FEEDBACK CONTROL SYSTEMS
LAB MANUAL

ROLL # ________________

DEPARTMENT OF ELECTRICAL ENGINEERING,
FAST-NU, LAHORE
## Table of Contents

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Description</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>List of Equipment</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Experiment No.1, Transfer Function &amp; System Response</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Experiment No.2, Introduction to MATLAB SIMULINK</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>Experiment No.3, Mathematical Modeling of Physical Systems</td>
<td>17</td>
</tr>
<tr>
<td>5</td>
<td>Experiment No.4, MATLAB Experiment – Feedback in Control Systems</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>Experiment No.5, Second Order System Analysis using MATLAB</td>
<td>28</td>
</tr>
<tr>
<td>7</td>
<td>Experiment No.6, QNET DC Motor Control Trainer</td>
<td>31</td>
</tr>
<tr>
<td>8</td>
<td>Experiment No.7, QNET DC Motor Speed Control</td>
<td>35</td>
</tr>
<tr>
<td>9</td>
<td>Experiment No.8, QNET DC Motor Position Control</td>
<td>41</td>
</tr>
<tr>
<td>10</td>
<td>Experiment No.9, QNET Rotary Pendulum Trainer</td>
<td>45</td>
</tr>
<tr>
<td>11</td>
<td>Experiment No.10, QNET-ROTPENT Balance Control Design</td>
<td>49</td>
</tr>
<tr>
<td>12</td>
<td>Experiment No.11, QNET-ROTPENT Swing-Up Control</td>
<td>52</td>
</tr>
<tr>
<td>13</td>
<td>Experiment No.12, QNET-ROTPENT Energy Control</td>
<td>56</td>
</tr>
<tr>
<td>14</td>
<td>Experiment No.13, Root Locus Analysis</td>
<td>59</td>
</tr>
<tr>
<td>15</td>
<td>Experiment No.14, Design of lead and lag compensator using Root Locus</td>
<td>62</td>
</tr>
<tr>
<td>16</td>
<td>Appendix A: Lab Evaluation Criteria</td>
<td>70</td>
</tr>
<tr>
<td>17</td>
<td>Appendix B: Safety around Electricity</td>
<td>71</td>
</tr>
<tr>
<td>18</td>
<td>Appendix C: Guidelines on Preparing Lab Reports</td>
<td>73</td>
</tr>
</tbody>
</table>
## List of Equipment

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Computer</td>
</tr>
<tr>
<td>2</td>
<td>NI ELVIS II Trainer</td>
</tr>
<tr>
<td>3</td>
<td>QNET DC Motor Control Trainer</td>
</tr>
<tr>
<td>4</td>
<td>QNET Rotary Pendulum Trainer</td>
</tr>
</tbody>
</table>
EXPERIMENT 1

Transfer Function & System Response

Objective:

1. To understand the MATLAB functions used to define the transfer function and response of a system.
2. To understand the basic MATLAB inbuilt functions used to solve complicated polynomials.
3. To find the inverse Laplace transform and to compute partial fraction expansion of the ratio of two polynomials.

1.1 Transfer Function with MATLAB

Use the following MATLAB commands in all in-Lab tasks. Use MATLAB ‘help’ to find the purpose of these commands

a. tf(num, den)
b. bode(sys)
c. step(sys)
d. impulse(sys)

Tasks:

1. For the following circuits, Find

i. Transfer Function
ii. Identify whether it is low pass, high pass, band pass or band reject filter
iii. Step response
iv. Impulse response

Use $R = 3$, $L = 4$ and $C = 2$
2. For the following circuits, Find

i. Transfer Function
ii. Identify whether it is low pass, high pass, band pass or band reject filter
iii. Step response
iv. Impulse response

Use $R_L = 3; \ L = 4; \ C = 2; \ R_1 = R_2 = 4; \ R_3 = 2; \ R_4 = 4$
1.2 Partial Fraction Expansion with MATLAB

When trying to find the inverse Laplace transfer (or inverse z transform), it is helpful to be able to break a complicated ratio of two polynomials into forms that exist on the Laplace Transform (or z transform) table. The “residue” function of MATLAB can be used to compute the partial fraction expansion (PFE) of the ratio of two polynomials. The “tf2zp” function returns poles, zeros and K of a transfer function. The “zp2tf” function returns the numerator and denominator coefficients when zeros, poles and K are sent as input parameters.

a. \([r,p,k] = \text{residue}(\text{num}, \text{den})\)
b. \([z,p,K] = \text{tf2zp}(\text{num}, \text{den})\)
c. \([\text{num,den}] = \text{zp2tf}(z,p,K)\)
**Question 1:**

Obtain the inverse Laplace transform of the following $F(s)$. [Use MATLAB to find the partial fraction expansion of $F(s)$]. Write the inverse Laplace transform in the text box below

$$F(s) = \frac{s^5 + 8s^4 + 23s^3 + 35s^2 + 28s + 3}{s^3 + 6s^2 + 8s}$$

**Question 2:**

Given the zero(s), pole(s), and gain $K$ of $B(s)/A(s)$, obtain the function $B(s)/A(s)$ using MATLAB. Consider the three cases below. Write the transfer function of each in the text box below:

- There is no zero. Poles are at $-1+2j$ and $-1-2j$, $K=10$
- A zero is at 0. Poles are at $-1+2j$ and $-1-2j$, $K=10$
- A zero is at $-1$. Poles are at $-2, -4$ and $-8$. $K=12$.

**Question 3:**

A function $B(s)/A(s)$ consists of the following zeros, poles, and gain $K$:

- Zeros at $s=-1, s=-2$
- Poles at $s=0, s=-4, s=-6$
- Gain $K=5$

Obtain the expression for $B(s)/A(s) = \text{num} / \text{den}$ with MATLAB and write it in the box below:
**Question 4:**
Obtain the partial fraction expansion of the following function with MATLAB:

\[ F(s) = \frac{10(s+2)(s+4)}{(s+1)(s+3)(s+5)^2} \]

Then, obtain the inverse Laplace transform of \( F(s) \). Write the inverse Laplace transform in the box below:


**Question 5:**
Consider the following function \( F(s) \):

\[ F(s) = \frac{s^4 + 5s^3 + 6s^2 + 9s + 30}{s^4 + 6s^3 + 21s^2 + 46s + 30} \]

Using MATLAB, obtain the partial fraction expansion of \( F(s) \). Then, obtain the inverse Laplace transform of \( F(s) \) and write it in the box below:


**Post Lab**

**Question 1:**
Solve the following differential equation using MATLAB

\[ \ddot{x} + 2\dot{x} + 10x = e^{-t}, \ x(0) = 0, \ \dot{x}(0) = 0 \]

The function \( e^{-t} \) is given at \( t=0 \) when the system is at rest.
**Question 2:** Find the transfer function of following circuit.

