```
clear;
close all;


%constants taken from http://psas.pdx.edu/rollcontrol/
Izz = 0.08594;       %m2*kg
kd = .002;           %damping coefficient
kt = .9141;          %Nm/rad  torsion rod spring constant exper. determined
lf = .0762;          %m distance from fin to z axis


%State Space equation
%assuming partial-state feedback (angular vel.)
%model equation from http://psas.pdx.edu/rollcontrol/
a = [-kd/Izz 0; 1 0];
b = [lf/Izz; 0];
c = [0 1];
d = 0;

sys_ss = ss(a,b,c,d,'statename',{'Angular Accel','Angular Vel'});

%Check it out
figure
pzmap(sys_ss)
title('Pole and Zero map of State Space Model')

figure
step(sys_ss)
title('Initial Step Response of State Space Equation')

%convert State Space to Laplace Transfer Function
[num den] = ss2tf(a,b,c,d);

%create root locus plot
figure
rlocus(num,den)
title('Continuous Initial Root locus')


%Create proportional controller

s = tf('s');

Gp = tf(num,den)                %Gp is the system process

kp = .000153;                   %proportional gain chosen from rl graph

Gc = kp;                        %Gc is the Controller process

cltf = feedback(Gc*Gp,1);       %closed loop response

figure
```

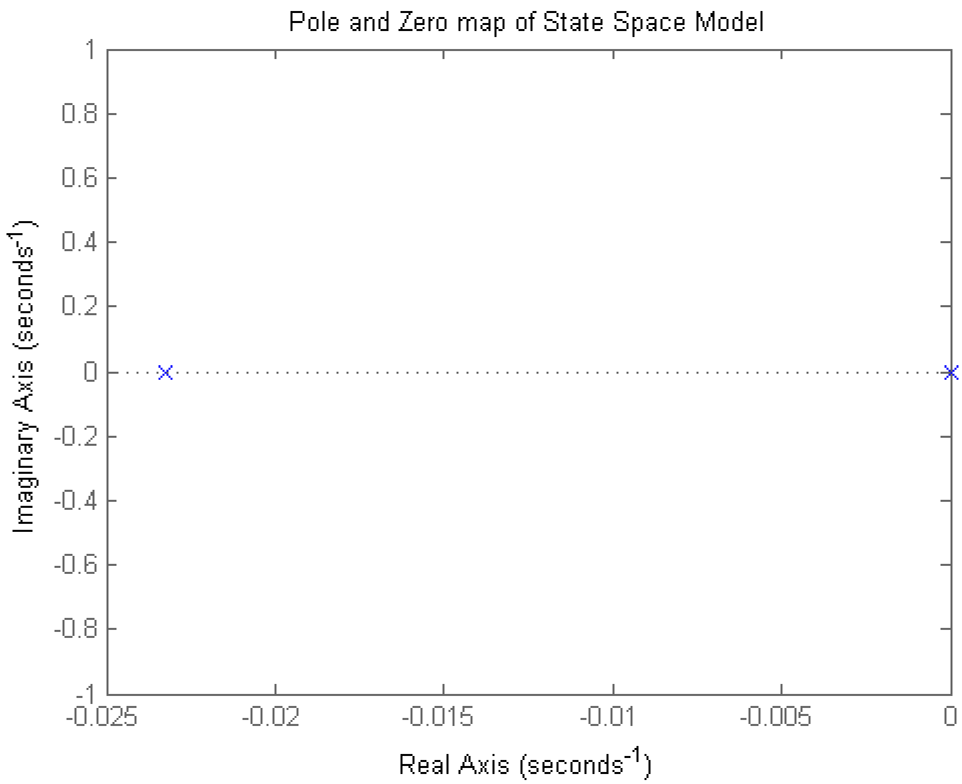
```
step(cltf)
title('Continuous system with kp')


%Now add a derivitive gain to speed things up

figure
rlocus(Gc*Gp)
title('Continuous system kp Root Locus')


kd = 100;                        %derivitave gain

Gc = kp+kd*s                     %new controller process

cltf = feedback(Gc*Gp,1)         %new closed loop response

figure
step(cltf)
title('Continuous system PD controller Step Response')

figure
impulse(cltf)
title('Continuous system PD controller Impulse Response')

%{
No Integrator required
Sys is arleady type 2,
steady state error goes to zero
if integrator is added the sys blows up
%}


%now model as discrete time and hope we can get close to the continuous


Ts = .01;                        %sample time of 10 ms chosen.
                                 %Not sure what the actual sensors can do

z = tf('z',Ts);

sysd = c2d(Gp,Ts,'zoh')          %convert continous to discrete

figure
rlocus(sysd)
title('Initial Discrete System Root locus')

kp = .000153;                    %new proportional gain chosen from rl graph

Gc = kp;                         %new controller process

cltf = feedback(Gc*sysd,1)       %new closed loop response

figure
```

