Project 1 - Control Systems Design - Fall 2021

Alan Chacko

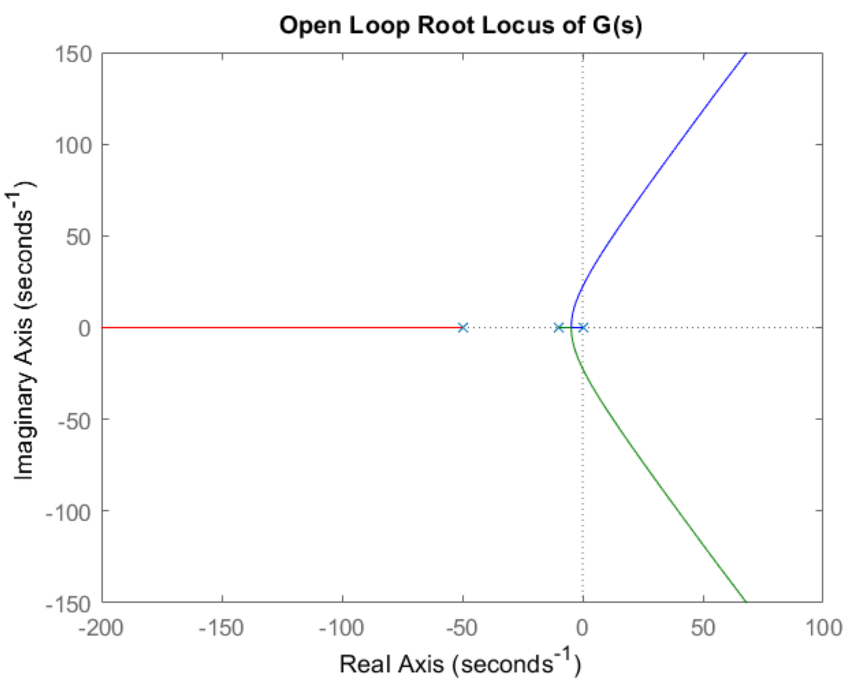
10/08/2021

1 Root Locus of G(s)

When plotting the root locus of the transfer function $G(s)$:

$$G(s) = \frac{K}{s(s+10)(s+50)}$$

We can see the movement of the poles and zeros of $G(s)$ as the gain K increases. The system becomes unstable once a gain of $K > 30000$ is achieved.



2 Parameters

We are told that overshoot is less than 7.5% and settling time of $t_s \leq 0.4s$, (assuming 5%). From this information we can derive the desired poles, or the poles for the root locus to include in order to meet the requirement.

$$\zeta = \sqrt{\frac{\ln(0.075)^2}{\pi^2 + \ln(0.075)^2}} = 0.636$$

$$\omega_n = \frac{3}{\zeta t_s} = 11.7895$$

$$x = \omega_n \zeta = 7.5000$$

$$y = \omega_d = \omega_n \sqrt{1 - \zeta^2} = 9.0963$$

From the parameters, we get the desired poles of:

$$\lambda_{1,2} = 7.5000 \pm 9.0963j$$

3 PD Controller

When designing a PD controller, we introduce an additional zero to the controller:

$$G_{c(PD)} = K(s + z_{cD})$$

Where K is the overall gain, and z_{cD} is the zero introduced. To find z_{cD}, we can use the root locus method to determine the total contributed phase and relate it to the angle of z_{cD}.

Given a pole-zero n, we refer to its angle as θ_{-n}.

The original transfer function has the poles -0, -10, and -50, and no zeros. So the contribution of each pole to the phase of the desired pole λ_1 or λ_2.

Using the rule:

$$180° + \sum \angle(s + z_n) - \sum \angle(s + p_n) = 0$$

The angles for poles are:

$$\theta_{50} = \tan^{-1}(\frac{9.1}{50 - 7.5}) = 12.09°$$

$$\theta_{10} = \tan^{-1}(\frac{9.1}{10 - 7.5}) = 74.64°$$

$$\theta_0 = 180° - \tan^{-1}(\frac{9.1}{7.5 - 0}) = 129.49°$$

The contributed phase needed is:

$$\theta_c = 180^\circ - (\theta_{50} + \theta_{10} + \theta_0) = -36.22^\circ$$

The zero z_{cD} should introduce the needed $+36.22^\circ$ of phase.