![Circuit Diagram](image)

**Figure 1.2**

**Question 3:**

a) How can LTI filters be uniquely identified by their impulse response?

b) How can you identify the order of a system from its differential equation?

c) How can Laplace transform offer a convenient method for the solution of linear, time-invariant differential equations?
EXPERIMENT 2

Introduction to MATLAB SIMULINK

Objective:

1. To understand MATLAB SIMULINK and implement system’s transfer function using it.
2. To solve the system equations and obtain the response of the system for different inputs.

Getting started with MATLAB SIMULINK

2.1 SIMULINK Tutorial
SIMULINK is the Graphical User Interface (GUI) for MATLAB. This section presents a brief tutorial on how to use SIMULINK to create an open-loop block diagram.

1. Start MATLAB and at the prompt type “SIMULINK” (all lower case) or you can click on the icon located on toolbar.

![MATLAB Icon](image)

Figure 2.1

2. If installed, the SIMULINK Library Browser will soon pop up.

3. Click on the new icon, identical to a MS Word new file icon. That is your space to work in. After creating a model it can be saved (using the save icon).
4. To build simulation models, you will be creating block diagrams. In general all blocks are double-clickable to change the values within. You can connect the ports on each block via arrows easily by clicking and dragging with the mouse. You can also double-click any arrow (these are the control variables) to label what it is. Same with all block labels (SIMULINK will give a default name that you can change).
5. SIMULINK uses EE lingo. Sources are inputs and sinks are outputs. If you click around in the SIMULINK Library Browser, you will see the possible sources, blocks, and sinks you have at your disposal.

Example:

Now let us create a simple one-block transfer function and simulate it subject to a unit step input.

The given open-loop transfer function is \( G(s) = \frac{1}{s^2+2s+8} \)

a. Click the new icon in the SIMULINK Library Browser to get a window to work in (untitled with the SIMULINK logo).

b. Double-click the “Continuous” button in the SIMULINK Library Browser to see what blocks are provided for continuous control systems. Grab and slide the “Transfer Fcn” block to your workspace.
c. Double-click the block in your workspace and enter 1 in Numerator coefficients and 1 2 8 in Denominator coefficients and close by clicking OK. SIMULINK will update the transfer function in the block, both mathematically and visually.

d. Go ahead and save your model as name.mdl (whatever name you want, as long as it is not a reserved MATLAB word).

e. Click the Sources tab in the SIMULINK Library Browser to see what source blocks are provided. You will find a Step, Ramp, Since Wave, etc. Grab and slide the Step block to your workspace.
Double-click the Step block in your workspace and ensure 1 is entered as the final value (for a unit step) and that 0 is the initial value. Close by clicking OK.

f. Draw arrows

g. Click the Sinks tab in the SIMULINK Library Browser to see what sink blocks are provided. Grab and slide the Scope block to your workspace.

![SIMULINK Library Browser](image)

Figure 2.6

h. Draw an arrow from the “Transfer Fcn” block to the Scope block by using the mouse, the same method as before.

i. To run the model (solve the associated differential equation numerically and plot the output results vs. time automatically), simply push play (the solid black triangle button in your workspace window).

j. After it runs, double-click on your Scope to display the results.

Your final model will look like this:

![Final Model](image)

Figure 2.7
Exercise 1:

Generate the following MATLAB SIMULINK model and simulate its step response.

![SIMULINK Model](image)

**Figure 2.8**

Exercise 2:

a. Obtain the unit impulse response of the following system using SIMULINK.

\[
\frac{B(s)}{A(s)} = \frac{1}{s^2 + 0.2s + 1}
\]

b. Obtain the unit step response of the following system using SIMULINK.

\[
\frac{B(s)}{A(s)} = \frac{s}{s^2 + 0.2s + 1}
\]

Explain why the results in a and b are same.
POST LAB

Create a SIMULINK model with a first order system, with gain, $K = 1$, and time constant, $T = 0.1$ sec. Simulate a square wave input with unit amplitude and frequency of 0.3 Hz. The sample time is 0.001 sec. View the reference position, $x_r(t)$, input, $u(t)$, and actual position, $x(t)$, through a scope, as in Figure below. Experiment with different values of $K_p$ and observe how the system response changes. Plot the results.

Figure 2.9
EXPERIMENT 3

Mathematical Modeling of Physical Systems

Objective:

1. To understand the role of mathematical models of physical systems in design and analysis of control systems.
2. To learn MATLAB functions in solving and simulating such models.

3.1 Mechanical System: Mass-Spring System Model

Consider the following Mass-Spring system shown in the figure below.

![Figure 3.1](image)

K is spring constant.
B is friction coefficient
x(t) is the displacement
F_a(t) is the applied force

**Question1:**

a) Derive the second order differential equation of the above mass-spring system and write it down in the text box below:
b) Write the transfer function of the system in the text box below:

\[ \frac{X(s)}{F(s)} = \]

\[ \]

c) Write the state space equation of the above system in the text box below:

d) Construct a SIMULINK diagram to calculate the response of the Mass-Spring system. The input force increases from 0 to 8 N at \( t = 1 \) s. The parameter values are \( M = 2 \) kg, \( K = 16 \) N/m, and \( B = 4 \) N.s/m. (Draw a block diagram to represent this equation and draw the corresponding SIMULINK diagram before implementing it on SIMULINK).

3.2 Electrical System: RLC circuit

**Question 2:**
Consider the electrical circuits shown in the figures below.

![RLC Circuit](image)

(a)  

(b)
Figure 3.2

a) Derive the differential equation of the above systems and write it down in the text box below, also state its order:

b) Write the transfer function of the systems in the text box below:

c) Write the state space equation of the above system in the text box below:
d) Construct a SIMULINK diagram to calculate the response of the above systems. Use $R = 3$, $L = 4$ and $C = 2$. (Draw a block diagram to represent the equations and draw the corresponding SIMULINK diagram before implementing it on SIMULINK).

