ECSE 403

Lab 1 Report

Ismail Faruk 260663521

1. Transfer function between the input voltage and the speed of the motor shaft,

$$G(s) = \frac{\omega(s)}{v_a(s)} = \frac{0.002}{0.01s+0.00104}.$$

2. Step Response:

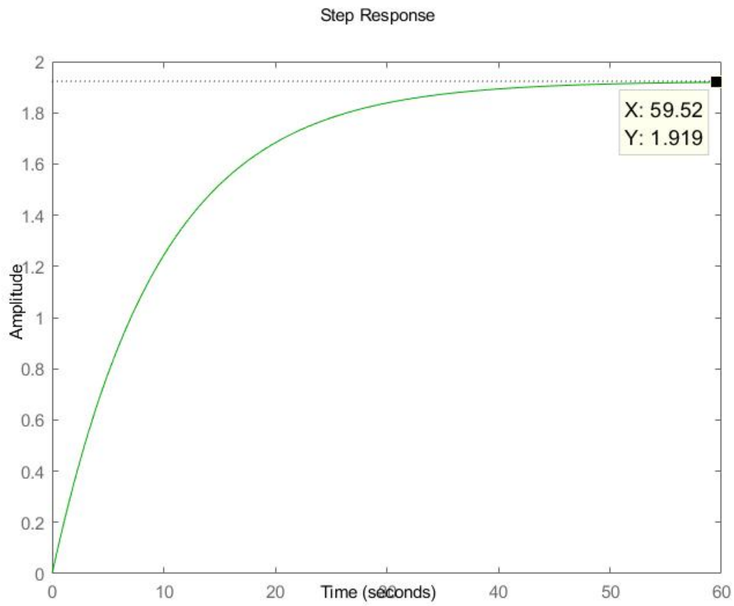

Fig 1

Steady State = 1.919

Time constant = $\frac{0.01}{0.00104} = 9.615\ s$

3. Step response information of G(s)

S <1x1 struct>

Field	Value	Min	Max
RiseTime	21.1283	21.1283	21.1283
SettlingTime	37.6188	37.6188	37.6188
SettlingMin	1.7318	1.7318	1.7318
SettlingMax	1.9223	1.9223	1.9223
Overshoot	0	0	0
Undershoot	0	0	0
Peak	1.9223	1.9223	1.9223
PeakTime	75.1570	75.1570	75.1570

Fig 2

Rise time = 21.1283 s

Settling time = 37.6188 s

4. Steady State value can be obtained as time t → ∞, which in Laplace domain would mean s → ∞. Thus, $\frac{w(s)}{v_a(s)} = \frac{0.002}{0.00104} = 1.923$. Steady State value found in Matlab = 1.919. This difference is because the Steady State value in Matlab is approximate and is taken at time t = 59.52 s, whereas the theoretical value is considered to be Steady State value as t → ∞.

5. Transfer function between the shaft's angel and input voltage,

$$H(s) = \frac{\theta(s)}{v_a(s)} = \frac{0.002}{s(0.01s+0.00104)}$$. This is a second order system.

6. Block diagram of unity feedback loop to the system: ω/v_a

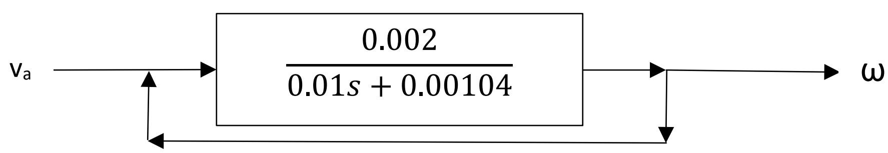

Fig 3

Closed loop transfer function, $\frac{\omega(s)}{v_a(s)} = \frac{0.002}{0.01s+0.00304}$

