Studio 9: Root Locus with MATLAB

Goal

The goal of this studio session is to use MATLAB to analyze and design control systems based on the root locus method. Everything in **red** is intended to be typed at a MATLAB command prompt. Results that should be displayed from MATLAB are listed in **blue**.

MATLAB commands used in this session:

In this studio, you will practice some root locus-specific commands in MATLAB. The following MATLAB commands will be used. You can use "help command" to get information on the usage of a specific command.

Basic Functions:

```
help                  - MATLAB help function
```

Root Locus Functions:

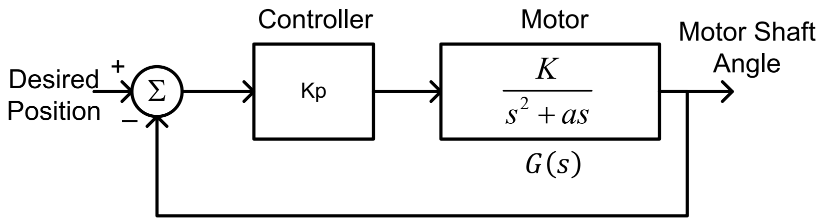
```
rlocus(sys)          - Calculates and plots the locus of the roots of sys
rlocfind             - Finds the root locus gains for a given set of roots
pzmap                - Pole-zero map of a linear system
sgrid                - Plot lines of constant damping ratio
```

Root Locus Design GUI Tools:

```
rltool               - Opens the SISO Design Tool for root locus design.
```

1.1 Motivation for Root Locus Plots

The root locus of a closed-loop system T(s) is the locus of all possible locations of the closed-loop system poles as a specified parameter of the system is varied. A common application of the root locus technique is to observe the variation in closed-loop poles as a linear gain element, K, is varied from zero to infinity. For example, this could help us determine a gain that will meet certain transient response specifications.
For example, consider the closed-loop system:



The closed-loop transfer function is

$$T(s) = \frac{K_pG(s)}{1+K_pG(s)} = \frac{\frac{K_pK}{s(s+a)}}{1+\frac{K_pK}{s(s+a)}} = \frac{K_pK}{s(s+a)+K_pK} = \frac{K_pK}{s^2+as+K_pK}$$

The poles of the closed-loop system are those values of s which satisfy

$$1 + K_pG(s) = 0$$

$$\text{or} \quad s^2 + as + K_pK = 0$$

We would like to see how the behavior of this system changes as K_p changes – let's try some different values of K_p. We could obviously play around with different values of K_p (trial and error approach) to get the desired step response for this system, but this can take a while. Another approach would be to analytically calculate the poles of this transfer function and solve for a value of K_p that will place the poles at the point on the s-plane for a desired transient response.

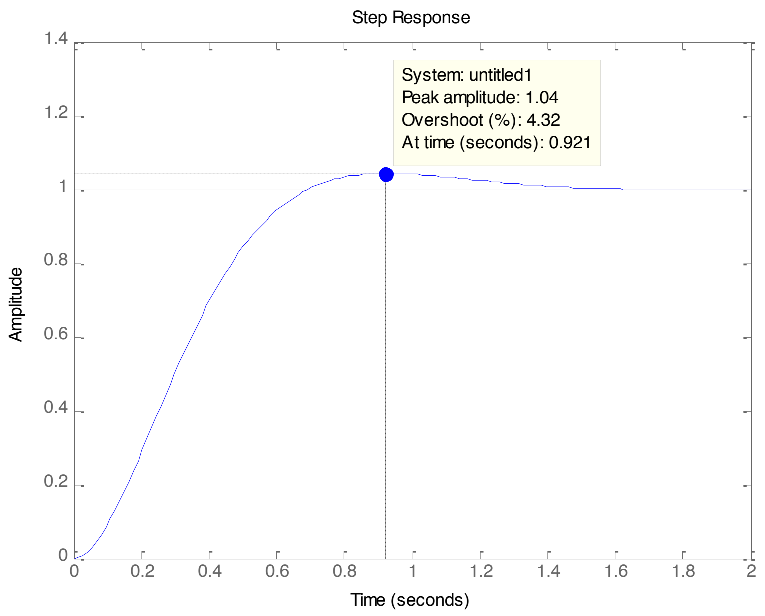
For example:

$$s = -\frac{a}{2} \pm \frac{j\sqrt{4K_pK - a^2}}{2}$$

For an underdamped response with damping ratio $\zeta = 1/\sqrt{2}$, we can set the magnitude of the real and imaginary parts of these roots equal to each other.

$$a = \sqrt{4K_pK - a^2} \rightarrow K_p = \frac{a^2}{2K} \rightarrow T(s) = \frac{a^2/2}{s^2 + as + a^2/2}$$

The step response with this K_p value provides a relatively fast response with a small overshoot as is expected from this ζ value. However, this approach becomes more difficult if we have more complex systems (i.e., more complex transfer functions).



A better approach for finding good values of K_p is to use the root locus method.

1.2 Basics of Root Locus Using MATLAB

The root locus plots the path of the closed-loop (CL) system poles, i.e., roots of the denominator polynomial of the CL transfer function, as a parameter, usually a gain like K_p above, changes. In the lecture, we already discussed how to manually sketch the root locus diagram – a valuable skill since it allows the control system designer to develop the intuition necessary to see how changes to the loop transfer function $K_pG(s)$ will affect the closed-loop system dynamics. However, as you have seen, this is a time consuming task, and MATLAB offers several powerful tools for plotting and analyzing the root locus. You will learn these tools in this studio.

