College of Electrical & Mechanical Engineering (CEME), NUST- Pakistan

Department of Electrical Engineering

LINEAR CONTROL SYSTEMS (EE-371)

PROJECT REPORT

DC MOTOR POSITION CONTROL USING PID CONTROLLER

Group Members			
Mashhood Ahmad (338931)			
Noman Aftab (340743)			
Abdul Hadi (332460)			
Degree	**42-EE**	**Syndicate**	**B**

Submitted on: 07-06-2023

Submitted to:

AZMAT SAEED

Objective

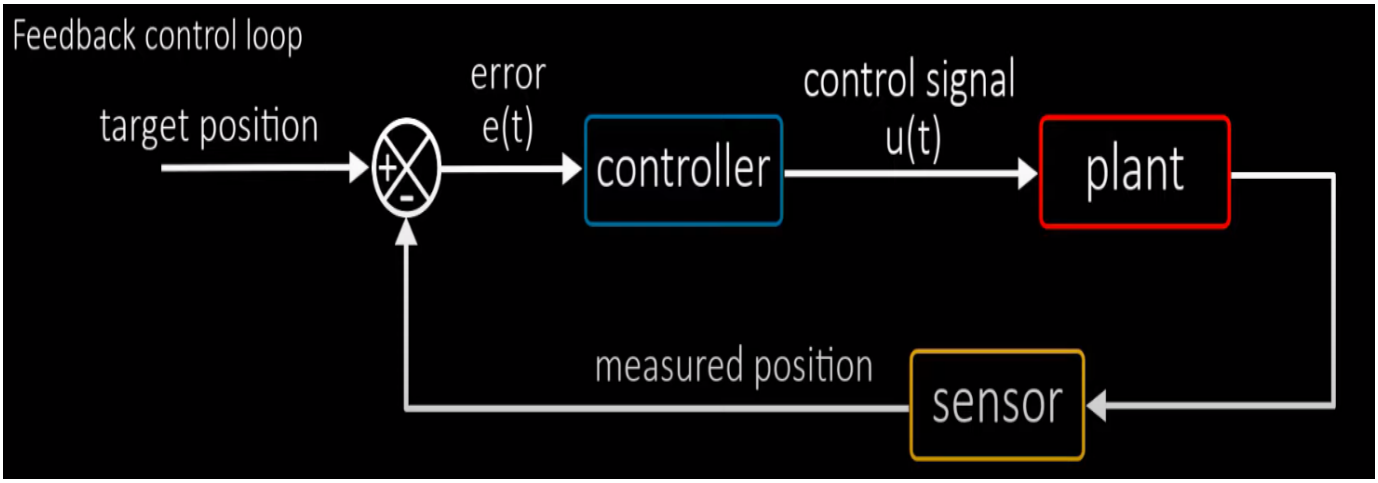
The core objective of this project was to design and implement a closed loop system to control the position of a DC Motor. Our main task was to design a controller for DC motor position control using the knowledge which we covered during Linear Control systems course.
The main requirements of the project were the following.

- Designing of a PID Controller by techniques which we covered in the course.
- Implementing its Digital model on Hardware
- Arduino Interfacing with Matlab
- GUI Designing in Matlab to directly control position through Matlab

Literature Review

A closed loop system, also known as a feedback control system, is a system that uses feedback to maintain or adjust its operation. In a closed loop system, the output of the system is continuously monitored and compared to a desired or reference value. The difference between the output and the reference value is called the error signal. This error signal is then fed back to the controller as an input, and then controller generates output which is a control signal for the system to achieve desired performance.

The feedback loop allows the system to dynamically respond to changes and disturbances, continuously striving to minimize the error and bring the output closer to the desired value. By continuously comparing the output with the reference value and making adjustments based on the feedback, a closed loop system can maintain stability, accuracy, and desired performance.



Closed loop systems are widely used in various fields, including engineering, control systems, electronics, and automation. Examples of closed loop systems include temperature control systems, cruise control systems in automobiles, and automatic voltage regulators in power systems.

