC of the signal
    c. Digital filtering –smoothing (Low pass digital filter)
    d. Calibration using Android phones lux sensor.

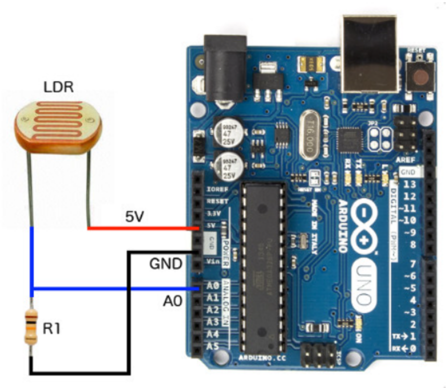
2. Pre-Requisites/Components Required

- LDR: 5mm
- Resistor:10KΩ
- Lux Meter(Android APP)
- Arduino Uno with Programming Cable
- Arduino IDE 1.8.15
- Breadboard and Jumper wires.

3. Circuit Diagram



Name: Aditya Shah

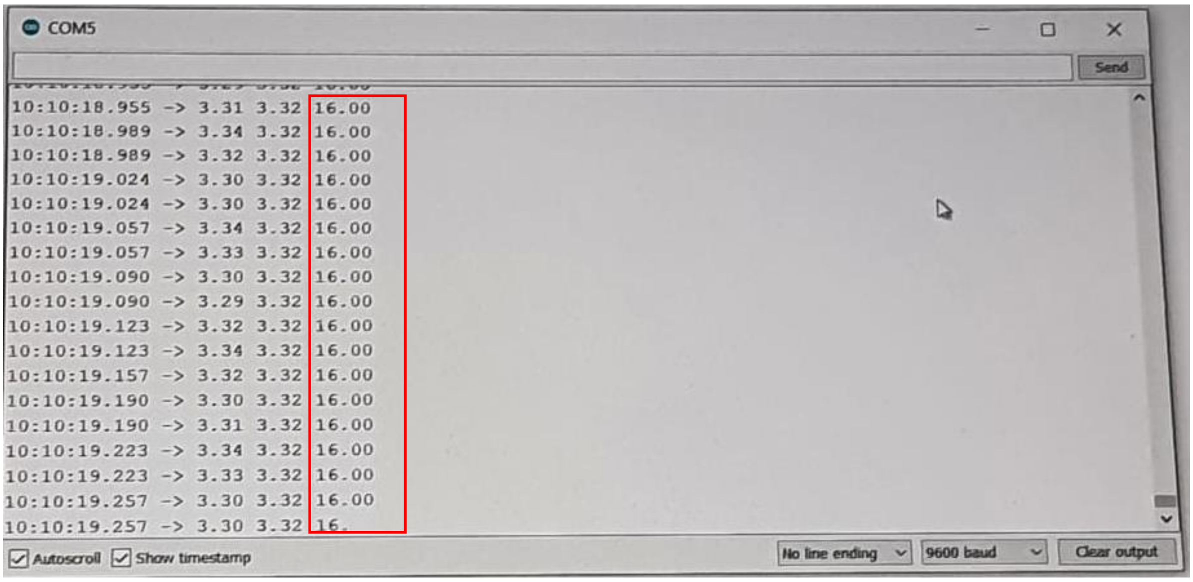


Schematics Source: http://cactus.io/hookups/sensors/light/ldr/hookup-arduino-to-ldr-sensor

4. Procedure

- **Connect the circuit as shown in circuit diagram section.**
- **The light intensity was measured using LDR, being a passive element , we have to give an external supply and the other terminal of LDR ,we gave it as i/p to Arduino A0 as shown in circuit diagram section..**
- **Then code was written in IDE, to convert the analog value to digital values, and then Low Pass filter is applied by a smoothening function, to get smooth values and then it is calibrated using Android phones lux sensor, taken as standard.**
- **Comments are included in the code.**
- **Demonstration video is made. Link given at last.**

