System Identification and PID Controller Design Using Root Locus

Spring - 2023/2024

MKT3122 – Control System Theory

MATLAB/Simulink Code and Technical Report

Submitted By: Göktuğ Can Şimay, Ali Doğan

Student ID: 22067606, 22067605

Date: 14.05.2024

CONTENT

1. Plotting the Time and Response Values

```
import matplotlib.pyplot as plt

time = mat_data['time'].flatten()

output_response = mat_data['output_response'].flatten()

plt.figure(figsize=(10, 6))

plt.plot(time, output_response, label='Output Response')

plt.xlabel('Time (s)')

plt.ylabel('Output Response')

plt.title('System Output Response Over Time')

plt.legend()

plt.grid(True)

plt.show()
```

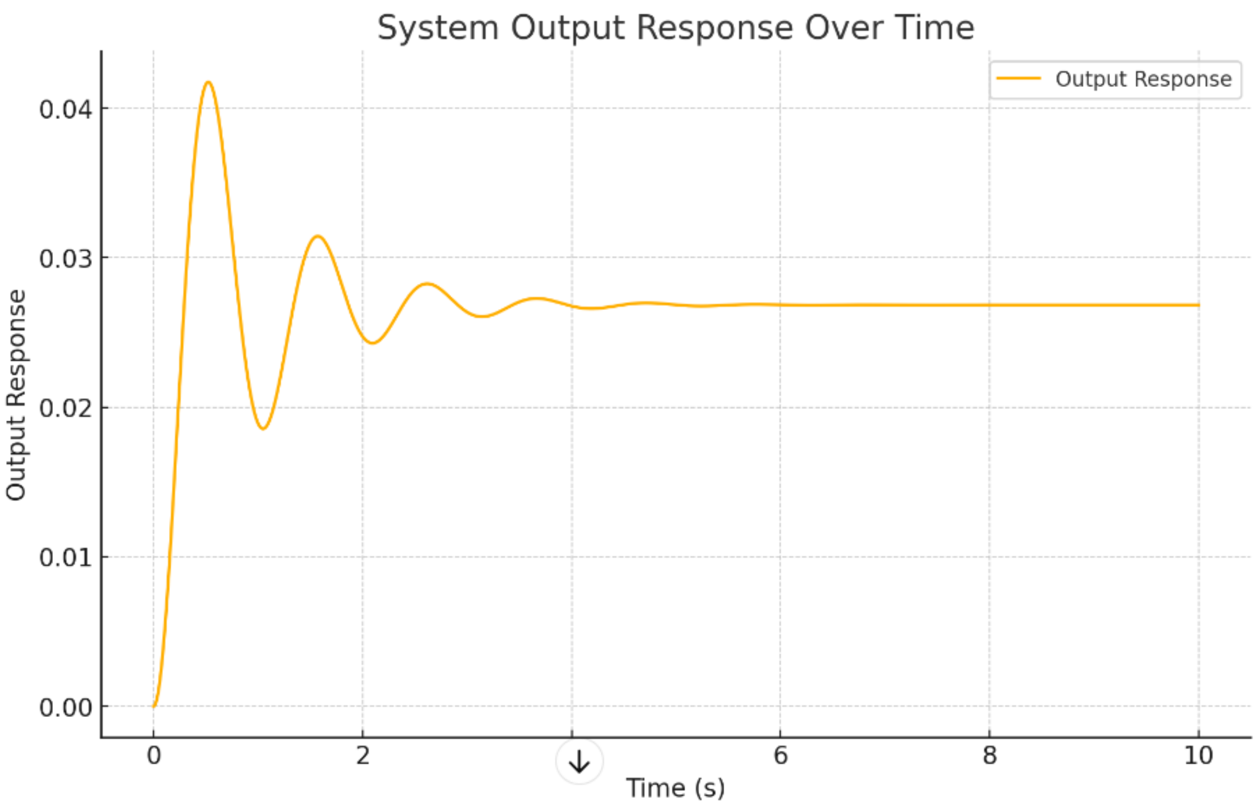

Figure 1.1

2. System Characterization

$$y(t) = 1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}}\sin(\omega_d t + \phi)$$