**Post Lab**

A permanent magnet dc motor is a mechanism which converts electrical power to mechanical power via magnetic coupling. The electrical power is provided by a voltage source, while the mechanical power is provided by a spinning rotor. A very basic dc motor is constructed of two main components: the rotor or armature and the stator. The armature rotates within the framework of the stationary stator. A simple illustration of a dc motor is shown in figure below:

![Figure 3.3](image)

The equivalent electrical circuit of a dc motor is illustrated in figure below:

![Figure 3.4](image)

The voltage across the inductor is proportional to the change of current through the coil with respect to time.

a) Find the differential equation of the motor, transfer function and state space equations.
b) Make the block diagram of the transfer function and implement it in MATLAB SIMULINK. You can reduce the block diagram according to your understanding and you are required to explain each step of reduction in detail.
c) Write detailed analysis on the step response of the DC motor. State the assumptions in your report.
EXPERIMENT 4

MATLAB Experiment – Feedback in Control Systems

Objective:

1. To obtain transfer functions of complex block diagrams through MATLAB.
2. To plot the responses of systems

4.1 Transfer Functions of Block Diagrams and Step Response

In control systems analysis, we frequently need to simplify a network of interconnected transfer functions into a single transfer function which is then used in subsequent calculations for analysis purposes. There are three different types of connections between transfer function that are usually encountered in practice: cascade-connected, parallel-connected and feedback-connected (closed-loop) transfer functions. MATLAB has convenient commands to obtain these transfer functions. To obtain the transfer functions of the cascaded, parallel, feedback and unity feedback systems, the following commands are used, respectively:

[num, den] = series (num1, den1, num2, den2)
[num, den] = parallel (num1, den1, num2, den2)
[num, den] = feedback (num1, den1, num2, den2)
[num, den] = cloop (num1, den1, -1)

Q1: Obtain \( Y(s)/X(s) = \frac{\text{num}}{\text{den}} \) for each of the arrangement of \( G_1(s) \) and \( G_2(s) \) and write the transfer function of each part in the box below.

1. \[ G_1(s) = \frac{10}{s^2 + 2s + 10} \]

2. \[ G_2(s) = \frac{5}{s + 5} \]
Figure 4.1
Q2: Evaluate the transfer function of the feedback system shown in the figure below using MATLAB. \(G_1(s) = 4, \ G_2(s) = 1/(s+2), \ H(s) = 5s\). Write the code and the transfer function in the box below

![Figure 4.2](image1)

Q3: Determine the transfer function of the following diagram. Check your answer with MATLAB

![Figure 4.3](image2)

**Transfer function:**
4.2 Transformation of Mathematical Models with MATLAB

\[
\begin{align*}
[A,B,C,D] &= \text{tf2ss}(\text{num}, \text{den}) \\
[\text{num,den}] &= \text{ss2tf} (A,B,C,D) \\
[z,p,k] &= \text{ss2zp} (A,B,C,D,i) \\
[\text{num, den}] &= \text{tfdata} (T,'v')
\end{align*}
\]

**Q4:** Transform the transfer function into the state space representation using MATLAB

(a) \( \frac{Y(s)}{U(s)} = \frac{s}{(s+10)(s^2+4s+16)} \)

(b) \( \frac{Y(s)}{U(s)} = \frac{s^2+7s+2}{s^3+9s^2+26s+24} \)

Write the state space representation of each transfer function in the box below.

**Q5:** Obtain the transfer function defined by the state space equations using MATLAB. Write the factored form in the box below

(a) \[
\begin{align*}
\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -6 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\
y &= \begin{bmatrix} 8 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}
\end{align*}
\]
(b)

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3 \\
\end{bmatrix} = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-1 & -2 & -3 \\
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
\end{bmatrix} + \begin{bmatrix}
10 \\
0 \\
0 \\
\end{bmatrix} u(t)
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 \\
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
\end{bmatrix}
\]

**Q6:** Obtain the closed-loop transfer function and the function to obtain the closed-loop state space model of the system shown below, using MATLAB.

![Figure 4.4](image)
Transfer function:

State-space Equation:

Post Lab

Figure 4.5

Given a mass-spring damper system

B = damping constant

M = mass

K = spring constant
F = u = force

The state space model for this system is shown below

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} = \begin{bmatrix}
0 & 1 \\
-k/m & -c/m
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} + \begin{bmatrix}
0 \\
1/m
\end{bmatrix} u
\]

\[
y = [1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}
\]

a) Define the state-space model above using the ss function in MATLAB.

Set c=1, m=1, k=50.

b) Use MATLAB to simulate the result. Explain each step in detail.

c) Find the transfer function from the state-space model.

d) Find the responses of the system using SIMULINK and insert the screen shots.
EXPERIMENT 5

Second Order System Analysis using MATLAB

Objective:

1. To implement the systems in MATLAB
2. To understand the system transient and steady-state responses.

The steady-state response means the manner in which the system output behaves as t approaches infinity. The transient response means that which goes from the initial state to the final state.

5.1 Transient Response

Q1: Write MATLAB program to obtain the unit-step response curves for the following systems:
(a).
\[
\begin{bmatrix}
  \dot{x}_1 \\
  \dot{x}_2 \\
  y
\end{bmatrix} = \begin{bmatrix}
  0 & 0 & 0 \\
  -6 & -1 & -3 \\
  8 & 1 & 0
\end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2 \\
  y
\end{bmatrix} + \begin{bmatrix}
  0 \\
  1 \\
  0
\end{bmatrix} u(t)
\]

b).
\[
\begin{bmatrix}
  \dot{x}_1 \\
  \dot{x}_2 \\
  \dot{x}_3 \\
  y
\end{bmatrix} = \begin{bmatrix}
  0 & 0 & 0 \\
  0 & 0 & 0 \\
  -1 & -2 & -3 \\
  1 & 0 & 0
\end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  y
\end{bmatrix} + \begin{bmatrix}
  0 \\
  0 \\
  0 \\
  10
\end{bmatrix} u(t)
\]

Q2: For \(\zeta=0.4\) and \(\omega_n = 5\, rad/sec\). Obtain the standard second order system using MATLAB. Write the transfer function in the box below:

Q3: For the system found in Question 2, write a MATLAB program to find the rise time, peak time, maximum overshoot and the settling time. Use \texttt{ltiview()} command.
Peak Time:
Maximum Overshoot:
Rise Time:
Settling Time:

5.2 Steady State Response

Q4: Obtain the partial fraction expansion of $C(s)$ using MATLAB when $R(s)$ is a unit step function. Hence write down the time response $c(t)$ in the space below:

$$
\frac{C(s)}{R(s)} = \frac{3s^3 + 25s^2 + 72s + 80}{s^4 + 8s^3 + 40s^2 + 96s + 80}
$$

Q5: When the closed-loop system involves a numerator dynamics, the unit-step response curve may exhibit a large overshoot. Comment on this statement after obtaining the unit-step response and the unit-ramp response of the following system with MATLAB.

$$
\frac{C(s)}{R(s)} = \frac{10s + 4}{s^2 + 4s + 4}
$$

Comment:

Q6: Using MATLAB, obtain the unit-ramp response of the closed-loop control system whose closed-loop transfer function is given below. Also, obtain the response of this system when the input is given by: $r = e^{-0.5t}$

$$
\frac{C(s)}{R(s)} = \frac{s + 10}{s^3 + 6s^2 + 9s + 10}
$$

Q7: Consider the system show in figure below. Determine the value of $k$ such that damping ratio is 0.5. Using MATLAB obtain the rise time, peak time, maximum overshoot, and settling time in the unit-step response and fill the following table.
Rise Time:  
Peak Time:  
Maximum overshoot  
Settling Time  
Value of k