$$\theta_{z_{cD}} = 36.22^\circ$$

From the angle, we derive the real world zero:

$$\tan(\theta_{z_{cD}}) = \frac{9.1}{z_{cD} - 7.5} \Rightarrow z_{cD} = 19.92$$

To get overall gain K, we use the Magnitude rule of root locus, where:

$$K = \frac{\prod |s - p_n|}{\prod |s - z_n|}$$

Representing the magnitude of the poles/zeros n by notation L_{-n}, we can find the magnitude of all poles and zeros below:

$$L_{50} = \sqrt{(50 - 7.5)^2 + 9.1^2} = 43.463$$
$$L_{10} = \sqrt{(10 - 7.5)^2 + 9.1^2} = 9.434$$
$$L_0 = \sqrt{(7.5 - 0)^2 + 9.1^2} = 11.790$$
$$L_{z_{cD}} = \sqrt{(19.92 - 7.5)^2 + 9.1^2} = 15.395$$

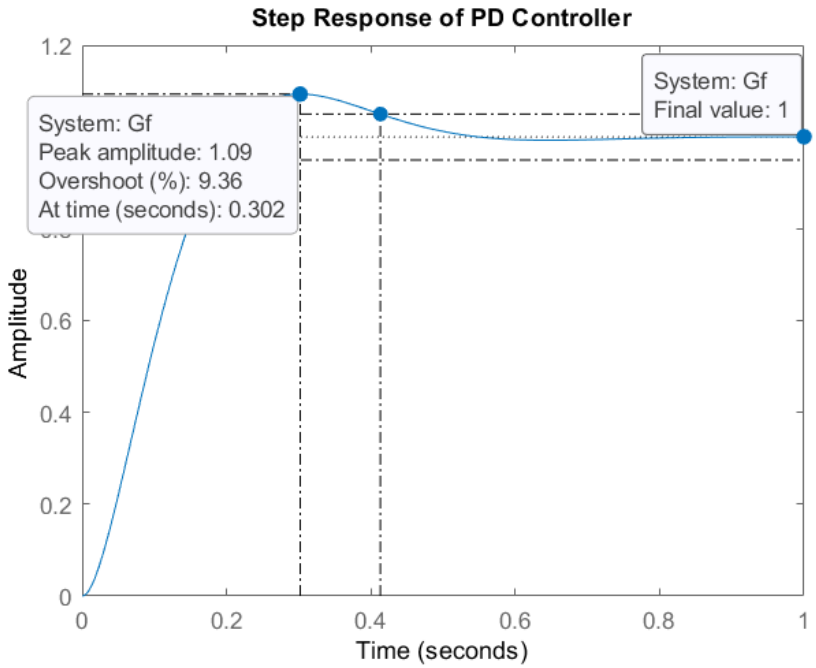
Then we plug into the gain K formula:

$$K = \frac{L_{50} L_{10} L_0}{L_{z_{cD}}} = 314$$

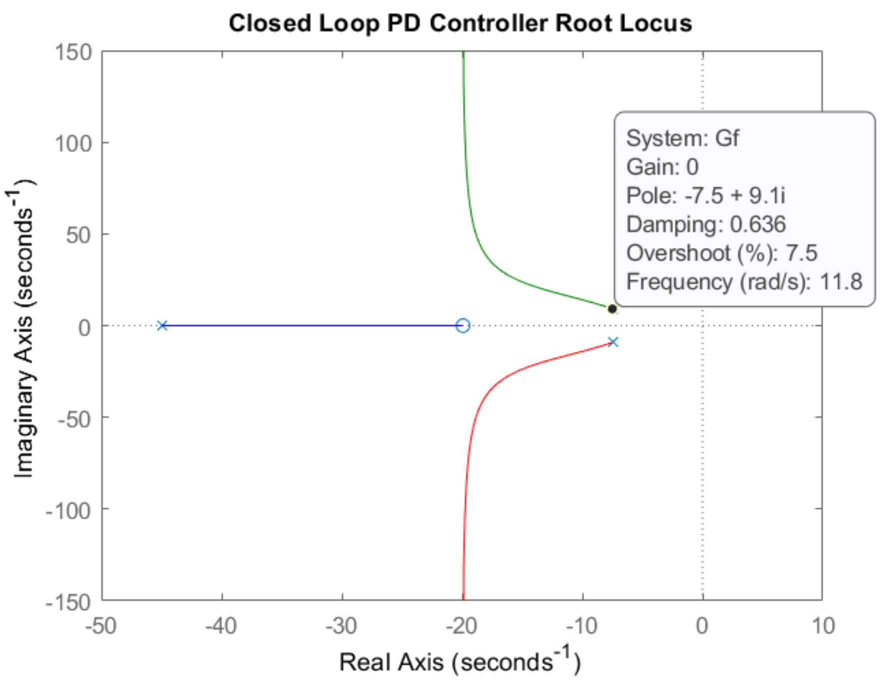
So the final PD controller is:

$$G_{c(PD)} = 314(s + 19.92)$$

Below we plot the PD step response:



Along with the root locus of the unity feedback loop of the cascade $G_{c(PD)}(s)G(s)$:



4 PID Controller

The PI contoller is to improve the steady state error. Since the system $G_{c(PD)}(s)G(s)$ is a type one system, its ramp steady state error is:

$$e_{ss(ramp)} = \frac{1}{K_v}$$

Where $K_v = \lim_{s\to 0}\{sG_{c(PD)}(s)G(s)\}$

$$K_v = \frac{314(19.92)}{(10)(50)} = 12.5$$

$$\Rightarrow e_{ss(ramp)} = 0.08$$

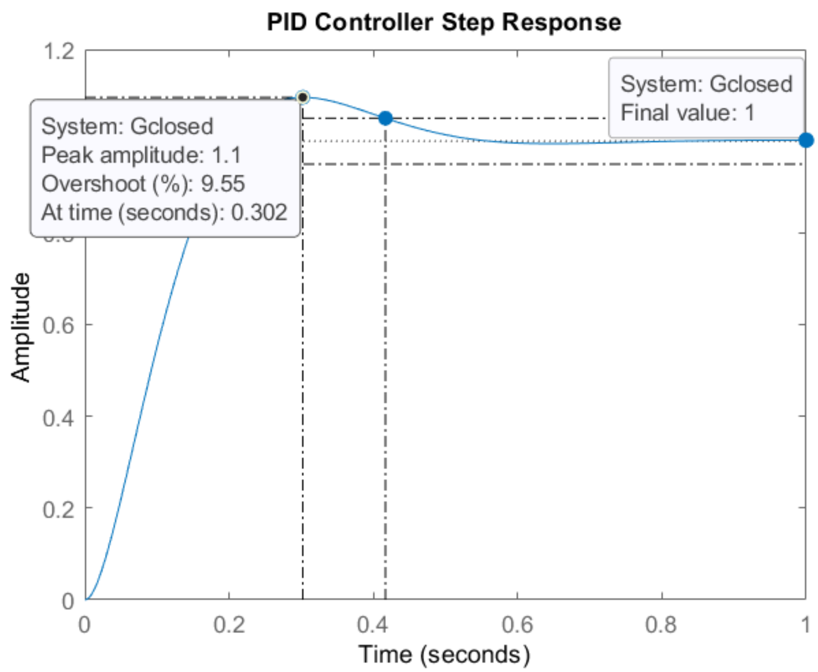
The PI controller simply introduces a pole at the origin and a zero near the origin to cancel it almost out, so steady-state ramp error decreases as time passes since we increased system type. So we need an arbitrary zero near the origin.