In this project we are going to design a closed loop system for DC Motor position control. As in the block diagram of closed loop system:

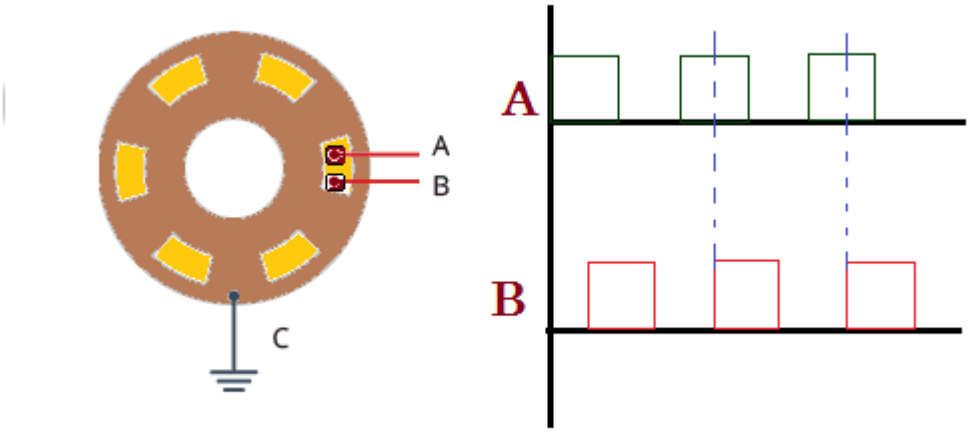
- Plant = DC Motor + Motor Driver IC.
- Controller = Arduino
- Sensor = Encoder
- Reference signal will be provided from GUI

Encoder is a device that is often used in conjunction with a DC motor to provide feedback about the motor's position, speed, or both. It helps in achieving precise control and monitoring of the motor's operation.
The encoder consists of two main components: a sensor and an encoder disc. The sensor is typically a photoelectric or magnetic device, while the encoder disc has slots or markings.

As the motor rotates, the encoder disc spins along with it. The sensor, positioned near the disc, detects the slots or markings as they pass by.

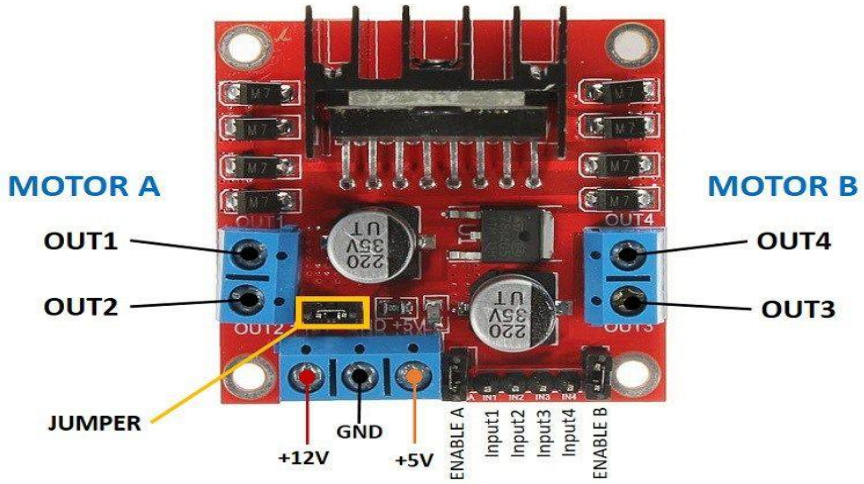
The sensor detects the slots or markings on the encoder disc, producing electrical signals. These signals are typically in the form of square waves and are often referred to as channels A and B. The signals from channel A and channel B are phase-shifted to determine the direction of rotation. By counting the number of pulses and tracking their

phase relationship, the motor's position and speed can be determined.



The encoder signals are then fed back to a control system, such as a microcontroller or a motor controller. The control system interprets these signals and adjusts the motor's operation accordingly. This closed-loop feedback allows for accurate control of the motor's position, speed, and direction.

Motor Driver IC L298N is a popular dual H-bridge motor driver IC that allows you to control the direction and speed of DC motors or control the rotation of stepper motors. It is commonly used in robotics, automation, and other applications that require motor control.



By properly configuring the logic control pins and adjusting the enable pins, you can control the direction and speed of the motors connected to the L298N. The IC handles the switching and control of the motor's power supply, allowing you to conveniently control the motors using a microcontroller or other control circuitry.

