Department of Electronic and Telecommunication Engineering

University of Moratuwa

Assignment -1
System Identification and PID Control

Wijetunga W.L.N.K - 200733D

This report is submitted as a partial fulfillment of the module

EN2143 – Electronic Control Systems

5th of June 2023

1. Introduction

System identification is the process of creating mathematical representations of a system's dynamic behaviour. It entails gathering data from the actual system, interpreting the data, and determining the parameters or model structure that best captures the behaviour of the system.

2. Transfer Function Used for Analysis

$$G(s) = 10e^{(-0.04s)} \frac{(s^2 + 4\,s + 3)}{(s^3 + 6s^2 + 5s + 16)}$$

3. Using System Identification Application According to the Given Instructions

System Identification Toolbox™ in MATLAB® provides MATLAB® functions, Simulink® blocks, and an app for dynamic system modelling, time-series analysis, and forecasting. It can be used to learn dynamic relationships among measured variables to create transfer functions, process models, and state-space models in either continuous or discrete time while using time- or frequency-domain data. [1]

- **Question 1** - Import time domain data, set input and output, set start time and sample time.

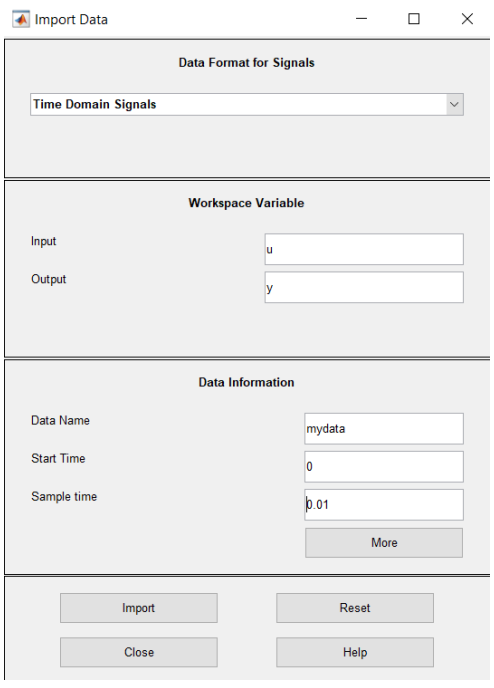

Figure 1

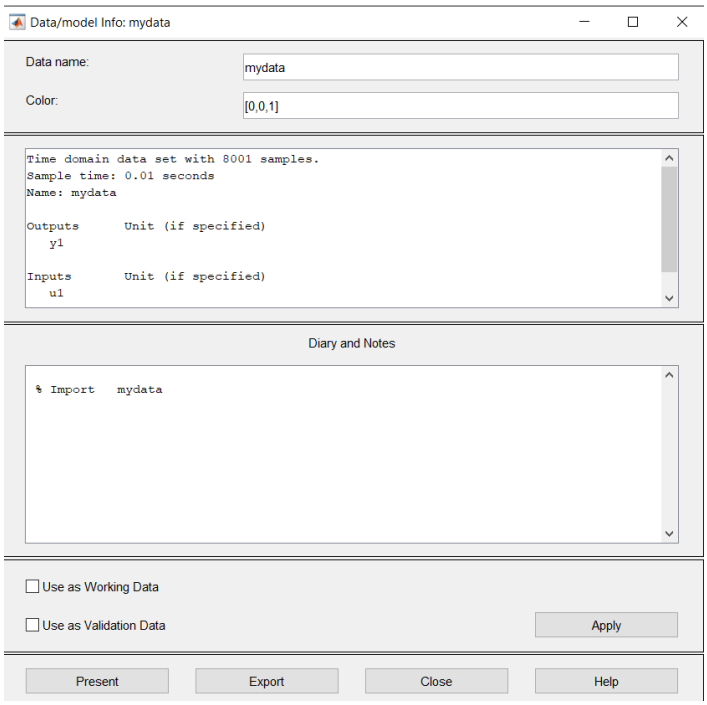

Figure 2

- **Question 2** - Visualize whether the imported data is as same as generated data