Post Lab

Q1: Using MATLAB, obtain the unit-step response, unit-ramp response, and unit impulse response of the system defined below. Where $R(s)$ and $C(s)$ are LAPLACE transform of the input $r(t)$ and output $c(t)$ respectively.

$$\frac{C(s)}{R(s)} = \frac{10}{s^2 + 2s + 10}$$

Q2: Consider the differential equation system given by:

$$\dot{y} + 3\dot{y} + 2y = 0, \quad y(0) = 0.1, \quad \dot{y}(0) = 0.05$$

Using `syms` and `dsolve` commands, obtain the response $y(t)$ subject to the given initial condition. Use MATLAB help to understand these commands.
**EXPERIMENT 6**

**QNET DC Motor Control Trainer**

**Objective:**

The DC Motor Control Trainer illustrates the fundamentals of DC motor controls using the NI ELVIS platform and NI LabVIEW. The experiment is designed to provide the understanding of how to use the virtual interface to take speed and voltage measurements of the responses. The components in the trainer are marked by the ID number and explained in the table below.

![Image of QNET DC Motor Control Trainer](image)

**Figure 6.1**

<table>
<thead>
<tr>
<th>ID#</th>
<th>Description</th>
<th>ID#</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NI ELVIS II Trainer</td>
<td>6</td>
<td>Connection btw PC and ELVIS II</td>
</tr>
<tr>
<td>2</td>
<td>Prototyping Board Power switch</td>
<td>7</td>
<td>QNET DC Motor Control Trainer</td>
</tr>
<tr>
<td>3</td>
<td>Power LED</td>
<td>8</td>
<td>QNET Power LEDs</td>
</tr>
<tr>
<td>4</td>
<td>Ready LED</td>
<td>9</td>
<td>Power Cable for QNET</td>
</tr>
<tr>
<td>5</td>
<td>Power Cable for ELVIS II</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The DCMCT components in the figure above are marked by an ID number and explained in the table below:

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DC motor</td>
</tr>
<tr>
<td>2</td>
<td>High resolution encoder</td>
</tr>
<tr>
<td>3</td>
<td>Motor metal chamber</td>
</tr>
<tr>
<td>4</td>
<td>Inertial Load</td>
</tr>
</tbody>
</table>

### 6.1 Modeling:

The DCMCT Modeling VU, runs the DC motor in open-loop and plots the corresponding speed and input voltage responses. This VI can be used to take speed and voltage measurements of the responses and runs a simulation of the DC motor in parallel. The table below describes the main elements of the QNET_DCMCT Model virtual instrument front panel.

<table>
<thead>
<tr>
<th>Speed $w_m$</th>
<th>Motor output speed numeric display rad/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current $I_m$</td>
<td>Motor armature current numeric display A</td>
</tr>
<tr>
<td>Voltage $V_m$</td>
<td>Motor input voltage numeric display</td>
</tr>
<tr>
<td>K</td>
<td>Motor model steady-state gain input box rad/(V.s)</td>
</tr>
<tr>
<td>Tau</td>
<td>Motor model time constant input box</td>
</tr>
<tr>
<td>Sampling rate</td>
<td>Sets the sampling rate of VI</td>
</tr>
<tr>
<td>Stop</td>
<td>Stops the LABVIE VI from running</td>
</tr>
</tbody>
</table>
6.1.1: Bumptest

1. Open the QNET_DCMCT_Modeling.vi.
2. Ensure the correct Device is chosen as shown in figure 1.

3. Run the QNET_DCMCT_Modeling.vi. The DC motor should begin spinning.
4. In the Signal Generator section set:
   - Amplitude = 2.0 V
   - Frequency = 0.40 Hz
   - Offset = 3.0 V

5. Once you have collected a step response, click on the Stop Button to stop running the VI.
6. Select the Measurement Graphs tab to view the measured response.
7. **Exercise 6.1.1a:** Use the responses in the speed (rad/s) and Voltage (V) graphs to compute the steady-state gain of the DC motor and fill out the values in the following table. See the Bumptest Method section in the QNET Practical Control Guide for details on how to find the steady-state gain from a step response. Finally you can use the Graph Palette for zooming functions and the Cursor Palette to measure data.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steady-state motor speed</td>
<td>(\omega_{ss})</td>
<td>rad/s</td>
<td></td>
</tr>
<tr>
<td>Initial step motor speed</td>
<td>(\omega_i)</td>
<td>rad/s</td>
<td></td>
</tr>
<tr>
<td>Input step amplitude</td>
<td>(A_s)</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Measured steady-state gain using bumptest</td>
<td>(K_{ab})</td>
<td>rad/(V.s)</td>
<td></td>
</tr>
</tbody>
</table>

8. **Exercise 6.1.1b:** Based on the Bumptest Method, find the time constant. Make sure you complete the table below and see the section in the QNET Practical Control Guide for information on how to find the time constant of the step response.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decay speed</td>
<td>(\omega_{b0}(t_1))</td>
<td>rad/s</td>
<td></td>
</tr>
<tr>
<td>Initial step time</td>
<td>(t_0)</td>
<td>s</td>
<td></td>
</tr>
<tr>
<td>Decay step time</td>
<td>(t_1)</td>
<td>s</td>
<td></td>
</tr>
<tr>
<td>Measured time constant using bumptest</td>
<td>(\tau_{ab})</td>
<td>s</td>
<td></td>
</tr>
</tbody>
</table>
9. Enter the steady-state gain and time constant values found in this section in the table below. These are called the bumptest model parameters.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-Lab: Bumptest Modeling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open-Loop Steady-State Gain</td>
<td>$K_{a,b}$</td>
<td></td>
<td>rad/(V.s)</td>
</tr>
<tr>
<td>Open-Loop Time Constant</td>
<td>$\tau_{a,b}$</td>
<td></td>
<td>s</td>
</tr>
</tbody>
</table>

6.1.2: Model Validation

1. Open the QNET_DCMCT_Modeling.vi.
2. Ensure the correct Device is Chosen.
3. Run the QNET_DCMCT_Modeling.vi. You should hear the DC motor begin running.
4. In the Signal Generator section set:
   - Amplitude = 2.0 V
   - Frequency = 0.40 Hz
   - Offset = 3.0 V

5. In the Model Parameters section of the VI, enter the bumptest model parameters, $K$ and $\tau$, that were found in section 6.1.1. The blue simulation should match the red measured motor speed more closely. **How well does your model represent the actual system? If they do not match, name one possible source of discrepancy**

   **Answer:**

6. **Exercise 6.1.2a:** Tune the steady-state gain, $K$, and the time constant $\tau$ in the model Parameters section so the simulation matches the actual system better. Enter both the bumptest and tuned model parameters in the following table.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-Lab: Bumptest Modeling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open-Loop Steady-State Gain</td>
<td>$K_{a,b}$</td>
<td></td>
<td>rad/(V.s)</td>
</tr>
<tr>
<td>Open-Loop Time Constant</td>
<td>$\tau_{a,b}$</td>
<td></td>
<td>s</td>
</tr>
<tr>
<td>In-Lab: Model Validation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open-Loop Steady-State Gain</td>
<td>$K_{a,v}$</td>
<td></td>
<td>rad/(V.s)</td>
</tr>
<tr>
<td>Open-Loop Time Constant</td>
<td>$\tau_{a,v}$</td>
<td></td>
<td>s</td>
</tr>
</tbody>
</table>
EXPERIMENT 7

QNET DC Motor Speed Control

Objective:

The experiment is designed in order to provide full understanding of PI controllers. The aim is to give you the knowledge to simulate and validate the controllers.

