Control lab 2

I. A second order pendulum system

Figure 1

Likening the leg to a link, referring Figure.1 above. It has a length, l, and is free to rotate about O, as shown. The mass is m and the centre of mass is located at G which in turn is situated at $\frac{l}{2}$, in the middle of the link. The moment of inertia, I, about the pivot point, O, is given by the relationship:

$$I = ml^2$$

as the mass is centred at $\frac{l}{2}$ then we have $I = m\left(\frac{l}{2}\right)^2$. The knee-extensor muscles contract to hold the system in equilibrium at angle θ_e, by application of a torque, τ_e, acting around O. The moment (force x distance) required to maintain equilibrium, τ_e, is:

$$\tau_e = \left(mg\frac{l}{2}\right)\sin\theta_e$$

Assume now that the knee moves to a new angle of displacement, θ. There will be a damping coefficient, c, as shown in Figure.1. This damping occurs naturally and is caused by physical properties of the muscle and tendon-bone connection. When the body is moving there is now a viscous damping moment that's proportional to the object velocity, or $\dot{\theta}$. This damping moment is $c\dot{\theta}$. The displacement angle is shown in Figure. 2

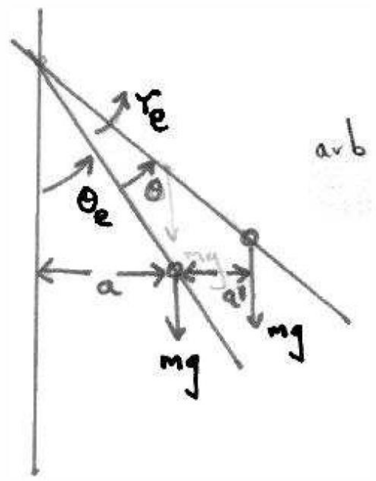


Figure 2

Let's now derive the differential equation that describes this system. From Newton's Law: $M = I\alpha$, where M is the sum of all external moments about the centre of mass, I is the moment of inertia of the mass and α is the angular acceleration of the mass. The knee displacement is θ, angular acceleration is $\ddot{\theta}$ and therefore the total inertial moment is $I\ddot{\theta}$. Balancing all moments:

$$I\ddot{\theta} = \tau_e - c\dot{\theta} - mg\frac{l}{2}\sin(\theta_e + \theta)$$

The final equation of motion is:

$$\ddot{\theta}(t) + \frac{c}{I}\dot{\theta}(t) + \frac{mgl}{2I}\theta(t) = \frac{1}{I}\tau(t)$$

The transfer function of this system in standard form is therefore:

$$P(s) = \frac{\Theta(s)}{T(s)} = \frac{\frac{1}{I}}{s^2 + 2\zeta\omega_n s + \omega_n{}^2}$$

1. Convert the equation of motion from the time, t, domain to s domain and rewrite the transfer function as shown above but in the s domain.
2. Using the standard form and the characteristic equation derive expressions for ω_n (undamped natural frequency) and ζ (damping ratio).
3. Let's assume the following physical parameters for the link shown above:
   $m = 5$ kg (mass);
   $l = 0.4$ m (length);
   $c = 0.5$ (damping coefficient);
   $g = 9.81$ m/s^2 (gravity)

Using these parameters and the transfer function derived in #1 above, let's use Matlab to experiment and investigate how these physical properties relate to the dynamic behaviour of the system and also how they relate to the poles of the system, which, as we know, set the dynamic behaviour.

1. Open the file “Control_lab2_student.m”
2. This file contains all the equations and the transfer function definition to allow you to start running the following experiments:
3. Vary the physical parameters one by one (m, l and c) and notice their impact on the step response of the system. Explain clearly why each parameter below is impacted (or maybe not) by the changes you make in each of the physical parameters.
    a. M_P, output overshoot.
    b. ts, output settling time.
    c. tp, time where the maximum output peak occurs.
    d. tr, rise time
    e. ω_n, the natural undamped frequency.
    f. ω_d, the damped frequency.

4. Comment on the steady state error of the output for each of the changes you make in the physical parameters. Is this output as expected? Explain why.
5. By using the “*pzplot*” function plot the poles and note how their positions vary as you vary the physical parameters. Correlate the position of the poles to the transient response of the output and give full explanations why the pole positions vary as they do and why the correlation to the output behaves accordingly.