    - **Imported Data**

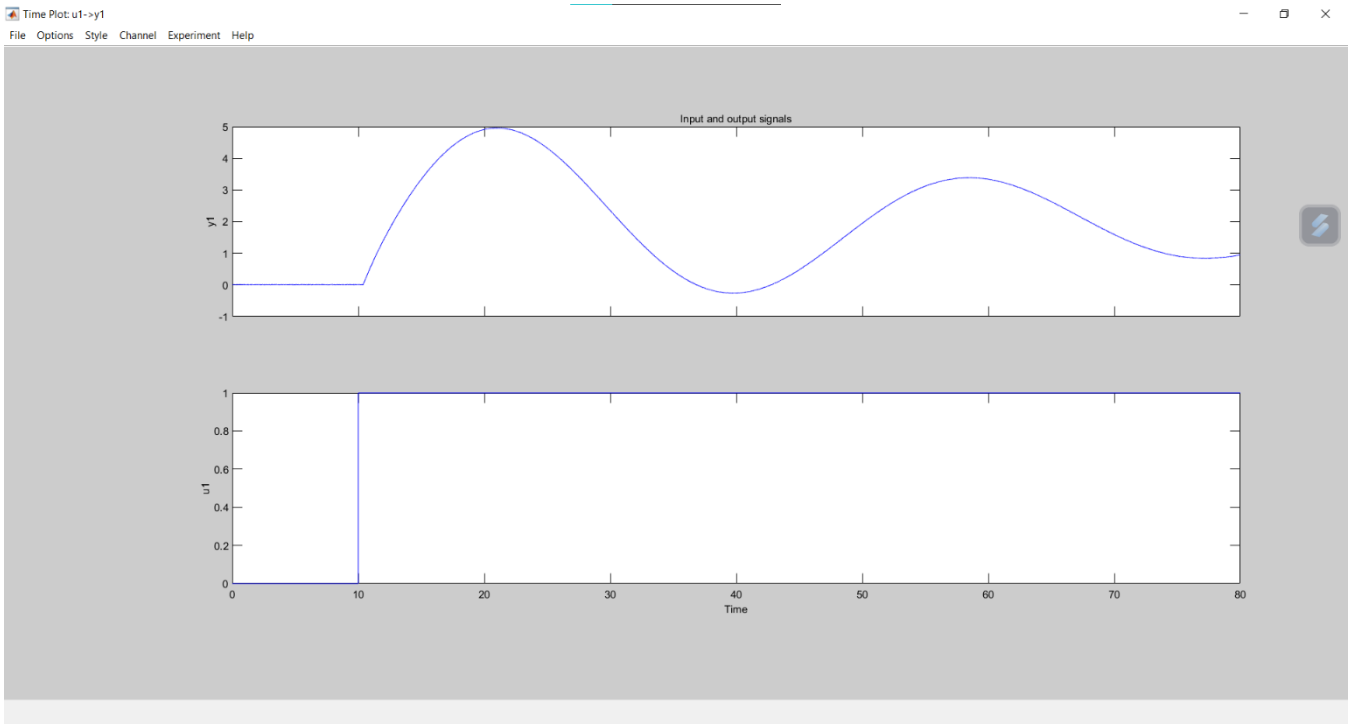

Figure 3

    - **Generated Data**

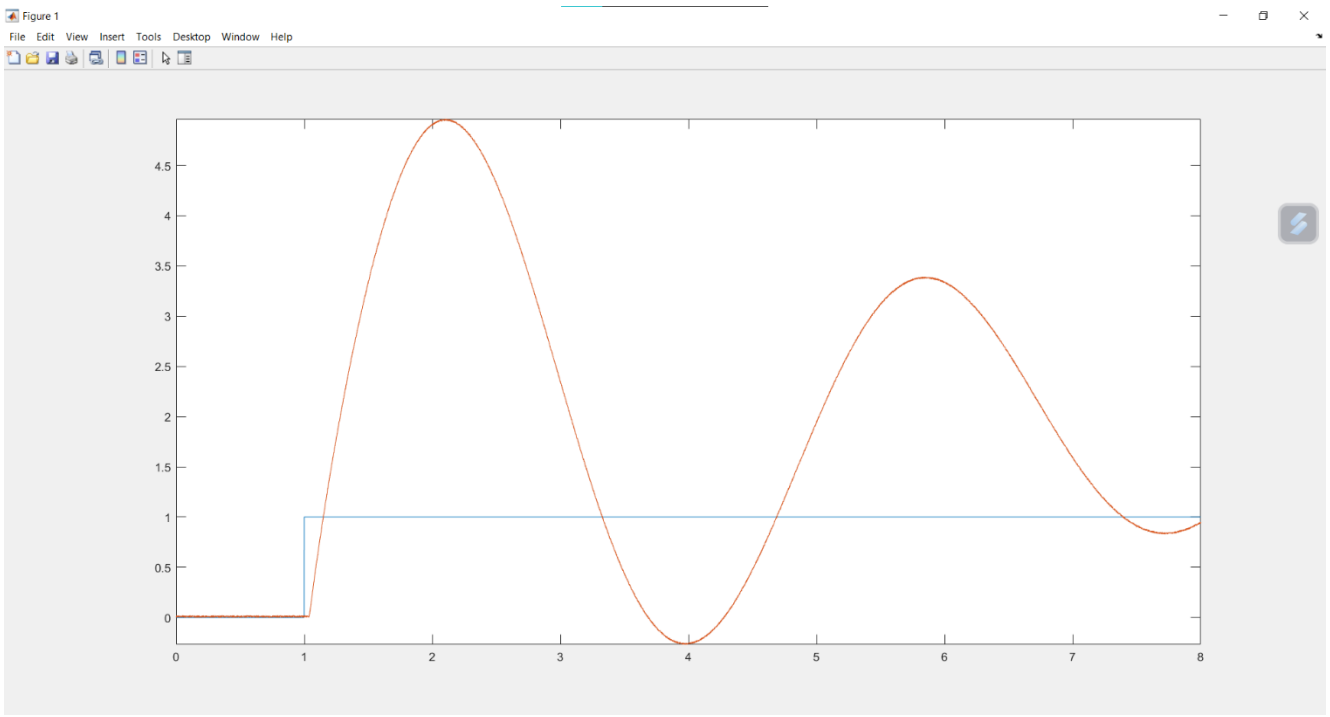

Figure 4

Comment - Generated data is similar to the imported data.

- **Question 3** - In estimate -> area, select transfer function models, select number of poles and zeros, select continuous time, set delay.

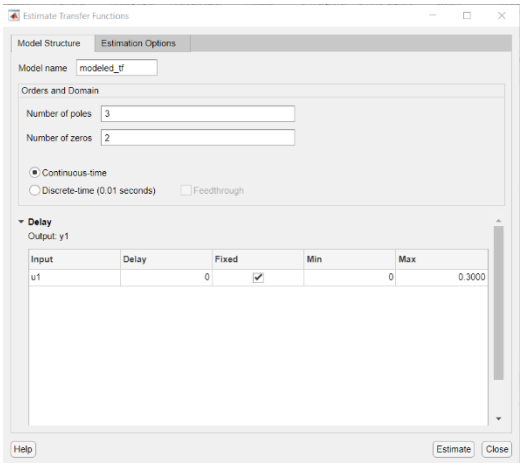

Figure 5

- Selected Parameters
  - Number of poles - 3
  - Number of zeros - 3
  - Delay - 0

- **Question 4** - Check model output and data fitting percentage.