MATLAB's rlocus command uses a slightly different system architecture as below.

```
            +------+
----->O------>| SYS  |----+---->
     -|       +------+    |
      |                   |
      |        +-----+    |
      +--------| Kp  |<---+
               +-----+
```

In this case, if sys represents G(s), the closed-loop transfer function T(s) is represented by:

$$\mathrm{T(s)} = \frac{\mathrm{G(s)}}{1 + \mathrm{K_pG(s)}}$$

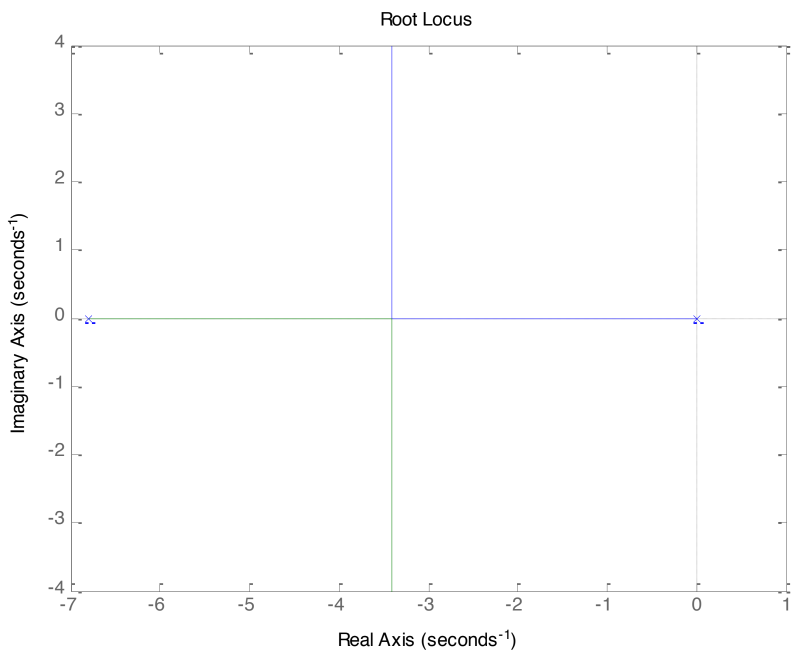
and thus the poles of the closed loop system are still those values of s which satisfy

$$1 + \mathrm{K_pG(s)} = 0$$

Using MATLAB's rlocus command, plot the root locus for the system above using K=1005 and a=6.8.

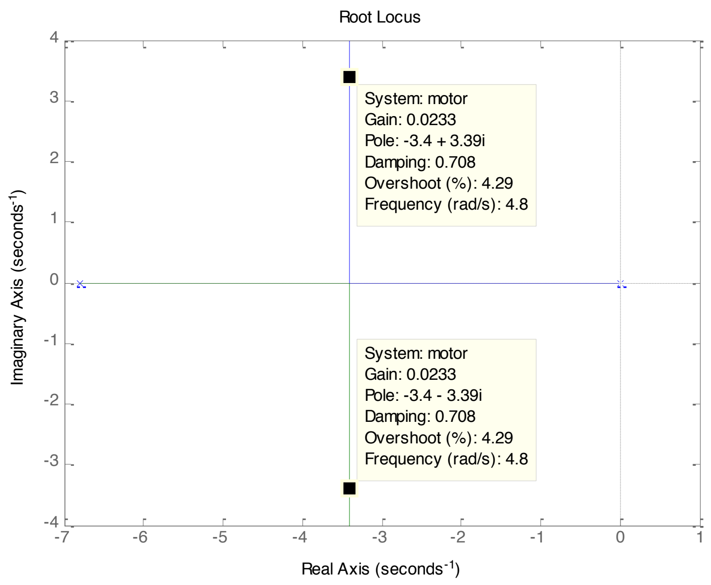
```
K = 1005
a = 6.8
G = tf(K,[1 a 0])
      Transfer function:
          K
      ---------
      s^2 + a s

rlocus(G)    % plots the root locus of the system
```

A representative result is shown below. This plot shows all possible closed-loop pole locations for our pure gain K_p. The plot, however, does not give directions of the loci (what are they?). In this case, the starting points for the roots are shown by the two small 'x' marks at $s = 0$ and $s = -6.8$. These starting points are given when K_p=0 (** note that K_p=0 breaks the closed-loop transfer function and gives the roots of G(s) instead). As K_p increases, the two roots move horizontally toward each other, meeting at $s = -3.4$, and then move vertically away (break-away point) from each other.



Root locus of previous system.

The K_p value corresponding to points on this graph isn't plotted directly, but is easily found. Right click on the green line in the rlocus plot above. You should get a black square on the plot that you can then drag to different locations. You can do the same thing on the blue line. For example, find the gain K_p at pole locations where the real and imaginary parts are equivalent ($s = -3.4 \pm 3.4j$ on this example plot). By clicking on this location, we find that gain K_p = 0.0233 (which = $a^2/2K$). This is the same K_p value that we calculated above!



You can also use the MATLAB command rlocfind to do the same thing. This function takes the system G and a vector of the desired pole locations as arguments and returns the K value.

```
rlocfind(G,[-3.4+3.4j,-3.4-3.4j])
```

```
ans =

    0.0234    0.0234
```

This tells us that we want a K_p value of 0.0234 for a pole at -3.4+3.4j and a gain of 0.0234 for a pole value of -3.4-3.4j. Since a second order function will always have complex conjugates, we could only use one of these poles to find K_p.

1.3 Design Example

We'll start with a generic root locus design example and then you can go through the same analysis for your motor exercise in the assignment below.

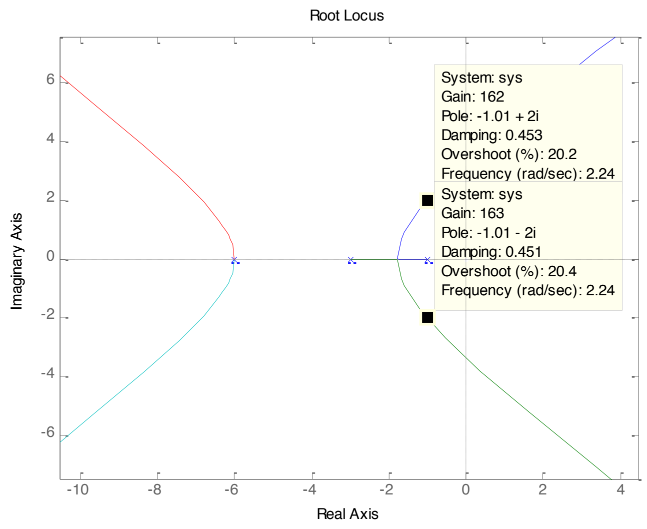
Given the following transfer function G(s), pick a value of K_p that will yield second-order response of the closed-loop transfer function T(s) with a settling time of T_s = 4 sec (assume negative feedback is used).

