제어시스템 모델링

[실습2-2 PD_control]

이름 : 김 용 현

학번 : 2017006262

실습이론

2차 표준형 시스템(2차 전달함수)

$$\frac{\dot{\theta}_m(s)}{\dot{\theta}_d(s)} = \frac{\frac{A k_p k_t}{R_a J}}{s^2 + \frac{A k_p k_t + k_e k_t}{R_a J} s + \frac{A k_p k_T}{R_a J}}$$

Frequency domain to Time domain

$$U(z) = K_D \frac{1-z^{-1}}{T} \theta_m(z) + k_p E(z)$$

$$= \frac{K_D}{T} \theta_m(z) - \frac{K_D}{T} z^{-1} \theta_m(z) + K_p E(z)$$

} frequency domain

$$\Downarrow$$

$$u(n) = \frac{K_D}{T} \theta_m(n) - \frac{K_D}{T} \theta_m(n-1) + k_p e(n)$$ Time domain

실습에서는 time domain을 이용하여 PI-control을 진행한다.

$$u(t) = k_p e(t) + k_D \frac{de(t)}{dt}$$

$$\frac{U(s)}{E(s)} = k_p + k_D s = k_p + K_D \frac{1-z^{-1}}{T}$$

$$U(z) = k_p E(z) + \frac{K_D}{T} E(z) - \frac{K_D}{T} z^{-1} E(z)$$

$$\Rightarrow u(n) = k_p e(n) + \frac{k_D}{T} e(n) - \frac{K_D}{T} e(n-1)$$

E(n) = error, E(n-1) = prev_error

실습코드(아두이노)

```
#include <SSD1306.h>
#include <MsTimer2.h>
// OLED Setup
#define OLED_DC 5
#define OLED_CLK 8
#define OLED_MOSI 7
#define OLED_RESET 6
#define PI 3.141592
SSD1306 oled(OLED_MOSI, OLED_CLK, OLED_DC, OLED_RESET, 0);

//MOTOR driver pin
#define PWM 9
#define IN1 10
#define IN2 11

//Encoder Pin
#define ENCODER_A 3
#define ENCODER_B 2

uint32_t oled_pow(uint8_t m,uint8_t n)
{
  uint32_t result=1;
  while(n--)result*=m;
  return result;
```

```
}
void OLED_ShowNumber(uint8_t x,uint8_t y,uint32_t num,uint8_t
len)
{
   u8 t,temp;
   u8 enshow=0;
   oled.drawchar(x-6,y,' ');

   for(t=0;t<len;t++)
   {
      temp=(num/oled_pow(10,len-t-1))%10;
      oled.drawchar(x+6*t,y,temp+'0');
   }
}
void OLED_ShowNumber_Minus(uint8_t x,uint8_t y,uint32_t
num,uint8_t len)
{
   u8 t,temp;
   u8 enshow=0;
   oled.drawchar(x-6,y,'-');

   for(t=0;t<len;t++)
   {
      temp=(num/oled_pow(10,len-t-1))%10;
      oled.drawchar(x+6*t,y,temp+'0');
   }
```

```
}

void Set_PWM(int PWM_value)
{
  if (PWM_value< 0) digitalWrite(IN1, HIGH),
digitalWrite(IN2, LOW);
  else digitalWrite(IN1, LOW),
digitalWrite(IN2, HIGH);
  analogWrite(PWM, abs(PWM_value));
}

float t = 0;
float t_step = 0.001;

int cart_encoder=0;
float target_cart_position = 1;
float prev_cart_position = 0;
float prev_cart_velocity = 0;
float error = 0;
float prev_error=0;
float enc2rad = 1/520.0*PI;

float Kp = 2138.1;
float Kd = 40.4914;

void control(){
```

```
      float cart_position = float(cart_encoder)*enc2rad;
      float cart_velocity = (cart_position-prev_cart_position)/t_step;


    // Fill in the Missing Code : HINT) error, u
    float error = target_cart_position - cart_position;
    float u = (Kp * error) + (Kd / t_step * error) - (Kd / t_step * prev_error) ;
    //


      prev_cart_position = cart_position;
      prev_cart_velocity = cart_velocity;
      prev_error = error;



      if(abs(u)>=255){
         if(u>=0)
               u = 255;
         else
               u = -255;
      }
      Set_PWM(u);
      t +=t_step;
```

```
      byte * time_ = (byte *) &t;
      Serial.write(time_,4);
      byte * b = (byte *) &cart_position;
      Serial.write(b,4);
}


void setup() {
  // OLED SETUP
  int fff = 1;
  TCCR1B =(TCCR1B & 0xF8) | fff;

   oled.ssd1306_init(SSD1306_SWITCHCAPVCC);
   oled.clear();     // clears the screen and buffer

   //motor driver setup
   pinMode(IN1, OUTPUT);
   pinMode(IN2, OUTPUT);
   pinMode(PWM, OUTPUT);

   //Encoder Setup
   pinMode(ENCODER_A, INPUT);
   pinMode(ENCODER_B, INPUT);

  Serial.begin(2000000);
```