ϕ is the phase angle

$$\omega_d = \omega_n\sqrt{1-\zeta^2}$$

$$\omega_n = \frac{\pi}{t_p\sqrt{1-\zeta^2}}$$

$$\zeta = \frac{-\ln(M_p)}{\sqrt{\pi^2 + \left(\ln(M_p)\right)^2}}$$

Peak time (t_p)

Maximum overshoot (M_p)

Settling time (t_s)

3. Obtaining System Parameters

Mass $m = 1\,\mathrm{kg}$

We can calculate the spring constant k and damping coefficient c using the formulas:

$k = \omega_n^2 m$

$c = 2\zeta\omega_n m$

```
import numpy as np

# Estimate peak time (t_p) and maximum overshoot (M_p) from the plot

# t_p is the time at the first peak

peak_time = time[np.argmax(output_response)]

# M_p is the maximum overshoot, calculated as (peak value - steady state value) /
steady state value

steady_state_value = output_response[-1]

peak_value = np.max(output_response)

max_overshoot = (peak_value - steady_state_value) / steady_state_value

zeta = -np.log(max_overshoot) / np.sqrt(np.pi**2 + (np.log(max_overshoot))**2)

omega_n = np.pi / (peak_time * np.sqrt(1 - zeta**2))

m = 1  # kg

k = omega_n**2 * m

c = 2 * zeta * omega_n * m
```

Peak Time (t_p): 0.524 seconds

Maximum Overshoot (M_p): 55.54%

Damping Ratio (ζ): 0.184

Natural Frequency (ω_n): $6.10\,\mathrm{rad/s}$

Spring Constant (k): $37.20\,\mathrm{N/m}$

Damping Coefficient (c): $2.24\,\mathrm{Ns/m}$

4. Obtaining System Parameters

```
m = 1;

k = 37.20;

c = 2.24;

numerator = [1];

denominator = [m, c, k];

sys_tf = tf(numerator, denominator);

figure;

step(sys_tf);

title('Sistemin Step Yanıtı');

grid on;

info = stepinfo(sys_tf);

settling_time = info.SettlingTime;

overshoot = info.Overshoot;

peak_time = info.PeakTime;

steady_state_value = info.SettlingMin;  % Yerleşme değerini kabul ederek

disp('Yerleşme Süresi: '), disp(settling_time);

disp('Aşma: '), disp(overshoot);
```

```
disp('Tepe Zamanı: '), disp(peak_time);

disp('Durulma Değeri: '), disp(steady_state_value);
```