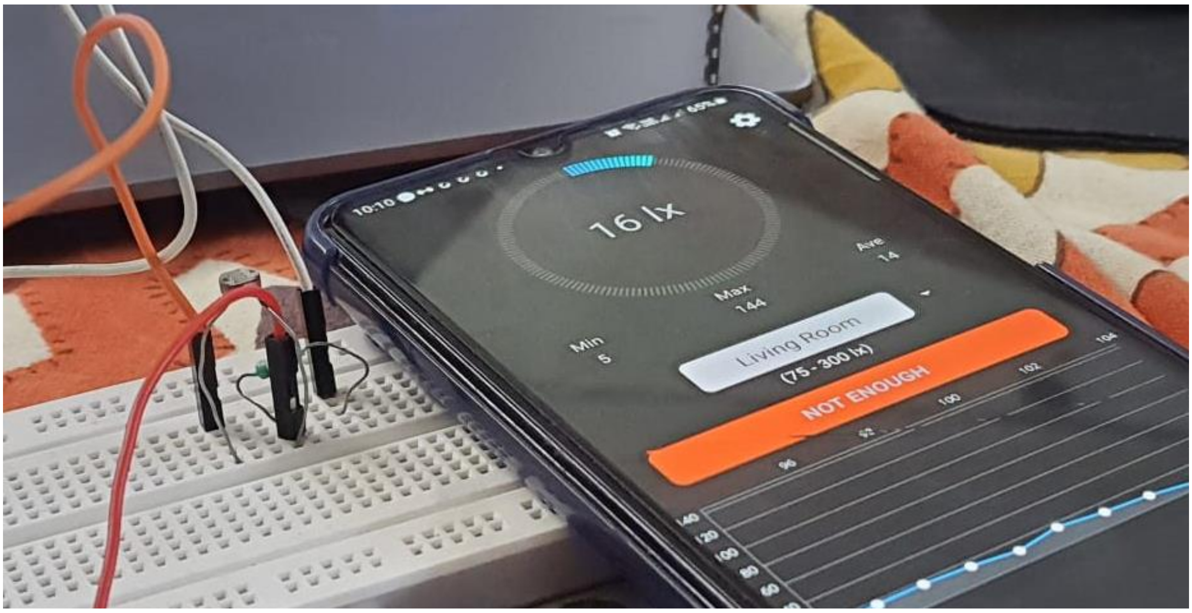
5. Results



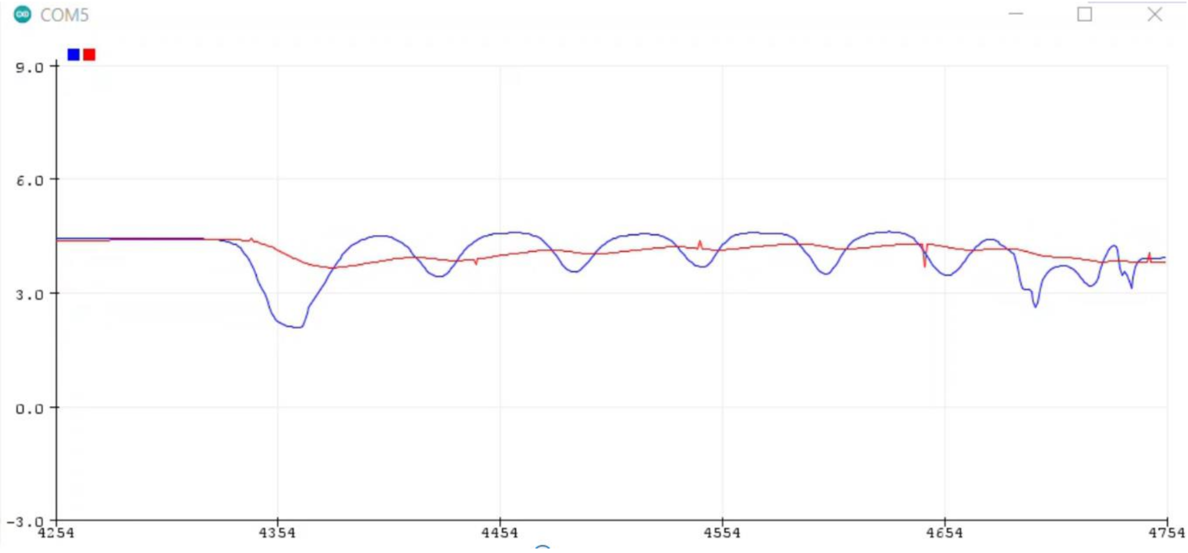
```
10:10:18.955 -> 3.31 3.32 16.00
10:10:18.989 -> 3.34 3.32 16.00
10:10:18.989 -> 3.32 3.32 16.00
10:10:19.024 -> 3.30 3.32 16.00
10:10:19.024 -> 3.30 3.32 16.00
10:10:19.057 -> 3.34 3.32 16.00
10:10:19.057 -> 3.33 3.32 16.00
10:10:19.090 -> 3.30 3.32 16.00
10:10:19.090 -> 3.29 3.32 16.00
10:10:19.123 -> 3.32 3.32 16.00
10:10:19.123 -> 3.34 3.32 16.00
10:10:19.157 -> 3.32 3.32 16.00
10:10:19.190 -> 3.30 3.32 16.00
10:10:19.190 -> 3.31 3.32 16.00
10:10:19.223 -> 3.34 3.32 16.00
10:10:19.223 -> 3.33 3.32 16.00
10:10:19.257 -> 3.30 3.32 16.00
10:10:19.257 -> 3.30 3.32 16.
```