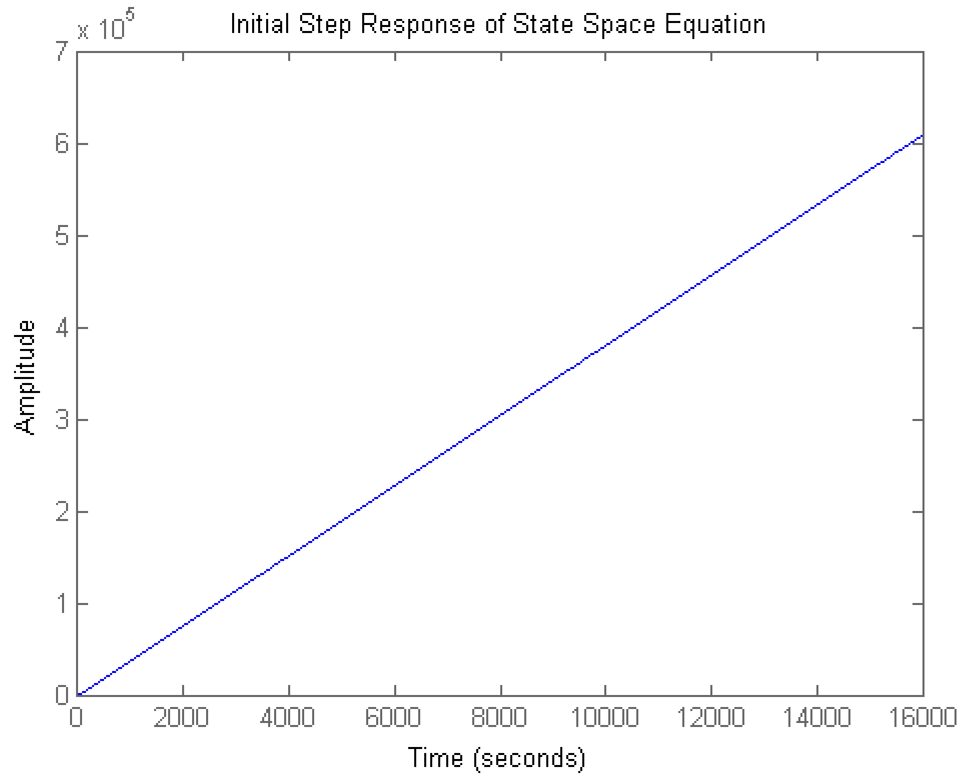
```
step(cltf)
title('Discrete system P controller Step Response')



%Now add a derivitive gain to speed things up

figure
rlocus(Gc*sysd)
title('Discrete system kp Root Locus')

kd = 1000;                      %new derivative gain chosen from rl graph
                                %acceptable values are 100-9000
                                %they depened on the systems capabilities
                                %larger value = faster rise time

Gc = kp+kd*((z-1)/z)            %new controller process

cltf = feedback(Gc*sysd,1)      %new closed loop response

figure
step(cltf)
title('Discrete system PD controller Step Response')

figure
impulse(cltf)
title('Discrete system PD controller Impulse Response')

final_gp = d2c(sysd,'tustin')
final_gc = d2c(Gc,'tustin')
```