The speed of the DC motor is controlled using proportional integral control system. The PI control also includes set point weight. The transfer function representing the DC-motor speed voltage relation is used to design the PI controller. The table below describes the main elements of the QNET_DCMCT Speed virtual instrument front panel

<table>
<thead>
<tr>
<th>Speed $w_m$</th>
<th>Motor output speed numeric display rad/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current $i_m$</td>
<td>Motor armature current numeric display A</td>
</tr>
<tr>
<td>Voltage $V_m$</td>
<td>Motor input voltage numeric display</td>
</tr>
<tr>
<td>Disturbance $V_{sd}$</td>
<td>Apply simulated disturbance voltage</td>
</tr>
<tr>
<td>$K_p$</td>
<td>Controller proportional gain input box</td>
</tr>
<tr>
<td>$K_i$</td>
<td>Controller integral gain input box</td>
</tr>
<tr>
<td>Bsp</td>
<td>Controller set point weight input box</td>
</tr>
</tbody>
</table>

7.1 Qualitative PI Control

1. Open the QNET_DCMCT_Speed_Control.vi.
2. Ensure the correct Device is chosen.
3. Run the QNET_DCMCT_Speed_Control vi. The motor should begin rotating.

4. In the Signal Generator section set:
   
   * Signal type = ‘square wave’
   * Amplitude = 25.0 rad/s
   * Frequency = 0.40 Hz
   * Offset = 100.0 rad/s

5. In the Control Parameters section set:
   
   * $k_p = 0.050 \text{ V.s/rad}$
   * $k_i = 1.00 \text{ V/rad}$
   * $bsp = 0.00$

6. Exercise 7.1a: Examine the behavior of the measured speed, shown in red, with respect to the reference speed, shown in blue, in the Speed (rad/s) scope. Explain what is happening.
7. Increment and decrement $kp$ by steps of 0.005 V.s/rad.

8. **Exercise 7.1b:** Look at the changes in the measured signal with respect to the reference signal. Explain the performance difference of changing $kp$.

9. Set $kp$ to 0 V.s/rad and $ki$ to 0 V/rad. The motor should stop spinning.

10. Increment the integral gain, $ki$, by steps of 0.05 V/rad. Vary the integral gain between 0.05 V/rad and 1.00 V/rad.

11. **Exercise 7.1c:** Examine the response of the measured speed in the Speed (rad/s) scope and compare the result with $ki$ is set low to when it is set high. Explain what is happening?

12. Stop the VI by clicking on the Stop button.

**7.2 PI Control According to Specifications**

1. **Exercise 7.2a:** Using the equations outlined in the Peak time and Overshoot section of the QNET Practical Control Guide, calculate the expected peak time, $tp$, and percentage overshoot, $PO$, given the following Speed Lab Designs (SLD) specifications:

   $\zeta = 0.75$

   $\omega_o = 16.0 \text{ rad/s}$

   Optional: You can also design a VI that simulates the DC motor first-order model with a PI control and have it calculate the peak time and overshoot.
2. **Exercise 7.2b:** Calculate the proportional, \( kp \), and the integral, \( ki \), control gains according to the model parameters found in modeling section and SLD specifications.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural frequency specification</td>
<td>( \omega_0 )</td>
<td>16.0</td>
<td>rad/s</td>
</tr>
<tr>
<td>Damping ratio specification</td>
<td>( \zeta )</td>
<td>0.75</td>
<td></td>
</tr>
<tr>
<td>Peak time</td>
<td>( t_p )</td>
<td></td>
<td>s</td>
</tr>
<tr>
<td>Percentage overshoot</td>
<td>PO</td>
<td></td>
<td>%</td>
</tr>
</tbody>
</table>

3. Run the QNET_DCMCT_Speed_Control.vi. The motor should begin spinning.

4. In the Signal Generator set
   - Signal type = ‘square wave’
   - Amplitude = 25.0 rad/s
   - Frequency = 0.40 Hz
   - Offset = 100.0 rad/s

5. In the Control Parameters section, enter the SLD PI control gains found in Exercise 7.2 b and make sure bsp = 0.00.

6. Stop the VI when you obtain two sample cycles by clicking on the Stop button.

7. **Exercise 7.2c:** Capture the measured SLD speed response. Make sure you include both the Speed (rad/s) and the control signal Voltage (V) scopes.

8. **Exercise 7.2d:** Measure the peak time and percentage overshoot of the measured SLD response. Are the specifications satisfied?
9. **Exercise 7.2e:** What effect does increasing the specification zeta have on the measured speed response? How about on the control gains? Use the damping ratio equation given the in the peak time and Overshoot section of the QNET Practical Control Guide for more help if needed.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Behaviour</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peak time</td>
<td>$t_p$</td>
<td></td>
<td>$s$</td>
</tr>
<tr>
<td>Percentage overshoot</td>
<td>PO</td>
<td></td>
<td>%</td>
</tr>
<tr>
<td>Proportional gain</td>
<td>$k_p$</td>
<td></td>
<td>V.s/rad</td>
</tr>
<tr>
<td>Integral gain</td>
<td>$k_i$</td>
<td></td>
<td>V/rad</td>
</tr>
</tbody>
</table>

10. **Exercise 7.2f:** What effect does increasing the specification $\omega_o$ have on the measured speed response and the generated control gains? Use the natural frequency equation found in the Peak Time and Overshoot section of the QNET Practical Control Guide for more help if needed.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Behaviour</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peak time</td>
<td>$t_p$</td>
<td></td>
<td>$s$</td>
</tr>
<tr>
<td>Percentage overshoot</td>
<td>PO</td>
<td></td>
<td>%</td>
</tr>
<tr>
<td>Proportional gain</td>
<td>$k_p$</td>
<td></td>
<td>V.s/rad</td>
</tr>
<tr>
<td>Integral gain</td>
<td>$k_i$</td>
<td></td>
<td>V/rad</td>
</tr>
</tbody>
</table>