PID Controller

A PID controller, also known as proportional-integral-derivative controller, is a control algorithm to regulate a process or system. The PID controller combines three control actions:

1. Proportional (P) Action: The proportional term generates a control output that is directly proportional to the error. It responds to the present error and can help reduce the steady-state error. The proportional gain **Kp** determines the strength of the proportional action.

2. Integral (I) Action: The integral term takes into account the accumulated past errors over time. It helps in eliminating the steady-state error by integrating the error signal. The integral action continuously adjusts the control output based on the integral of the error. The integral gain **Ki** determines the strength of the integral action.

3. Derivative (D) Action: The derivative term considers the rate of change of the error signal. It anticipates the future trend of the error and provides a damping effect, reducing overshoot and improving system stability. The derivative gain **Kd** determines the strength of the derivative action.

Description of actions performed by Kp, Ki and kd is summarized in this table:

	Rise Time	Overshoot	Settling Time	Steady State Error
K_p	Decrease	Increase	Small Increase	Decrease
K_i	Small Decrease	Decrease	Decrease	Small Change
K_d	Small Decrease	Increase	Increase	Large Decrease

The control output of the PID controller is calculated as the sum of the proportional, integral, and derivative terms:

Control Output = (Kp * Error) + (Ki * Integral of Error) + (Kd * Derivative of Error)

Modeling

For the designing of motor controller we need mathematical model or Transfer Function of the motor which in general has this form.

$$G(s) = \frac{K}{s(\tau s + 1)}$$

Where τ is the time constant of the DC motor and K is the gain or motor's constant.

Motor Specifications:

1rev of Encoder=11 pulses
1rev of Motor=10 rev of Encoder
1rev of Motor=110 pulses
360 degree=110 pulses
1 degree=0.3066

We took data by changing the voltage on motor and observing the speed on oscilloscope. Then, we used **tfestimate** command of matlab for deriving the Transfer function of motor and controller as:

```
clear close all;
v=[0.8 1 1.2 1.4 1.6 1.8 2 2.2 2.4 2.6 2.8 3 3.2 3.4 3.6 3.8 4 4.2 4.4 4.6 4.8 5 5.2 5.4 5.6 5.8
6 6.2 6.4 6.6 6.8 7 7.2 7.4 7.6 7.8 8 8.2 8.4 8.6 8.8 9 9.2 9.4 9.6 9.8 10 10.2 10.4 10.6 10.8
11 11.2 11.4 11.6 11.8 12];
v=v';
f=[10.72 29.54 48.37 67.19 86.01 104.8 123.7 142.5 161.3 180.1 198.9 217.8 236.6 255.4
274.2 293.1 311.9 330.7 349.5 368.3 387.2 406 424.8 443.6 462.4 481.3 500.1 518.9 537.7
556.6 575.4 594.2 613 631.8 650.4 669.5 688.3 707.1 726 744.8 763.6 782.4 801.2 820.1
838.9 857.7 876.5 895.4 914.2 933 951.8 970.6 989.5 1008 1027 1046 1065];
f=f.*(60/100);
f=f';
z = iddata(f,v,0.2);
 sys = tfest(z,1)
```

Controller Transfer Function:

```
Tunable Block
Name: C
Sample Time: 0
Value:
  10000 (s+0.01584) (s+0.003156)
  ------------------------------
                s
```

Motor Transfer Function:

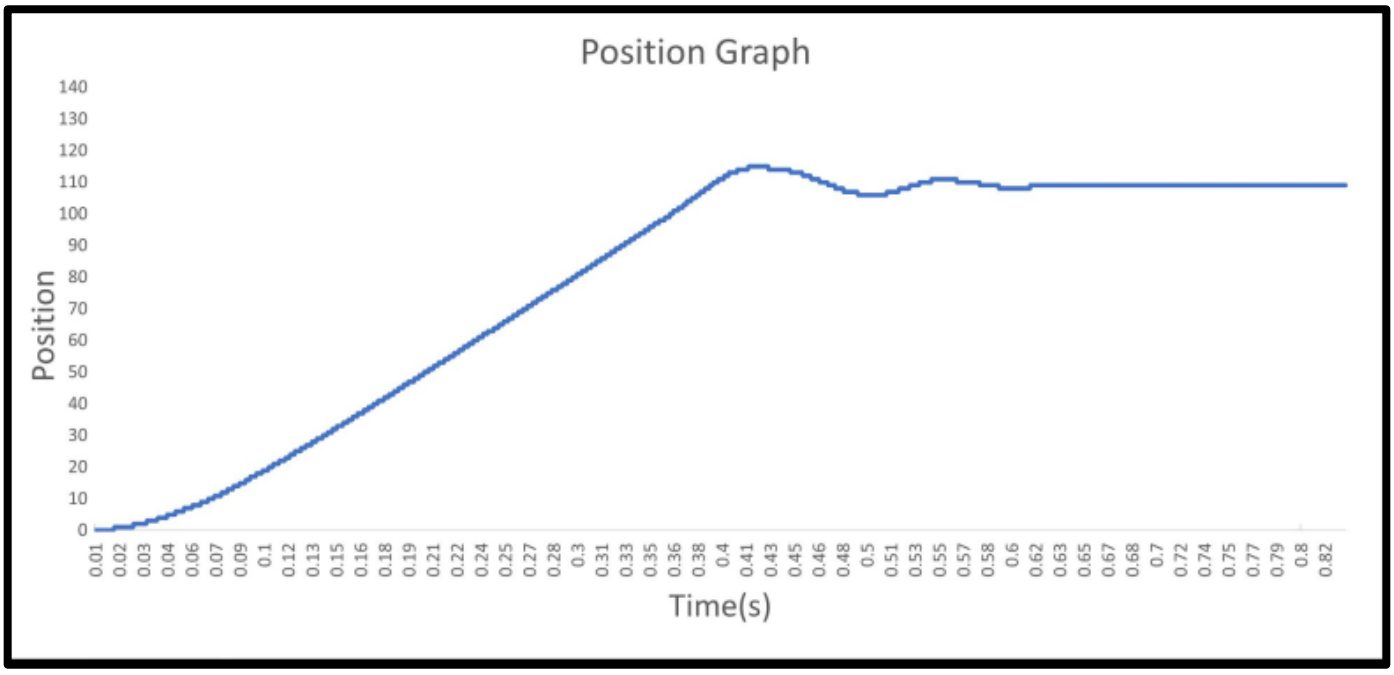
```
Fixed Block
Name: G
Sample Time: 0
Value:
          0.1228
  ----------------------
  s^2 + 9.667 s + 0.1228
```

Arduino Serial Monitor:

We used ardunio serial monitor data for the hardware and simulation tracking system. We online extract data from Arduino serial plotter and then plot it in Colab software and found following sketch:

A	B
2	151.86
2.5	214.5
3	239.04
3.5	284.34
4	322.26
4.5	368.1
5	405.36
5.5	454.5
6	504.18
6.5	540.54
7	588.18
7.5	603.6
8	690
8.5	738
9	768
9.5	834
10	870

This data is plotted in colab and obtained following graph;