$$G(s) = \frac{1}{(s+1)(s+3)(s+6)^2}$$

First, recall that settling time for a second-order system is defined by Ts = 4/σ, where σ is the absolute value of the real part of the dominant second-order poles ($\sigma = \omega_n\zeta$ for an underdamped stable 2nd-order system). Thus, to achieve a settling time of 4 sec, σ = 1, i.e. the dominant poles must have real parts equal to -1. To find suitable closed-loop pole locations, we will use rlocus() to graphically select the poles and find the value of K_p which will put the poles at this point of the root locus:

```
sys = zpk([], [-1 -3 -6 -6], 1)
rlocus(sys)
```

When you graphically pick your poles with real parts of -1, the result should look like this:



You may need to zoom in on your plot to find the exact value of K_p that results in a real part of -1. This plot tells us that $K_p \approx 163$ will yield the desired settling time of 4s.

Note that you should also check that the 'approximate 2nd-order' assumption is valid. You can use rlocfind without a desired pole location to display all of the pole locations at a selected point.

```
 [kp,poles] = rlocfind(sys)
```

Using rlocfind in this way will ask you to select a point in your rlocus plot. Select a point so that the real value of the dominant poles is approximately -1.

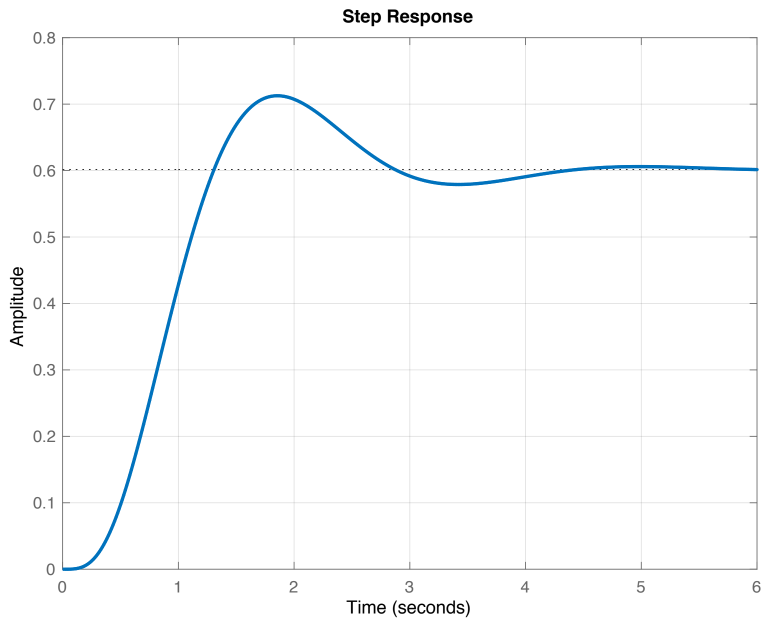
```
Select a point in the graphics window

selected_point =

  -1.0545 - 2.0031i


kp =

  159.3061


poles =

  -6.9659 + 2.1526i
  -6.9659 - 2.1526i
  -1.0341 + 1.9898i
  -1.0341 - 1.9898i
```

Are the non-dominant poles on the leftmost loci at least 5x further into the left half plane than the two dominant poles? If so, you can assume the system will behave approximately like a true 2nd-order system.

To validate your newly designed system, let's plot the time-domain response of the closed-loop system. First build the closed-loop system T(s). To do this, assume that the feedback is unity gain and G(s) is in the forward path along with our newly selected controller gain K_p.

```
G = zpk([], [-1 -3 -6 -6], 1)
Kp=163
T = feedback(Kp*G,1)
step(T)
```

This code should result in the following step response plot:



The settling time T_s is defined as the time to reach and stay within 2% of the final steady-state value. Estimate your settling time (you can also right click, select Characteristics → Settling Time and move the mouse over the dot that appears on the plot). Did you meet the performance specification with your controller?

It is also possible to graphically plot the poles and zeros of a closed-loop system without drawing the entire root locus plot by using the pzmap() command. For example, if you want to see the location of the poles and zeros of a closed-loop system T(s), simply enter to display the result.

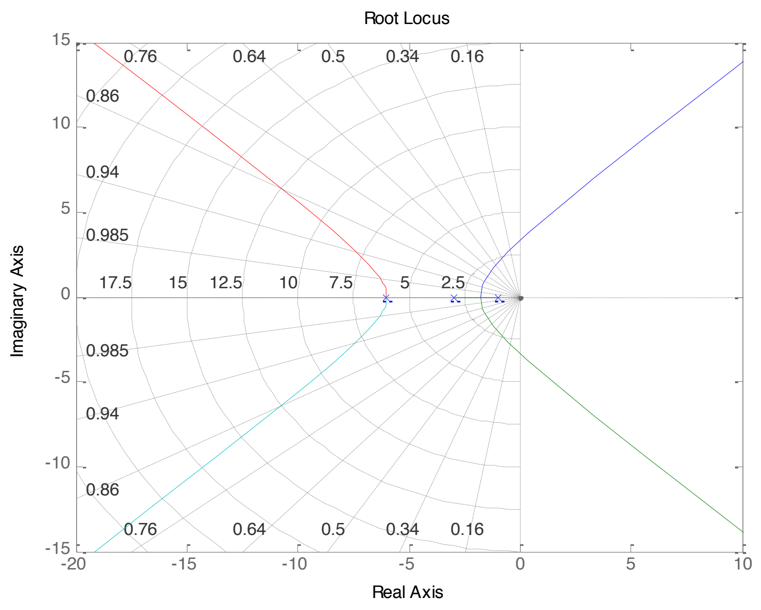
```
pzmap(T)
```

Of course in this case you need to use T(s) in the function pzmap(), unlike the root locus plot which only uses the open-loop transfer function G(s). Note the locations of the non-dominant poles, which are far enough into the left-half plane compared to the dominant poles to significantly affect the dynamic response.