☑ Autoscroll ☑ Show timestamp No line ending 9600 baud

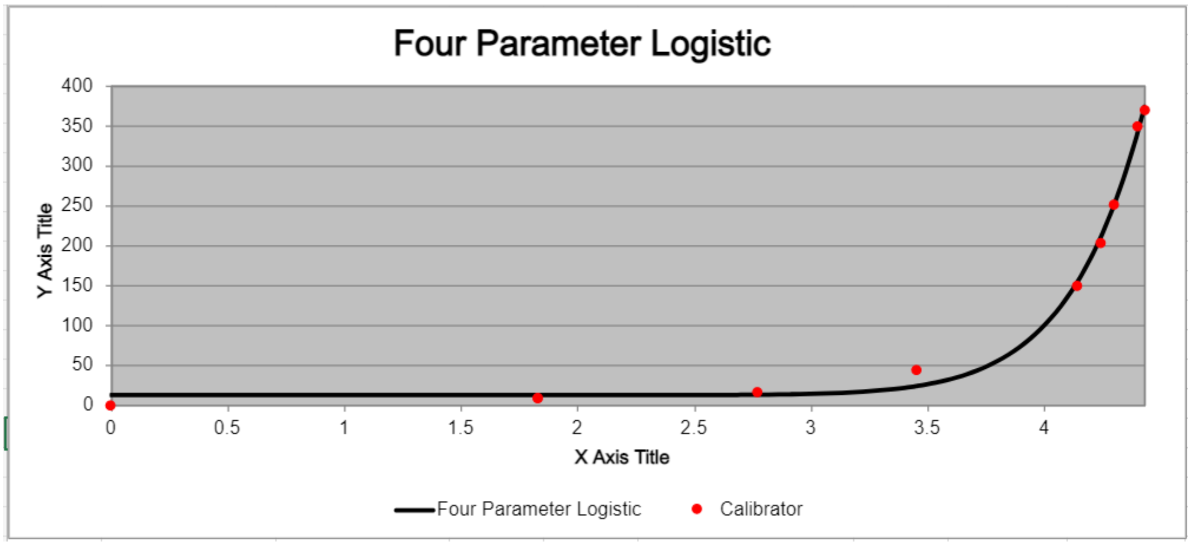
Calibrated results



Phone Lux meter app results



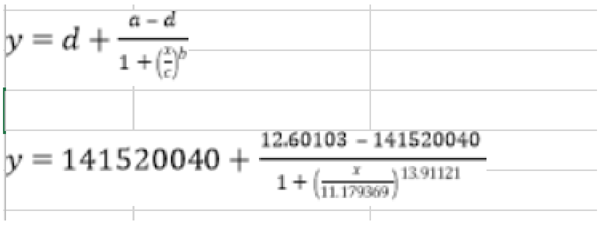
Smoothening Effect (Red Color)



Smoothed sensor value versus android lux sensor reading

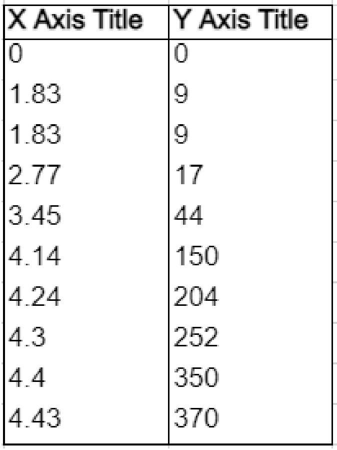
Name: Aditya Shah Roll: 2011mt02

Used below formulation for calibration: -

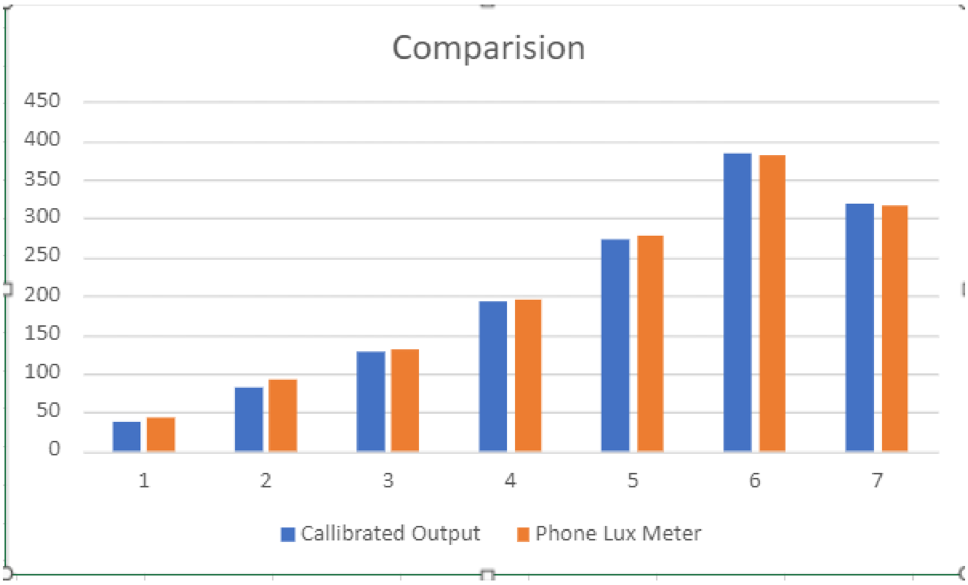


$$y = d + \frac{a - d}{1 + \left(\frac{x}{c}\right)^{b}}$$

$$y = 141520040 + \frac{12.60103 - 141520040}{1 + \left(\frac{x}{11.179369}\right)^{13.91121}}$$



X Axis Title	Y Axis Title
0	0
1.83	9
1.83	9
2.77	17
3.45	44
4.14	150
4.24	204
4.3	252
4.4	350
4.43	370