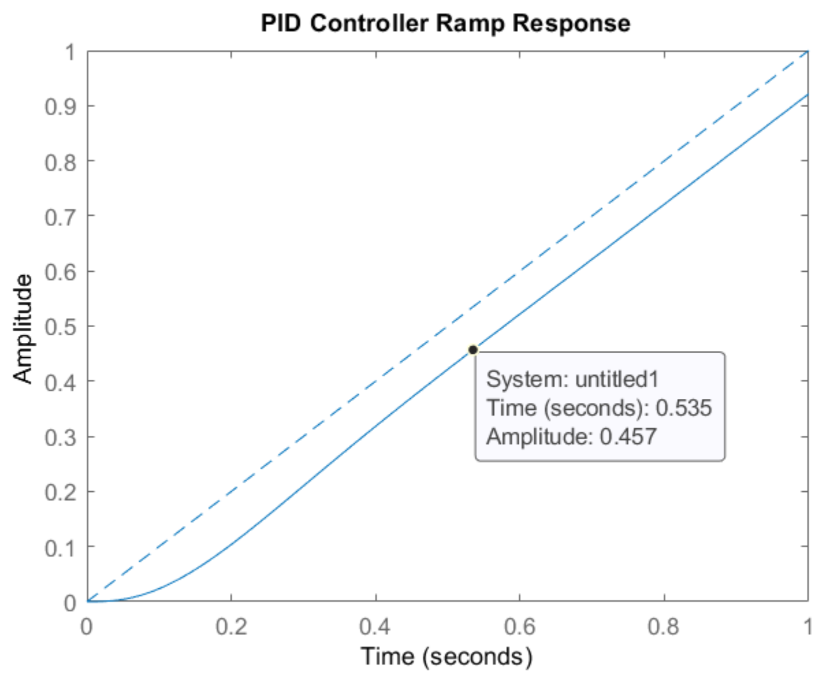
$$G_{c(PI)} = \frac{(s+0.01)}{s}$$

So our final PID controller is:

$$G_c = \frac{314(s+19.92)(s+0.01)}{s}$$

The unit step and unit ramp responses from unity feedback are below. Error is introduced due to the introduced poles and zeros not being fully accounted for. The dashed lines represent the steady state at $t \to \infty$.





5 Phase-Lead Controller

The phase-lead controller plays the same role as the PD controller. Both improve the system transient response. The desired poles derived in the Parameters section above are referenced here in calculation.

A phase-lead controller is defined as:

$$G_{\text{lead}} = \frac{K(s+z_l)}{s+p_l}$$

Since G_{lead} introduces both a zero and a pole, both a positive and negative phase contribution are made to the root locus. We can use the zero to cancel out a non-dominant pole, based on the pole contributions to total phase.

Since,

$$\theta_c = 180^\circ - \theta_{50} - \theta_{10} - \theta_0 = -36.21^\circ$$

We can arbitrarily set $z_l = 10$, since the pole θ_{10} contributes 74.64° which leads to the net negative contribution. The other non-dominant pole

θ_{50} only contributes 12.08° phase, so canceling it out with a zero would require another zero to make θ_c positive.

If $z_l = 10$, then the net contribution equals the desired contribution for the pole:

$$\theta_{p_l} = \theta_c = 180° - \theta_{50} - \theta_0 = 38.41°$$

From θ_{p_l}, we can derive $p_l = 18.9712$.

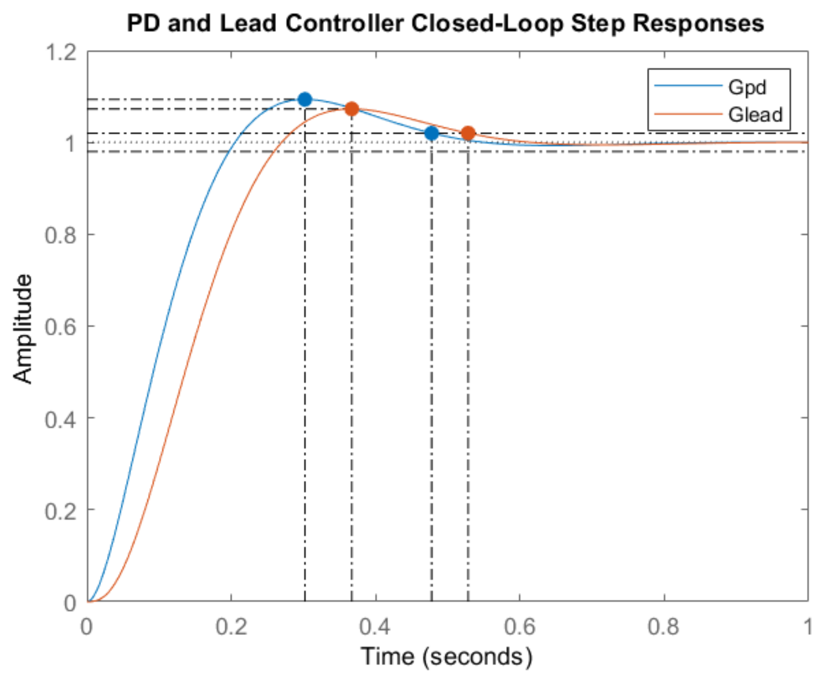
For overall gain K, we can use the magnitude formula used earlier. Note L_{10} is cancelled out.