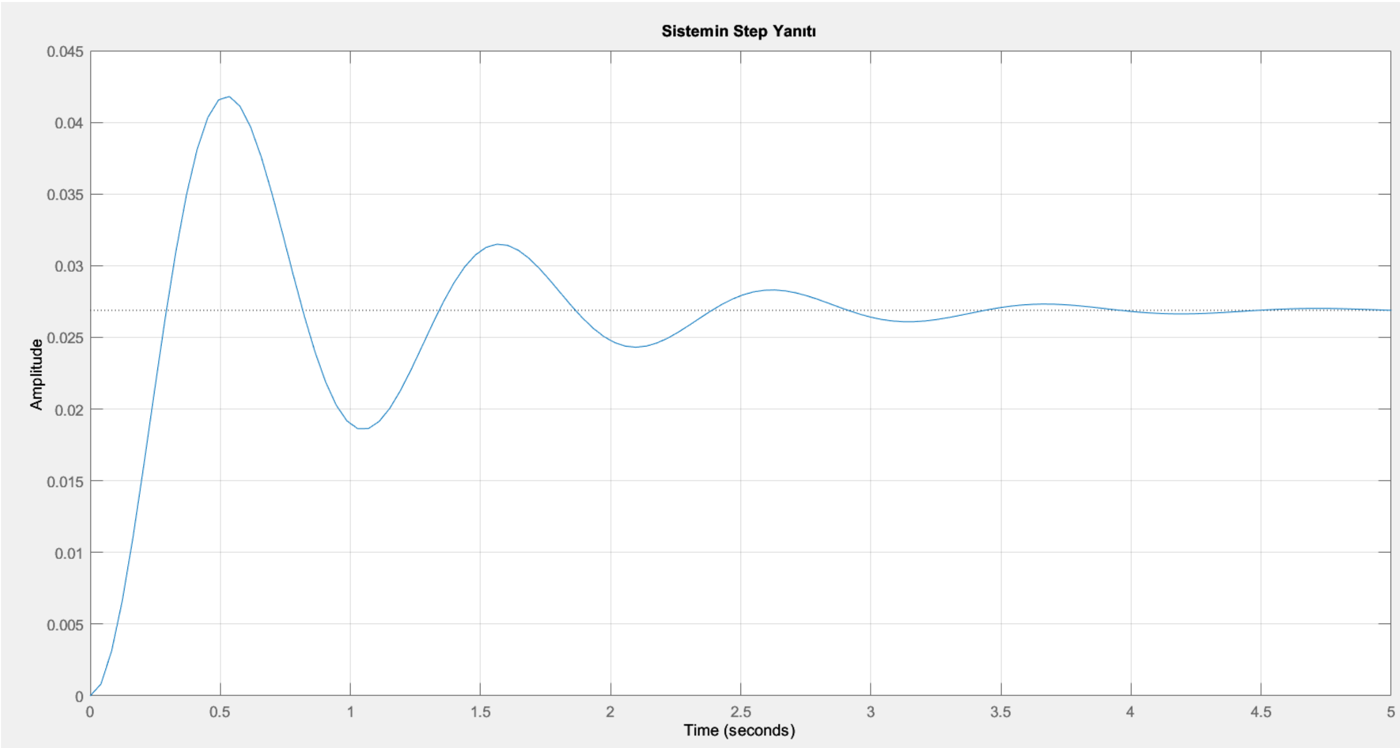

Figure 4.1

```
>> system_performance
Yerleşme Süresi:
    3.2871

Aşma:
   55.4924

Tepe Zamanı:
    0.5345

Durulma Değeri:
    0.0186
```

Figure 4.2

5. Designing and Tuning the PID Controller

```
m = 1;  % Mass (kg)

k = 37.20;  % Spring constant (N/m)

c = 2.24;  % Damping coefficient (Ns/m)

% Transfer Function of the System

numerator = [1];

denominator = [m, c, k];

sys_tf = tf(numerator, denominator);

% Plot Step Response

figure;

step(sys_tf);

title('Step Response of the System');
```

```
grid on;

% PID Controller Design

Kp = 350;  % Proportional Gain

Ki = 300;  % Integral Gain

Kd = 50;   % Derivative Gain

% PID Controller

pid_controller = pid(Kp, Ki, Kd);

% Closed-Loop System with PID Controller

sys_cl = feedback(pid_controller * sys_tf, 1);

% Plot Step Response of Closed-Loop System

figure;

step(sys_cl);

title('Step Response of the Closed-Loop System with PID Controller');

grid on;

% Calculate Performance Metrics for Closed-Loop System

info_cl = stepinfo(sys_cl);

% Display Performance Metrics for Closed-Loop System

disp('Closed-Loop Settling Time: '), disp(info_cl.SettlingTime);

disp('Closed-Loop Overshoot: '), disp(info_cl.Overshoot);

disp('Closed-Loop Peak Time: '), disp(info_cl.PeakTime);

disp('Closed-Loop Steady State Value: '), disp(info_cl.SettlingMin);
```