6. Inference.

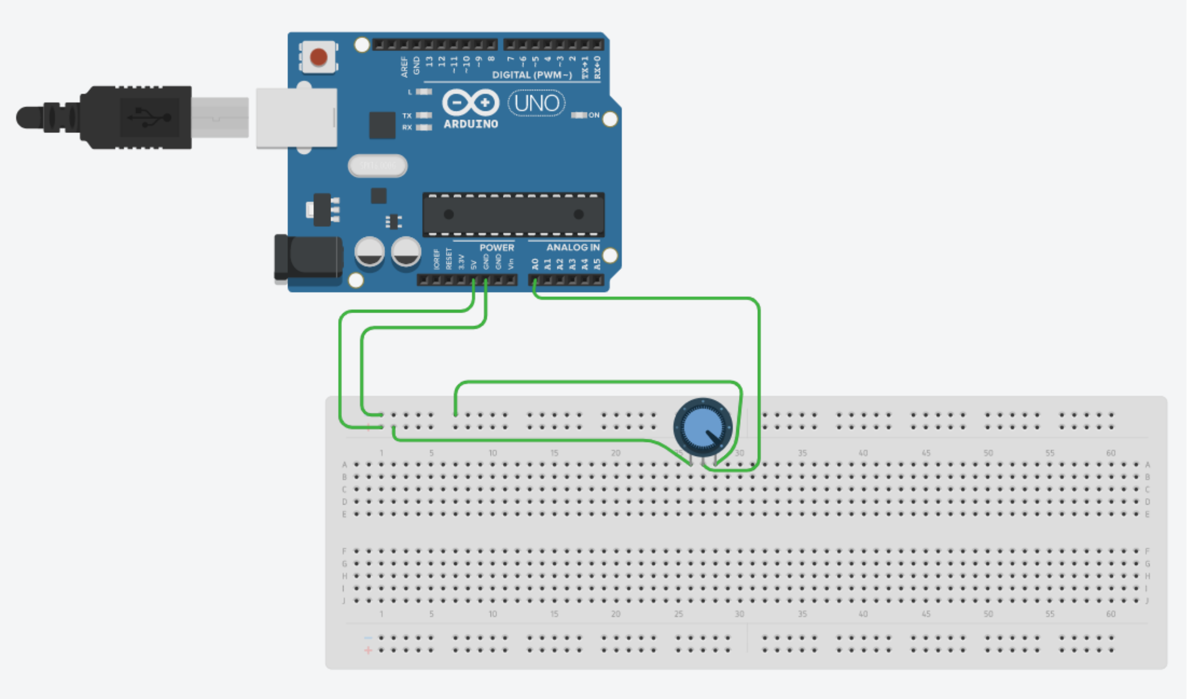
I learnt how to use LDR as a LUX meter. I found difficulty in calibration part. The LDR variation with LIGHT intensity variation was highly nonlinear. To tackle this ,I used an curve fitting technique, for better calibration. I used Non-Linear 4PL technique for calibration, which give good result. During experimentation, it was found for low LUX value, the calibrated value were little lower than phone lux meter app and opposite was observed for higher LUX value case. Also, since the ambient light sensor of phone is located at top, near the camera. For different light intensity, there was little deviation of distance, which may be one of the reasons for the deviation of reading of the two. And since the ADC is of 10 bits, we can get a very good resolution, which is capable to sense 5/1023 V=4.88mV variation coming due to light intensity variation recorded by LDR.

TINKERCAD

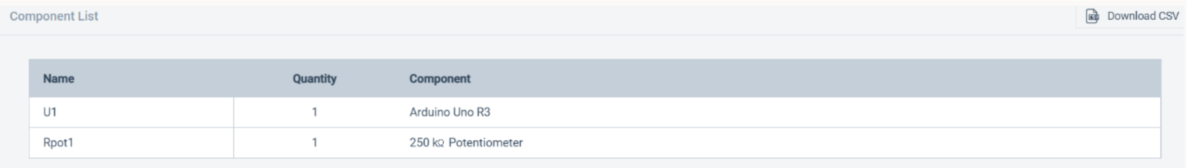
1. **Calibrate the rotary potentiometer to get angles in degree ?**
2. **Control DC motor speed using PWM.**
3. **Attach the DC motor & potentiometer, and control the angle of the DC motor.**
   •Proportional control = error * Kp
   •Use the calibrated potentiometer code and control dc motor code to get this task done.