1.4 Plotting Constant Damping Ratio Lines

If the above design problem had required finding closed-loop poles with a particular damping ratio (or %OS), it would have been a bit more challenging to get the correct answer since the root locus plot does not show lines of constant damping. This is easily corrected using the sgrid command. To see how this works, re-plot the root locus for the previous example, and then apply the sgrid command:

```
rlocus(G)
sgrid
```



Note that both constant damping (ζ) and constant natural frequency (ω_n, NOT damped frequency ω_d) lines are displayed in the left-half plane, i.e. both radial lines with constant slopes (constant ζ) and circular lines with constant radii (constant ω_n). The values of ζ for each radial line are displayed around the periphery of the root locus plot.

1.5 Design Example Using rltool

MATLAB also has a special interactive design tool using the root locus. Let's go through the previous design example (settling time of 4s) using the rltool.

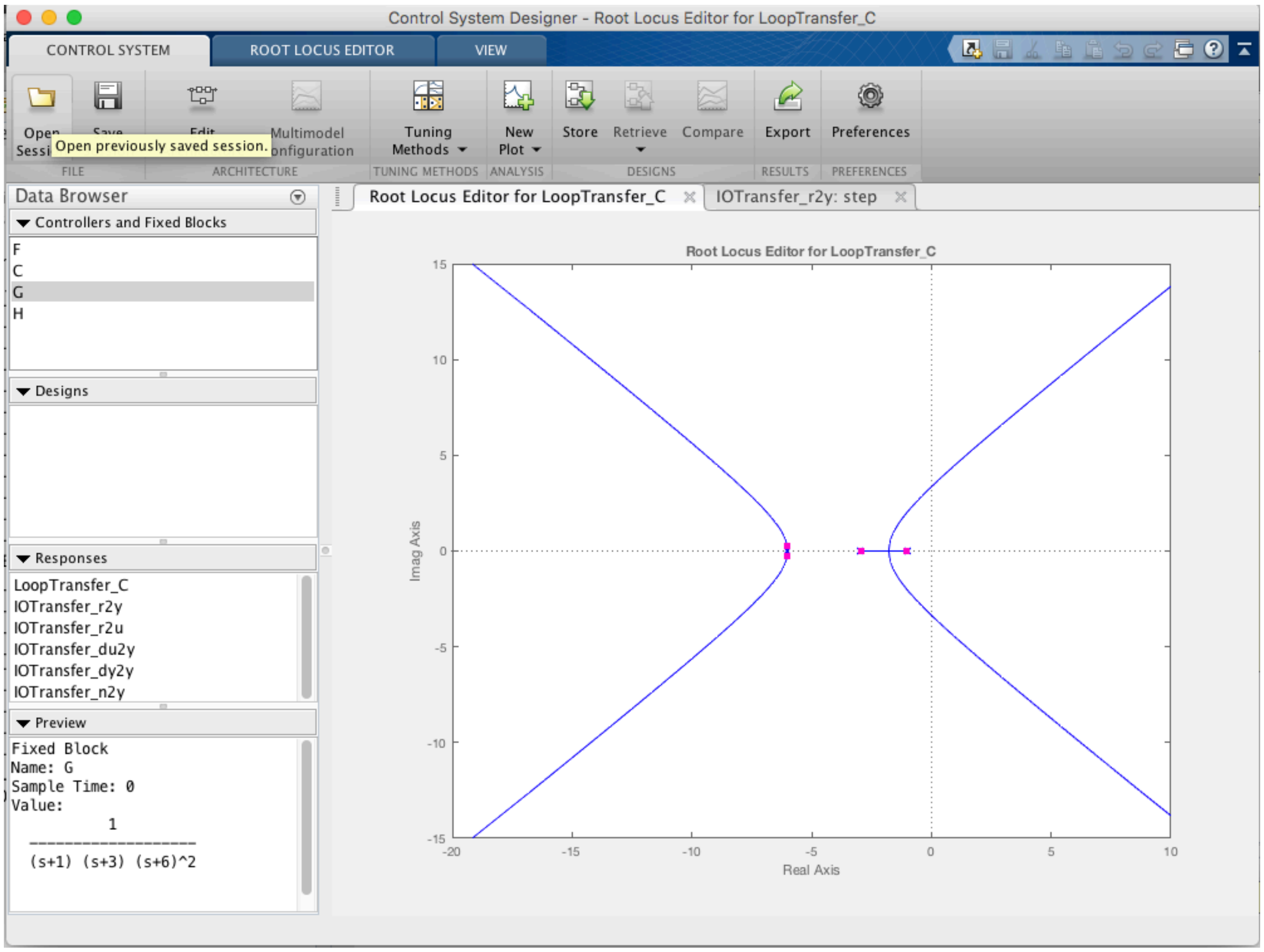
First, define the open-loop transfer function as before:

```
G = zpk([], [-1 -3 -6 -6], 1)
```

then, open rltool with G as an argument

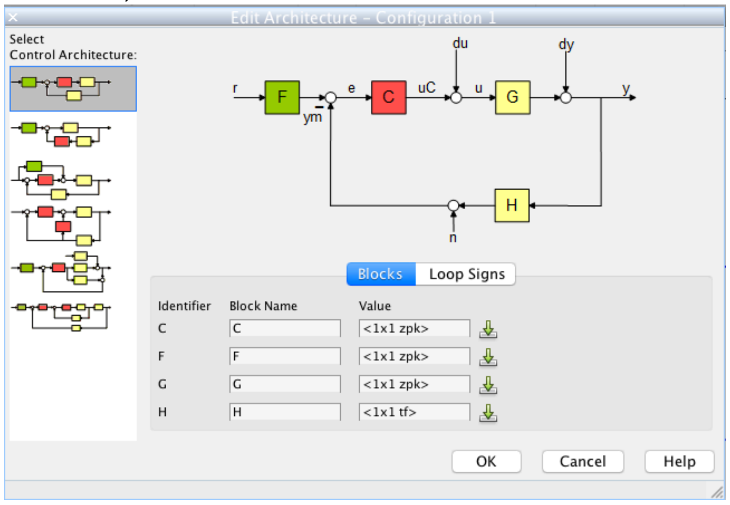
```
    rltool(G)
```

A graphical user interface with the root locus plot and a "Control System Designer – Root Locus Editor for LoopTransfer_C"" window should pop up as shown below:



```
Fixed Block
Name: G
Sample Time: 0
Value:
             1
  ---------------------
  (s+1) (s+3) (s+6)^2
```