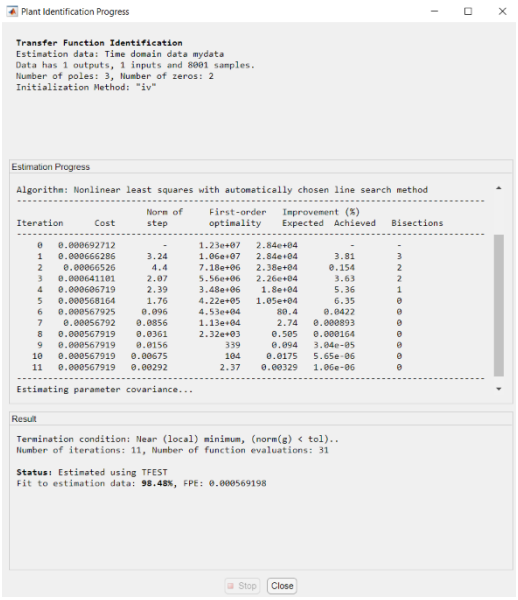

Figure 6

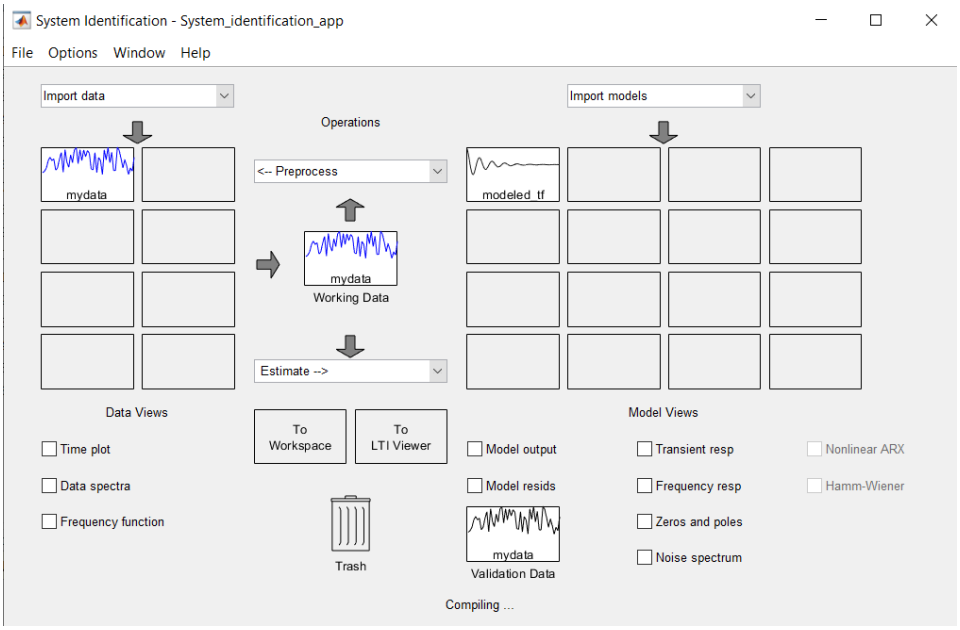

Figure 7

- **Step Response of the Model Output**

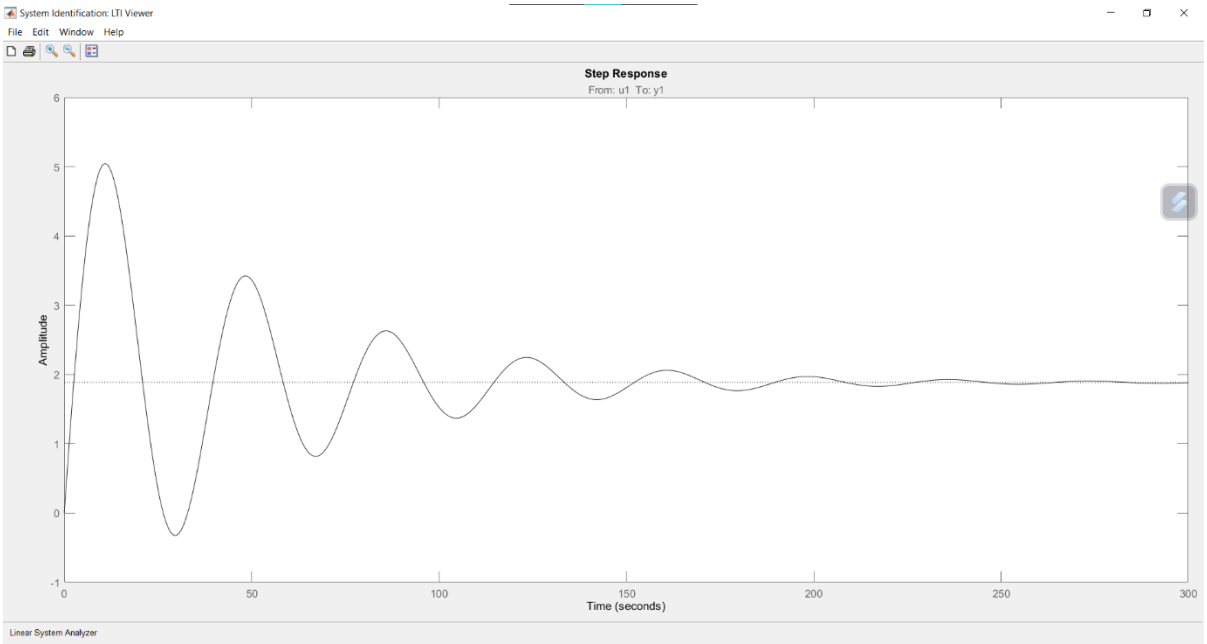

Figure 8

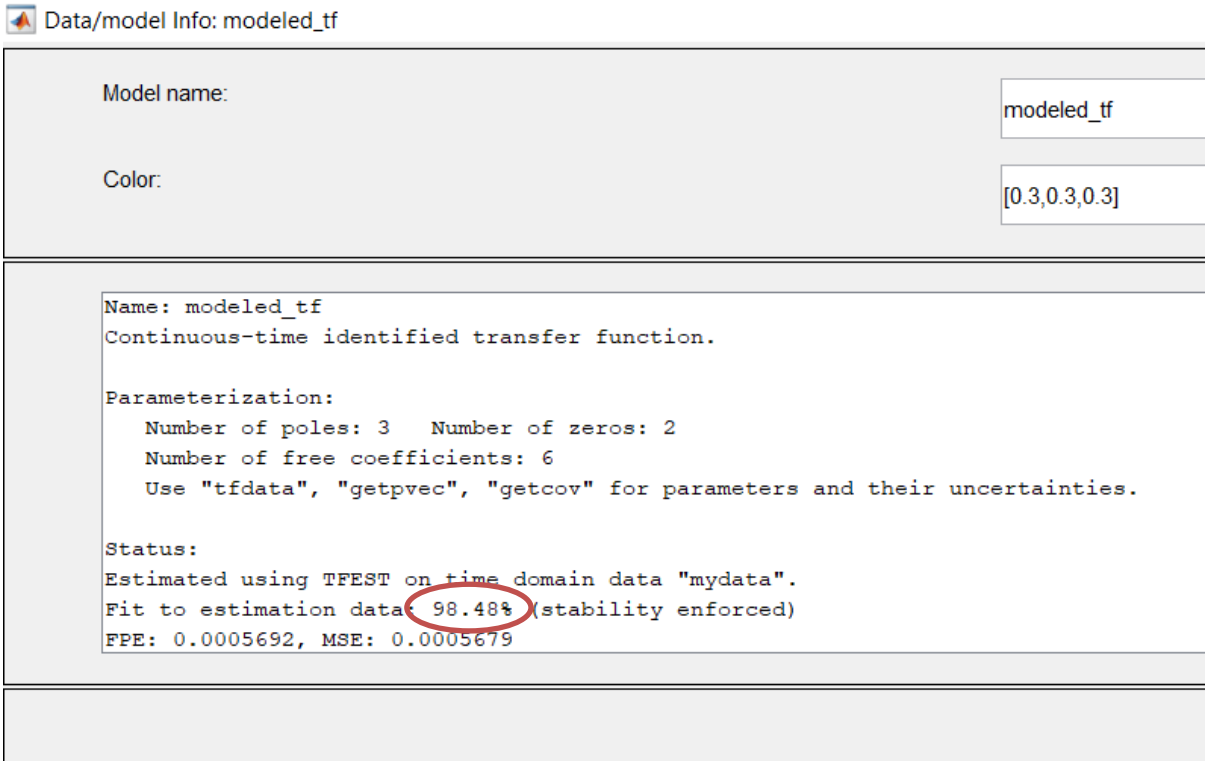

Figure 9

  - Data fitting percentage - **98.48%**

- **Question 5** - Bring the modelled transfer function to workspace

4. Use of PID controller for the model

- **Question 6** - Design a PID controller for the model and verify/comment on the performance