Answer

1.) Circuit Diagram: -



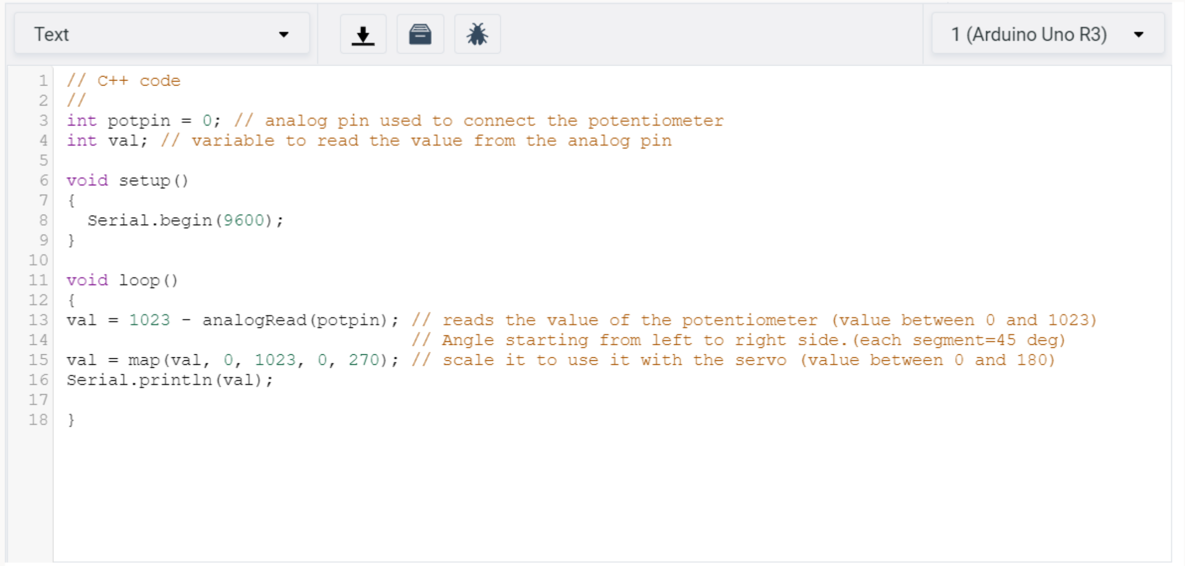
Components: -



Component List

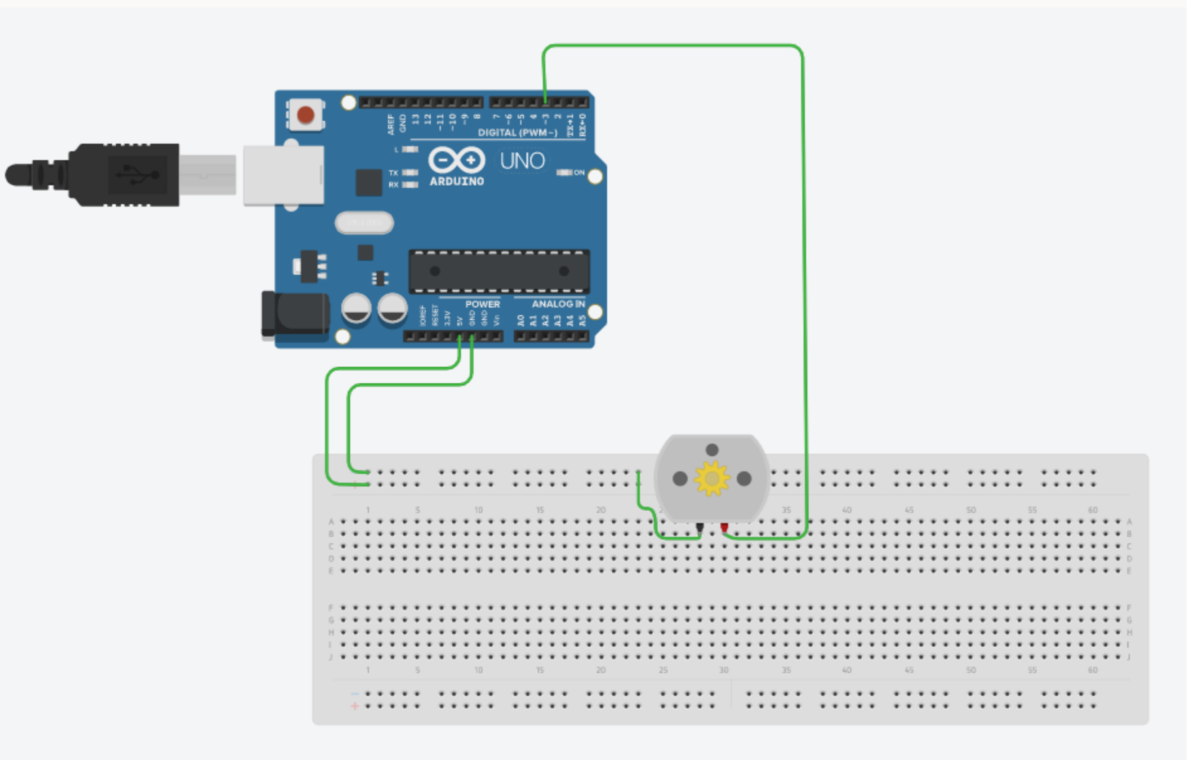
Name	Quantity	Component
U1	1	Arduino Uno R3
Rpot1	1	250 kΩ Potentiometer