7. Step response of the unity feedback closed loop system KG(s)

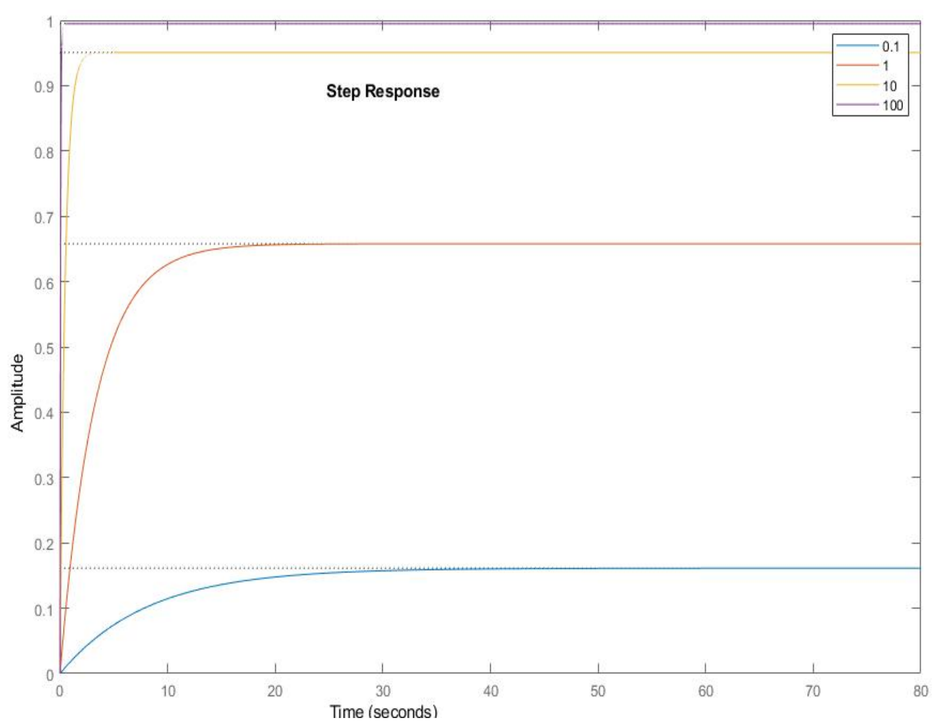

Fig 4

Step response information of KG(s)

StepInfo
3x5 string

	1	2	3	4	5
1	Proportional Gain	0.1	1	10	100
2	RiseTime	17.7178	7.227	1.0442	0.10928
3	SettlingTime	31.549	12.8687	1.8594	0.19459

Fig 5

Fig 4 and **Fig 5** shows the effect of proportional controller on the system. As the proportional gain increases, the rise time decreases and settling time decreases. As this is a first order system, there was no overshoot.

8. Step response of the unity feedback closed loop system KH(s)

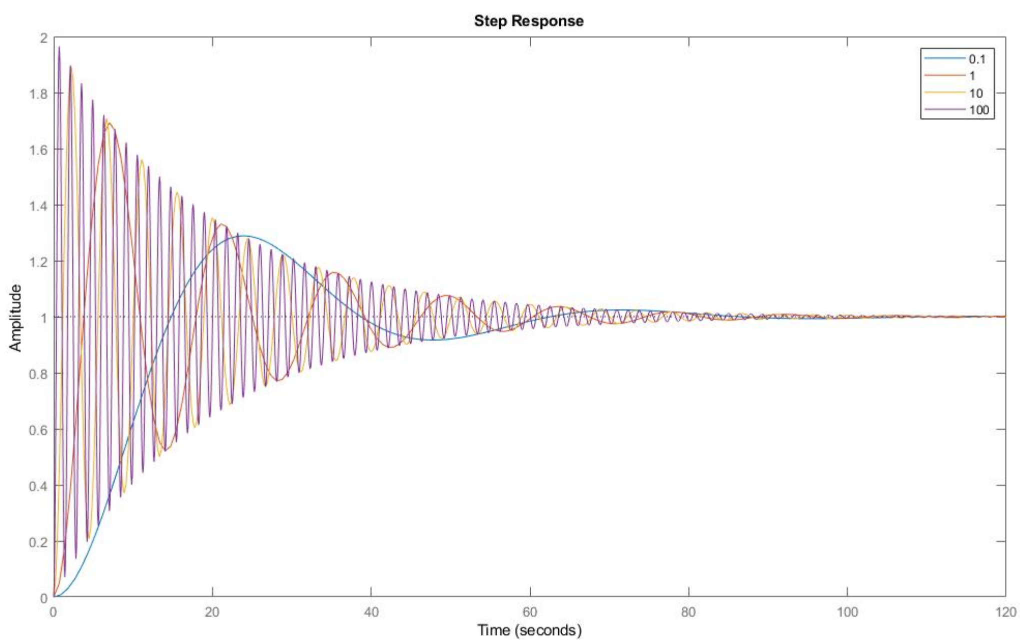

Fig 6

Step response information of KH(s)