**7.3 Effect of Set-Point Weight**

1. Run the QNET_DCMCT_Speed_Control vi. The motor should begin rotating.
2. In the Signal Generator section set:
   - Signal type = 'square wave'
   - Amplitude = 25.0 rad/s
   - Frequency = 0.40 Hz
   - Offset = 100.0 rad/s
3. In the Control Parameters section set:
   - $kp = 0.050$ V.s/rad
   - $ki = 1.00$ V/rad
   - bsp = 0.00
4. Increment the set-point weight parameter bsp in steps of 0.05. Vary the parameter between 0 and 1.
5. **Exercise 7.3a:** Examine the effect that raising bsp has on the shape of the measured speed signal in the Speed (rad/s) scope. Explain what the set-point weight parameter is doing.
6. Stop the VI by clicking on the Stop button

7.3.1. Tracking Triangular Signals
1. Run the QNET_DCMCT_Speed_Control vi. The motor should begin rotating.
2. In the Signal Generator section set:
   Signal type = ‘triangular wave’
   Amplitude = 50.0 rad/s
   Frequency = 0.40 Hz
   Offset = 100.0 rad/s
3. In the Control Parameters section set:
   kp = 0.020 V.s/rad
   ki = 0.00 V/rad
   bsp = 1.00
4. Exercise 7.3.1a: Compare the measured speed and the reference speed. Explain why there is a tracking error.

5. Increase ki to 0.1 V/rad and examine the response. Vary ki between 0.1 V/rad and 1.0 V/rad.
6. Exercise 7.3.2b: What effect does increasing ki have on the tracking ability of the measured signal? Explain using the observed behavior in the scope.

7. Stop the VI by clicking on the Stop button.
Post Lab

a) What do you understand by inertia, friction and dampening?

b) What kind of controller is used to control the speed of the motor? Give reasons to support your answer.

c) Explain the purpose of proportional, integral and derivative gains and how do they affect the speed of the motor.
EXPERIMENT 8

QNET DC Motor Position Control

Objective:

The aim is to make you understand the system, commission the system, run and evaluate it. Control of motor position is a natural way to introduce the benefits of derivative action. In this experiment performance of proportional-derivative controller is observed. $k_p$ is the proportional control gain and $kd$ is the derivative control gain.

8.1 Qualitative PD Control

1. Open the QNET_DCMCT_Position_Control.vi.
2. Ensure the correct Device is chosen.
3. Run the QNET_DCMCT_Speed_Control vi. The motor should begin rotating.
4. In the Signal Generator section set:
   - Amplitude = 2.00 rad
   - Frequency = 0.40 Hz
   - Offset = 0.00 rad
5. In the Control Parameters section set:
   - $k_p$ = 2.00 V.s/rad
   - $k_i$ = 0.00 V/rad
   - $kd$ = 0.00 V.rad.
6. Change the proportional gain, $k_p$, by steps of 0.25 V.rad. Try the following gains $k_p = 0.5, 1, 2, and 4$ V.rad.
7. Exercise 8.1a: Examine the behavior of the measured position (red line) with respect to the reference position (blue line) in the Position (rad) scope. Explain what is happening.

8. Exercise 8.1b: Describe the steady-state error to a step input.

9. Increment the derivative gain, $kd$, by steps of 0.01 V.s/rad.
10. **Exercise 8.1c**: Look at the changes in the measured position with respect to the desired position. Explain what is happening.

11. **Exercise 8.1d**: Using the equations outlined in the *Peak Time* and *Overshoot* section of the QNET Practical Control Guide, calculate the expected peak time, $t_p$, and percentage overshoot, $PO$, given the following specifications:
   
   - Zeta $= 0.75$
   - $w_0 = 16.0$ rad/s

12. **Exercise 8.1e**: Calculate the proportional, $k_p$, and integral, $k_i$, control gains according to the model parameters found in previous experiment and the SLD specifications.

13. Stop the VI by clicking on the Stop button.

### 8.2 PD Control according to Specifications

1. **Exercise 8.2a**: Using the equations in the *Peak Time* and *Overshoot* section of the QNET Practical Control Guide, calculate the expected peak time, $t_p$, and percentage overshoot, $PO$, given
   
   - Zeta $= 0.60$
   - $w_0 = 25.0$ rad/s
   - $p_0 = 0.0$

2. **Exercise 8.2b**: Calculate the proportional, $k_p$, and derivative, $k_d$, control gains according to the model parameters found in previous and the specifications above.
3. Run the QNET_DCMCT_Position_Control.vi. You should see the DC motor rotating back and forth.

4. In the Signal Generator section set:
   - Amplitude = 2.00 rad
   - Frequency = 0.4 Hz
   - Offset = 0.00 rad

5. In the control Parameters section set the PD gains found in previous experiment.

**Exercise 8.2c:** Measure the peak time and percentage overshoot of the measured position response. Are the specifications satisfied? If they are not, then given on possible reason why there would be discrepancy.

6. **Exercise 8.2d:** What effect does changing the specification \( \zeta \) have on the measured position response and the generated control gains? See the Peak Time and Overshoot section of the QNET Practical Control Guide for more help.

7. Stop the VI by Clicking on the Stop button.
Post Lab

a) What kind of controller is used to control the position of the motor? Give reasons to support your answer.

b) Explain the purpose of proportional, integral and derivative gains and how do they affect the position of the motor. You can write the equations to support your answer
EXPERIMENT 9

QNET Rotary Pendulum Trainer

Objective:
Rotary pendulum system is chosen for you to understand a task-based controller. The experiment begins by modeling the system and determines strategies to dampen the oscillations of the system. Furthermore it makes you understand the relationship between stability and inertia of the system.

9.1 Simple Modeling
The QNET-ROTPENT Simple Modeling runs the DC motor connected to the pendulum arm in open-loop and plots the corresponding pendulum arm and link angles as well as the applied input motor voltage.

Figure 9.1
9.1.1. Dampening
1. Open the QNET_ROTPENT_Simple_Modeling.vi.
2. Ensure the correct Device is chosen.
3. Run the QNET_ROTPENT_Simple_Modeling.vi.
4. Hold the arm of the rotary pendulum system stationary and manually perturb the pendulum.
5. While still holding the arm, examine the response of Pendulum Angle (deg) in the Angle (deg) scope. This is the response from the pendulum system.
6. Repeat Step 3 above and release the arm after several swings.
7. **Exercise 9.1.1a:** Examine the Pendulum Angle (deg) response when the arm is not fixed. This is the response from the rotary pendulum system. Given the response from these two systems pendulum and rotary pendulum – which converges faster towards angle zero? Why does one system dampen faster than the other?
8. Stop the VI by clicking on the Stop button.