Code: -

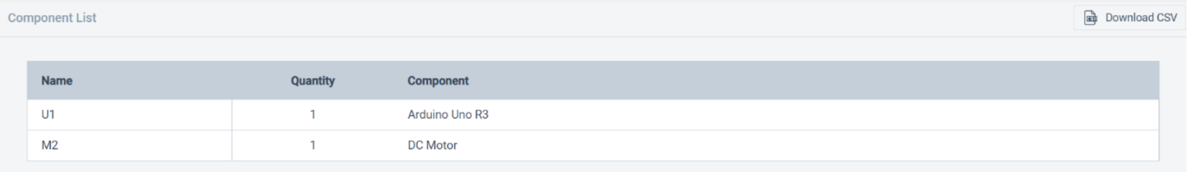


```
// C++ code
//
int potpin = 0; // analog pin used to connect the potentiometer
int val; // variable to read the value from the analog pin

void setup()
{
  Serial.begin(9600);
}

void loop()
{
val = 1023 - analogRead(potpin); // reads the value of the potentiometer (value between 0 and 1023)
                                 // Angle starting from left to right side.(each segment=45 deg)
val = map(val, 0, 1023, 0, 270); // scale it to use it with the servo (value between 0 and 180)
Serial.println(val);

}
```

2.)Circuit Diagram: -



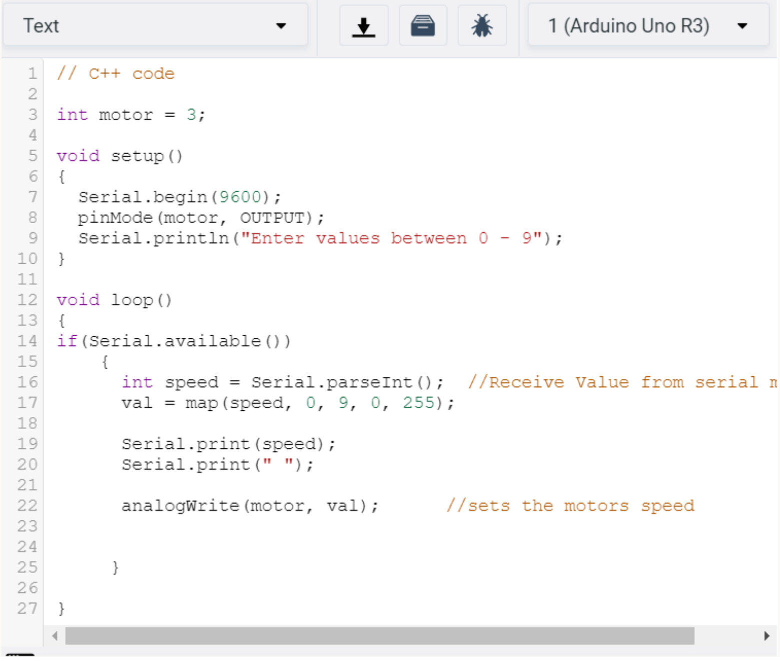