StepInfo
4x5 string

	1	2	3	4	5
1	Proportional Gain	0.1	1	10	100
2	Overshoot	28.8756	69.2106	89.0836	96.4128
3	RiseTime	10.0198	2.5544	0.75805	0.23509
4	SettlingTime	76.1727	72.2175	73.6468	75.1704

Fig 7

Fig 6 and **Fig 7** shows the effect of proportional controller on the system. Increase in the proportional gain increases the overshoot, reduces the rise time and reduces the settling time.

9. For overshoot of 20%, to find the maximum value of K, we need to go through a few steps.

Step 1: Find the value of the damping ratio for 20% overshoot

$e^{-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}} = 0.2$, $\zeta = \sqrt{\frac{(\ln(0.2))^2}{\pi^2+(\ln(0.2))^2}} = 0.2079$

Step 2: Find the closed loop transfer function with proportional controller K, KH(s)

$$\frac{0.2\mathrm{K}}{s^2 + 0.104s + 0.2K}$$

Step 3: Apply Routh-Hurwitz stability criterion

s^2	1	0.2K
s^1	0.104	0
s^0	b_1	

$b_1 = -\begin{vmatrix} 1 & 0.2K \\ 0.104 & 0 \end{vmatrix} = 0.0208K$

K > 0 for all times for the system to be stable, for which there will be no change in sign in the first column.

Step 4: Equate and solve with the following equation

$$\frac{\mathrm{A}}{s^2 + 2\zeta\omega_n s + {\omega_n}^2}$$

Natural frequency, $\omega_n = 0.104/(2*0.2079) = 0.25\ s^{-1}$

$K = 0.25^2/0.2 = 0.3125$

Thus, maximum value of K = 0.3125

10. For rise time of 4 seconds, to find the value of K, we need to go through a few steps

Step 1: Find the natural frequency

$\omega_n = \frac{\pi}{t_r} = 0.7854\ s^{-1}$

Step 2: Find the closed loop transfer function with proportional controller K, KH(s)

