Mechatronics System Design: Analysis and Control – Solar Tracker

MECN4029A – Mechatronics II Assignment –

SIMULATION DOCUMENT

Group - 22

Sibusiso Sithole	1424744
Kabelo Moshoma	1346560
Sohil Sugreem	845552
Tshepiso Ngwato	841628

Lecturer: Dr. A. Panday

A Mechatronics II report submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, Johannesburg, in partial fulfilment of the requirements for the degree of Bachelor of Science in Engineering (Mechanical).

Johannesburg, May 2024

1. Bode Plot – Uncontrolled

```
                    %% Transfer Function for Uncontrolled linear
system

Num_1 = 112.93;
Den_1 = [1 2500 377 112.93]; % Coefficients from Denominator

Sys_1 = tf(Num_1, Den_1);

% Bode Plot
figure;
margin(Sys_1); % To indicate Gain and Phase margins on Bode Plot
```

2. Bode Plot – PID Controlled

```
                                   %% PID Controlled
Num_2 = [1.268e05 2.76e04 1479];
Den_2 = [1 2576 1.909e05 1.556e05 2.76e04 1479]; % Coefficients from
Denominator

Sys_2 = tf(Num_2, Den_2);

% Bode Plot
figure;
margin(Sys_2); % To indicate Gain and Phase margins on Bode Plot
```

3. Nyquist Plot – Uncontrolled

```
                   %% Uncontrolled
                    % Transfer Function
Num_1 = 112.93;
Den_1 = [1 2500 377 112.93];
sys_1 = tf(Num_1, Den_1);

                    % The Nyquist plot
figure;
nyquist(sys_1);
title('Nyquist Plot');
grid on;

                % Stability analysis using margins
[GM, PM] = margin(sys_1);

                        % Dsiplaying Margins
disp('Gain Margin (dB):');
```

```
disp(20*log10(GM));
disp('Phase Margin (degrees):');
disp(PM);

                % COmment on system's stability
if GM > 1 && PM > 0
    disp('The system is stable.');
else
    disp('The system is unstable.');
end
```

4. Nyquist Plot – PID Controlled

```
                    %% PID Controlled
                 % Transfer Function
Num_2 = [1.268e05 2.76e04 1479];
Den_2 = [1 2576 1.909e05 1.556e05 2.76e04 1479]; % Coefficients from
Denominator
sys_2 = tf(Num_2, Den_2);


                    % The Nyquist plot
figure;
nyquist(sys_2);
title('Nyquist Plot');
grid on;

                % Stability analysis using margins
[GM, PM] = margin(sys_2);

                    % Dsiplaying Margins
disp('Gain Margin (dB):');
disp(20*log10(GM));
disp('Phase Margin (degrees):');
disp(PM);

                % COmment on system's stability
if GM > 1 && PM > 0
    disp('The system is stable.');
else
    disp('The system is unstable.');
end
```

5. Pole Zero Plot – Uncontrolled

```
               %% Uncontrolled
                    %Transfer Function
Num_1 = 112.93;
```

```
Den_1 = [1 2500 377 112.93];
sys_1 = tf(Num_1, Den_1);

                                %Pole-Zero plot
figure;
pzmap(sys_1);
title('Pole-Zero Plot');
grid on;

                                %Poles and Zeros
poles = pole(sys_1);
zeros = zero(sys_1);
hold on;

plot(real(poles), imag(poles), 'rx', 'MarkerSize', 10, 'LineWidth',
2);
plot(real(zeros), imag(zeros), 'bo', 'MarkerSize', 10, 'LineWidth',
2);
hold off;

                             % Stability Analysis
if all(real(poles) < 0)
    disp('The system is stable.');
elseif any(real(poles) == 0)
    disp('The system is marginally stable.');
else
    disp('The system is unstable.');
end
```

6. Pole Zero Plot – PID Controlled

```
                                  %% PID Controlled
                                     %Transfer Function
Num_2 = [1.268e05 2.76e04 1479];
Den_2 = [1 2576 1.909e05 1.556e05 2.76e04 1479]; % Coefficients from
Denominator
sys_2 = tf(Num_2, Den_2);

                                %Pole-Zero plot
figure;
pzmap(sys_2);
title('Pole-Zero Plot');
grid on;

                                %Poles and Zeros
poles = pole(sys_2);
zeros = zero(sys_2);
hold on;

plot(real(poles), imag(poles), 'rx', 'MarkerSize', 10, 'LineWidth',
2);
```

```
plot(real(zeros), imag(zeros), 'bo', 'MarkerSize', 10, 'LineWidth', 2);
hold off;

                        % Stability Analysis
if all(real(poles) < 0)
    disp('The system is stable.');
elseif any(real(poles) == 0)
    disp('The system is marginally stable.');
else
    disp('The system is unstable.');
end
```

7. Inputs

```
%Simulink Input Data For Solar Tracker


%% Gain Constants
Kpos1 = 0.0667; %UNits????
Kpos2 = 0.0667; %Position Sensor Gain in V/degree
Kb = 0.085; %Back EMF in V/rad/s
Kt = 0.09; %Motor Torque Constant in Nm/A
La = 0.002; %Armature Inductance in H
Ra = 5; %Armature Resistance in hms
Controller = 1; %Unity for uncontrolled system


%% Solar Panel Variables
Mpanel=30; %mass of the solar panel in Kg
Wpanel = 1; %width of the panel in metres
d = 0.05; %thickness of the solar panel in metres
DegBeta= 45;
beta = deg2rad(DegBeta); %the inclination angle of the solar panel
lpanel = 1.5; %the length of the solar panel

N1 = 2400;
N2 = 12;
Kr = N1/N2; %Gear Ratio

%To calculate the moment of inertia of the panel
Jpanel = (Mpanel/12)*(lpanel^2*cos(beta)^2 + d^2*sin(beta)^2 + Wpanel^2);

%Shaft Variables
Mshaft = 5;  %Shaft mass in kg
lshaft = 2.5; %Shaft height in meters

%Shaft moment of inertial
Jshaft = (1/12)*(Mshaft)*(lshaft^2);
```

```
%Motor Rotor/Shaft Moment of inertia
Ja = 0.1;  %moment of inertia of motor in kgm^2

%Load Equivalent Moment of inertia
%JL =  (0)*(Jpanel + Jshaft); %Equivalent Moment of Inertia
JL= Jpanel;

%System Equivalent Moment of Inertia
Jm = (Ja + JL*(N1/N2)^2);


%% Damping

Da = 0.05; % Motor Damping Constant in Nm/rad/s
DL = 0.75; %Damping caused by friction on the shaft and gear (in
Nm/s)

%Equivalent Damping COnstant
%Dm = Da + DL*(N1/N2)^2;
Dm= Da + DL;
```