Components: -


Component List

Download CSV

Name	Quantity	Component
U1	1	Arduino Uno R3
M2	1	DC Motor

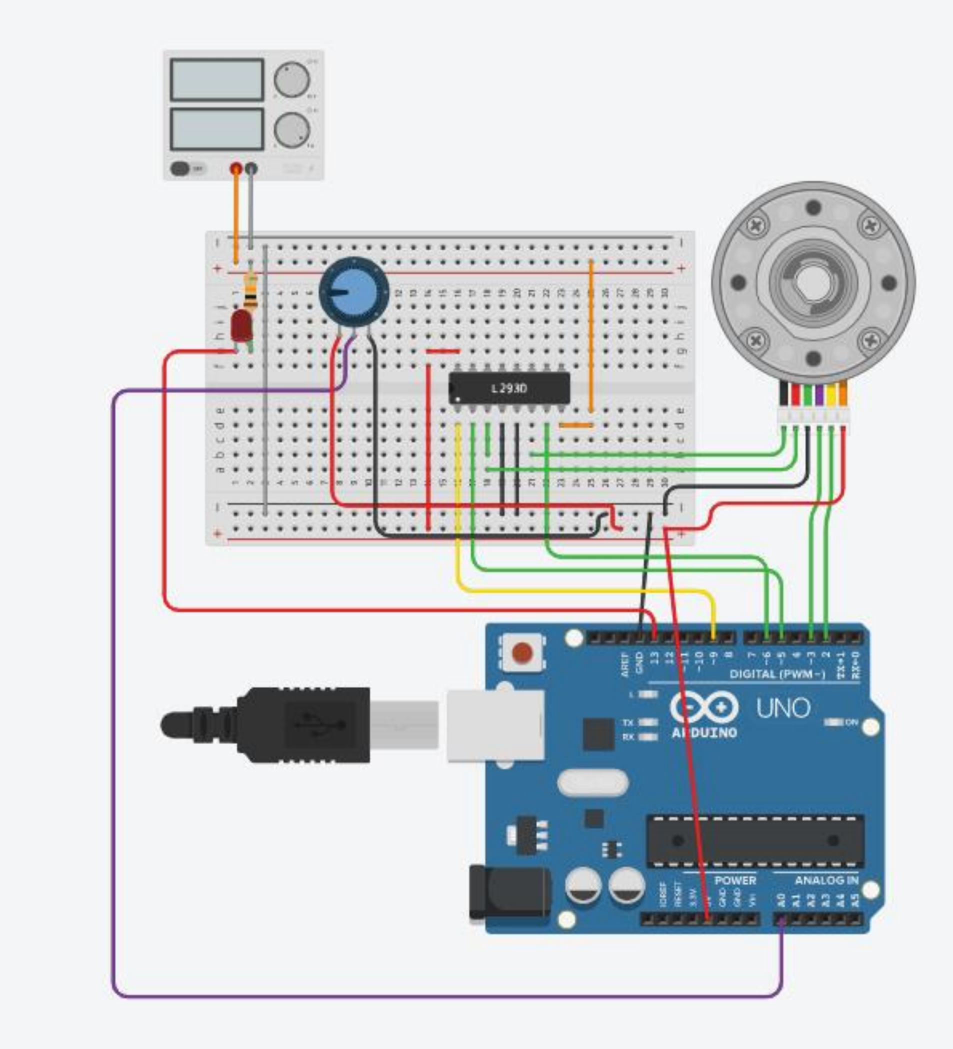
Code: -


Text

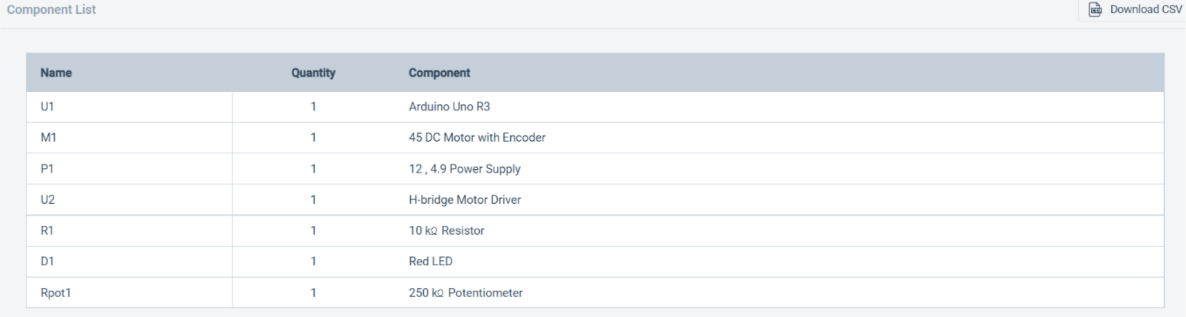
1 (Arduino Uno R3)

```
// C++ code

int motor = 3;

void setup()
{
  Serial.begin(9600);
  pinMode(motor, OUTPUT);
  Serial.println("Enter values between 0 - 9");
}

void loop()
{
if(Serial.available())
    {
      int speed = Serial.parseInt();  //Receive Value from serial m
      val = map(speed, 0, 9, 0, 255);

      Serial.print(speed);
      Serial.print(" ");

      analogWrite(motor, val);      //sets the motors speed


     }

}
```