You can also display the transfer function or any other variable just by directly typing the variable name in this window: for example note from the following screenshot how to display the transfer function:

II. Proportional gain – Open loop transfer function

1. For this lab exercise, we are now going to add proportional gain, Kp, to the plant pendulum transfer function and notice the impact of this on open loop behaviour.
2. Make the appropriate edit to the file “PendulumOpenLoop_Tut1a” and remember to save the new file to another name, such as “PendulumOpenLoop_Tut2a” or something appropriate.
3. Hint: add a new variable called Kp to the script:

The plant transfer function is:

*sys = tf (1/I,[1 2*xi*wn wn^2]*

Now create a new variable called sysKp, which will be the transfer function of the gain Kp

Kp=100; (this value will be part of your experiments)
sysKp = tf(Kp,1)

The new transfer function will be the product of the original transfer function sys and the newly added proportional gain sysKp

*sysP = sys*sysKp*

4. In block diagram form, you are now making the system look like Figure 3 below:

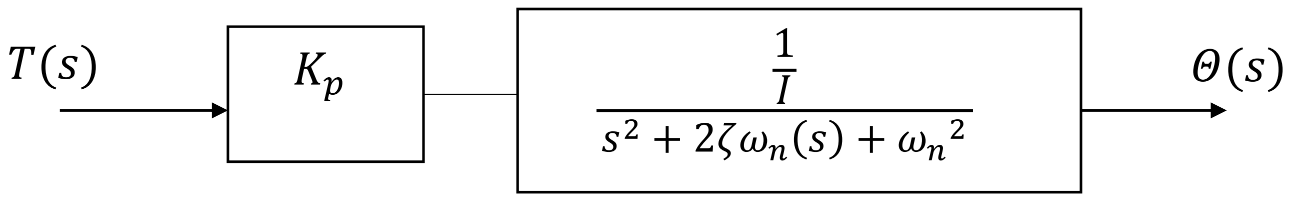


Figure 3

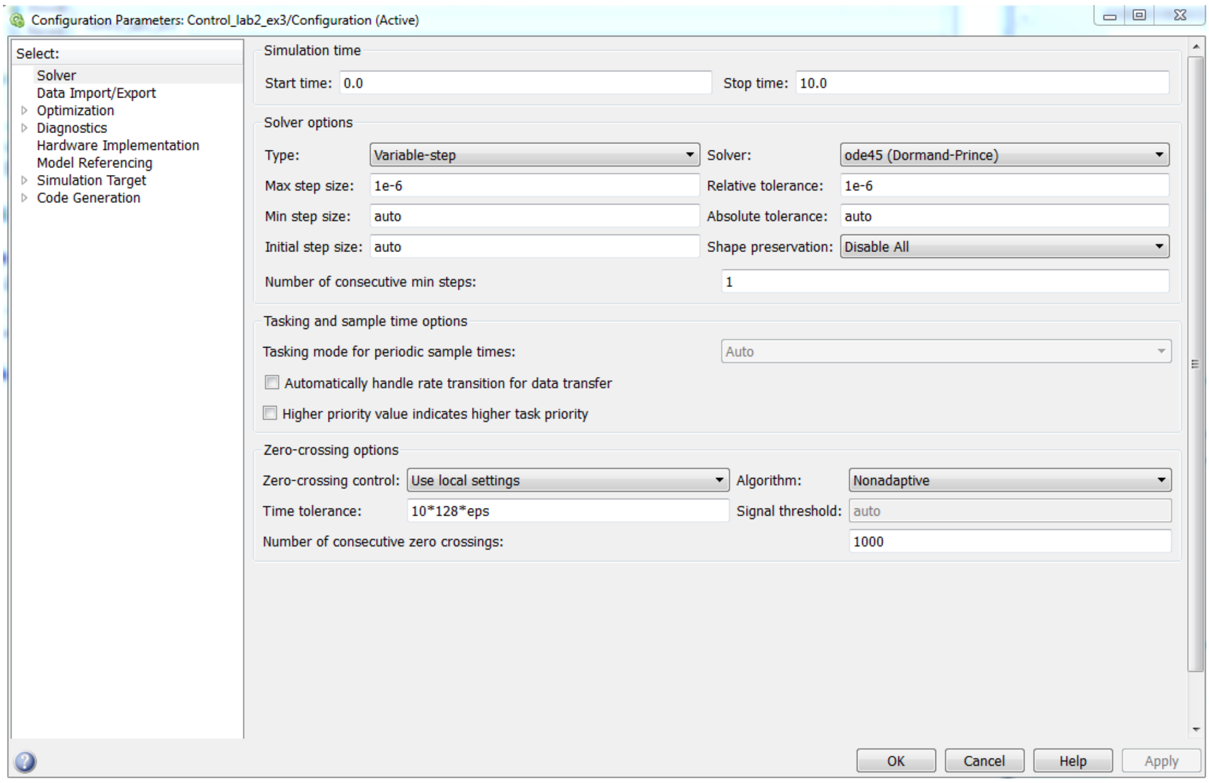
5. Vary the proportional gain, Kp, and notice its impact on the step response of the system. Explain clearly, why each parameter below is impacted (or maybe not) by the changes made in each of the physical parameters.
    a. M_P, output overshoot.
    b. ts, output settling time.
    c. tp, time where the maximum output peak occurs.
    d. tr, rise time.
    e. ω_n, the natural undamped frequency.
    f. ω_d, the damped frequency.
6. Comment on the steady state error of the output for each of the changes you make in the proportional gain, Kp. Is this output as expected? Explain why.
7. By using the "*pzplot*" function plot the poles and note how their positions vary as you vary the proportional; gain, Kp. Correlate the position of the poles to the transient response of the output and give full explanations why the pole positions vary or not vary.