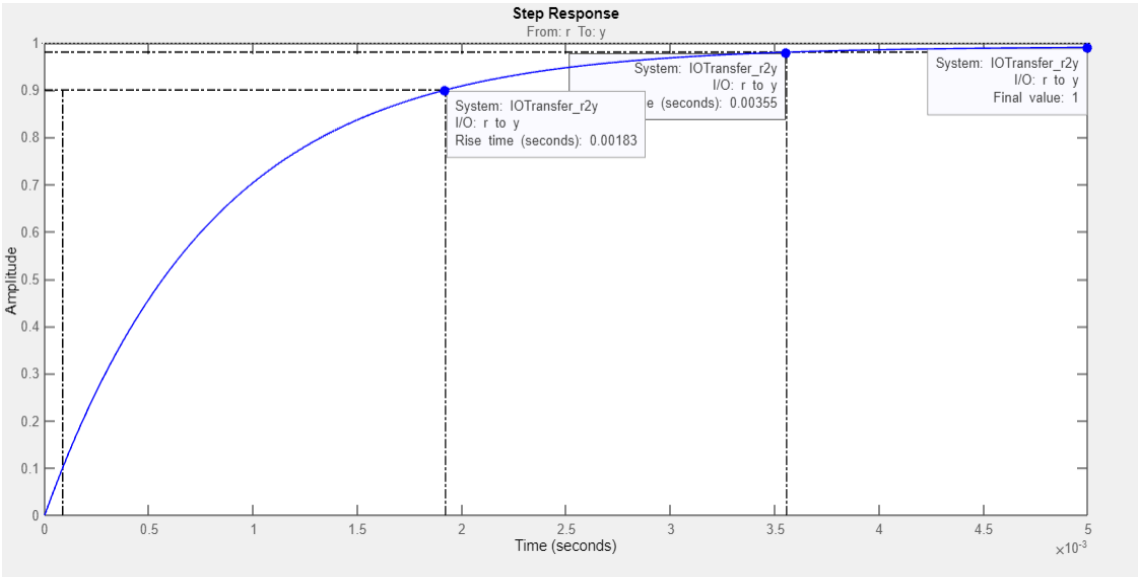


In above figure we have a little bit overshoot which we decreased by varying kp, ki and kd on hit and trial method.

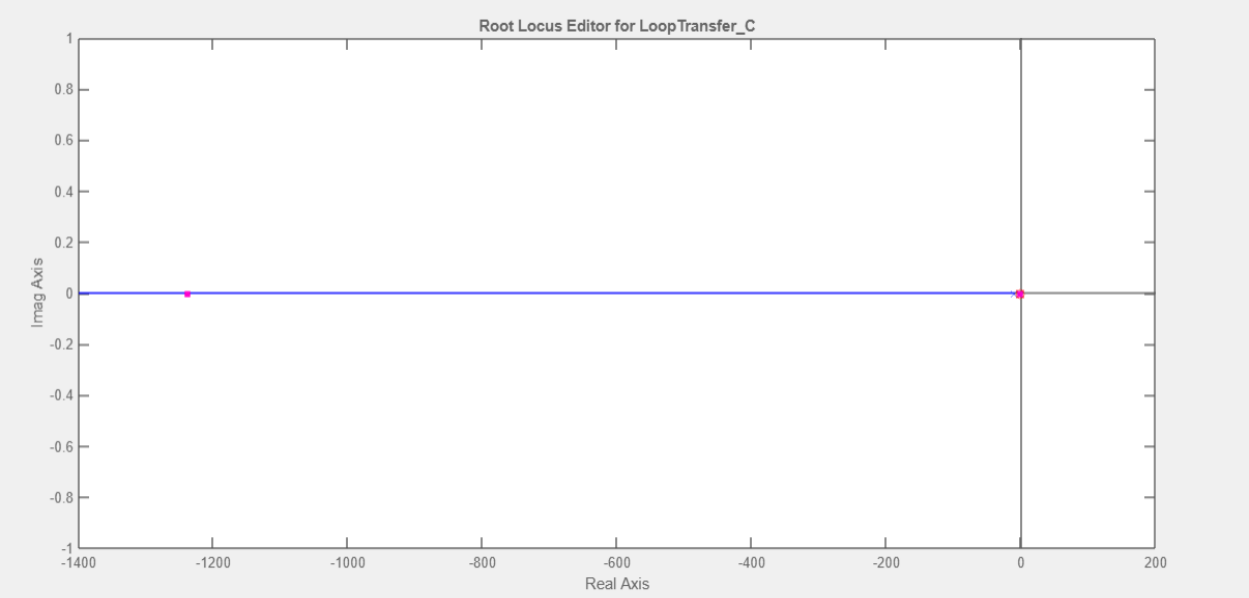
Matlab Verification:

At last we used matlab **Sisotool** for plotting our step response and root locus for closed loop system. We obtained following :

Step Response



Root Locus



Code

Including Libraires:

```
#include <util/atomic.h> // For the ATOMIC_BLOCK macro-> code will be excuted
automatically
```

Defining Pins:

```
#define ENCA 2 // Yellow
#define ENCB 3 // Sea Green
#define PWM 5  // from H Bridge
#define IN2 6  // from H Bridge -> Adjacent to PWM
#define IN1 7  // from H Bridge -> Adjacent to IN2
```

Initializing Variables:

```
volatile int posi = 0;
long prevT = 0;
float eprev = 0;
float eintegral = 0;
int angle=0;
String position;
```

```
float value;
```