3.)Circuit Diagram: -



Components: -

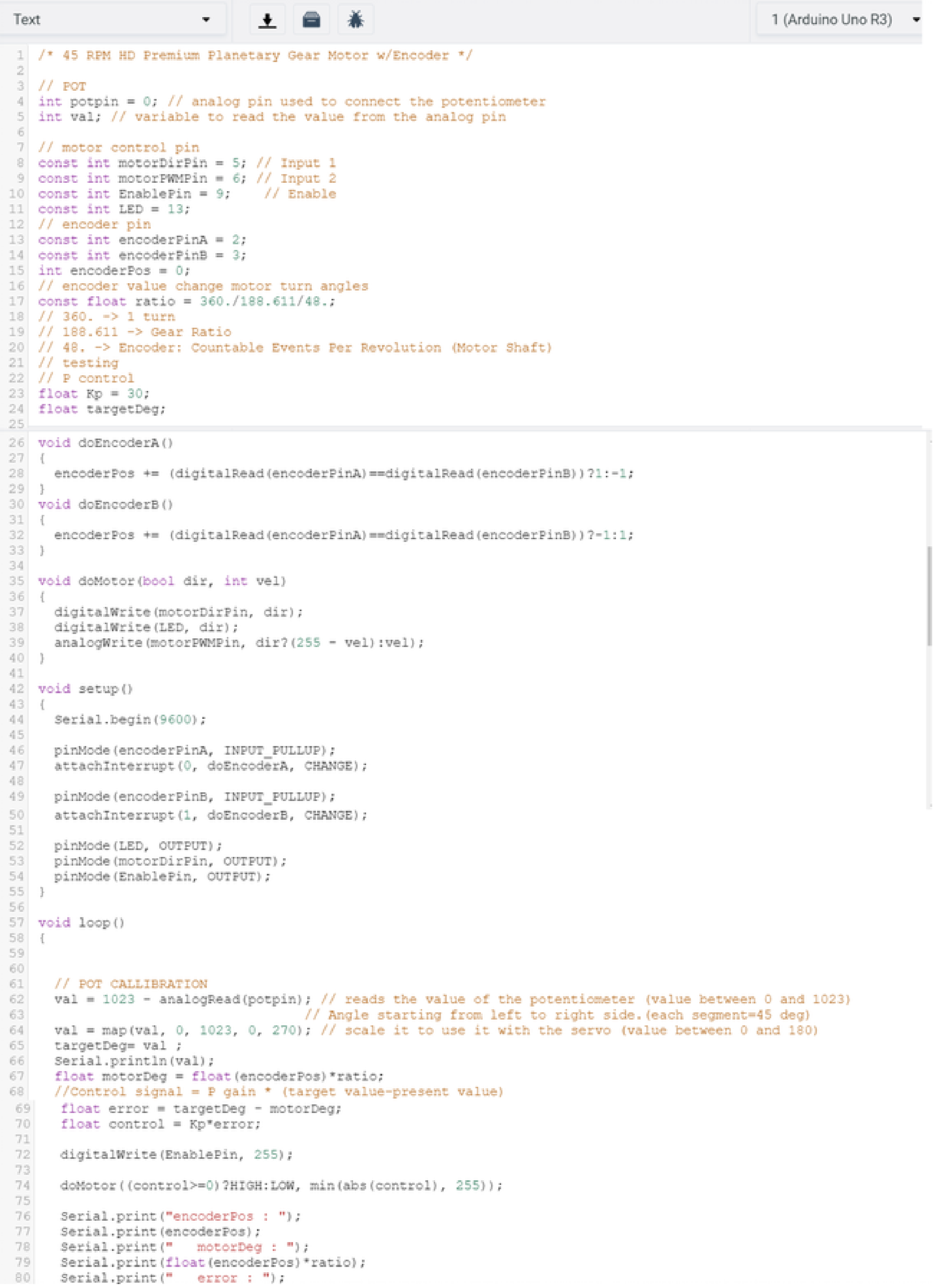


Component List

Download CSV

Name	Quantity	Component
U1	1	Arduino Uno R3
M1	1	45 DC Motor with Encoder
P1	1	12 , 4.9 Power Supply
U2	1	H-bridge Motor Driver
R1	1	10 kΩ Resistor
D1	1	Red LED
Rpot1	1	250 kΩ Potentiometer

Code: -



```
/* 45 RPM HD Premium Planetary Gear Motor w/Encoder */

// POT
int potpin = 0; // analog pin used to connect the potentiometer
int val; // variable to read the value from the analog pin

// motor control pin
const int motorDirPin = 5; // Input 1
const int motorPWMPin = 6; // Input 2
const int EnablePin = 9;    // Enable
const int LED = 13;
// encoder pin
const int encoderPinA = 2;
const int encoderPinB = 3;
int encoderPos = 0;
// encoder value change motor turn angles
const float ratio = 360./188.611/48.;
// 360. -> 1 turn
// 188.611 -> Gear Ratio
// 48. -> Encoder: Countable Events Per Revolution (Motor Shaft)
// testing
// P control
float Kp = 30;
float targetDeg;

void doEncoderA()
{
  encoderPos += (digitalRead(encoderPinA)==digitalRead(encoderPinB))?1:-1;
}
void doEncoderB()
{
  encoderPos += (digitalRead(encoderPinA)==digitalRead(encoderPinB))?-1:1;
}

void doMotor(bool dir, int vel)
{
  digitalWrite(motorDirPin, dir);
  digitalWrite(LED, dir);
  analogWrite(motorPWMPin, dir?(255 - vel):vel);
}

void setup()
{
  Serial.begin(9600);

  pinMode(encoderPinA, INPUT_PULLUP);
  attachInterrupt(0, doEncoderA, CHANGE);

  pinMode(encoderPinB, INPUT_PULLUP);
  attachInterrupt(1, doEncoderB, CHANGE);

  pinMode(LED, OUTPUT);
  pinMode(motorDirPin, OUTPUT);
  pinMode(EnablePin, OUTPUT);
}

void loop()
{


  // POT CALLIBRATION
  val = 1023 - analogRead(potpin); // reads the value of the potentiometer (value between 0 and 1023)
                                 // Angle starting from left to right side.(each segment=45 deg)
  val = map(val, 0, 1023, 0, 270); // scale it to use it with the servo (value between 0 and 180)
  targetDeg= val ;
  Serial.println(val);
  float motorDeg = float(encoderPos)*ratio;
  //Control signal = P gain * (target value-present value)
   float error = targetDeg - motorDeg;
   float control = Kp*error;

   digitalWrite(EnablePin, 255);

   doMotor((control>=0)?HIGH:LOW, min(abs(control), 255));

   Serial.print("encoderPos : ");
   Serial.print(encoderPos);
   Serial.print("   motorDeg : ");
   Serial.print(float(encoderPos)*ratio);
   Serial.print("   error : ");
```

To View the Demonstration, Output and Circuit Image's: -
Click on the following Link: -

- Exp-1.1- Control LED brightness using PWM
- Exp-1.2- Oscillate LED brightness in a sinusoidal wave
- Exp-1.3- Emulate PWM using delayMicroseconds() instruction
- Exp-1.4- Using LDR as a lux sensor
- Extra-1- Calibrate the rotary potentiometer
- Extra-2- Control DC motor speed using PWM
- Extra-3- Control the angle of the DC motor