Mechatronics Lab
Final Project report
Closed loop position and speed control of DC motor using PID controller.

Purvang lapsiwala UIN:662689378

Date: May, 13, 2018

Contents

1 Summary

In this experiment we learnt to build a close loop control of the DC Motor using the angular feedback.The feedback is then used for position and speed control. We learnt the principle of the incremental encoders and the function of the opto-interrupter. We fabricated a low resolution incremental encoder (4 slits) and 3D Printed the platform for the entire setup. The objective of this experiment is to project is the following

- Build the encoder with phase shift of 90deg.
- Get angular feedback using interrupt (pin interrupt)
- Use angular feedback to achieve the position control.
- Use position feedback to control angular velocity of the motor.
- use PID as the controller. Tune the PID gains to get best results.

2 List of Components

1. DC Motor
2. Opto-Interruptor-2
3. Encoder (3-D printed)
4. Disk with 4 holes
5. Potentiometer
6. Diodes
7. Arduino
8. Jumper Wires
9. Mosfets (510 and 9520)
10. Bread Board

3 Description of the experiment

3.1 Purpose and Objective

Purpose of the experiment is to understand the Operating principle of opto-interrupter and encoder and its closed loop control using the micro-controller. Second part of the experiment to control the speed and velocity of the motor using the PID controller.

3.2 Theory

3.2.1 Encoders

The main part of building a close loop control system includes building the position sensor. The position sensor is built using an incremental encoder and hence understanding the principle behind is very important part. The incremental encoder can be constructed at home. It requires a disk which has four holes. For a higher resolution the number of holes can be increased but for a home made fabrication a four hole disk can be used as it is mechanically difficult to assemble. The disk is usually made of plastic and is opaque, i.e its main purpose is to obstruct the light to enter. The light only passes through the holes on the disks.

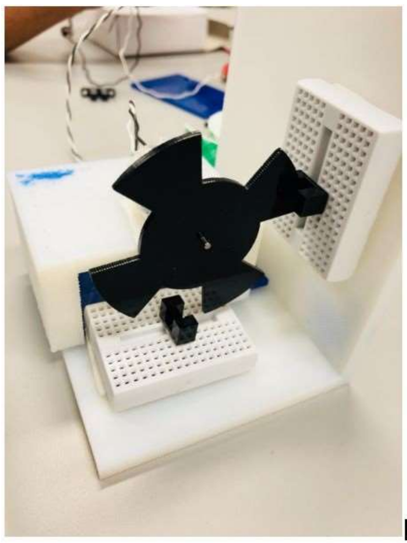


Figure 1: Actual encoder setup with 3D printed stand

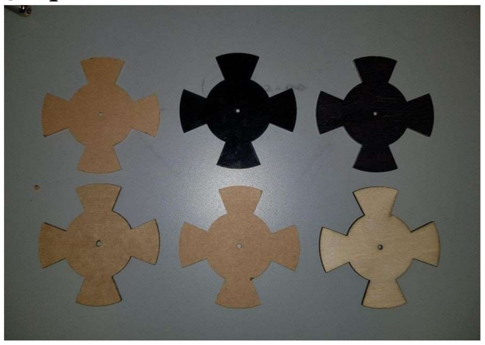


Figure 2: Different configuration of the encoders used.

3.2.2 opto interrupter

two opto-interrupters as placed at 90 degrees phase difference and the disk is setup on the platform such that it passes through two opto-interrupters. When the disk rotates, the light gets interrupted by the opaque solid section and the light passes only through those holes. By using one opto-interrupter the position can be detected but the direction or the speed of the rotating disk cannot be detected. Hence the second opto-interrupter is introduced and is placed at a mechanical 90 degrees phase angle relative to the first opto-interrupter over the disk as shown in the below picture. Here 360 degrees is considered to be one cycle. If the disk is rotating in the clockwise direction the digital obtained from the opto-interrupter-1 would lead by 90 degrees by the signal from the second opto-interrupter. And if the rotation is anti-clockwise, the second signal leads by 90 degrees to the first signal.