Setup Function:

```
void setup() {
  Serial.begin(115200)
  pinMode(ENCA,INPUT);
  pinMode(ENCB,INPUT);
  attachInterrupt(digitalPinToInterrupt(ENCA),readEncoder,RISING);

  pinMode(PWM,OUTPUT);
  pinMode(IN1,OUTPUT);
  pinMode(IN2,OUTPUT);
}
```

Inside Loop Function:

- **Taking Value from Serial Port & setting Target:**

```
// Taking Value from Serial port of Arduino
    if (Serial.available()>0)
    {
      position = Serial.readStringUntil('\n');
        value = position.toFloat();
    }
    //angle=180;
    float target=(0.3055*value);
```

- **Control Signal:**

```
 float u = kp*e + kd*derivative + ki*eintegral;
```

- **Limition PWM value:**

```
float power = fabs(u);  // Take the absolute value of 'u' and assign it to 'power'
if (power > 255) {     // Check if 'powerr' is greater than 255
  power = 255;         // Limit 'power' to 255 if it exceeds that value
}
```

- **Motor Control:**

```
// Function for setting motor specs
void setMotor(int dir, int pwmVal, int pwm, int in1, int in2){
  analogWrite(pwm,pwmVal);
```

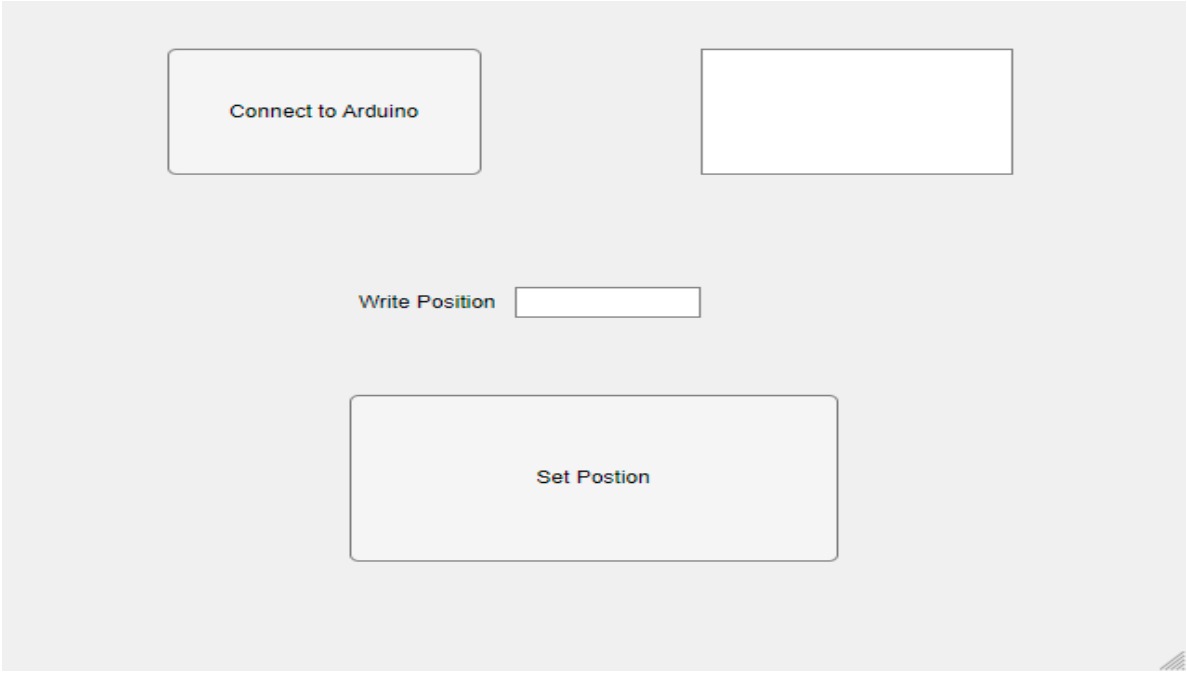
```
  if(dir == 1){
    digitalWrite(in1,HIGH);
    digitalWrite(in2,LOW);
  }
  else if(dir == -1){
    digitalWrite(in1,LOW);
    digitalWrite(in2,HIGH);
  }
  else{
    digitalWrite(in1,LOW);
    digitalWrite(in2,LOW);
  }
}
```

- **Position Calculation:**

```
// Reading Encoder Value fOR Position Calculation
void readEncoder(){
  int b = digitalRead(ENCB);
  if(b > 0){
    posi++;
  }
  else{
    posi--;
  }
}
```

Graphical User interface & Matlab

For the purpose of interfacing Arduino with Matlab we used serial communication method and for GUI we used App designer in Matlab.



