제어시스템 모델링

[실습2-1 PI\_control]

이름 : 김 용 현

학번 : 2017006262

|  |
| --- |
| 실습이론 |
| 2차 표준형 시스템(2차 전달함수)  Frequency domain to Time domain  실습에서는 time domain을 이용하여 PI-control을 진행한다.  U(n) = u, U(n-1) = prev\_u  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  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;  float End\_Time = 2;  int cart\_encoder=0;  float target\_cart\_velocity = 20;  float prev\_cart\_position = 0;  float prev\_cart\_velocity = 0;  float error = 0;  float prev\_error=0;  float alpha = 0.1;  float enc2rad = 1/520.0\*3.141592;  float prev\_u= 0;  float Ki = 2138.091260222951;  float Kp = 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\_velocity - cart\_velocity;  float u = prev\_u + (Kp + t\_step \* Ki) \* error - Kp \* prev\_error ;    //u = prev\_u + (Kp + t\_step \* Ki) \* error - Kp \* prev\_error ;    prev\_cart\_position = cart\_position;  prev\_cart\_velocity = cart\_velocity;  prev\_error = error;  prev\_u = u;    if(abs(u)>=255){  if(u>=0)  u = 255;  else  u = -255;  }  if(abs(u)<=10)  u = 0;    if(t >=End\_Time){  target\_cart\_velocity = 0;  Set\_PWM(0);  }else{  Set\_PWM(u);  }  t +=t\_step;  byte \* time\_ = (byte \*) &t;  Serial.write(time\_,4);  byte \* b = (byte \*) &cart\_velocity;  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);  // motor stop  digitalWrite(IN1, 0);  digitalWrite(IN2, 0);  digitalWrite(PWM, 0);    oled.drawstring(00,1,"=====PI\_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; % 동기 폴리가 모터 샤프트를 중심으로 회전할 때의 관성 모먼트  A = motor\_voltage/bit\_resolution % 모터 전압 / 모터 드라이버 최대출력전류/PWM 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)); % 이 부분을 알맞게 수정하시오  G = (w\_n^2/(s^2 + (2\*zeta\*w\_n\*s) + w\_n^2)); % 이 부분을 알맞게 수정하시오  G\_d = c2d(G,0.001,'zoh');  step(G,'-',G\_d,'--')  axis([0 time\_list(end) 0 max(vel\_list./target\_value)]) |

|  |
| --- |
| 결론 |
| 아두이노 코드 실행 시 PI\_CONTROL  화면 출력과 함께 모터가 돌아간다.  Matlab을 이용하여 구한 Kp, Ki값을 코드에 대입하여 작동 시 다음과 같은 그래프를 얻을 수 있었다.  전달함수 G에는 2차 표준형 시스템 함수를 대입하여 그래프를 그렸다.  실습 초반에는 Kp, Ki값을 matlab을 이용하여 값을 구한 후 대입해야 한다는 점을 몰라 어려움이 있었으나 이후 감쇠비, 고유 진동수 등을 입력하여 두 값을 구해 실습을 완료할 수 있었다.  본 실험을 통해 2차 표준형 시스템 함수를 활용해 볼 수 있는 기회가 되었다. 0.2초부터 0.6초 간격으로 비슷한 파형의 그래프가 보이는데, u가 10 미만일 경우 모터가 동작하지 않아 발생하는 오차로 보인다. |