제어시스템 모델링

[실습1-3 Bode Plot]

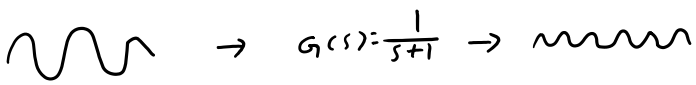
이름 : 김 용 현

학번 : 2017006262

실습이론

Bode Plot

시스템 입력 주파수를 증가시켜 출력 주파수의 크기 비율을 Logscale로 그린 그래프를 bode plot이다.



실습에서 주파수를 바꾸면서 그래프를 그리지 않고 chirp signal 사용.

*chirp signal: 주파수가 시간에 따라 선형적으로 변하는 신호.

$x(t) = A\sin(w_0 t + \theta_0)$, $y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$

$= A|H(jw)|\sin(w_0 t + \angle H(jw))$

$f(t) = ct + f_0 \quad c = \frac{f_T - f_0}{T}$

$x(t) = \sin\left[\phi_0 + 2\pi\left(\frac{c}{2}t^2 + f_0 t\right)\right]$

주파수 영역 분석

$$Y(k) = \sum_{n=0}^{N-1} y(n) e^{-j\left(\frac{2\pi k}{N}\right)n} \quad , (k = 0, 1 \cdots N-1)$$

$$y(n) = \frac{1}{N}\sum_{n=0}^{N-1} Y(k) e^{j\left(\frac{2\pi k}{N}\right)n} \quad , (k = 0, 1 \cdots N-1)$$

$$|H(k)| = \frac{|Y(k)|}{|X(k)|} \quad , (k = 0, 1 \cdots N-1)$$

$$|H(k)| = \frac{|AH(k)|}{|A|} \quad , (x(t) = \text{sinusoidal})$$

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

float t = 0;
float u = 0;
float Sampling_Time = 0.001;
float Start_Frequency = 0;
float End_Frequency = 200;
float End_Time = 20.0;
int cart_encoder = 0;
float cart_position = 0.0;
```

```
float prev_cart_position = 0.0;
float prev_filtered_velocity = 0.0;
float cart_velocity = 0;
float filtered_velocity = 0;

void Set_PWM(int motor)
{
  if (motor >= 0)
  {
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    analogWrite(PWM, motor);
  }
  else
  {
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    analogWrite(PWM, motor);
  }
  // Fill in the Missing Code
}
float getChirpSignal(float Start_Frequency, float End_Frequency,
float Sampling_Time, float End_Time , float t) {

  // Fill in the Missing Code, HINT: f(t) = ct+f0 , c = f1-
f0/End_Time , x(t) = sin(2pi(c/2*t^2+f0*t)
```

```
  //
  float value = 0;
  float c = (End_Frequency - Start_Frequency) / (End_Time);
  return sin(2 * 3.141592 * (c/2*t*t));
}



void control() {
  cart_position = float(cart_encoder) / 520 * PI;
  float cart_velocity = (cart_position - prev_cart_position) /
Sampling_Time;
  filtered_velocity = cart_velocity;
  prev_cart_position = cart_position;
  // Fill in the Missing Code


  //
  u = getChirpSignal(Start_Frequency, End_Frequency,
Sampling_Time , End_Time, t);
  if (t > End_Time)
    Set_PWM(0);
  else
    Set_PWM(u * 255);
  t += Sampling_Time;
  // Serial Write
  byte * time_ = (byte *) &t;
```

```
  byte * filtered_velocity_ = (byte *) &filtered_velocity;
  Serial.write(time_, 4);
  Serial.write(filtered_velocity_, 4);