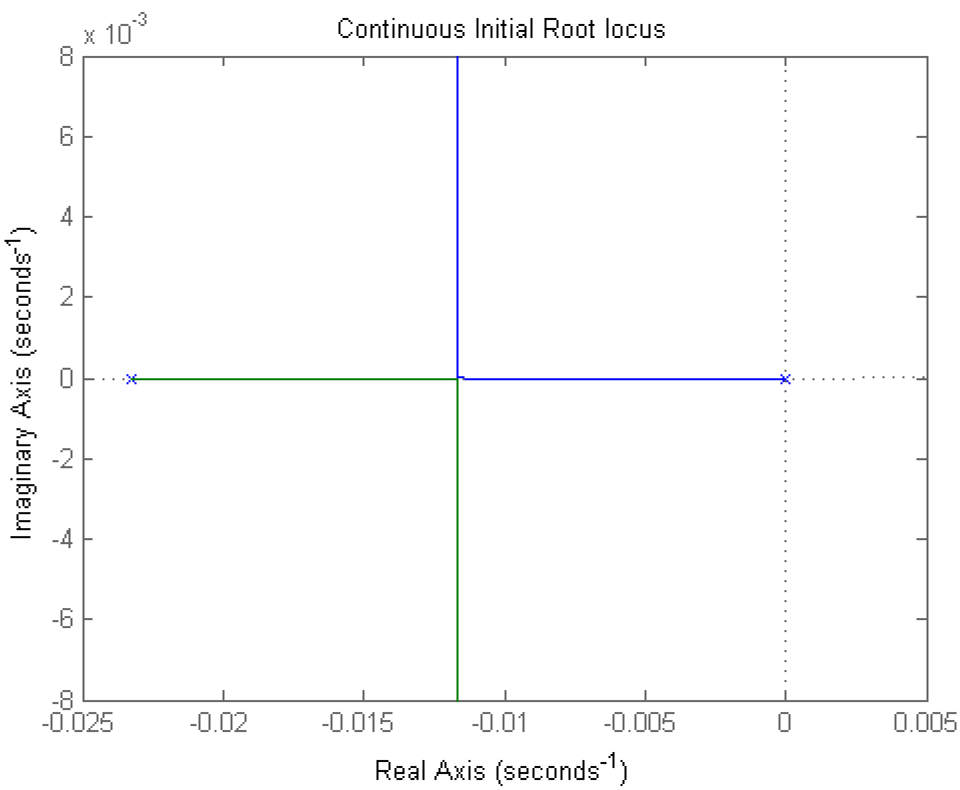
```
Gp =

      0.8867
  ---------------
  s^2 + 0.02327 s

Continuous-time transfer function.


Gc =

  100 s + 0.000153

Continuous-time transfer function.


cltf =

     88.67 s + 0.0001357
  -------------------------
  s^2 + 88.69 s + 0.0001357
```

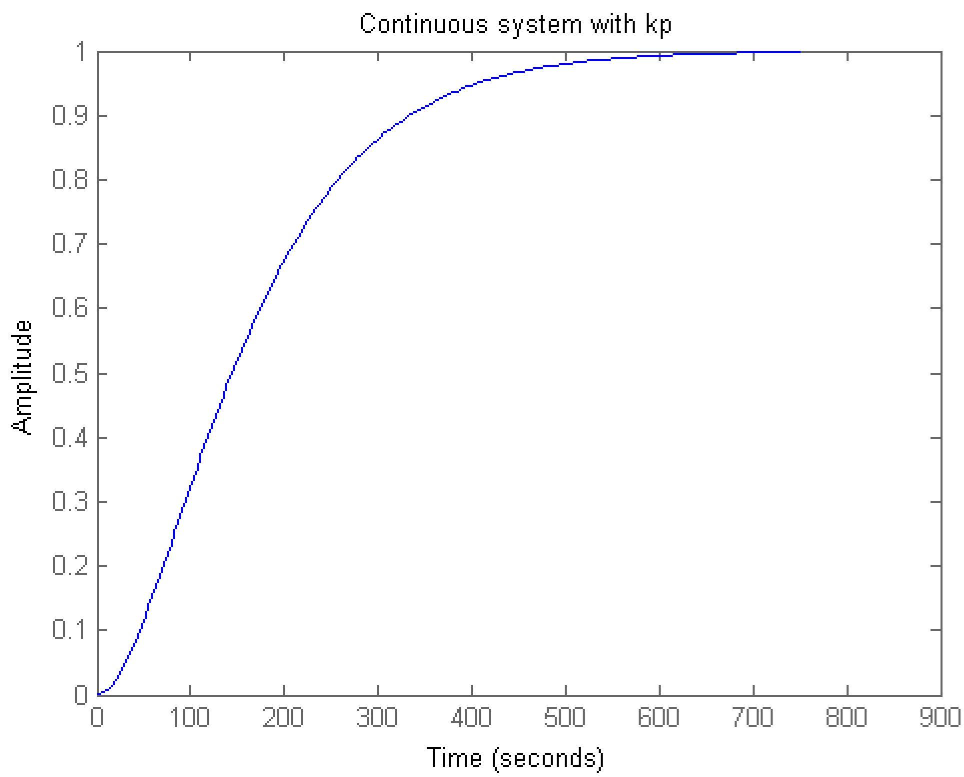
```
Continuous-time transfer function.


sysd =

  4.433e-05 z + 4.433e-05
  -----------------------
    z^2 - 2 z + 0.9998

Sample time: 0.01 seconds
Discrete-time transfer function.


cltf =

  6.782e-09 z + 6.782e-09
  -----------------------
    z^2 - 2 z + 0.9998

Sample time: 0.01 seconds
Discrete-time transfer function.


Gc =

  1000 z - 1000
  -------------
        z

Sample time: 0.01 seconds
Discrete-time transfer function.


cltf =

  0.04433 z^2 - 3.432e-06 z - 0.04433
  -----------------------------------
     z^3 - 1.955 z^2 + z - 0.04433

Sample time: 0.01 seconds
Discrete-time transfer function.


final_gp =

  -8.598e-10 s^2 - 0.004433 s + 0.8867
  ------------------------------------
      s^2 + 0.02327 s + 9.947e-14

Continuous-time transfer function.


final_gc =
```