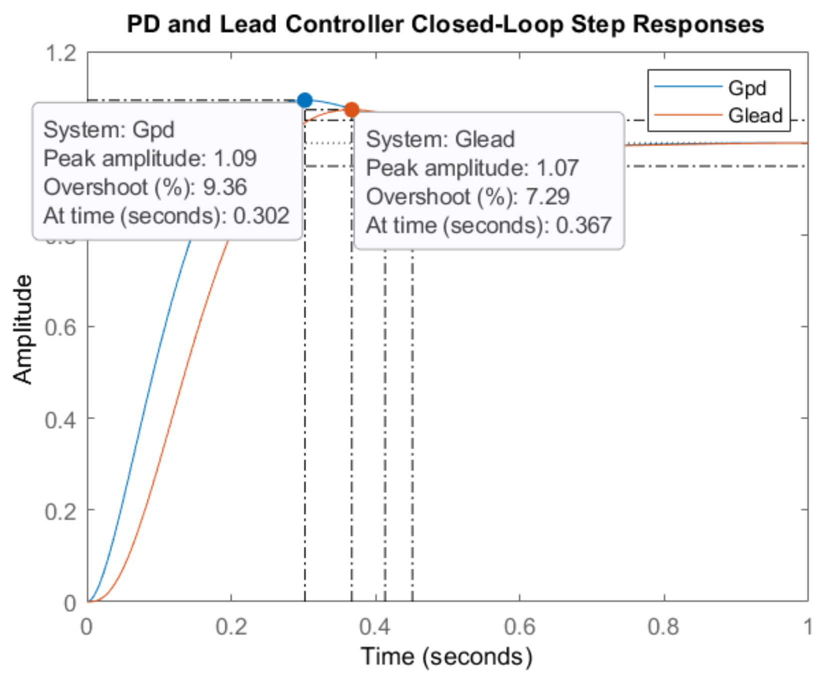
$$L_{p_l} = \sqrt{(18.97 - 7.5)^2 + 9.1} = 14.64$$
$$K = L_{50} L_{p_l} L_0 = 7501.6$$

The lead controller is:

$$G_{\text{lead}} = \frac{7502(s + 10)}{s + 18.97}$$

Below we plot the PD-controlled step response and the phase-lead step response (assuming phase-lag was a typo in the project description). Note PD introduces only a zero, while phase-lead introduces both a pole and zero. This leads to a shift in settling time for the lead controller, but a better overshoot in comparison to the PD controller.


PD and Lead Controller Closed-Loop Step Responses
Amplitude
Time (seconds)
Gpd
Glead
0
0.2
0.4
0.6
0.8
1
1.2


PD and Lead Controller Closed-Loop Step Responses
Amplitude
Time (seconds)
Gpd
Glead
System: Gpd
Peak amplitude: 1.09
Overshoot (%): 9.36
At time (seconds): 0.302
System: Glead
Peak amplitude: 1.07
Overshoot (%): 7.29
At time (seconds): 0.367

6 Phase-Lead-Lag Controller

For the phase lag controller, we define:

$$G_{\text{lag}} = \frac{s + z_{\text{lag}}}{s + p_{\text{lag}}}$$

Where the ratio $\frac{z_{\text{lag}}}{p_{\text{lag}}}$ should be as large as possible. Here the steady state ramp error is different:

$$e_{ss(ramp)} = \frac{1}{K_v}$$

Where $K_v = \lim_{s\to 0}\{sG_{\text{lead}}(s)G(s)\}$

$$K_v = \frac{7502}{(18.97)(50)} = 7.91$$
$$\Rightarrow e_{ss(ramp)} = 0.1264$$

Since our lead/lag steady state should be near 0, we can select an arbitrary value near 0, like 0.001. The we can define the ratio $\frac{z_{\text{lag}}}{p_{\text{lag}}}$ below:

$$\frac{z_{\text{lag}}}{p_{\text{lag}}} = \frac{e_{\text{lead}}}{e_{\text{lead-lag}}} = \frac{0.1264}{0.001}$$