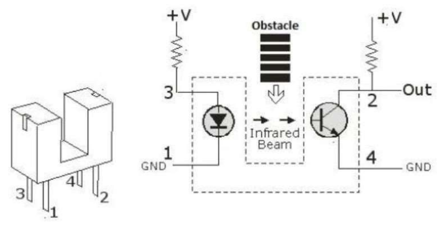


Figure 3: opto-interrupter circuit

3.2.3 H-bridge

1. The H-bridge is constructed using the power transistors IRF 9520 (used as p-channel) and IRF 511 (used as n-channel), four diodes which are connected from d-pin to s-pin in the transistors. The motor is connected to the D-pin of the transistors. The source terminals of the p-channel are connected to a 9V power supply and the source terminals of n-channel are connected to the ground.

2. The two PWM signals from the Arduino powers up the transistor, where one PWM signal activates top left and bottom right pair of transistors and other pair of transistors are activated by the second PWM signal.

3. The two transistors on one side of the circuit must never be turned on at the same time since it may cause short circuit which eventually destroys the transistors.

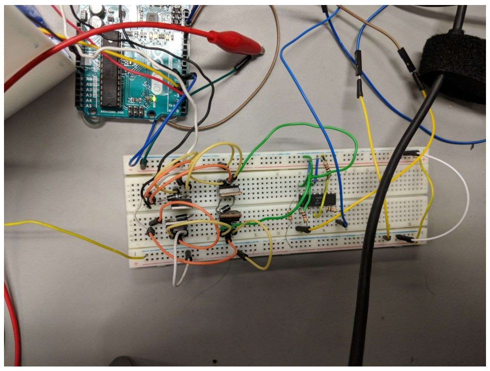


Figure 4: H-Bridge actual circuit

4 Circuits Diagrams and Images

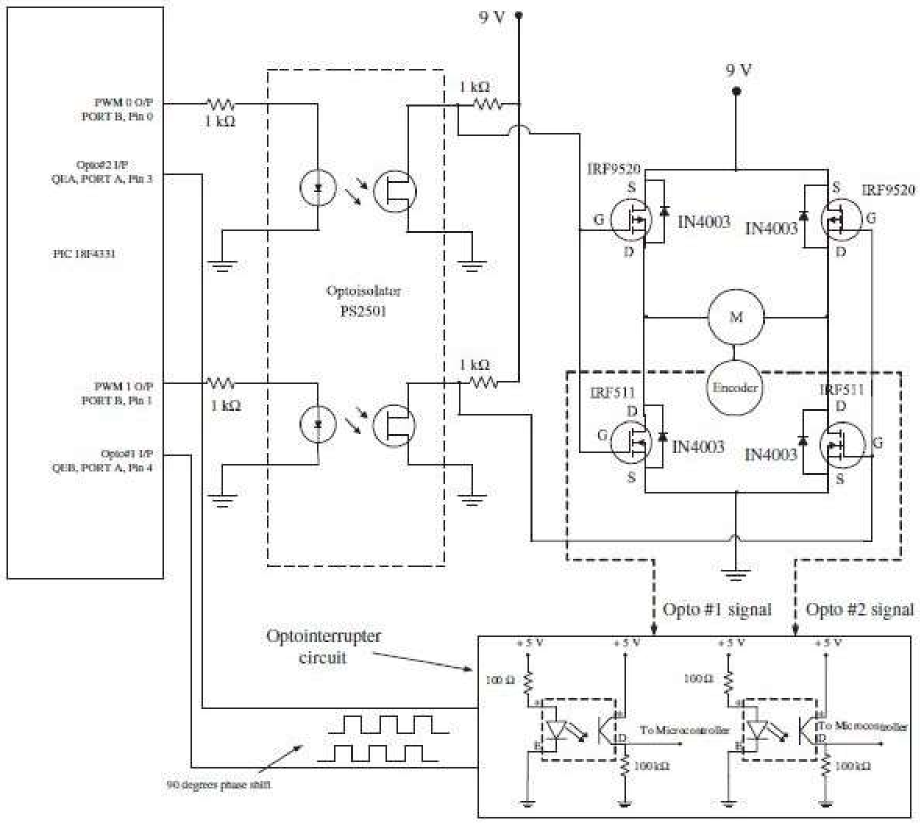


Figure 5: Complete circuit diagram

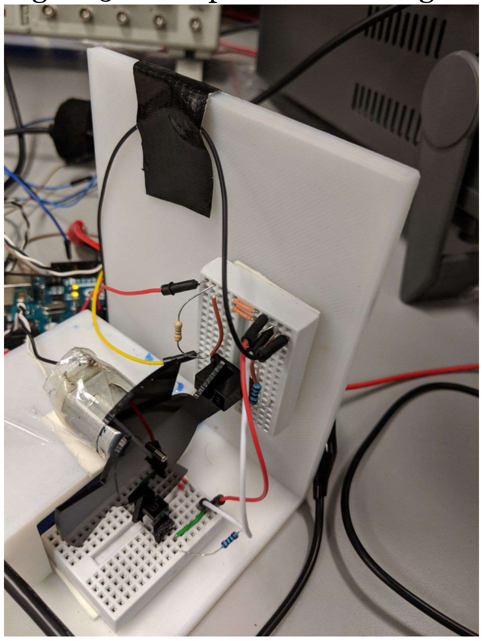
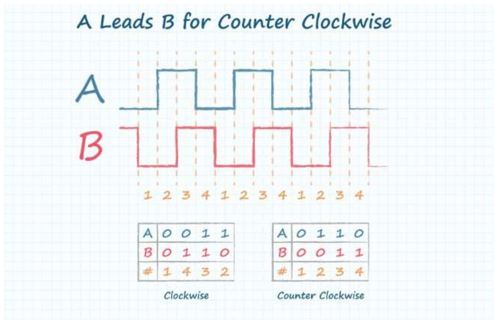


Figure 6: Sensor and encoders

5 Encoders working



A	0	0	1	1
B	0	1	1	0
#	1	4	3	2

A	0	1	1	0
B	0	0	1	1
#	1	2	3	4

Figure 7: Encoder working logic

Encoders are in 90 degrees out of phase because of placement of the opto-interrupter. Here each combination of the A and B is considered as a State.
The states were stored and compared with the previous state. Based on the previous state the direction of the state is decided. This means that encoder ISR will be called every time there is change in the A value and change count will give the angle change. when going in CCW the count value will decrease.