III. Disturbance

1. For this lab exercise, we are now going to add a step disturbance to the output of our open loop plant system as developed in section II.
2. Launch "*simulink*"
3. Open the file "Control_lab2_ex3.slx", it will automatically use the variable values from Matlab workspace so you don't need to specify the values for *wn*, *xi* or *Kp*. If you cleared the memory run the previous Matlab script to repopulate the workspace.

   Go to the tab simulation> Model configuration parameters

   Enter the following parameters and apply to configure your model



Read the help documentation on simulink:
01 – Simulink_user_guide
02 – Simuling_guide_to_blocks

For an overview of the functions available in Simulink.
Make sure you explore the model so that you understand how it is setup.
And where to find the blocks in the Simulink libraries
Hint: only 4 libraries used for this model: Sources, Sinks, Math operations and Continuous

4. For this exercise, we are now going to use Simulink in Matlab. The schematic you are going to create will look something like that shown in Figure 4 below:

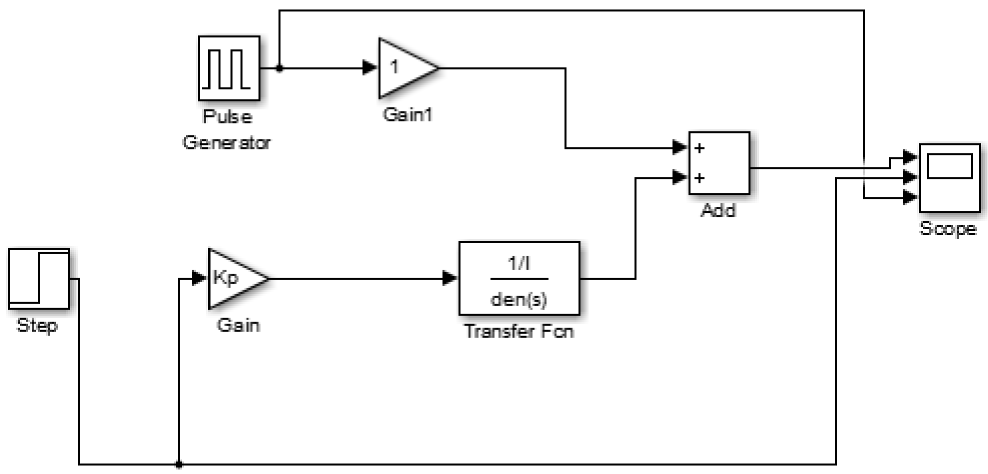


Figure 4

5. The above setup will impress a disturbance on the output of the plant which has an amplitude of 0.1V with a period of 1s and 50% duty cycle.
6. Make a copy of your previous script and call it “Control_lab2_ex3b.m”. You should then run this script before you run the simulation in *Simulink*. This is because the script will define all variables shown in the Simulink schematic such as Kp, wn and so on. You should change the values in the script for your experiments, run the script and then run the simulink simulation each time you make the change to the script to see the changes.
Hint: don’t name the script and the m-file the same, Matlab will not like it
7. By observing the output response, as before to a unit step input, what do you observe about the output of the system during the times that the disturbance is applied and then removed?
8. Modify the gain, *Kp*, and note if changing the gain has any impact on the output behaviour during the disturbance.
9. Modify parameter, *c* and observe the behaviour of the system output. Explain why the output behaves the way it does.
10. Explain what you are observing and if this behaviour is as you would expect and why.