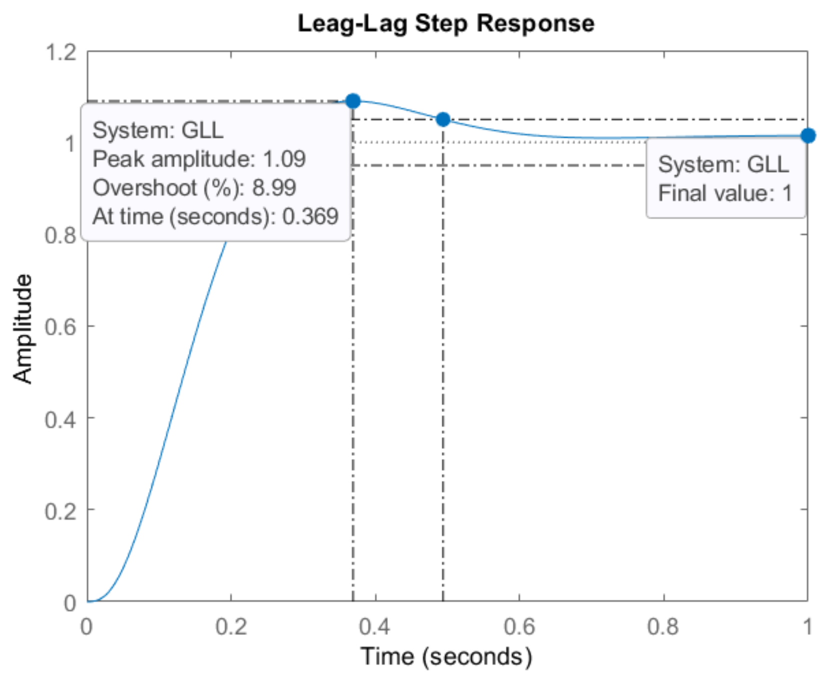
So our lag controller is:

$$G_{\text{lag}} = \frac{s + 0.1264}{s + 0.001}$$

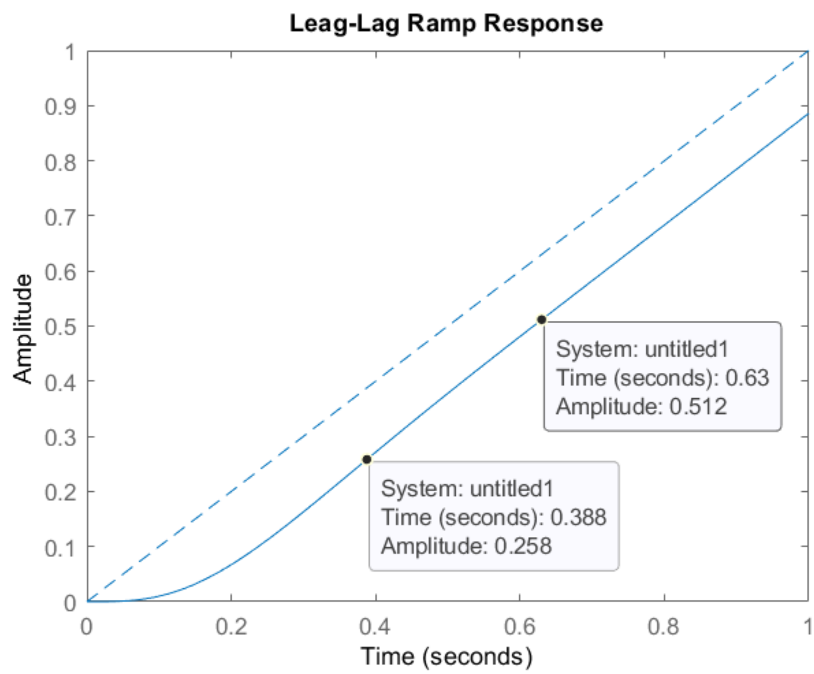
Combining it with the lead controller builds our final lead-lag controller:

$$G_c = \frac{7502(s + 10)(s + 0.1264)}{(s + 18.97)(s + 0.001)}$$

The closed-loop step response is:

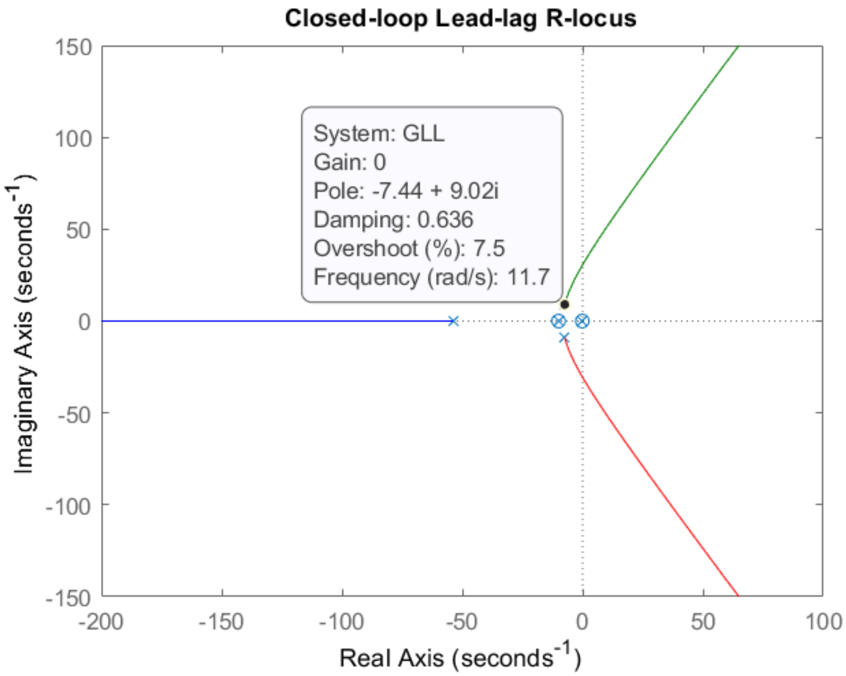


The closed-loop ramp response is:



The ramp response still displays initial error, but this is reduced to 0 as time increases. The two points indicate a decreasing error, with steady state occuring at a slope of 1 (where time equals amplitude).

Note, the overshoot in the step response of the lead-lag is off by a margin when compared to the lead only step response. This is due to error from not accounting the additional zeros and poles added by the lag controller. We can see the requirements are met from the root locus of the final system at the desired poles.