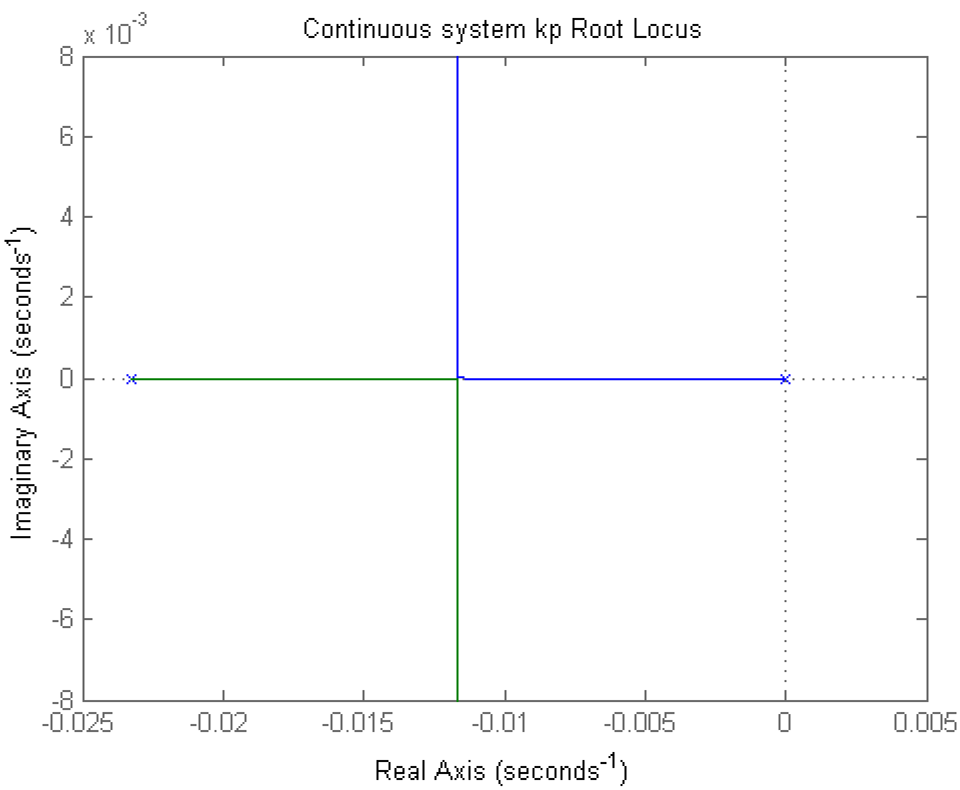
```
2000 s + 0.0306

---------------
     s + 200

Continuous-time transfer function.
```