}



void setup() {
  //display 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);
  //Timer Setup
```

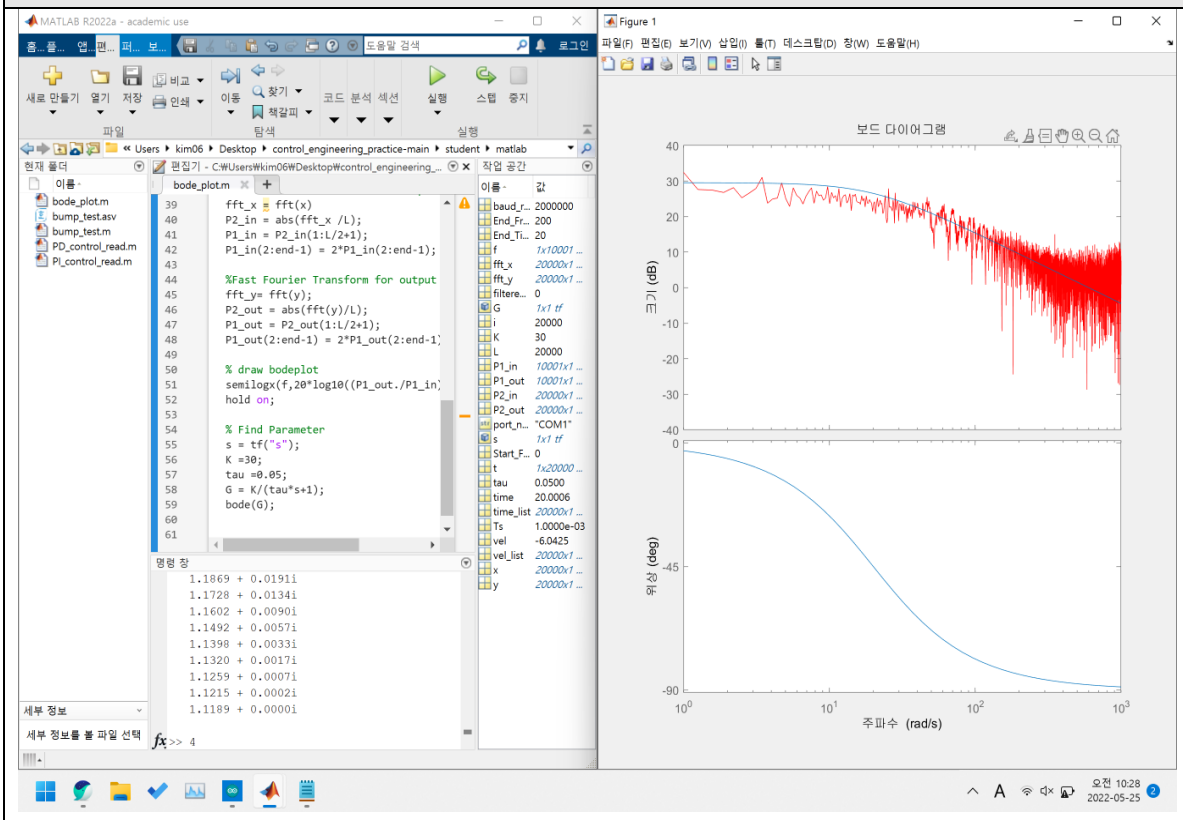
```
  MsTimer2::set(1, control);
  MsTimer2::start();
  //Interrupt Setup
  attachInterrupt(0, doEncoderA, CHANGE);
  attachInterrupt(1, doEncoderB, CHANGE);

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



  oled.drawstring(00, 2, "BODE_PLOT");
  oled.display();

}

void loop() {
}
void doEncoderA() { // Fill in the Missing Code
        if(digitalRead(ENCODER_A) == digitalRead(ENCODER_B))
        {
          cart_encoder =   cart_encoder +1;
        }
        else
        {
```

```
            cart_encoder = cart_encoder -1;
        }

}
void doEncoderB() { // Fill in the Missing Code
        if(digitalRead(ENCODER_A) == digitalRead(ENCODER_B))
        {
            cart_encoder =   cart_encoder -1;
        }
        else
        {
            cart_encoder = cart_encoder +1;
        }

}
```

실습코드(matlab)

```
clear
close all;
%% Data Acquisition

% Serial Port Setup
port_name = "COM1";
baud_rate = 2000000;
s = serialport(port_name,baud_rate);

% Parameter Setup
Ts = 0.001;
End_Time = 20;
Start_Frequency = 0;
End_Frequency = 200;

% Data list
time_list = zeros(End_Time/Ts,1);
vel_list = zeros(End_Time/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
Ts = 0.001;
t =  0:Ts:(End_Time-Ts);
x =chirp(t,Start_Frequency,End_Time,End_Frequency)';
y = vel_list;
L = length(y);
f = linspace(0,(1/Ts)*pi,L/2+1);
fft_x = fft(x)
P2_in = abs(fft_x /L);
P1_in = P2_in(1:L/2+1);
P1_in(2:end-1) = 2*P1_in(2:end-1);

%Fast Fourier Transform for output signal
fft_y= fft(y);
P2_out = abs(fft(y)/L);
P1_out = P2_out(1:L/2+1);
P1_out(2:end-1) = 2*P1_out(2:end-1);

% draw bodeplot
semilogx(f,20*log10((P1_out./P1_in)),"r")
hold on;

% Find Parameter
s = tf("s");
K =30;
tau =0.05;
G = K/(tau*s+1);
bode(G);
```

결론



Chirp신호 입력, 출력속도에 대한 Bode plot을 출력하였고 이에 대한 K와 tau를 추정한 결과 K=30, tau=0.05가 나오게 되었다.

실습 1-2의 아두이노 코드에서 pwm, control함수를 bode plot에 맞게 수정하는 것으로 실습을 진행할 수 있었다.

Frequency response 함수를 이용하여 bode plot 실습을 진행하였다. F(t), X(t) 함수를 아두이노 코드에 대입하여 input signal FFT의 magnitude 값과 output signal FFT의 magnitude값을 이용하여 bode plot을 그릴 수 있었다.