9.1.2. Friction
1. Run the QNET_ROTPENT_Simple_Modeling.vi.
2. In the Signal Generator section set:
   - Amplitude = 0.00 V
   - Frequency = 0.25 Hz
   - Offset = 0.00 V
3. Change the Offset in steps of 0.10 V until the pendulum begins moving. Record the voltage at which the pendulum moved.
4. Repeat Step 3 above for steps of -0.10 V.
5. **Exercise 9.1.2a:** Enter the positive and negative voltage values needed to get the pendulum moving in. Why does the motor need a certain amount of voltage to get the motor shaft moving?
6. Stop the VI by clicking on the Stop button.

9.1.3. Moment of Inertia


2. Run the QNET_ROTPEND_Simple_Modeling.vi
3. In the Signal Generator section set:
   - Amplitude = 1.00 V
   - Frequency = 0.25 Hz
   - Offset = 0.00 V

4. Click on the Disturbance toggle switch to perturb the pendulum and measure the amount of time it takes for the pendulum to swing back-and-forth in a few cycles (e.g. 4 cycles).

5. Exercise 9.1.3b: Find the frequency and moment of inertia of the pendulum using the observed results. See QNET Practical Control Guide to see how to calculate the inertia experimentally and make sure you fill the following Table.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive Coulomb Friction Voltage</td>
<td>$V_f$</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Negative Coulomb Friction Voltage</td>
<td>$V_f$</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Cycles</td>
<td>$n_{sys}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Duration</td>
<td>$\Delta t$</td>
<td></td>
<td>s</td>
</tr>
<tr>
<td>Frequency</td>
<td>$f$</td>
<td></td>
<td>Hz</td>
</tr>
<tr>
<td>Pendulum moment of inertia</td>
<td>$J_{p,exp}$</td>
<td></td>
<td>kg.m$^2$</td>
</tr>
</tbody>
</table>
6. **Exercise 9.1.3c:** Compare the moment of inertia calculated analytically in Exercise 9.1.3a and the moment of inertia found experimentally. Is there a large discrepancy between them?

7. Stop the VI by clicking on the Stop button.

**Post Lab**

a) What is rotary inverted pendulum system?

b) Define dampening and friction?
EXPERIMENT 10

QNET-ROTPENT Balance Control Design

Objective:
The experiment illustrates one of the important methods for finding the parameters of control strategies. It will give you a deeper understanding of LQR technique that is suitable for finding the parameters of the balance controller.

10.1 Model Analysis
1. Open the QNET_ROTPENT_Control_Design.vi, shown in Figure 9.
2. Run the QNET_ROTPENT_Control_Design.vi.
3. Select the Symbolic Model tab.
4. The Model Parameters array includes all the rotary pendulum modeling variables that are used in the state-space matrices A, B, C, and D.
5. Select the Open Loop Analysis tab.
6. **Exercise 10.1a:** This shows the numerical linear state-space model and a pole-zero plot of the open-loop inverted pendulum system. What do you notice about the location of the open-loop poles? Recommended: In the Model Parameters section, it is recommended to enter the pendulum moment of inertia, $J_p$, determined experimentally in previous experiment.

7. In the Symbolic Model tab, set the pendulum moment of inertia, $J_p$, to 1.0e-5 kg.m$^2$.
8. **Exercise 10.1b:** Select the Open Loop Analysis tab. How did the locations of the open-loop poles change with the new inertia? Enter the pole locations of each system with a different moment of inertia in the table below. Are the changes of having a pendulum with a lower inertia as expected?
9. Reset the pendulum moment of inertia, Jp, back to 1.77e-4.
10. Stop the VI by clicking on the Stop button.

10.2 Control Design and Simulation
1. Open the QNET_ROTPENT_Control_Design.vi, shown in Figure 10.
2. Select the Simulation tab.
3. Run the QNET_ROTPENT_Control_Design.vi.
4. In the Signal Generator section set:
   Amplitude = 45.0 deg
   Frequency = 0.20 Hz
   Offset = 0.0 deg
5. Set the Q and R LQR weighting matrices to the following:
   Q (1, 1) = 10, i.e. set first element of Q matrix to 10.
   R = 1.00
6. Changing the Q matrix generates a new control gain.
7. Exercise 10.2a: The arm reference (in red) and simulated arm response (in blue) are shown in the Arm (deg) scope. How did the arm response change? How did the pendulum response change in the Pendulum (deg) scope?
8. Set the third element in the Q matrix to 0, i.e. Q (3, 3) = 0.
9. Exercise 10.2b: Examine and describe the change in the Arm (deg) and Pendulum (deg) scope.
10. By varying the diagonal elements of the Q matrix, design a balance controller that adheres to the following specifications:
   Arm peak time less than 0.75 seconds: \( t_p \leq 0.75 \) s
   Motor voltage peak less than \( \pm 12.5 \) V: \( |V_m| \leq 12.5 \) V
   Pendulum angle less than 10.0 degrees: \( |\alpha| \leq 10.0 \) deg

11. Exercise 10.3c: Enter the Q and R matrices along with and control gain used to meet the specifications.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q(1,1)</td>
<td>( Q_{1,1} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q(2,2)</td>
<td>( Q_{2,2} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q(3,3)</td>
<td>( Q_{3,3} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q(4,4)</td>
<td>( Q_{4,4} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>( R )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>K(1)</td>
<td>( k_{\text{to}} )</td>
<td>( V/\text{rad} )</td>
<td></td>
</tr>
<tr>
<td>K(2)</td>
<td>( k_{\text{yo}} )</td>
<td>( V/\text{rad} )</td>
<td></td>
</tr>
<tr>
<td>K(3)</td>
<td>( k_{\text{d0}} )</td>
<td>( V.\text{s/\text{rad}} )</td>
<td></td>
</tr>
<tr>
<td>K(4)</td>
<td>( k_{\text{d\alpha}} )</td>
<td>( V.\text{s/\text{rad}} )</td>
<td></td>
</tr>
</tbody>
</table>

**Post Lab**

a) How is inertia related to stability? What does the pole-zero plot tells you about stability of the inverted pendulum system?

b) How is the arm response of pendulum related to proportional and derivative gain?
EXPERIMENT 11

QNET-ROTPENT Balance Control System

Objective:

Balancing is a common control task. In this experiment we will find control strategies that balance the pendulum in upright position while maintaining a desired position of the arm. The QNET rotary pendulum trainer swing up control VI implements a state-feedback controller to balance the pendulum when in its upright position.