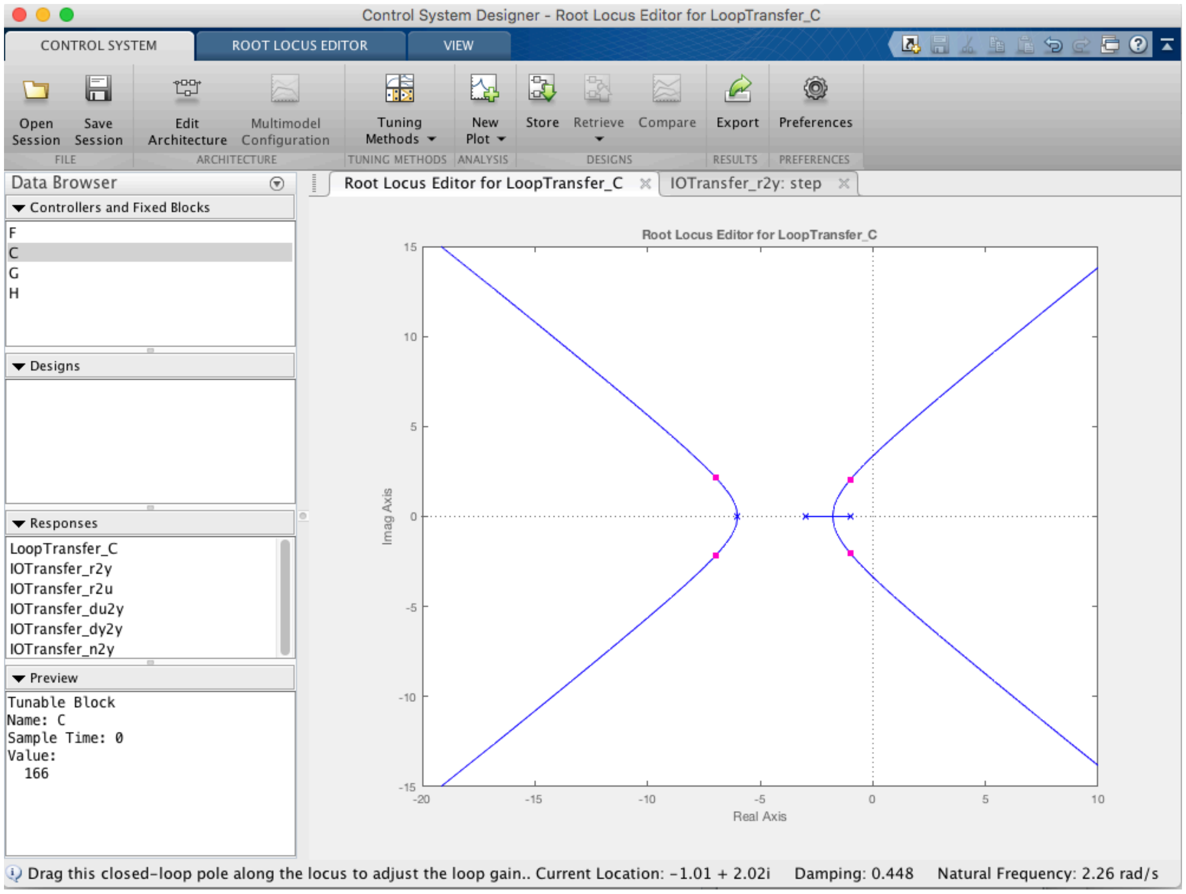
rltool defaults to a controller in the forward path as shown in the block diagram under Edit Architecture tab with F=1, C=K_p, G as defined above, and H=1:


Edit Architecture - Configuration 1
Select Control Architecture:
r
F
e
C
uC
du
u
G
dy
y
ym
H
n
Blocks
Loop Signs
OK
Cancel
Help

Identifier	Block Name	Value
C	C	<1x1 zpk>
F	F	<1x1 zpk>
G	G	<1x1 zpk>
H	H	<1x1 tf>

The pink squares on the root locus represent the closed-loop poles corresponding to the gain set point (note that the default K_p value is $\mathrm{Kp} = \mathrm{C} = 1$).

To find a K_p value that results in a real pole location of $\sigma = -1$ (for a settling time of 4s), drag one of the pink squares until you see a pole value of -1. The pole values are displayed on the bottom of the plot. The loop gain K_p is displayed in the Preview box in the lower left corner (make sure C is highlighted in the Controller and Fixed Blocks window).