    - The output of the plant before and after implementing the PID controller is as follows.

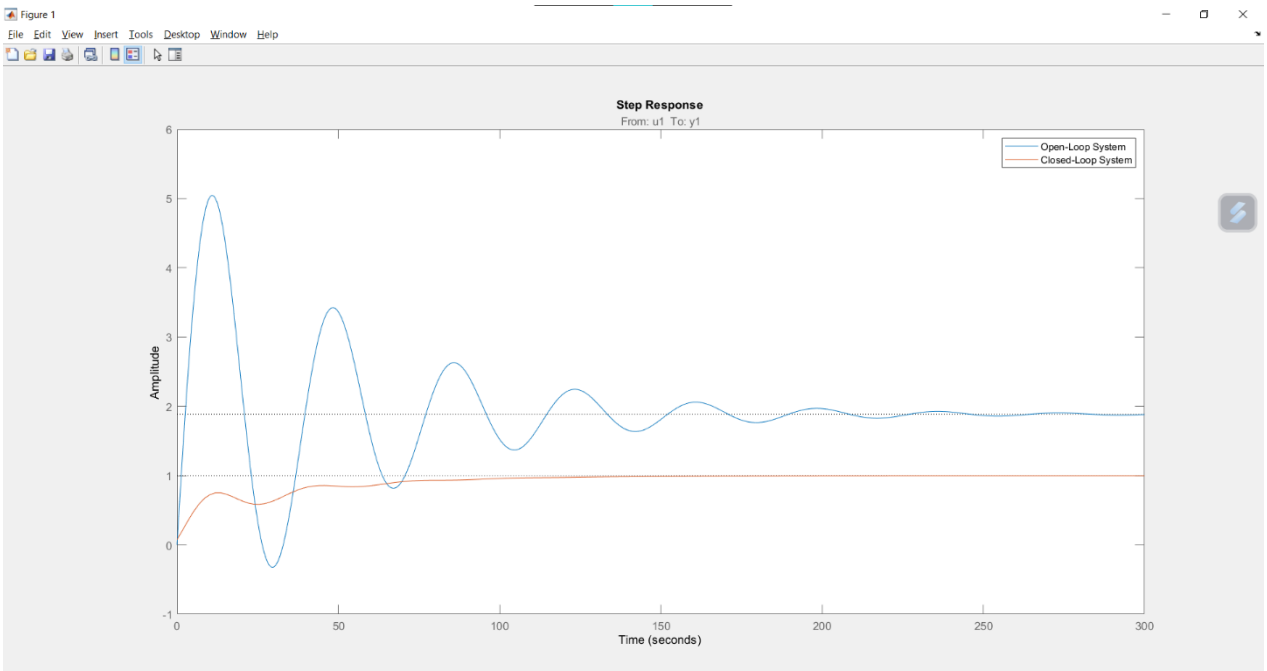

Figure 10

    - Open loop transfer function modelled by MATLAB is as follows.

$$\frac{0.7921\ s^2 + 0.2376\ s + 0.01599}{s^3 + 0.3366\ s^2 + 0.03998\ s + 0.008474}$$

    - Closed loop transfer function of the system with unity feedback which has the PID controller (where the above function is the open loop transfer function) is as follows.

$$\frac{0.08326\ s^4 + 0.1163\ s^3 + 0.04773\ s^2 + 0.007435\ s + 0.0003762}{1.083\ s^4 + 0.453\ s^3 + 0.08771\ s^2 + 0.01591\ s + 0.0003762}$$

    - It can be clearly seen that after implementing the PID controller, the oscillations of the system have been reduced.
    - Comparing with the initial step response of the system, the time taken by the system to reach stability is less after implementing the PID controller.

