Mectron/Sfwr Eng 4AA4 - Lab 5

PID Controller

Goals: Learn how to simulate a PID controller and a DC Motor using MATLAB and Simulink for determination of suitable values of K_p, K_i and K_d of the PID controller.

Note: Before you begin your lab sessions, please read the following documents at your own convenient time, in addition to the class notes:

- http://en.wikipedia.org/wiki/PID_controller
- http://igor.chudov.com/manuals/Servo-Tuning/PID-without-a-PhD.pdf
- https://www.mathworks.com/products/simulink.html
- https://www.mathworks.com/help/

Note:

"If you are using your own personal computer, before staring the lab, **MATLAB 2020a** or 2020b and **SIMULINK** library are needed to be installed. **Lower versions of MATLAB will not work for this lab**. McMaster is providing free academic licence of MATLAB for students:

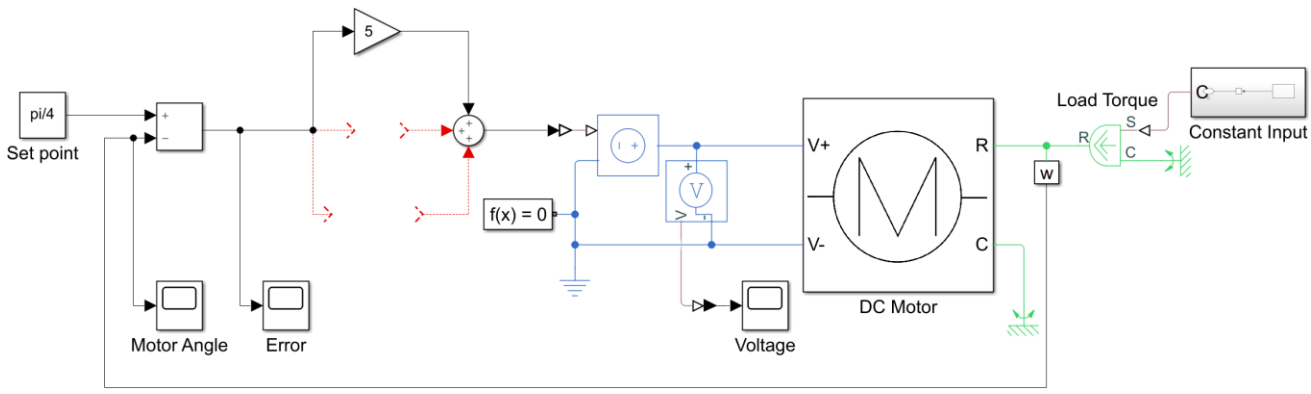
https://uts.mcmaster.ca/services/computers-printers-and-software/software-licensing/matlab/

They are already installed in the desktops of ITB235 as well."

Simulink Introduction

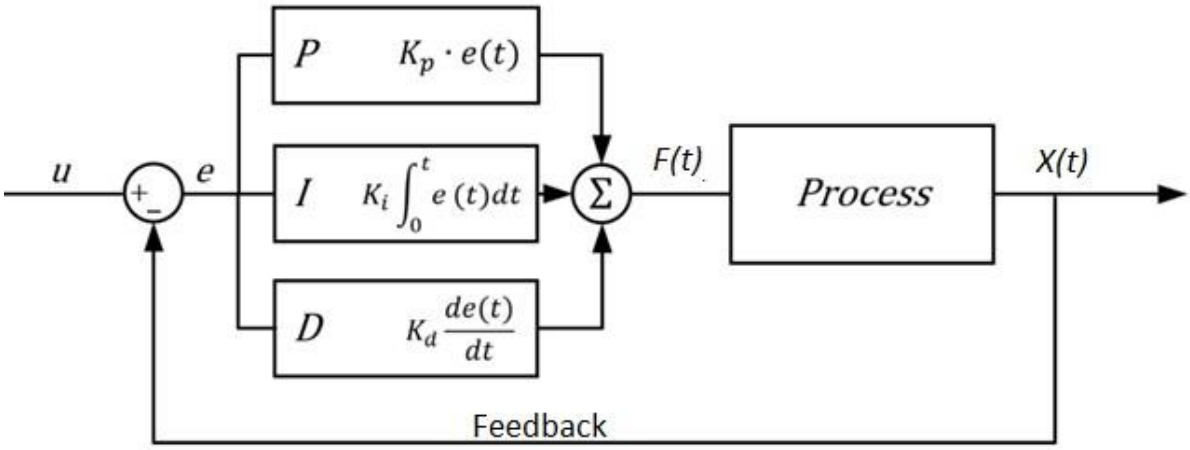
Simulink is a graphical editor of MATLAB software that can be used for constructing models of hybrid dynamical systems. Models can be assembled, loaded, saved, compiled, and simulated. It provides a Library browser that lists all standard blocks grouped by categories. You need to search and select suitable blocks and drag them into the workspace for constructing a model. The model can then be compiled and simulated. The data resulting from the simulation can be graphically viewed in real time. The software provides facilities covering a wide range of applications.