IV. Closed-loop transfer function

1. For this lab exercise, we are now going to close the loop and add a step disturbance to the output of our closed loop plant system as developed in Control Lab 3.
2. Make the appropriate edit to the file “PendulumOpenLoop_Tut3a” and remember to save the new file to another name, such as “PendulumClosedLoop_Tut4a” or something appropriate.
3. Hint: The edit you need to make is to create another transfer function in the script which can be done in different ways as illustrated below:
4. The original open loop transfer for the plant is given as:

*sys = tf (1/I,[1 2*xi*wn wn^2]*
CLTF = sys/(1+sys);

5. Edit the script to plot both the transient step response and the poles of the loop in order to be able to analyse the following questions:
6. By observing the output response, as before to a unit step input, what do you observe about the following compared to that measured in Control_lab2_ex1?
    a. M_P, output overshoot.
    b. ts, output settling time.
    c. tp, time where the maximum output peak occurs.
    d. tr, rise time.
    e. ω_n, the natural undamped frequency.
    f. ω_d, the damped frequency.
7. Comment on the steady state error.
8. Type “*CLTF*” in the command window and note the transfer function.
9. Type “*sys*” in the command window and note the transfer function.
10. What do you observe about the difference between the denominator of “*CLTF*” and “*sys*”?
11. By looking at the position of the poles, have the poles varied in position compared to the open loop case? Why do you so answer?
12. Vary the parameters “c” (the damping coefficient) and “l” (the length) and note the impact on the transient response and pole positions. Try and have the system go into oscillation, explain how you did it and how this impacted the position of the system poles.

V. Closed-loop transfer function with proportional gain

1. For this lab exercise, we are now going to close the loop and we will also add proportional gain, Kp to the closed-loop system.
2. Make the appropriate edit to the file "Control_lab2_ex5".

   *sys = tf (1/I,[1 2*xi*wn wn^2]*

   So you can add loop gain, Kp, in the following way:

   Kp=100;
   sysKp = tf(Kp,1)
   *OLTF = sysKp*sys*
   CLTF = OLTF/(1+OLTF)
3. Edit the script to plot both the transient step response from the OLTF and CLTF and the poles of the open and closed loops loop in order to be able to analyse the following questions:
4. Vary Kp and observe the following:
   a. Mp, output overshoot.
   b. ts, output settling time.
   c. tp, time where the maximum output peak occurs.
   d. tr, rise time from 10%-90% of final output value.
   e. ω_n, the natural undamped frequency.
   f. ω_d, the damped frequency.
5. Comment on the steady state error. Why is it different if we vary Kp?
6. Type *"sys"* in the command window and note the transfer function.
7. Type *"CLTF"* in the command window and note the transfer function.
8. What do you observe about the difference between the denominator of *"sys"* and *"CLTF"*?
9. What differences in the denominator of the transfer function occur as you increase Kp? For this example, create another polynomial, OLTF, in your script, which is merely the pendulum transfer function multiplied by Kp.
10. By looking at the position of the poles, have the poles varied in position compared to the open loop case and the closed loop case with unity gain?
11. How do the pole positions change as Kp is varied?
12. Will this be the reason for the change in dynamic performance as Kp is varied?

VI. Closed-loop transfer function with disturbance

1. For this lab exercise, we are now going to add a step disturbance to the output on previous exercise we carried out in section V.
2. Make the appropriate edit to the file “Control_lab2_ex6b.m”.
3. Modify “Control_lab2_ex3.slx” so that the closed-loop system resembles what is shown below and save it as “Control_lab2_ex6.slx” as shown below:

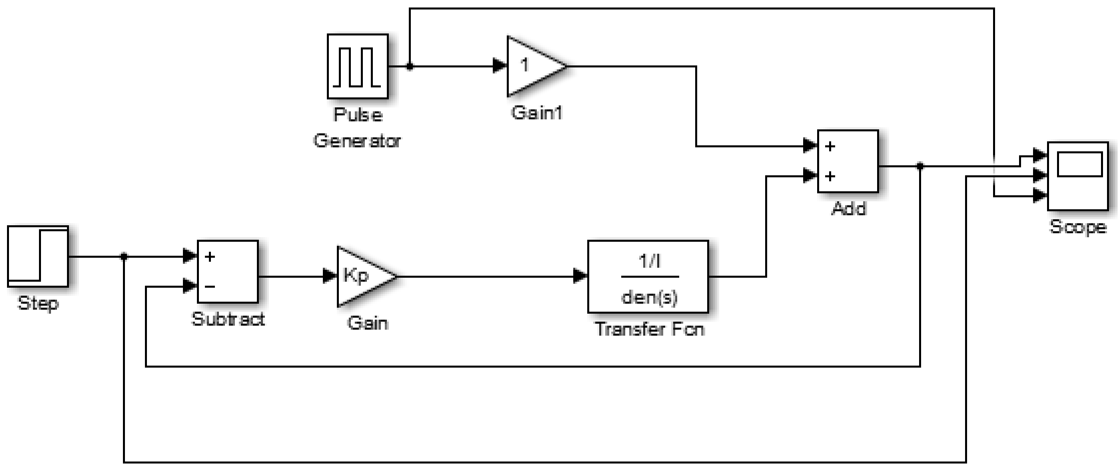


Figure 5

4. Run your Control_lab2_ex6b.m file as this will set up all of your variables. When changing any variables change using this script and rerun the script before rerunning the Simulink simulation.
5. By observing the output response, as before to a unit step input, what do you observe about the output of the system during the times that the disturbance is applied and then removed?
6. Modify the gain, Kp, and note if changing the gain has any impact on the output behaviour during the disturbance. Please explain why changing the gain, Kp, has influence on the response to disturbance, if any.
7. Tie in your response for question 7 by plotting the poles and tying in the pole positions to what you see on the transient response due to a step input and the output disturbance, for various physical parameters such as *Kp, c,* etc...
8. Modify parameter, *c* and observe the behaviour of the system output. Explain why the output behaves the way it does.
9. Explain what you are observing and if this behaviour is as you would expect and why.

VII. PI controller in closed-loop system

1. For this lab exercise, we are now going to add integral gain, Ki, to our proportional gain, Kp, in a closed loop system.
2. Make the appropriate edit to the file “Control_lab2_ex7b.m”.
3. The original open loop transfer for the plant is given as:

*sys = tf (1/I,[1 2*xi*wn wn^2]*

So you can assign two variables now, Kp and Ki. You should keep the script written in such a way that you can independently vary Kp and Ki to note the impact on the system response and the pole/zero positions.

For example:

Kp=1;
sysKp = tf(Kp,1)
Ki=1;
sysKi = tf(Ki,[1 0])
*OLTF = (sysKp+sysKi)*sys*
CLTF=sys/(1+sys)

CLTF is the closed loop transfer function, Kp and Ki can be varied separately.

4. Edit the script to plot both the transient step response and the poles of the loop in order to be able to analyse the following questions:
5. Note the response on the output as you vary both Kp and Ki for the following parameters:

   Rise time   Overshoot   Settling Time   Steady State error

   Please show which gain has the largest impact on which parameter and give clear reasons why this should be the case.

6. As you vary the gains in the system please take note of how this impacts the pole and zero positions and give explanations as to the behaviour of the time-domain response.
7. You should observe the transfer function, “CLTF”, as you vary the gains. You can print “CLTF” in the console window to verify this.

VIII.PD controller in closed-loop system

1. For this lab exercise, we are now going to add derivative gain, Kd, to our proportional gain, Kp, in a closed loop system.
2. Make the appropriate edit to the file “Control_lab2_ex8”
3. The original open loop transfer for the plant is given as:

*sys = tf (1/I,[1 2*xi*wn wn^2]*

So you can assign two variables now, Kp and Kd. You should keep the script written in such a way that you can independently vary Kp and Kd to note the impact on the system response and the pole/zero positions.

```
Kp=10;
sysKp = tf(Kp,1)
Kd=1;
sysKd = tf([Kd 0],1)
OLTF = (sysKp+sysKd)*sys
CLTF=sys/(1+sys)
```

CLTF is the closed loop transfer function, Kp and Kd can be varied separately.

4. Edit the script to plot both the transient step response and the poles of the loop in order to be able to analyse the following questions:
5. Note the response on the output as you vary both Kp and Kd for the following parameters:

Rise time Overshoot Settling Time Steady State error

Please show which gain has the largest impact on which parameter and give clear reasons why this should be the case.