5. Root locus of the controlled plant

- **Question 7** - Draw the root locus of the controlled plant and comment on the pole placement

    - **Pole Zero Plot of the controlled plant**

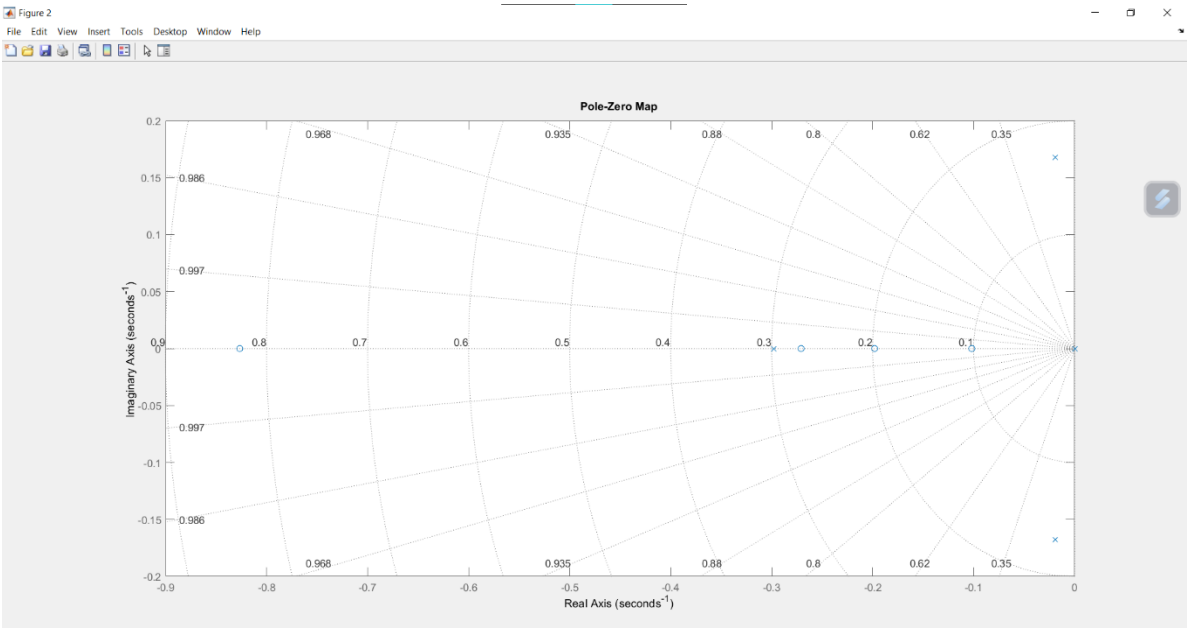

Figure 11

    - **Root Locus of the controlled plant**

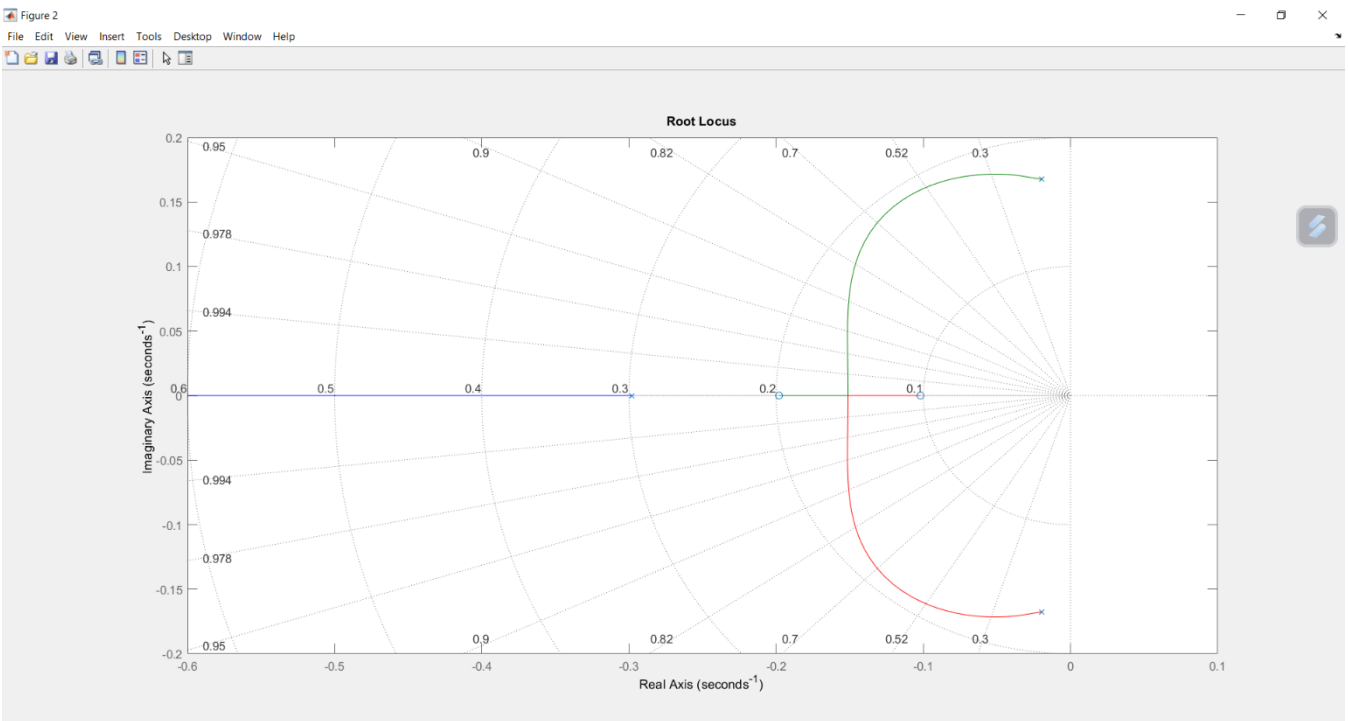

Figure 12

- **Comments**
    - All the poles are in the left-hand plane.
    - Therefore, the system is stable.

References

[1] The MathWorks, Inc, “System Identification Toolbox,” [Online]. Available: https://www.mathworks.com/products/sysid.html. [Accessed 5 6 2023].

6. Appendix

- **MATLAB code Used for Data Generation**

```
clc; close all; clear;
s = tf('s');

%num = conv([exp(0.1) 0],[1 1]);
%den = conv([1 5 6], conv([1 1],[1 3]));

G = (10*(s+3)*(s+1)*exp(-0.04*s))/(s^3 + 6*(s^2) + 5*s +
16);% write your transfer function for data generation
display(G);

dt = 0.001;% set your sampling interval

t = 0:dt:8;
u = ones(length(t),1);

u(1:1/dt)=0; % set first 10 samples to zero

y = lsim(G,u,t);
y = y + rand(length(t),1)*0.02;

plot(t,[u,y]); axis([0 8 0 1.2]);
```

- **MATLAB Code Used to Design the PID**

```
% Design a PID controller in CT
s = tf('s');
Gs = modeled_tf;% Matlab modelled transfer function
display(Gs);

Cs = pidtune(Gs,'pid');
Gc = feedback(Cs*Gs,1);% Unity feedback system
display(Gc);

%% Comparison of Closed loop and open loop transfer
functions

figure;
k = stepplot(Gs, Gc);
legend('Open-Loop System', 'Closed-Loop System');

%% Pole Zero plot
figure;
pzplot(Cs*Gs);
grid on;

%% Draw the root locus
figure;
rlocus(Gs);
grid on
```