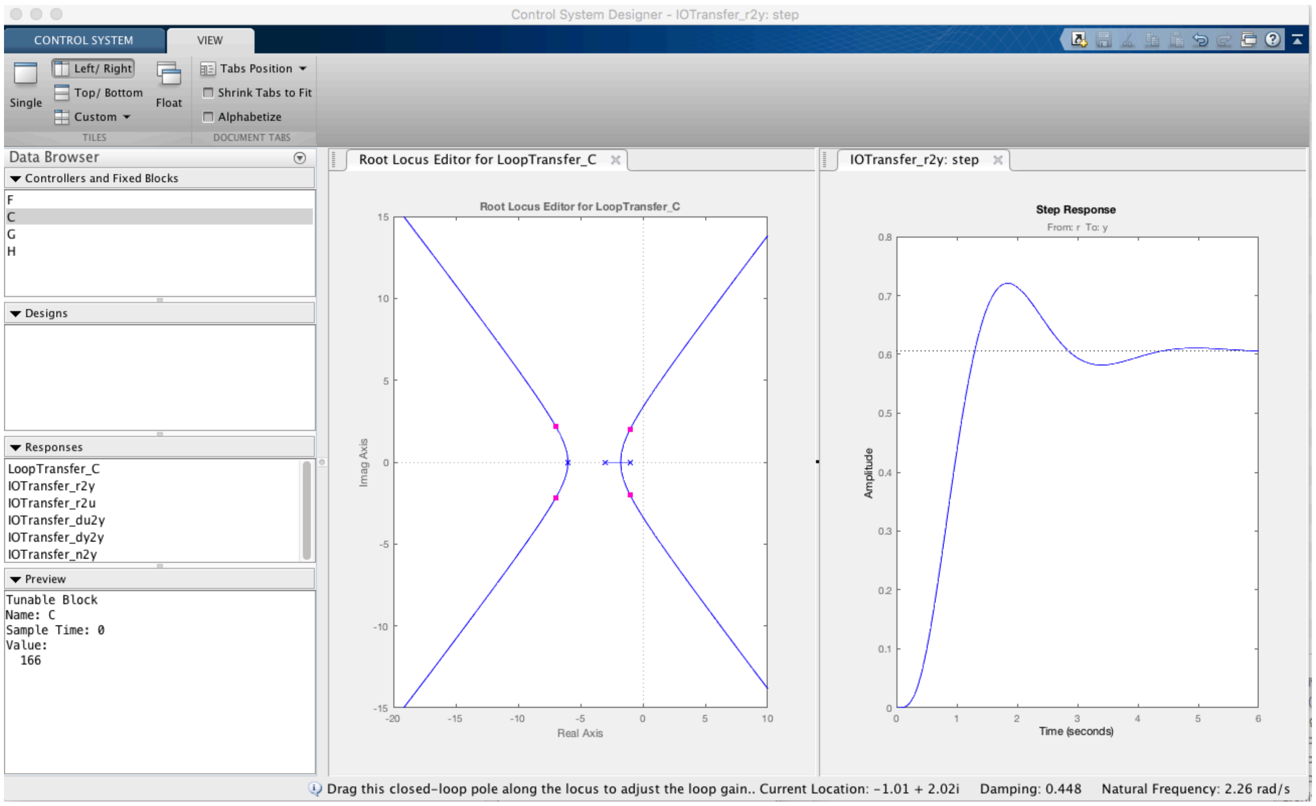


```
Tunable Block
Name: C
Sample Time: 0
Value:
  166
```

Drag this closed-loop pole along the locus to adjust the loop gain.. Current Location: -1.01 + 2.02i   Damping: 0.448   Natural Frequency: 2.26 rad/s

However, rltool provides an even easier way to find the K_p value that will result in a settling time of 4s.

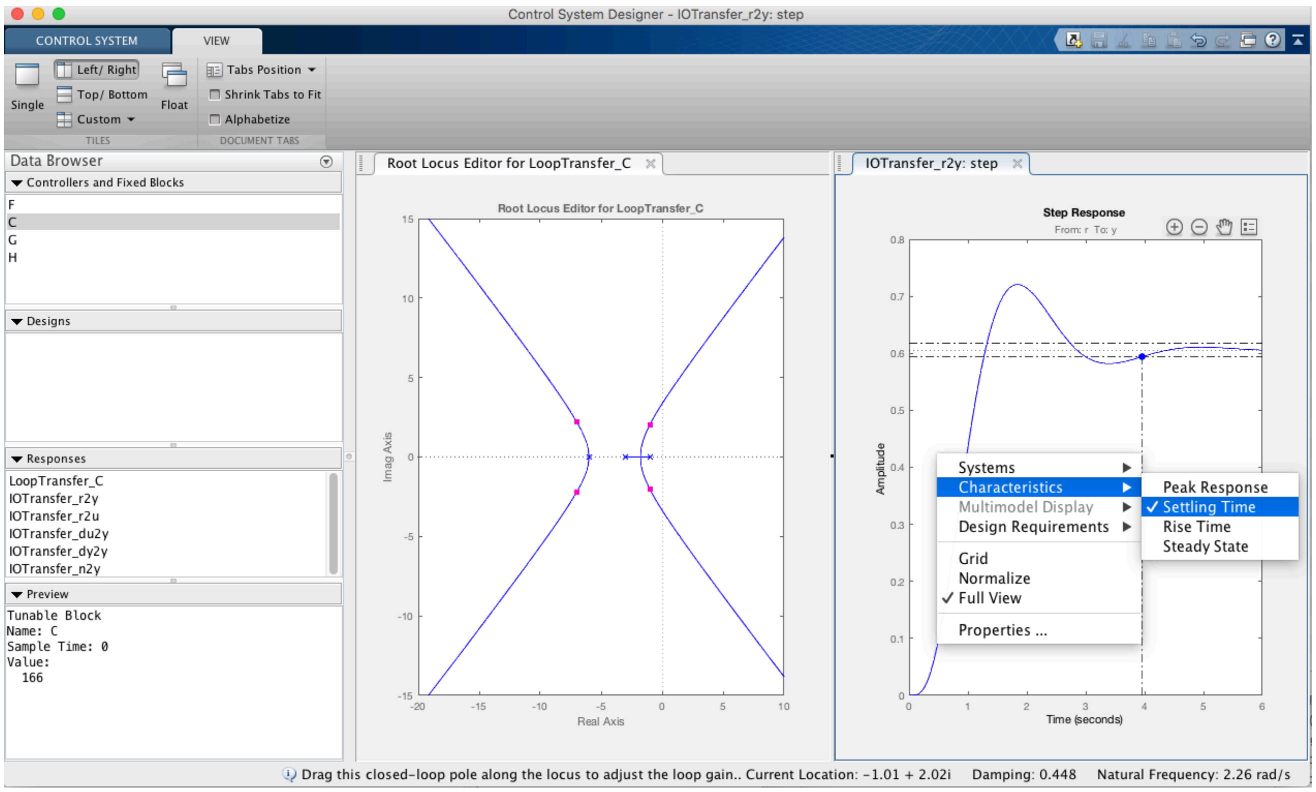
Click on the VIEW tab then click on Left/Right tab, this will result in a new subplot that shows the step response