6 Procedure

1. An L-shaped platform is setup using the 3D printed L-clamp and a base.
2. The opto-interruptors are now placed at 90 degrees to each other to facilitate the reading of speed and direction of the motor.
3. This whole setup is connected to the main circuit which consists of H-bridge (to control the DC motor) constructed using transistors and an opto-isolator is used to isolate the Arduino from power fluctuations in the circuit.
4. Next step is Programming the logic and extracting in the distance from the opto-interrupter signals(digital pins 2 and 3). For this step we Interrupt method and switch case inside in the ISR. Alternatively nested IF can be used but the if condition gets complicated.
5. Error value is calculated using the desired and actual angular value (actual values are calculated from the encoder ticks).
6. PID controller function was created and inputs to the block is error and output of this is mapped into -255 to 255. Because the PWM output in 8-bit.
7. PID gains are tunned to get the least error in the steady state.
8. **Angular velocity** similar in encoder ticks must be used to set the angular velocity. But since the arduino loop time is very small and the error of angular velocity will always be very low. To overcome this the signal is sampled at a large sampling rate using the 'if' condition and flags.
9. After calculating the error procedure is similar to the angular control.

NOTE: Here the encoders have 4 slits and count is taken change in signal. Hence the resolution of the encoders will be $resultion = \frac{360}{8}$