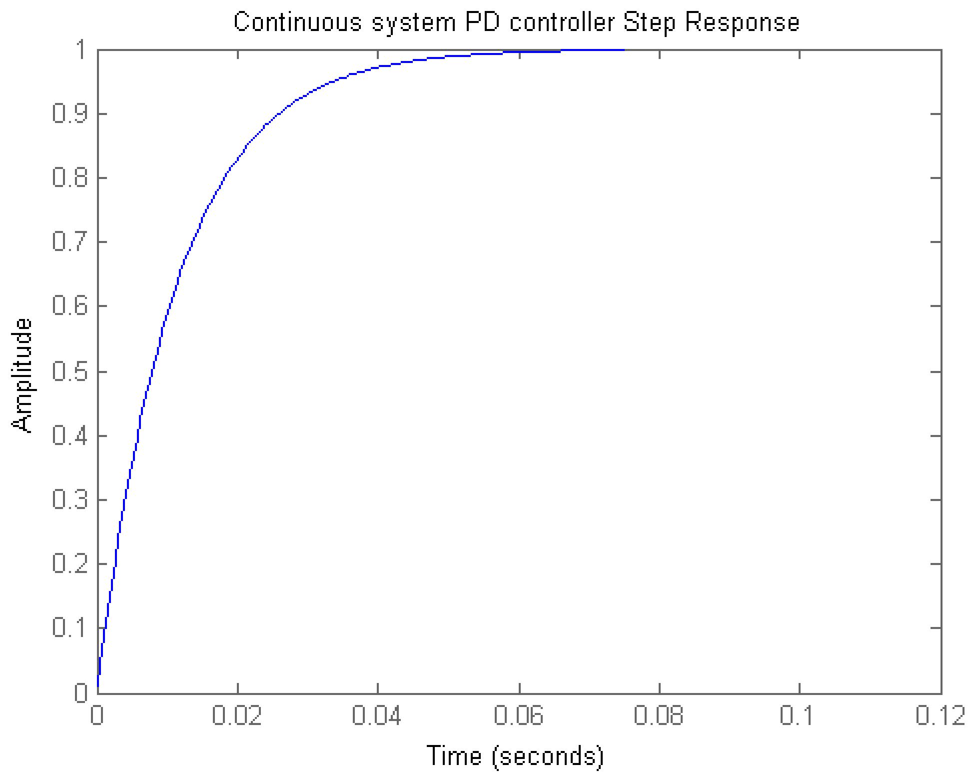


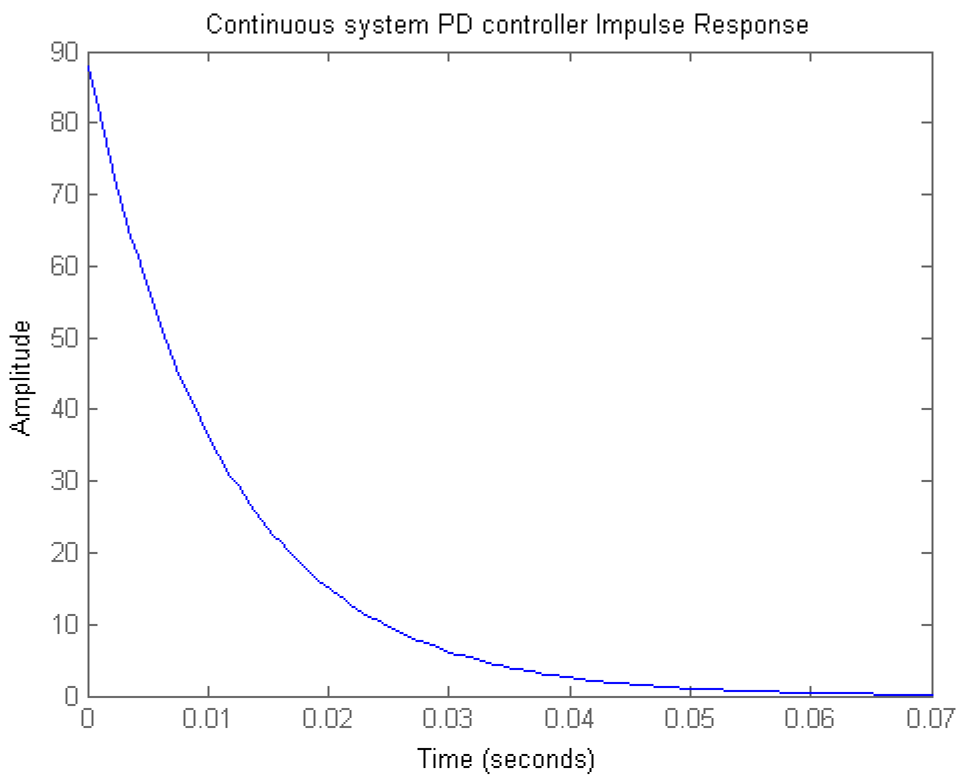

Initial Step Response of State Space Equation
x 10^5
Amplitude
Time (seconds)
0
1
2
3
4
5
6
7
0
2000
4000
6000
8000
10000
12000
14000
16000