By using suitable blocks, it is possible to simulate the rotation control of a DC Motor and use it in conjunction with a PID controller to observe the output as the parameters of the PID controller are changed.



PID Controller Introduction

The following block diagram is a PID controller system.



u: Set point, or target position degree
x(t): potentiometer reading, feedback degree
f(t): the volt value output to the DC motor

In time domain, a PID controller can be represented by the following differential equation:

$$F(t) = K_p e(t) + K_i \int_0^t e(\eta)d\eta + K_d \frac{de(t)}{dt}$$

Using Euler's approximation method:

$$\frac{dx}{dt} \approx \frac{x(k) - x(k-1)}{T}$$

For proportional part:

$$F_P(k) = K_p e(k)$$

For integral part:

$$F_I(t) = K_i \int_0^t e(\eta)d\eta$$

Take derivative on both sides:

$$\frac{dF_I(t)}{dt} = K_i e(t)$$

Using Euler's approximation:

$$\frac{F_I(k) - F_I(k-1)}{T} = K_i e(k)$$

Or

$$F_I(k) = F_I(k-1) + K_i T e(k)$$

This can be further simplified by considering that:

$$F_I(k-1) = F_I(k-2) + K_i T e(k-1)$$

Substituting it into earlier equation:

$$F_I(k) = F_I(k-2) + K_i T[e(k-1) + e(k)]$$

As the initial value $F_I(0) = 0$, the contribution from the integral component can be written as:

$$F_I(k) = K_i T \sum_{i=1}^{i=k} e(i)$$

The derivative part approximates to:

$$F_D(k) = \frac{K_d}{T}[e(k) - e(k-1)]$$

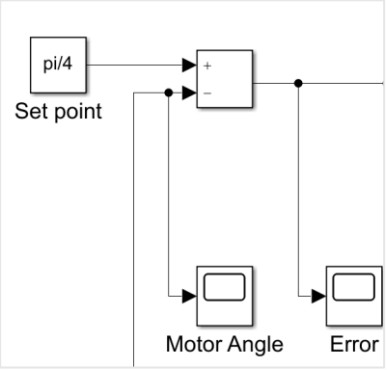
So, the PID controller approximates to:

$$\begin{aligned} F(k) &= F_P(k) + F_I(k) + F_D(k) \\ &= K_p e(k) + K_i T \sum_{i=1}^{i=k} e(i) + \frac{K_d}{T}[e(k) - e(k-1)] \end{aligned}$$

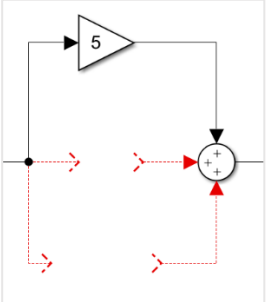
PID Controller Implementation in Simulink

In this lab you are required to design a PID controller in the Simulink. The task is that the system read the "set point" from users for desired motor shaft position. As the motor rotating, a "feedback" signal will be read from rotational motion sensor. Then, we employ a "Minus" block to calculate the "process error" between "set point" and the "feedback" signal. Next, the "process error" will be feed into the PID controller to generate the new voltage signal to drive the motor rotating.

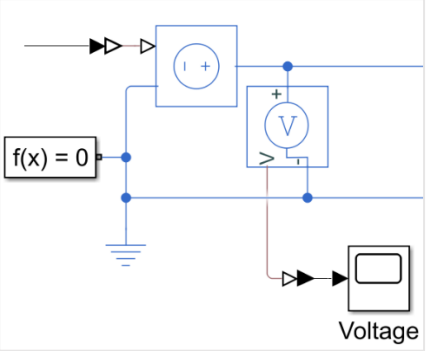
- Open "Simulink" after launching MATLAB. Open Simulink Models file "DCMotor4AA4.slx". Click "Run" on the menu bar and make sure there is no errors.
- There are 4 main parts of the Simulink model:
    - "set point" and "Minus" block: The "set point" is implemented by a Constant block, that is to provide a constant value. Here, we put $\frac{\pi}{4}$ in it, which is the set point for angle of the DC motor.