Control System Designer - IOTransfer_r2y: step

```
Tunable Block
Name: C
Sample Time: 0
Value:
  166
```

Drag this closed-loop pole along the locus to adjust the loop gain.. Current Location: −1.01 + 2.02i Damping: 0.448 Natural Frequency: 2.26 rad/s

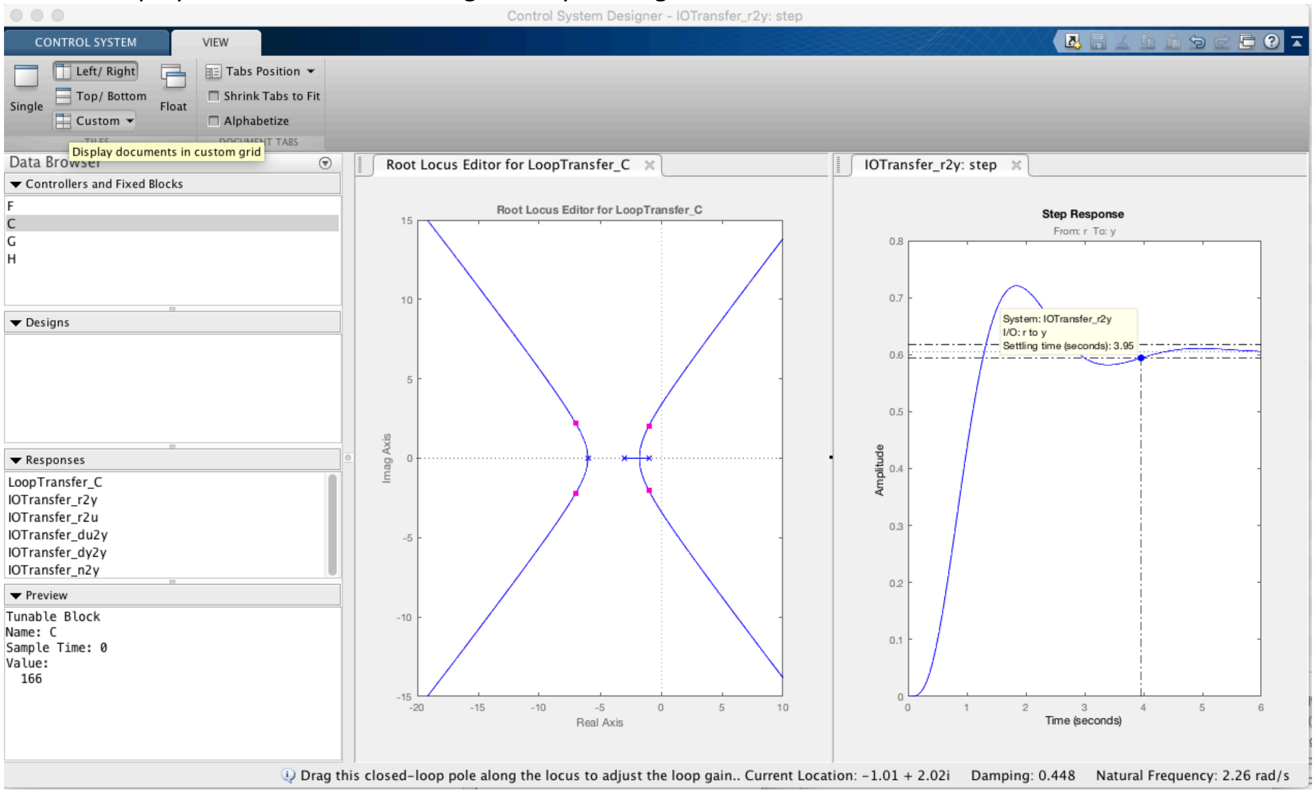
To find the settling time from the step response, right click on the step response plot, select Characteristics->Settling Time. The settling time will be shown with a blue dot and a dotted line to the time-axis.


Control System Designer - IOTransfer_r2y: step

```
Tunable Block
Name: C
Sample Time: 0
Value:
  166
```

Drag this closed-loop pole along the locus to adjust the loop gain.. Current Location: −1.01 + 2.02i Damping: 0.448 Natural Frequency: 2.26 rad/s

You can display the value of the settling time by clicking on the blue dot



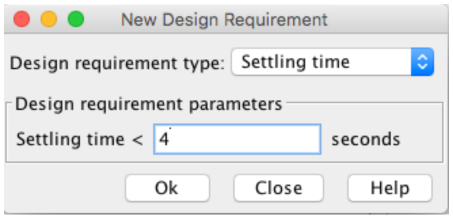
```
Tunable Block
Name: C
Sample Time: 0
Value:
  166
```

System: IOTransfer_r2y
I/O: r to y
Settling time (seconds): 3.95

Drag this closed-loop pole along the locus to adjust the loop gain.. Current Location: -1.01 + 2.02i   Damping: 0.448   Natural Frequency: 2.26 rad/s

Try dragging one of the pink squares (closed-loop poles) in the root locus window to a new location and watch how the step response (and settling time) changes based on the location of the closed-loop poles. Find an approximate value of K_p that results in the shortest settling time possible. Does this pole location make sense?

Another way to find values of K_p that will result in a settling time less than 4s is to use the design feature in rltool. Right click in the root locus plot and select “Design Requirements”->New. A window will pop up that allows you to select several transient response parameters. Select settling time < 4 seconds as the design guideline, click Ok.