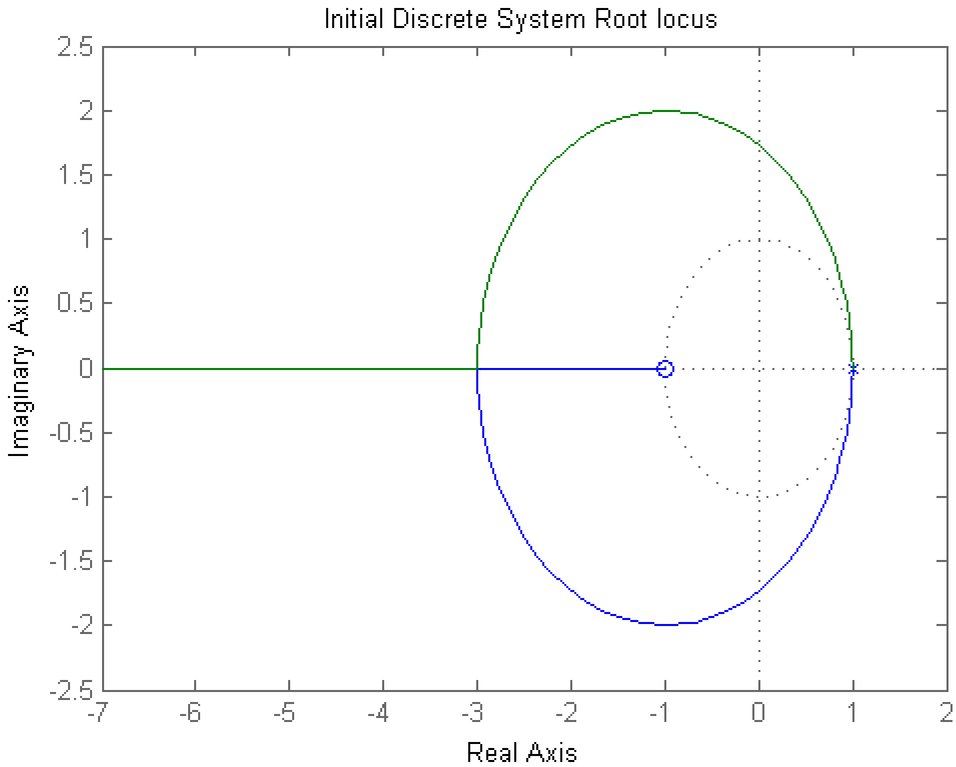

Continuous Initial Root locus
x 10^-3
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)
8
6
4
2
0
-2
-4
-6
-8
-0.025
-0.02
-0.015
-0.01
-0.005
0
0.005

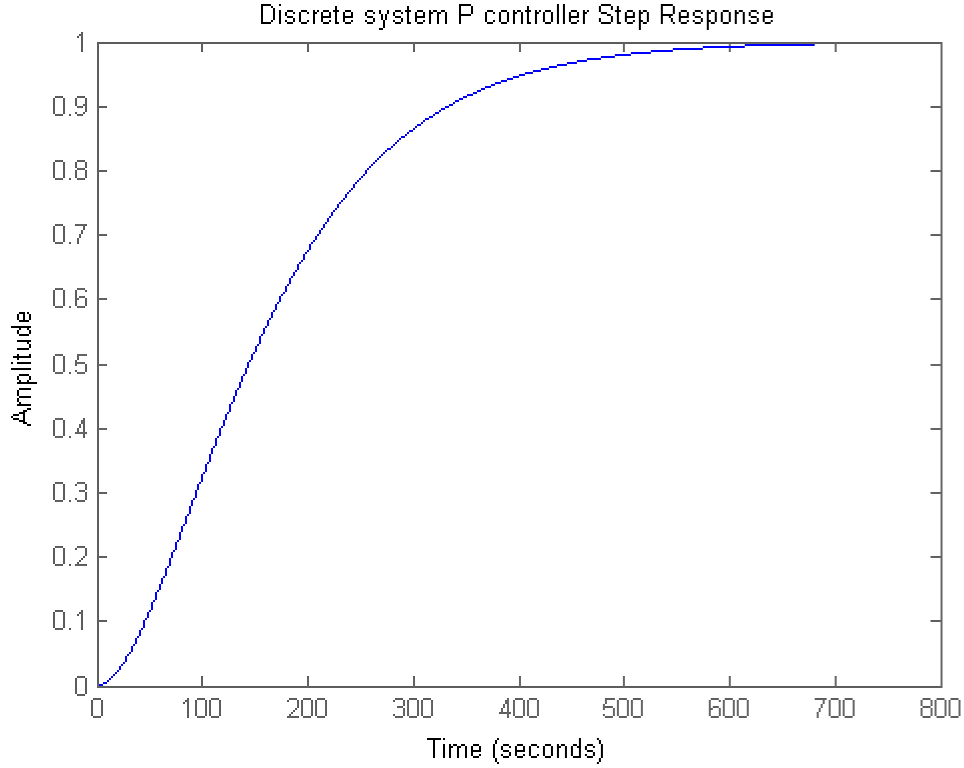

Continuous system with kp
Amplitude
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
100
200
300
400
500
600
700
800
900