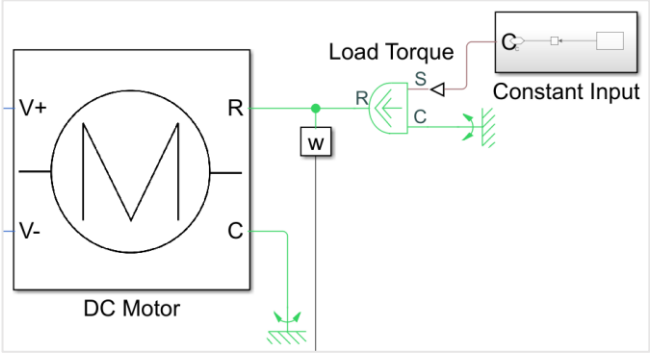
- PID controller: A Sum block which adds the three values together. Here, we just put a Gain block for now. The function of Gain block is to provide proportional value. As for "I" and "D", it leaves to you to finish.



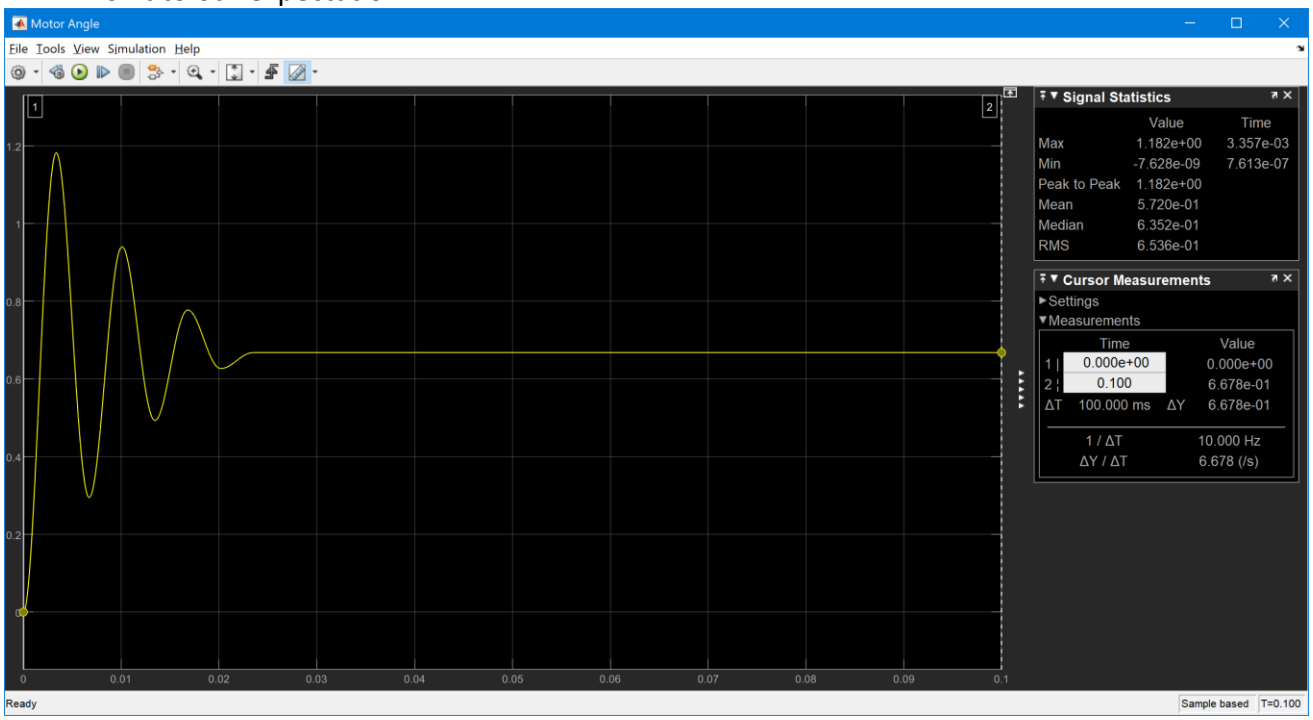
- Power module: A Controlled Voltage Source to generate the desirable voltage for DC motor. We also put a Voltage Sensor block and a Scope block to show the output voltage change.



- DC Motor and load: The Constant Input block provides a load to the motor and Block w is to measure the angle of the shaft (You can double click "w" for details).