7 Results and Code

7.1 position Control Code

```
// Initialising all the parameters
int count = 0;
int last_count;
float theta = 0;
int outputA, outputB;
float theta_desired = -720;
float Kp = 0.4;
float  Kd  =  0.25;
float Ki  =  0.001;
float last_error = 0;
float  error;
float total_error = 0;
float P_control, I_control, D_control;
float U_control;
float Motor_Power;
int  state;
int last_state;

// Inital setup
void setup()
{
   Serial.begin(9600);
   // Rotary encoder pins
   pinMode(3, INPUT);
   pinMode(4, INPUT);
   attachInterrupt(digitalPinToInterrupt(3), Encoder_Reading_A,  CHANGE);
   // Motor PWM pins
   pinMode(5, OUTPUT);
   pinMode(6,  OUTPUT);
}

// Loop
void  loop()
{
   Encoder_Reading_A();
   Angle();
   Controller();
   Motor_Control();
   Print_Data();
}

// Controller
void Controller()
{
   error = theta_desired - theta;
   P_control = Kp * error;
   D_control = Kd * (error - last_error);
   total_error = total_error + error;
   if(total_error > 255)
   {
    // AntiWind Up control
      total_error = 0;
   }
   I_control = Ki * total_error;
   U_control = P_control + I_control + D_control;
   U_control = constrain(U_control, -255, 255);
   last_error = error;
}

// Calculation of Angle using encoder reading /needs to be scaled/
void Angle()
{
   // Rise in reading means one blade has cut the opto isolator
   // once hence the blade has moved 45 degrees
   theta = theta + ((count - last_count) *  45);
}

// Motor Control
void  Motor_Control()
{
   if(U_control  >=  0)
   {
      // Indicates error is positive
      Motor_Power = floor(U_control);
      analogWrite(5, Motor_Power);
      analogWrite(6, 0);
   }
   else
   {
      // Indicates error is negative i.e. we over shoot
      Motor_Power = -1*floor(U_control);
      analogWrite(5, 0);
      analogWrite(6, Motor_Power);
   }
}

// Encoder  Reading
void Encoder_Reading_A()
{
   last_count =  count;
   outputA = digitalRead(3);
   outputB =  digitalRead(4);
   if(outputA == HIGH && outputB == HIGH)
   {
      state = 1;
   }
   if(outputA == HIGH && outputB == LOW)
```

```
{
  state = 2;
}
if(outputA == LOW && outputB == HIGH)
{
  state = 3;
}
if(outputA == LOW && outputB == LOW)
{
  state = 4;
}
switch(state)
{
  case 1:
  {
    if(last_state == 3)
    {
      count++;
    }
    if(last_state == 2)
    {
      count--;
    }
    break;
  }
  case 2:
  {
    if(last_state == 1)
    {
      count++;
    }
    if(last_state == 4)
    {
      count--;
    }
    break;
  }
  case 3:
  {
    if(last_state == 4)
    {
      count++;
    }
    if(last_state == 1)
    {
      count--;
    }
    break;
  }
  case 4:
    {
      if(last_state == 2)
      {
        count++;
      }
      if(last_state == 3)
      {
        count--;
      }
      break;
    }
  }
  last_state = state;
}

// Print all data
void Print_Data()
{
//   Serial.println("Angle \t");
//   Serial.println(theta);
//   Serial.println("\t P_control \t");
//   Serial.println(P_control);

//   Serial.println("D_control \t");
//   Serial.println(D_control);
//   Serial.println("I_control \t");
//   Serial.println(I_control);

//   Serial.println("U_control \t");
//   Serial.println(U_control);
 //Serial.println("Error \t");
     Serial.println(error);
//
//   Serial.println("Count \t");
//   Serial.println(count);
//   Serial.println("Motor Power \t");
//   Serial.println(Motor_Power);
}
```