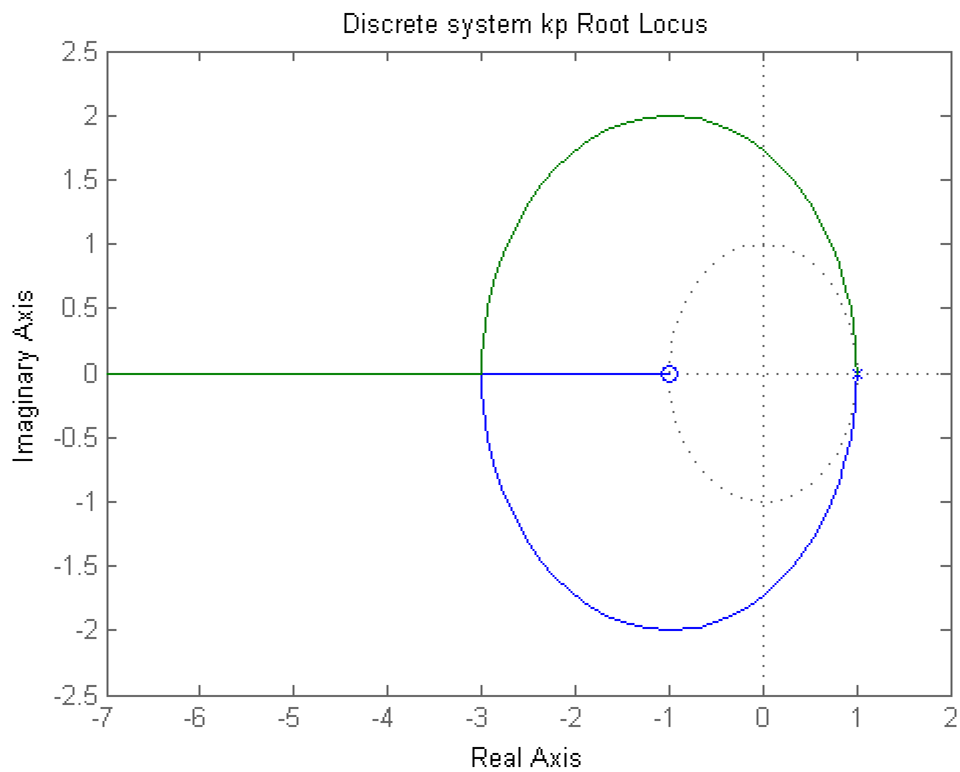

Continuous system kp Root Locus
x 10^-3
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)
8
6
4
2
0
-2
-4
-6
-8
-0.025
-0.02
-0.015
-0.01
-0.005
0
0.005