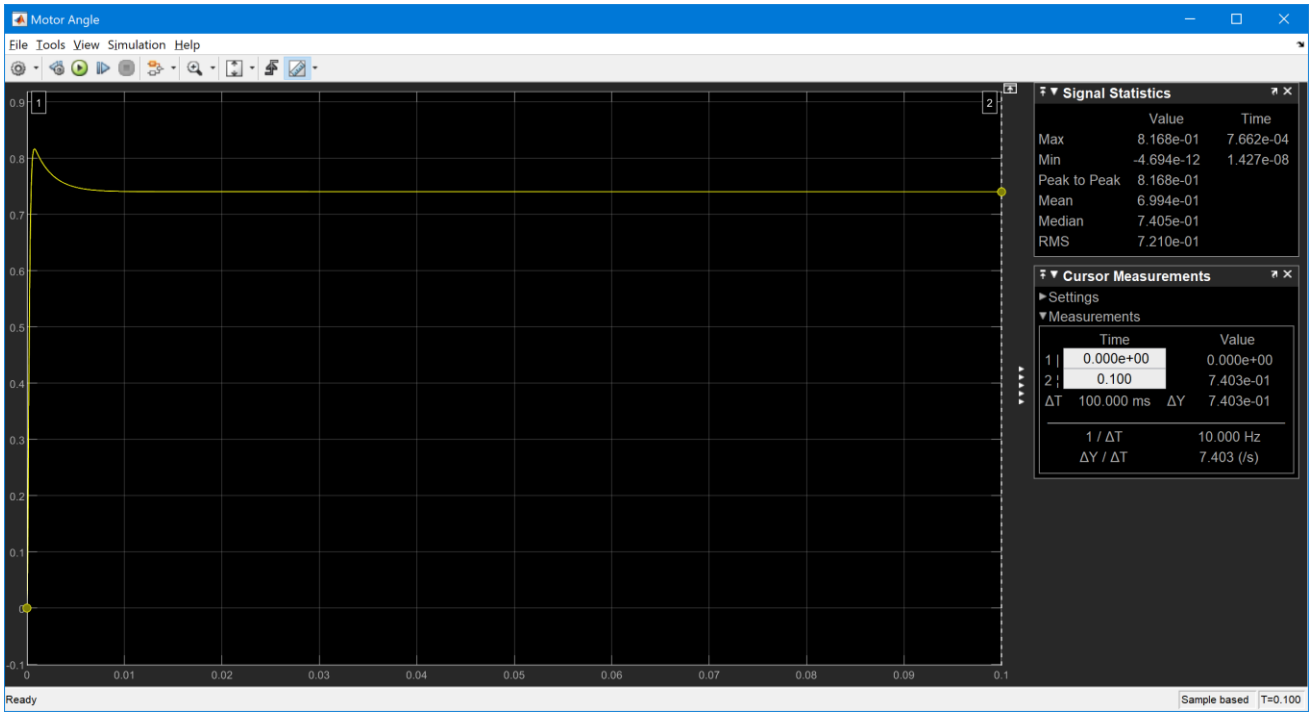


- Since you have clicked the Run button. The Simulink Model has finished the whole simulation process. You can double click "Motor Angle" block to view the result. See the following figure. As we can find, the start angle is 0 rad and the final angle is 0.6678 rad, which is very close to $\frac{\pi}{4}$. During the whole simulation process, the angle is oscillating around the set point. This result is fit to our expectation.



- Now, you are expected to finish the PID controller. Find the Discrete-Time Integrator block and Discrete Derivative block in the Library Browser. Drag them into the appropriate position in the workspace and connect them to the existed system.
- Set Sample time of Discrete-Time Integrator to 0.0001. Adjust Gain values for the 3 blocks of the PID controller, that is K_p, K_i and K_d. The limitation of the Gain values is between 0 and 10. Please do not use values beyond this interval. Also please do not change other parameter settings of the 3 blocks of PID controller and other parts of the system.
- There are some experiences to adjust these three values:
    - Start with $K_i = 0.0000001$, $K_d = 0$, and K_p be the default.
    - Gradually change the K_p to achieve the best control result. Too high K_p value will make the system unstable. Too low K_p value will make the system take too long time to rotate to the target position. So, use your judgement carefully.

- Once you get a good control with a K_p, adjust the value of K_i. Observe how the K_i will affect your control system. You may need to slightly adjust K_p while you are trying to find the best value for K_i.
- Once you have your satisfied K_p and K_i, you can work on K_d. Observe how the K_d will affect your control system.



Score

Demonstrate your work in a report including:

- Schematic of completed the PID controller: 25%
- A table that shows the selected K_p, K_i and K_d values, along with the overshooting and the final value at each step: 25%
- A screenshot of final Motor Angle result (Same as above figure). Your final simulation should not be worse than the above result, where the max overshooting value is 0.8168, the final value is 0.7403 and there is only 1 obvious oscillation of the angle.: 25%
- Explanation of your observation from the effect of changing K_p, K_i and K_d on the system response: 25%

Save your report in a PDF file named "*your macid*.pdf" and upload it in Avenue.