7 Appendix

The following section will contain the MATLAB code for the project. Each section is marked by the question letter used in the project report.

```
s = tf('s');
t = 0:0.001:1;

%% Question A %%

% Defining transfer function G(s) and plotting root locus
k1 = 1;
z1 = [inf;inf;inf];
p1 = [0;-10;-50];
[n1,d1] = zp2tf(z1,p1,k1);
G = tf(n1,d1);

figure;
rlocus(G);
title("Open Loop Root Locus of G(s)")

% Parameters
```

```
mpos = 0.075;
ts = 0.4;

zeta = sqrt(log(mpos)^2 / (pi^2 + log(mpos)^2));
natfreq = 3/(zeta * ts);
x = natfreq*zeta; % b characteristic
y = natfreq*sqrt(1 - zeta^2); % Damping frequency

% Poles
lambda1 = x + y*1i;
lambda2 = x - y*1i;

% Find controller zero zA
theta50 = atand(y/(50 - x));
theta10 = atand(y/(10 - x));
theta0 = 180 - atand(y/x);
thetap = theta50 + theta10 + theta0;
thetaA = -1*180 + thetap;
zA = y/tand(thetaA) + x;

% Find controller gain K
L50 = sqrt((50 - x)^2 + y^2);
L10 = sqrt((10 - x)^2 + y^2);
L0 = sqrt((x)^2 + y^2);
LA = sqrt((zA - x)^2 + y^2);
K = (L50*L10*L0)/LA;

% PD contoller
numPD = [ K K*zA ];
demPD = 1;
Gpd = tf(numPD, demPD);

% Open Loop
Gcpd = series(Gpd,G);
[nG, dG] = tfdata(Gcpd,'v');
figure;
rlocus(Gcpd);
title("Open Loop PD Controller Root Locus")

% Closed Loop
[nGf, dGf] = feedback(nG, dG, 1, 1, -1);
Gf = tf(nGf,dGf);
pG = pole(Gf);
zG = zero(Gf);
figure;
rlocus(Gf);
title("Closed Loop PD Controller Root Locus")

% Step Response
```

```
figure;
step(Gf, t);
title('Step Response of PD Controller')

%% Question B %%

% Ramp Steady State Error
% Since G(s) is a type1 system, ramp steady state equals 1/Kv
Kv = 1/(50*10);
ess = 1/Kv; % evaluates to 500

% PI Controller
numPI = [ 1 0.01 ];
demPI = [ 1 0 ];
Gpi = tf(numPI,demPI);
Gc = series(Gpi,Gcpd);
[nGc,dGc] = tfdata(Gc,'v');
[nGclose, dGclose] = feedback(nGc, dGc, 1, 1, -1);
Gclosed = tf(nGclose, dGclose);
figure;
rlocus(Gclosed);
title('Controller R-locus')

% Step Response
figure;
step(Gclosed,t);
title('PID Controller Step Response');

% Ramp Response
figure;
plot(t,t,'--');
hold on;
step(Gclosed/s, t);
title('PID Controller Ramp Response');

%% Question C %%

% Phase-Lead Controller
% Since the available contributed phase is less than zero (thetac = -36.31), we can cancel
% out one of the non-dominant poles, so the phase goes up to a positive
% value for the pole to contribute to. Arbitrarily, if zl = 10, then pl is
% below:

zlead = 10;
thetapl = 180 - (theta50 + theta0);
px = y/tand(thetapl) + x;
```

```
Lpl = sqrt((px - x)^2 + y^2);
K2 = L50*L0*Lpl;

nplead = [K2 K2*zlead];
dplead = [1 px];
plead = tf(nplead,dplead);

% Open Loop
Gopen = series(plead,G);
[nGopen, dGopen] = tfdata(Gopen,'v');
figure('Name', 'Open Loop Lead R-locus');
rlocus(Gopen);
title('Open Loop Lead R-locus');

% Closed Loop
[nLead, dLead] = feedback(nGopen,dGopen,1,1,-1);
Glead = tf(nLead,dLead);
pGlead = pole(Glead);
zGlead = zero(Glead);
figure('Name', 'Closed Loop Lead R-locus');
rlocus(Glead);
title('Closed Loop Lead R-locus')

% Step Response
figure('Name', 'Lead Step Response');
step(Gf, t);
hold on;
step(Glead, t);
title('PD and Lead Controller Closed-Loop Step Responses');

% Ramp Response
figure('Name', 'Lead Ramp Response');
plot(t,t,'--');
hold on;
step(Glead/s, t);
title('Lead Ramp Response');

%% Question D %%
% Given a type 1 system, the error is proportional to the
    desired ramp steady
% state error by 1/Kv
syms N(w) w
[limn, limd] = tfdata(s*Gopen,'v');
N(w) = simplify(poly2sym(limn,w)/poly2sym(limd,w));
Kv = vpa(N(0));
ess = double(1/Kv);

% once we add the controller below, ess is decreased by a
    factor of
```

```
% 1/(1000*ess) giving an improved steady state error of 0.001. We could
% decrease pc even close to 0, but avoid zero so system type doesn't change.
nlag = [1 ess];
dlag = [1 0.001];
Glag = tf(nlag,dlag);

% Open Loop
Gc = series(Glag,Gopen); % connects lead, lag, and G
[nGc, dGc] = tfdata(Gc,'v');

% Closed Loop
[nLL, dLL] = feedback(nGc,dGc,1,1,-1);
GLL = tf(nLL,dLL);
pGLL = pole(GLL);
zGLL = zero(GLL);
figure('Name', 'Closed-loop Lead-lag R-locus');
rlocus(GLL);
title('Closed-loop Lead-lag R-locus');

% Step Response
figure('Name', 'Leag-Lag Step Response');
step(GLL, t);
title('Closed-Loop Leag-Lag Step Response');

% Ramp Response
figure('Name', 'Leag-Lag Ramp Response');
plot(t,t,'--');
hold on;
step(GLL/s, t);
title('Closed-Loop Leag-Lag Ramp Response');
```