1) First of all, we will make connection of Matlab and Arduino by pressing "Connect to Arduino" push Button.
2) After Pressing this Button following callback will be executed.

```
app.arduino= serialport("COM6", 115200);
             configureTerminator(app.arduino, "LF");
             flush(app.arduino);
             if isvalid(app.arduino)
                 app.EditField.Value = "                connected";
             else
                 app.EditField.Value = "                 Error";
             end
```

This will give the status of connection wether connected or not connected in front edit filed box.

3) After making connection we will write angle that we want from the motor to turn in "Write Position" Box.
4) Then, we will press button "Set Position" and following command will be executed.

```
position = app.WritePositionEditField.Value;
writeline(app.arduino, position);



% Wait for a moment to ensure Arduino processes the data
pause(2);

% Close the serial connection
clear app.arduino;
delete app.arduino;
```

In Arduino code that was already burnt in hardware we will be searching for any data available on serial port and will fetch that data and give it as a target to motor.

```
// Taking Value from Serial port of Arduino
    if (Serial.available()>0)
    {
      position = Serial.readStringUntil('\n');
        value = position.toFloat();
    }
    //angle=180;
    float target=(0.3055*value);
```

Hardware Implementation

Closed loop system of our project is like the following block diagram:

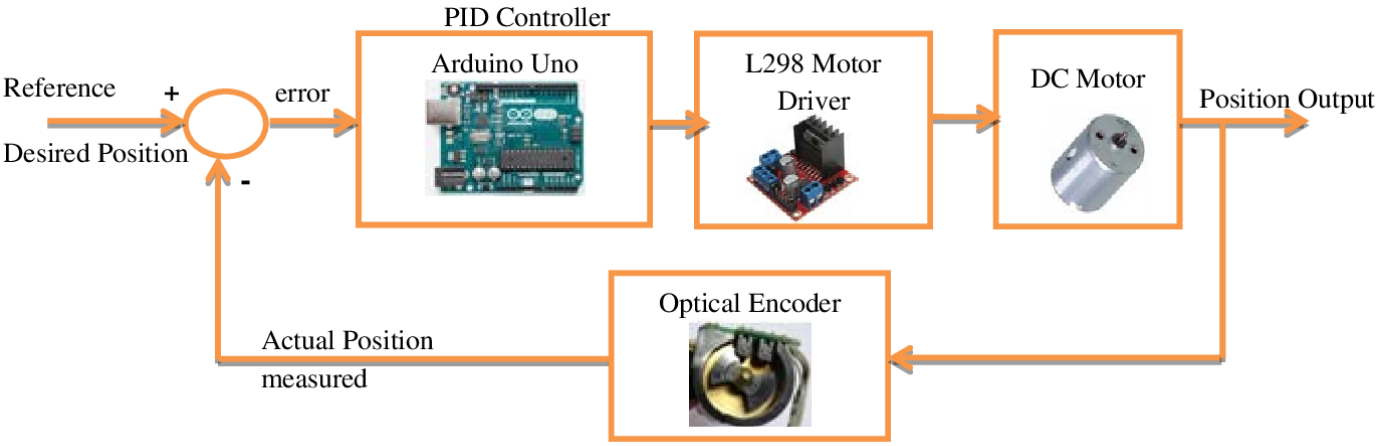


Figure 1: Block Diagram for DC Motor Position Control

We made the connections as follows:

- Arduino pin 2 & 3 is connected with Encoder of the motor.
- Pin 5 is connected with PWM pin of Driver IC.
- Pin 6 & 7 also connected to Driver IC for direction control of motor.
- We are providing 5V and ground to Encoder power pins.
- And motor power pins are attached to output of Driver IC.

Conclusion

In this project we successfully designed a closed loop system for DC motor position control. We successfully designed controller for this system and also verified it on Hardware as well. We are achieving high accuracy in hardware.