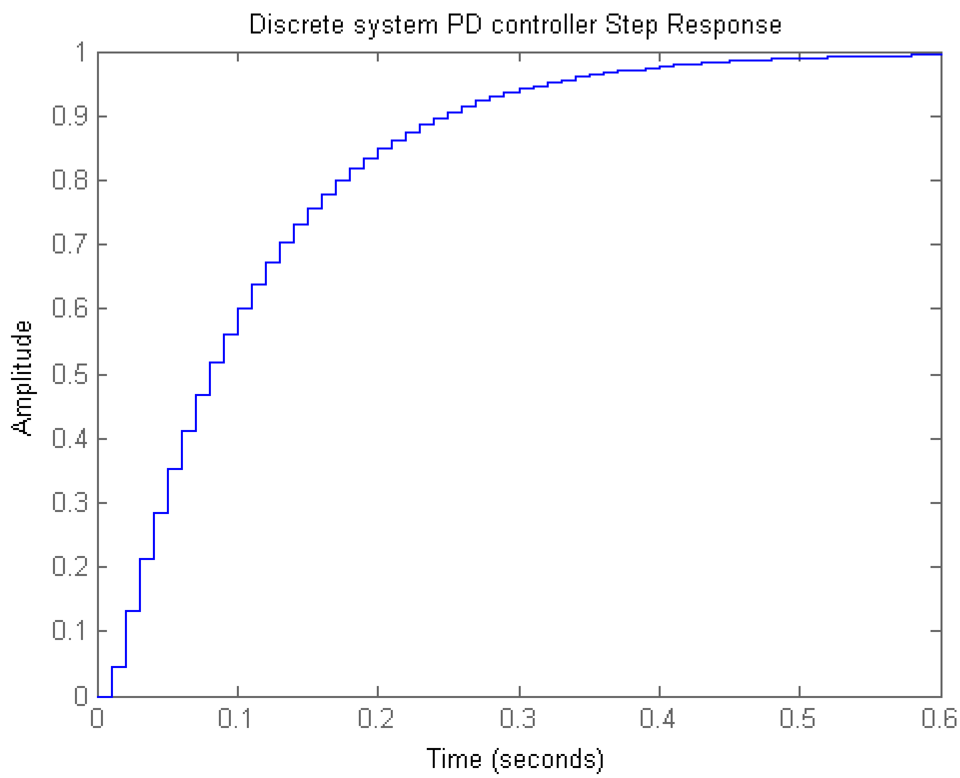

Continuous system PD controller Step Response
Amplitude
Time (seconds)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.02
0.04
0.06
0.08
0.1
0.12

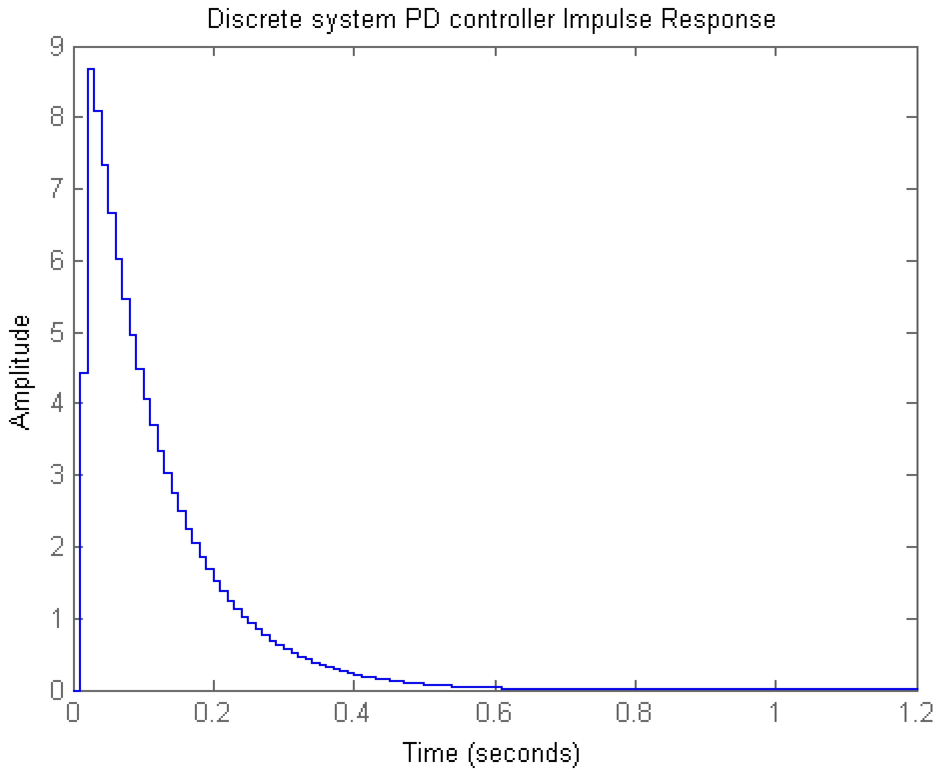

Continuous system PD controller Impulse Response
Amplitude
Time (seconds)
0
10
20
30
40
50
60
70
80
90
0.01
0.02
0.03
0.04
0.05
0.06
0.07


Initial Discrete System Root locus
Imaginary Axis
Real Axis
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-7
-6
-5
-4
-3
-2
-1
0
1
2


Discrete system P controller Step Response
Amplitude
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
100
200
300
400
500
600
700
800


Discrete system kp Root Locus
Imaginary Axis
Real Axis
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-7
-6
-5
-4
-3
-2
-1
0
1
2


Discrete system PD controller Step Response
Amplitude
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6



Published with MATLAB® R2013b