7.2 Velocity control code

```
// Initialising all the parameters
float old_time_countangle ;
int count = 0;
int i =0;
int last_count;
float theta = 0;
int outputA, outputB;
float angular_vel_desired = 200;
float Kp = 1;
float  Kd  =  0.25;
float Ki  =  0.001;
float last_error = 0;
float  error;
float total_error = 0;
float P_control, I_control, D_control;
float U_control;
float Motor_Power;
int  state;
int last_state;
float delta_time;
float t;
float last_theta = 0;
float angular_vel;
float countangle;
float counttime;
#define LOOPTIME 1000
float last_t =  0;
float old_time_counttime;
// Inital setup
void setup()
{
   Serial.begin(9600);
   // Rotary encoder pins
   pinMode(3, INPUT);
   pinMode(4, INPUT);
   attachInterrupt(digitalPinToInterrupt(3), Encoder_Reading_A,  CHANGE);
   // Motor PWM pins
   pinMode(5, OUTPUT);
   pinMode(6,  OUTPUT);
}

// Loop
void  loop()
{
   Encoder_Reading_A();
    if((millis()-last_t) >= LOOPTIME)
   {                                                                                    //
      last_t = millis();
      Angular_Velocity();                                                            // calc
   }
   //Angular_Velocity();
   Controller();
   Motor_Control();
   Print_Data();
}

// Controller
void Controller()
{
   error = angular_vel_desired - angular_vel;
   P_control = Kp * error;
   D_control = Kd * (error - last_error);
   total_error = total_error + error;
   if(total_error > 255)
   {
      // AntiWind Up control
      total_error = 0;
   }
   I_control = Ki * total_error;
   U_control = P_control + I_control + D_control;
   U_control = constrain(U_control, -255, 255);
   last_error = error;
}

// Calculation of Angle using encoder reading /needs to be scaled/
void Angular_Velocity()
{
   // Rise in reading means one blade has cut the opto isolator
   // once hence the blade has moved 45 degrees
//   delta_time = millis() - t;
//   t = t + delta_time;
   angular_vel = ((count - last_count)*(60*(1000/LOOPTIME)));
}

// Motor Control
void  Motor_Control()
{
   if(U_control  >=  0)
   {
      // Indicates error is positive
      Motor_Power = floor(U_control);
      analogWrite(5, Motor_Power);
      analogWrite(6, 0);
   }
   else
   {
      // Indicates error is negative i.e. we over shoot
      Motor_Power = -1*floor(U_control);
      analogWrite(5, 0);
      analogWrite(6, Motor_Power);
   }
}

// Encoder  Reading
void Encoder_Reading_A()
{
   last_count =  count;
   outputA = digitalRead(3);
   outputB =  digitalRead(4);
```

```
if(outputA == HIGH && outputB == HIGH)
{
   state = 1;
}
if(outputA == HIGH && outputB == LOW)
{
   state = 2;
}
if(outputA == LOW && outputB == HIGH)
{
   state = 3;
}
if(outputA == LOW && outputB == LOW)
{
   state = 4;
}
switch(state)
{
   case 1:
   {
      if(last_state == 3)
      {
         count++;
      }
      if(last_state == 2)
      {
         count--;
      }
      break;
   }
   case 2:
   {
      if(last_state == 1)
      {
         count++;
      }
      if(last_state == 4)
      {
         count--;
      }
      break;
   }
   case 3:
   {
      if(last_state == 4)
      {
         count++;
      }
      if(last_state == 1)
      {
          count--;
        }
        break;
      }
      case 4:
      {
        if(last_state == 2)
        {
          count++;
        }
        if(last_state == 3)
        {
          count--;
        }
        break;
      }
  }
  last_state = state;
}

// Print all data
void Print_Data()
{
  Serial.println("Delta time");
  Serial.println(delta_time);
  Serial.println("Theta ");
  Serial.println(theta);
//   Serial.println("\t P_control \t");
//   Serial.println(P_control);

//   Serial.println("D_control \t");
//   Serial.println(D_control);
//   Serial.println("\t I_control \t");
//   Serial.println(I_control);

//   Serial.print("\t U_control \t");
//   Serial.print(U_control);
//   Serial.println("Error \t");
//   Serial.println(error);
//
//   Serial.println("Count \t");
//   Serial.println(count);
//   Serial.println("Motor Power \t");
//   Serial.println(Motor_Power);
}
```

8 Conclusion

In Conclusion, in this experiment we understood the basic working principle of the DC speed and position control and also we create encoders with phase swift to recognize not only angle but the direction of movement.Using the interrupt the angle was calculated and the using the backward approximation method angular velocity was calculated. In the next part of experiment we implemented the PID algorithm and then tunned gains. We have also demonstrated the overshoot in the system in position control.

9 Future scope and improvements

Here we have 4 slits with more slits and optointerrupts there resolution and controllability will be improved. Lighter material can be used to for encoder.

10 Videos,Results images and Pseudo code

Please visithereor visit the pagehttps://photos.app.goo.gl/rZrAzSsHqVdT3cY72

11 References

1.Lab manual.