$$\frac{0.2\mathrm{K}}{s^2 + 0.104s + 0.2K}$$

Step 3: Equate and solve with the following equation

$$\frac{\mathrm{A}}{s^2 + 2\zeta\omega_n s + \omega_n{}^2}$$

K = 0.7854/0.2 = 3.927

11. Bode plot of transfer function H

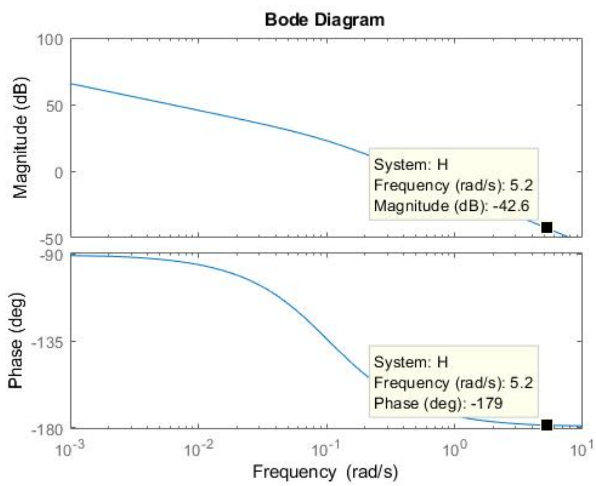

Fig 8: Gain Margin

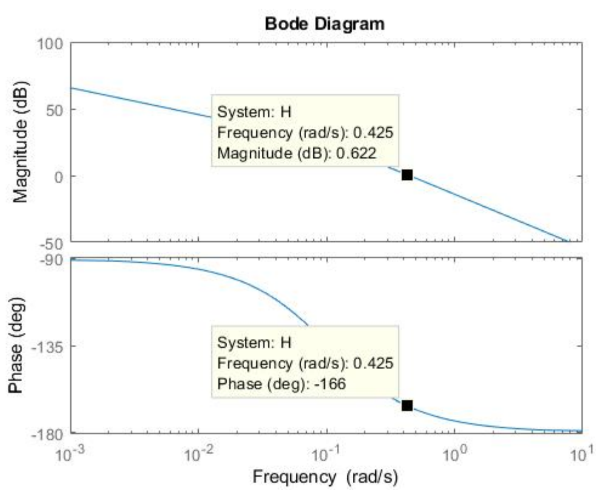

Fig 9: Phase Margin

Gain Margin = 42.6 dB, at ω = 5.2 s^{-1}

Phase Margin = -166+180 = 14^0, at ω = 0.425 s^{-1}

<u>Definitions</u>

Gain Margin - The gain margin is the amount of gain increase or decrease required to make the loop gain unity at the frequency, where the phase angle is –180°

Phase Margin - The phase margin is the difference between the phase of the response and –180° when the loop gain is 1.0

12. Root locus of the transfer function H

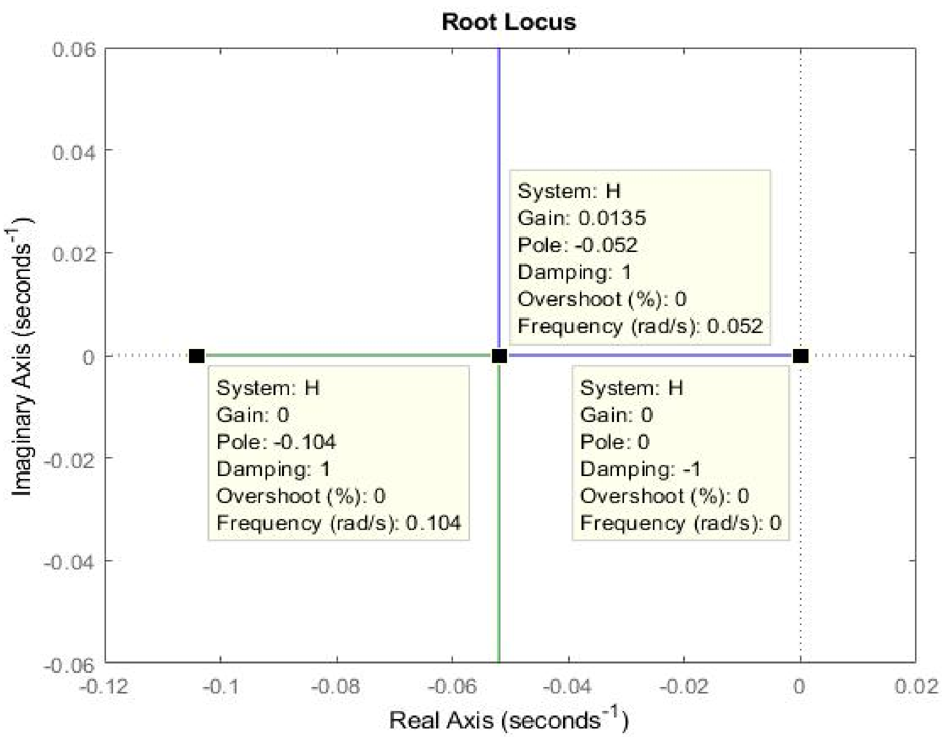

Fig 10

Root locus diagram is a graphical method for examining how the roots of a system change with variation of a certain system parameter.

From the **Fig 10**, we can observe that the system becomes unstable at a Gain of 0.0135.