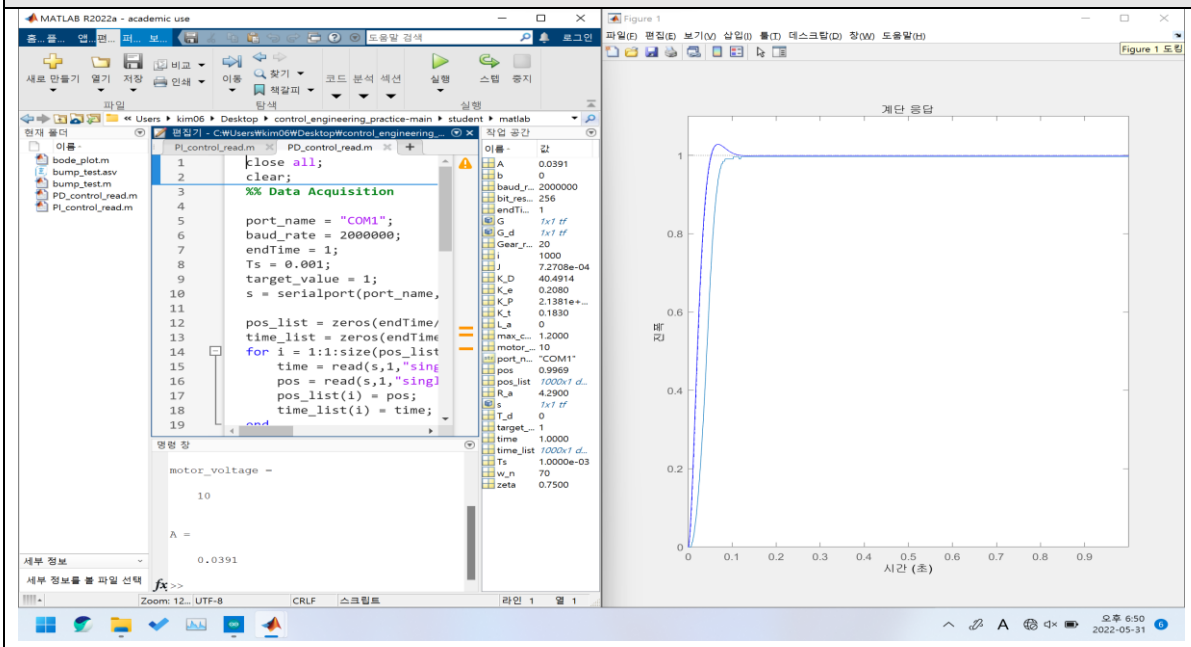
```
  delay(200);
  MsTimer2::set(1, control);
  MsTimer2::start();
  attachInterrupt(0, doEncoderA, CHANGE);
  attachInterrupt(1, doEncoderB, CHANGE);

  // motor stop
  digitalWrite(IN1, 0);
  digitalWrite(IN2, 0);
  digitalWrite(PWM, 0);

  oled.drawstring(00,1,"=====PD_CONTROL=====");
  oled.display();
}
void loop() {

}
void doEncoderA(){cart_encoder +=
(digitalRead(ENCODER_A)==digitalRead(ENCODER_B))?1:-1;}
void doEncoderB(){cart_encoder +=
(digitalRead(ENCODER_A)==digitalRead(ENCODER_B))?-1:1;}
```