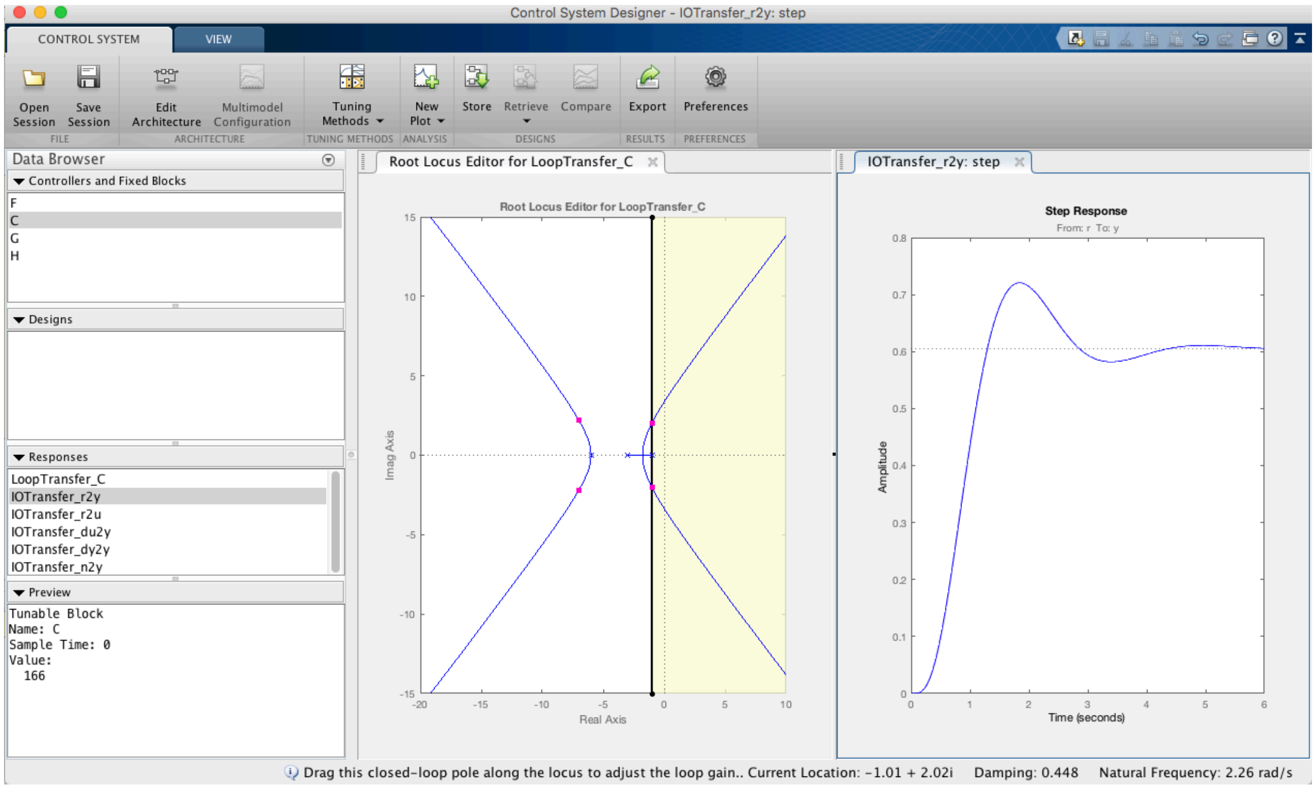


Design requirement type: Settling time

Design requirement parameters

Settling time < 4 seconds

The root locus plot will now include a yellow area that defines gains that will NOT meet your maximum 4s settling time. Does this yellow area make sense based on the value you found for a settling time of 4s above?



F
C
G
H

LoopTransfer_C
IOTransfer_r2y
IOTransfer_r2u
IOTransfer_du2y
IOTransfer_dy2y
IOTransfer_n2y

```
Tunable Block
Name: C
Sample Time: 0
Value:
  166
```

Drag this closed-loop pole along the locus to adjust the loop gain.. Current Location: -1.01 + 2.02i Damping: 0.448 Natural Frequency: 2.26 rad/s

Exercise

Each student shall complete the exercise below and get their work checked off by the studio instructor.

For remote students and students who do not finish within studio session:

Compile your answers and outputs of Exercises 1 along with the MATLAB code you used to answer the questions in a word file and name it LastNameFirstNameStudio09.docx and upload your file to Canvas under the Studio09 link. Your file shall include the solutions to parts (a) through (e).

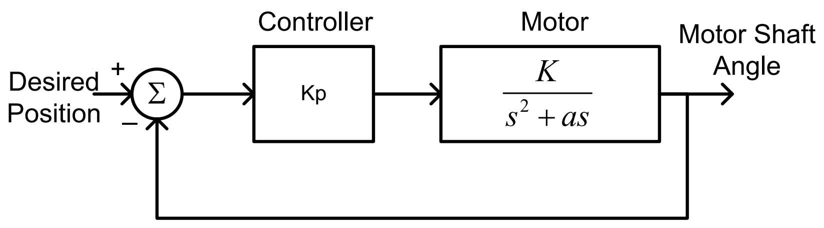
Exercise 1. Design a control system for a (motor) transfer function using what you learned above.

Let

$$G(s) = \frac{1005}{s(s+6.8)}$$

A proportional controller, i.e., a pure gain , is added to the forward path of a unity-gain negative feedback loop placed around the plant as shown in the block diagram below. The transfer function of this controller is:

$$G_c(s) = K_p$$



Block diagram for Motor and proportional controller, K=1005, a=6.8.

(a) Plot the root locus of this system using MATLAB's `rlocus` command.

(b) Find a value of K_p that will produce a closed-loop system with a damping ratio $\zeta = 0.7$ (this can be approximate). Provide your chosen K_p and the location of all closed-loop poles for this gain.

(c) Calculate T_s, T_p, T_r and %OS for the value of K_p and closed-loop poles found in part (b).

(d) Plot the step response for the closed-loop system (T(s) NOT Gc(s)*G(s)) for your chosen K_p to verify your results from (c).

(e) Is it possible to choose a gain that decreases the settling time in this system while maintaining a damping ratio < 0.7? Why or why not?