13. Root locus of the transfer function H, with controller block

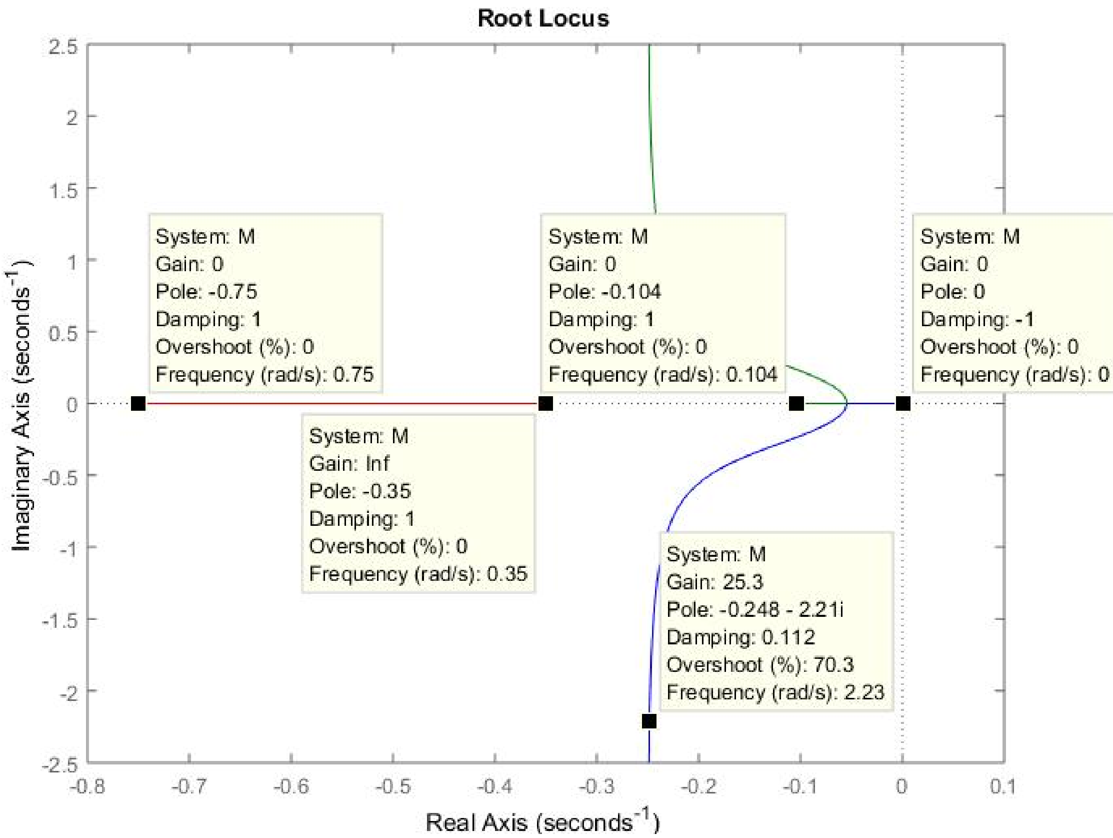

Fig 11

From the **Fig 11**, we can observe that the system is stable for all values of K because all the poles are on the left hand side plane.

Appendix

-Units are formatted as italic

Source code:

```
s = tf('s');

% Q2 to Q4
G = 0.002/(0.01*s+0.00104);
figure(1)
stepplot(G);
S = stepinfo(G);

% Q5, Q11
H = 0.002/(s*(0.01*s+0.00104));
figure(2);
stepplot(H);
T = stepinfo(H);

% Q7
StepInfoG = ["Proportional Gain",0,0,0,0;
             "RiseTime",0,0,0,0;
             "SettlingTime",0,0,0,0]
figure(3);
hold on;
K = [0.1 1 10 100];
for i = 1:4
        J = 0.002*K(i)/(0.002*K(i)+0.01*s+0.00104);
        stepplot(J);
        S = stepinfo(J);
        StepInfoG(1,i+1) = K(i);
        StepInfoG(2,i+1) = S.RiseTime;
        StepInfoG(3,i+1) = S.SettlingTime;
        legendInfo{i} = [num2str(K(i))];
end
legend(legendInfo);
hold off;

% Q8
StepInfoH = ["Proportional Gain",0,0,0,0;
             "Overshoot",0,0,0,0;
             "RiseTime",0,0,0,0;
```

```
             "SettlingTime",0,0,0,0]
figure(4);
hold on;
K = [0.1 1 10 100];
for i = 1:4
        L = 0.002*K(i)/(0.002*K(i)+s*(0.01*s+0.00104));
        stepplot(L);
        S = stepinfo(L);
        StepInfoH(1,i+1) = K(i);
        StepInfoH(2,i+1) = S.Overshoot;
        StepInfoH(3,i+1) = S.RiseTime;
        StepInfoH(4,i+1) = S.SettlingTime;
        legendInfo{i} = [num2str(K(i))];
end
legend(legendInfo);
hold off;

% Q11
figure(5);
bode(H);

% Q12
figure(6);
rlocus(H);

% Q13
figure(7);
Ks = (s+0.35)/(s+0.75);
M = Ks*H;
rlocus(M);
```