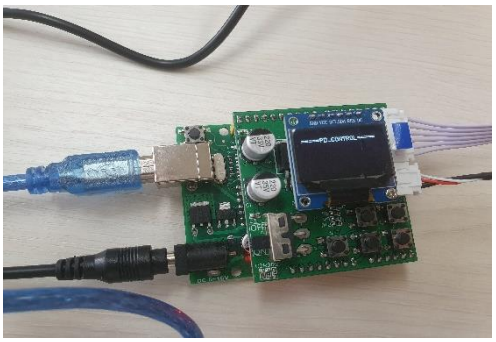
실습코드(matlab)

```
close all;
clear;
%% Data Acquisition

port_name = "COM1";
baud_rate = 2000000;
endTime = 2;
Ts = 0.001;
target_value = 20;
s = serialport(port_name,baud_rate);

time_list = zeros(endTime/Ts,1);
vel_list = zeros(endTime/Ts,1);
filtered_vel=0;
for i = 1:1:size(time_list)
    time = read(s,1,"single");
    vel = read(s,1,"single");
    vel_list(i) = vel;
    time_list(i) = time;
end
s.delete
windowSize = 50;
b = (1/windowSize)*ones(1,windowSize);
a = 1;
plot(time_list,filter(b,a,vel_list./target_value));
```

```
hold on;
T_d = 0; % 부하
R_a=4.29; % 전기자 저항
L_a = 0; % 전기자 인덕턴스
b = 0; % 모터 점성마찰계수
K_t=0.183; % 토크 상수
K_e=0.208; % back emf 상수
Gear_ratio = 20; % 기어비
max_current = 1.2 % 모터드라이버최대출력전류 A
motor_voltage = 10 % 모터 최대 전압
bit_resolution = 2^8; % PWM 출력 0~255 8bit
J=7.083e-06+1.8e-06*Gear_ratio^2;
A = motor_voltage/bit_resolution
K = 1/K_e % 1차 표준형 시스템 계수
tau = J*R_a/(K_t*K_e);% 1차 표준형 시스템 시상수
zeta = 0.75; % 감쇠비
w_n = 70; % 고유진동수
s = tf('s');
K_I= ((w_n^2*R_a*J)/(A*K_t)); % 이 부분을 알맞게 수정하시오
K_P= ((2*zeta*w_n*R_a*J-K_e*K_t)/(A*K_t));
G = (w_n^2/(s^2 + (2*zeta*w_n*s) + w_n^2));
G_d = c2d(G,0.001,'zoh');
step(G,'-',G_d,'--')
axis([0 time_list(end) 0 max(vel_list./target_value)])
```

결론





아두이노 코드를 실행하면 PD_control이라는 화면이 출력되면서 모터가 조금 돌아간 후 고정된다.

실습 2-1 PI_control에서 2차 표준형시스템, time domain의 변화에 맞춰 코드를 수정하여 적용하여 실습을 진행하였다.

실습을 진행하는 과정에서 Float가 아닌 int형을 잘못 사용하여 어려움이 있었다.

실습 2-1과 거의 비슷한 실습이어서 큰 어려움은 없었다. 이번 실험을 통해 2차 표준형시스템(2차 시스템 전달함수)을 더 익숙하게 다루게 되는 계기가 되었다.

2-1과 다르게 이번 실습에서는 그래프가 1에서 고정된 모습을 보아 target position인 1에서 고정되어 이전과 같은 떨림 현상이 없었던 것 같다.