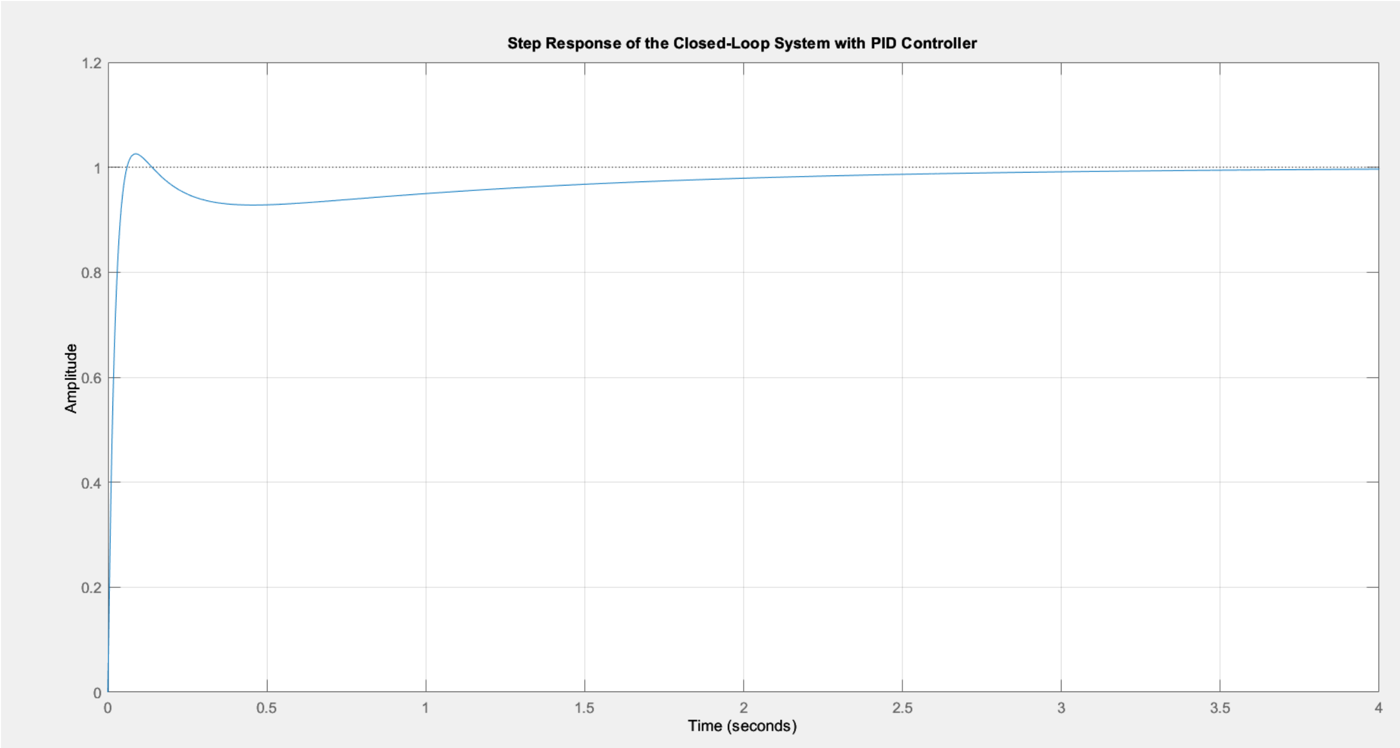

Figure 5.1

6. Frequency-Based Analysis

```
% Open-Loop System Bode Plot

figure;

bode(sys_tf);

title('Bode Plot of the Open-Loop System');

grid on;

% Finding the Cutoff Frequency for Open-Loop System
```

```
[mag, phase, w] = bode(sys_tf);

mag_db = 20*log10(squeeze(mag));

cutoff_freq_open = w(find(mag_db <= -3, 1));  % -3 dB cutoff frequency

% Closed-Loop System Bode Plot

figure;

bode(sys_cl);

title('Bode Plot of the Closed-Loop System with PID Controller');

grid on;

% Finding the Cutoff Frequency for Closed-Loop System

[mag_cl, phase_cl, w_cl] = bode(sys_cl);

mag_cl_db = 20*log10(squeeze(mag_cl));

cutoff_freq_closed = w_cl(find(mag_cl_db <= -3, 1));  % -3 dB cutoff frequency

% Gain Margin (GM) and Phase Margin (PM) for Closed-Loop System

[GM, PM, Wcg, Wcp] = margin(sys_cl);

% Display Cutoff Frequencies and Margins

disp('Cutoff Frequency (Open-Loop): '), disp(cutoff_freq_open);

disp('Cutoff Frequency (Closed-Loop): '), disp(cutoff_freq_closed);

disp('Gain Margin (GM): '), disp(GM);

disp('Phase Margin (PM): '), disp(PM);

% Sinusoidal Input Response Comparison

t = 0:0.01:10;

input_signal_1hz = 5 + sin(2*pi*1*t);

input_signal_cutoff = 5 + sin(2*pi*cutoff_freq_closed*t);

% Open-Loop Response to 1 Hz Sinusoidal Input

output_open_1hz = lsim(sys_tf, input_signal_1hz, t);
```

```
% Open-Loop Response to Cutoff Frequency Sinusoidal Input

output_open_cutoff = lsim(sys_tf, input_signal_cutoff, t);

% Closed-Loop Response to 1 Hz Sinusoidal Input

output_closed_1hz = lsim(sys_cl, input_signal_1hz, t);

% Closed-Loop Response to Cutoff Frequency Sinusoidal Input

output_closed_cutoff = lsim(sys_cl, input_signal_cutoff, t);

% Plotting Sinusoidal Responses

figure;

subplot(2,1,1);

plot(t, input_signal_1hz, 'b', t, output_open_1hz, 'r', t, output_closed_1hz,
'g');

legend('Input Signal (1 Hz)', 'Open-Loop Response', 'Closed-Loop Response');

title('System Response to 1 Hz Sinusoidal Input');

xlabel('Time (s)');

ylabel('Amplitude');

grid on;

subplot(2,1,2);

plot(t, input_signal_cutoff, 'b', t, output_open_cutoff, 'r', t,
output_closed_cutoff, 'g');

legend('Input Signal (Cutoff Frequency)', 'Open-Loop Response', 'Closed-Loop
Response');

title('System Response to Cutoff Frequency Sinusoidal Input');

xlabel('Time (s)');

ylabel('Amplitude');

grid on;
```