6. As you vary the gains in the system please take note of how this impacts the pole and zero positions and give explanations as to the behaviour.
7. You should observe the transfer function, “CLTF”, as you vary the gains. You can print CLTF in the console window to verify this.

IX. PID controller in closed-loop system

1. For this lab exercise, we are now going to simulate a PID system, with Kp, Ki and Kd gains operating together.
2. Make the appropriate edit to the file "Control_lab2_ex9"
3. The original open loop transfer for the plant is given as:

*sys = tf (1/I,[1 2*xi*wn wn^2]*

So you can assign 3 variables now, Kp, Kd and Ki. You should keep the script written in such a way that you can independently vary Kp, Kd and Ki to note the impact on the system response and the pole/zero positions.

```
Kp=10;
sysKp = tf(Kp,1)
Kd=1;
sysKd = tf([Kd 0],1)
Ki=1;
sysKi = tf(Ki,[1 0])
OLTF = (sysKp+sysKd+sysKi)*sys
CLTF=sys/(1+sys)
```

CLTF is the closed loop transfer function, Kp, Kd and Ki can be varied separately.

4. Edit the script to plot both the transient step response and the poles of the loop in order to be able to analyse the following questions:
5. Note the response on the output as you vary both Kp and Ki for the following parameters:

Rise time   Overshoot   Settling Time   Steady State error

Please show which gain has the largest impact on which parameter and give clear reasons why this should be the case.

6. As you vary the gains in the system please take note of how this impacts the pole and zero positions and give explanations as to the behaviour.
7. You should observe the transfer function, "CLTF", as you vary the gains. You can print CLTF in the Console window to verify this.

X. PID controller – Simulink

1. For this lab exercise, we are going to use Simulink and the available blocks that can be dragged from the Palette Browser under the “Continuous time systems” tab. You are going to create a schematic as shown in Figure 6.

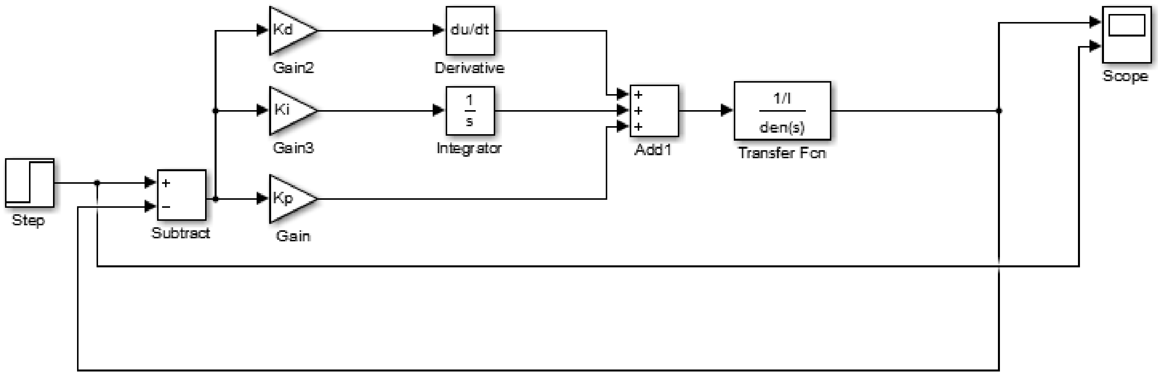


Figure 6

2. The PID block will allow you to enter the Kp, Ki and Kd gains directly and allow experimentation to quickly ascertain how each influences the output.
3. Let’s have a quick reminder of how each gain influences the output:

The proportional gain, Kp, is always assigned a non-zero value and plays a part in setting the rise time and steady state error, although it will never on its own remove the steady state error.

The derivative gain, Kd, reacts to changes in the controller input and therefore helps to reach steady state quicker. However, setting this gain too high could lead to stability issues.

The integral gain, Ki, eliminates the steady state error, but as it creates a pole at DC then if set too high could lead to oscillation and ultimately instability.

All 3 can be varied independently from each other to provide an optimal output and this is why PID controllers are so popular today.

4. As a first experiment, assume the controller transfer function above. By setting Kd and Ki to zero and influencing Kp, note what happens to the output response. Why does the output behave like this?

Hint: by calculating the Characteristic equation of the above loop you can plot the poles of the equation and understand the system behaviour.