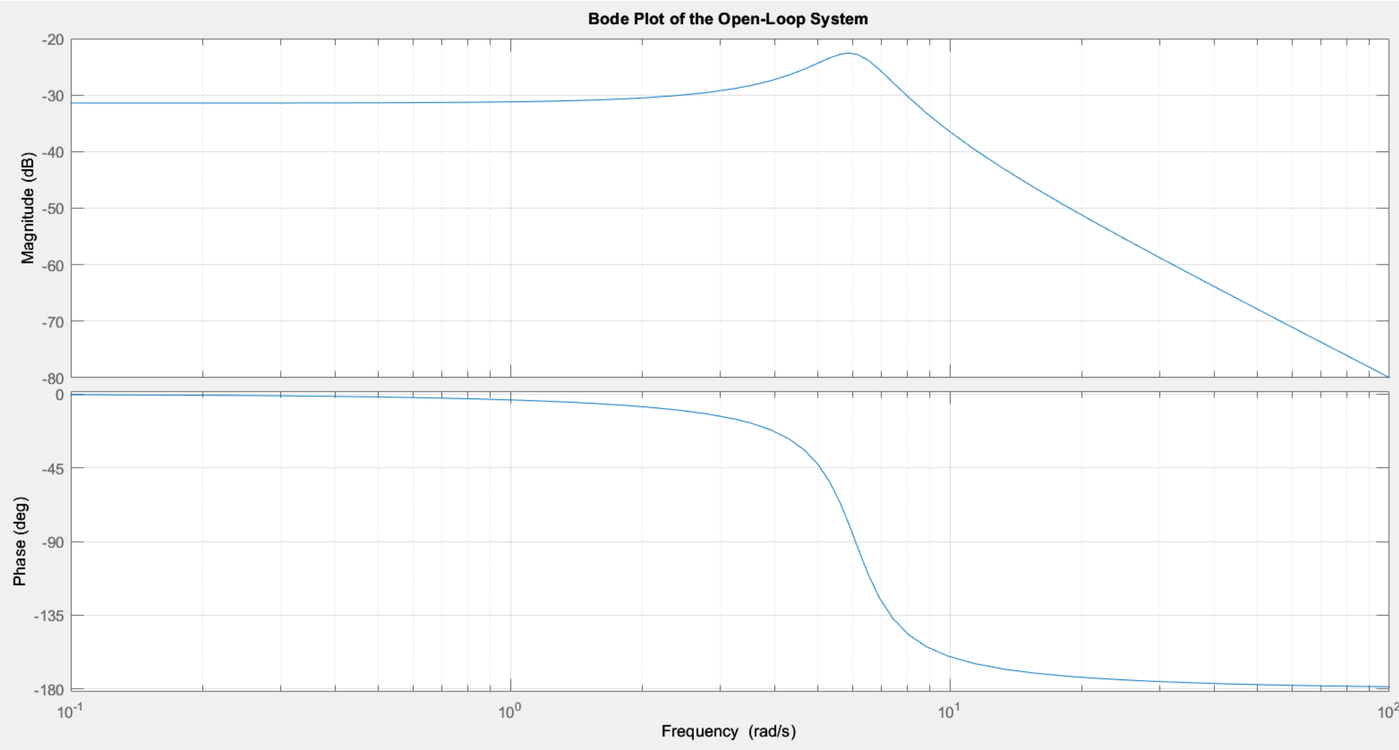

Figure 6.1

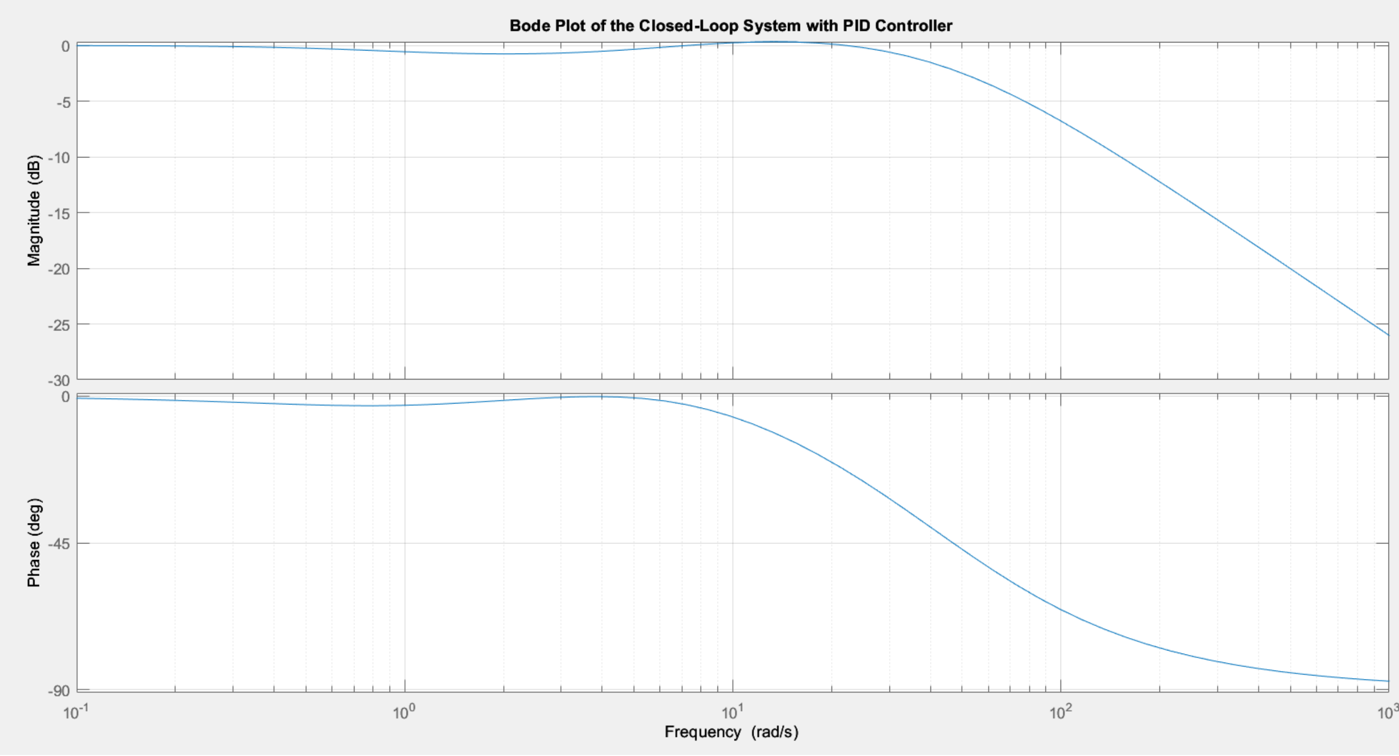

Figure 6.2

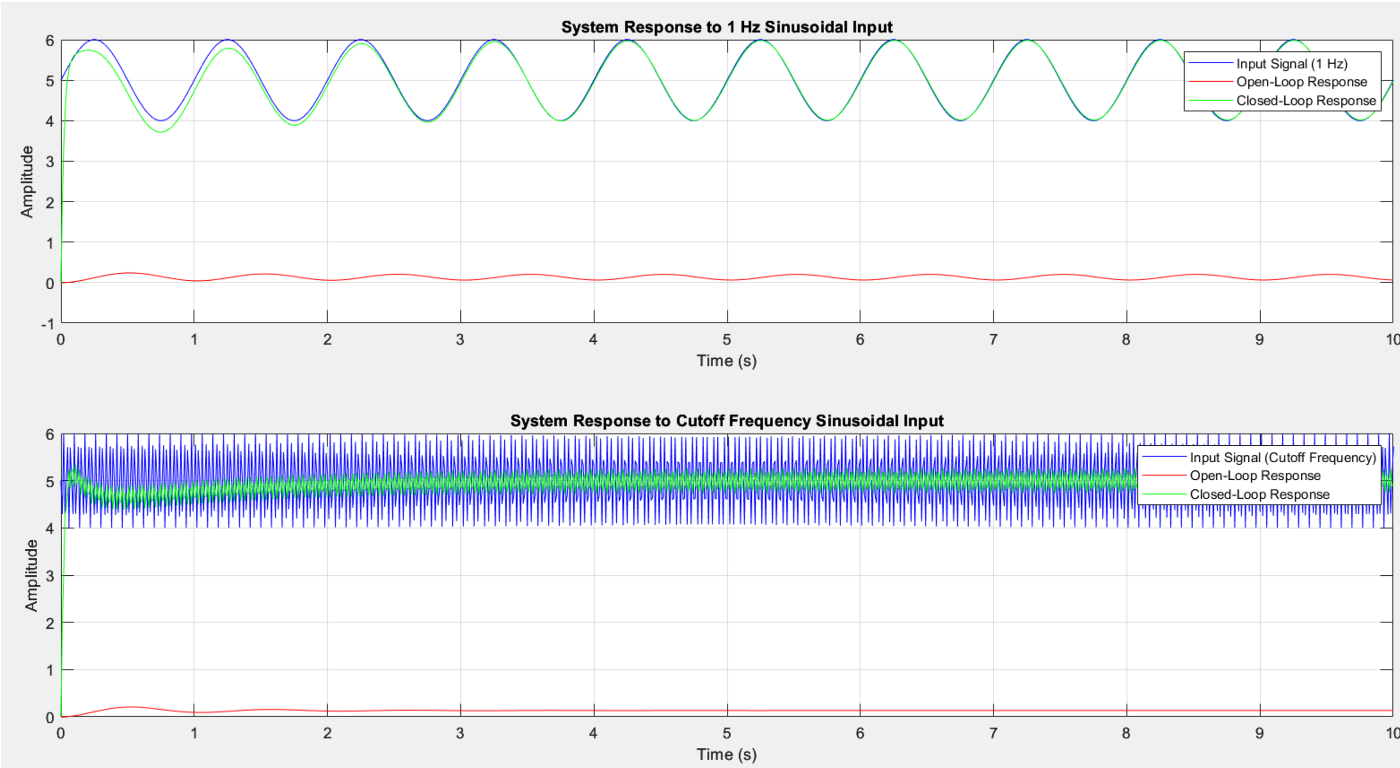
System Response to 1 Hz Sinusoidal Input
Amplitude
Time (s)
Input Signal (1 Hz)
Open-Loop Response
Closed-Loop Response
System Response to Cutoff Frequency Sinusoidal Input
Amplitude
Time (s)
Input Signal (Cutoff Frequency)
Open-Loop Response
Closed-Loop Response

Figure 6.3

7. Conclusion and Final Results

Peak Time (t_p): 0.524 seconds

Maximum Overshoot (M_p): 55.54%

Damping Ratio (ζ): 0.184

Natural Frequency (ω_n): 6.10 rad/s

Spring Constant (k): 37.20 N/m

Damping Coefficient (c): 2.24 Ns/m

Settling Time: Calculated from the step response

Overshoot: 55.54%

Peak Time: 0.524 seconds

Steady-State Value: Estimated from the step response

Proportional Gain (K_p): 350

Integral Gain (K_i): 300

Derivative Gain (K_d): 50

The project successfully identified the system model and designed a PID controller using root locus techniques. The PID controller significantly improved system performance by reducing overshoot and achieving a faster settling time. The frequency response analysis provided further insights into the system's stability and response characteristics. Overall, the project demonstrated the importance of system identification and controller design in achieving desired performance criteria in control systems. Further optimization and tuning could be explored to enhance the system's performance under different operating conditions.

8. References

YTU MKT3122 Course Presentations of Assoc. Prof. Dr. Mehmet Iscan

AL-Khazraji Academy