"pole(Characteristic Equation)" in OLTF and CLTF

5. Alternatively, you can create a script that will plot the poles to allow you to visualise easily where the poles move to depending on the system gain configuration.

So from this what do you conclude is the reason for the output behaving as it does with Kp=1, Ki and Kd=0?

6. Now in addition to Kp=1, add derivative gain, Kd=1 and note the change in behaviour. Explain this by again looking at the poles and zeros in the system using the console or edit your script that you generated above. Note also in this case the steady-state output error and explain this based on the closed loop transfer function calculated in the Console.
7. Finally add integral gain and experiment with the system output.
8. Outline your conclusions on how each of the gains impacts the system as outlined in points 4-7.
9. Now make a change to the transfer function that will ensure all poles are on the LHP and repeat the experiments from 4 to 7.
10. Let us now setup a somewhat more complex system and check disturbance and tracking attenuation and performance. Create a schematic in Simulink that represents the system as shown in Figure 7. This could be a joint that's subject to an external force N(s). J is the joint inertia and B is the frictional resistance.

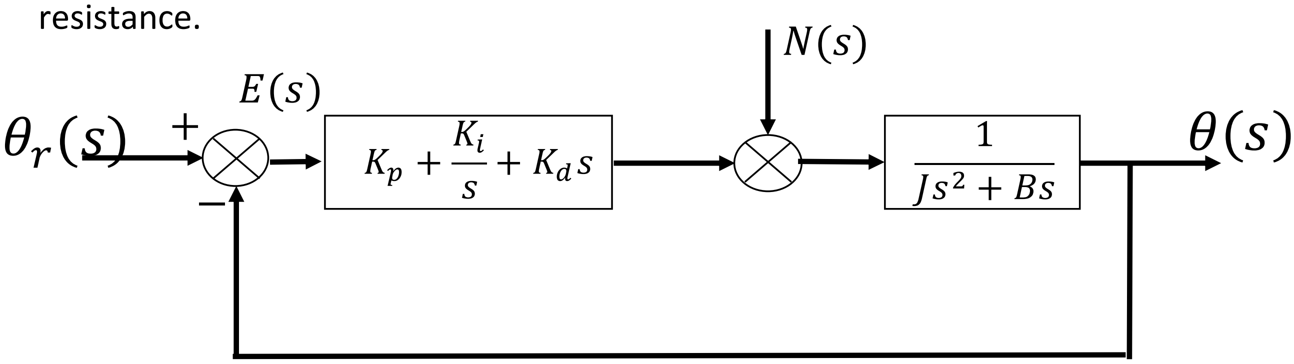


Figure 7

Let's assume J=10 and B=0.1.

- Set the N(s) disturbance to 20V and apply at 40s.
- Apply the reference input, θ_r, at 20s.
- Set the Kp gain to 1 and all others to zero.

- Observe the output and explain what is happening and why. You can plot the poles using the Console by entering the characteristic closed loop equation or use a script to plot the poles and zeros.
- What happens to the steady state error as you increase the gain Kp to say 100? Does it change the transient behaviour?
- What change in the plant transfer function would need to take place to change this behaviour?

11. Let us now start to calculate the optimal gains for this system. Let us set the integral gain, Ki to 0. This can be added at the end to remove any steady state error. Let us select optimal Kp and Kd gains. Let's assume we want an overall critically damped system with ζ = 1 and let us select a natural frequency, ω_n of 10 rad/sec.

- Firstly, calculate the Closed Loop Characteristic Equation. This isn't too difficult as we know the plant – in some cases the plant is unknown and the controller would have to be designed using the Zeigler Nichols method.
- Then equating the coefficients calculate the required Kp and Kd to provide the required specification as above.
- Using the values you have calculated simulate your system in Simulink and then try some experiments by selecting new values of ω_n (and hence Kd and Kp)and note how this changes the response of the output.
- Comment on the steady state error when using only a PD controller (Kp and Kd gain only). Can it be completely removed only using PD control?
- Now introduce integral control, Ki. Calculate the new closed loop Characteristic equation. Note that the integrator raises the order of the system. What is the best method for calculating the maximum allowable gain, Ki, before instability occurs?
- Use this method and simulate with the calculated gain values in Simulink to verify your calculation.
- Observe any ringing that occurs during the disturbance – does the frequency match your target ω_n?
- Does the output respond faster or slower as you increase ω_n?