11.1 Default Balance Control

1. Open the QNET_ROTPENT_Swing_Up_Control.vi.
2. Ensure the correct Device is chosen.
3. Run the QNET_ROTPENT_Swing_Up_Control.vi.
4. In the Signal Generator section set:
   Amplitude = 0.0 deg
   Frequency = 0.10 Hz
   Offset = 0.0 deg
5. In the Balance Control Parameters section set:
   kp_theta = -6.50 V/rad
   kp_alpha = 80 V/rad
   kd_theta = -2.75 V/(rad/s)
   kd_alpha = 10.5 V/(rad/s)
6. In the Swing-Up Control Parameters section set:
   mu = 55 m/s2/J
   Er = 20.0 mJ
   max accel = 10 m/s2
   Activate Swing-Up = OFF (de-pressed)
7. Adjust the Angle/Energy (deg/mJ) scope scales to see between -250 and 250 (see QNET user Manual for help)
8. Manually rotate the pendulum in the upright position until the In Range LED in the Control Indicators section turns bright green. Ensure the encoder cable does not interfere with the pendulum arm motion.
9. **Exercise 11.1a:** Vary Offset and observe the Arm Angle (deg) response in the Angle/Energy (deg/mJ) scope. Do not set the Offset too high or the encoder cable will interfere with the pendulum arm motion. What do you notice?
10. Exercise 11.1b: As the pendulum is being balanced, examine the red Arm Angle (deg) and blue Pendulum Angle (deg) responses in the Angle/Energy (deg/mJ) scope. What do you notice?

11. Click on the stop button to stop running the VI.

11.2 Implement Designed Balance Controller
1. Go through the ‘Control Design and Simulation’ Section in the previous experiment and design a balance control according to the given specifications. Remark: It is recommended to use the experimental determined pendulum moment of inertia that was found in ‘Moment of Inertia’ section in experiment 9.
2. Open the QNET_ROTPEND_Swing_Up_Control.vi.
3. Ensure the correct Device is chosen.
4. Run the QNET_ROTPEND_Swing_Up_Control.vi.
5. In the Signal Generator section set:
   - Amplitude = 45.0 deg
   - Frequency = 0.20 Hz
   - Offset = 0.0 deg
6. To implement your balance controller, enter the control gain found in ‘Control Design and Simulation’ (in the previous experiment) in kp_theta, kp_alpha, kd_theta, and kd_alpha in the Control Parameters section.
7. Manually rotate the pendulum in the upright position until the In Range? LED in the
Control Indicators section turns bright green. Ensure the encoder cable does not interfere with the pendulum arm motion.

8. Does your system meet the specifications given in the ‘Control Design and Simulation’ section in previous experiment?

9. Click on the Stop button to stop running the VI.

11.3 Balance Control with Friction Compensation

1. Go through steps 1-8 in Section 11.1 to run the default balance control.

2. In the Signal Generator section set:
   - Amplitude = 0.0 deg
   - Frequency = 0.10 Hz
   - Offset = 0.0 deg

3. In the Dither Signal section set:
   - Amplitude = 0.00 V
   - Frequency = 2.50 Hz
   - Offset = 0.00 V

4. Exercise 11.3a: Observe the behavior of Arm Angle (deg) in the Angle/Energy (deg/mJ) scope. Intuitively speaking, can you find some reasons why the arm is oscillating?

5. Increase the Amplitude in the Dither Signal section by steps of 0.1 V until you notice a change in the arm angle response.

6. Exercise 10.3b: From the Voltage (V) scope and the pendulum motion, what is the Dither signal doing? Compare the response of the arm with and without the Dither signal.
7. Increase the Frequency in the Dither Signal section starting from 1.00 to 10.0 Hz.
8. **Exercise 11.3c**: How does this affect the pendulum arm response?

9. Click on the Stop button to stop running the VI.

**Post Lab**

What do you understand by Balance controller and balance control with friction compensation?
EXPERIMENT 12

QNET-ROTPENT Energy Control

Objective:

The main objective of the experiment is to make you understand the balancing control law which performs the dual task of swinging up the pendulum and balancing it and to understand how switching is performed between the two control systems.

The QNET rotary pendulum trainer swing up control VI implements an energy-based control that swings up the pendulum to its upright vertical position and a state-feedback controller to balance the pendulum when in its upright position.

1. Open the QNET_ROTPENT_Swing_Up_Control.vi.
2. Ensure the correct Device is chosen.
3. Run the QNET_ROTPENT_Swing_Up_Control.vi.
4. In the Balance Control Parameters section ensure the following parameters are set:
   - kp_theta = -6.50 V/rad
   - kp_alpha = 80.0 V/rad
   - kd_theta = -2.75 V/(rad/s)
   - kd_alpha = 10.5 V/(rad/s)
5. In the Swing-Up Control Parameters section set:
   - mu = 55 m/s2/J
   - Er = 20.0 mJ
   - max accel = 10 m/s2
   - Activate Swing-Up = OFF (de-pressed)
6. Adjust the Angle/Energy (deg/mJ) scope scales to see between -250 and 250 (see QNET User Manual for help)
7. Manually rotate the pendulum at different levels and examine the blue Pendulum Angle (deg) and the green Pendulum Energy (mJ) in the Angle/Energy (deg/mJ) scope. The pendulum energy is also displayed numerically in the Control Indicators section.
8. Exercise 12.1a: What do you notice about the energy when the pendulum is moved at different positions? Record the energy when the pendulum is being balanced (i.e. fully inverted in the upright vertical position).
9. Click on the Stop button to bring the pendulum down to the gantry position and re-start the VI. In the Swing-Up Control Parameters section, set the Activate Swing-Up = ON (pressed) switch. If the pendulum is stationary, click on the Disturbance button in the Signal Generator section to perturb the pendulum.

10. **Exercise 12.1b**: In Swing-Up Control Parameters, change the reference energy $E_r$ between 5.0 mJ and 50.0 mJ. As it is varied, examine the control signal in the Voltage (V) scope as well as the blue Pendulum Angle (deg) and the red Pendulum Energy (mJ) in the Angle/Energy (deg/mJ) scope. What do you notice?

11. **Exercise 12.1c**: In Control Parameters fix $E_r$ to 20.0 mJ and vary the swing-up control gain $\mu$ between 10 and 100 m/s$^2$/J. Describe how this changes the performance of the energy control.

12. Click on Stop Control to disable the energy and balance controllers.

**12.1 Hybrid Swing-Up Control**

1. Open the QNET_ROOTPENT_Swing_Up_Control.vi.
2. Ensure the correct Device is chosen.
3. Run the QNET_ROOTPENT_Swing_Up_Control.vi.
4. In the Balance Control Parameters section verify the following parameters are set:
   
   $kp_{\theta} = -6.50$ V/rad
   $kp_\alpha = 80.0$ V/rad
   $kd_{\theta} = -2.75$ V/(rad/s)
   $kd_\alpha = 10.5$ V/(rad/s)
5. In the Swing-Up Control Parameters section set:

\[ \mu = 55 \text{ m/s}^2/\text{J} \]
\[ E_r = 20.0 \text{ mJ} \]
\[ \text{max accel} = 10 \text{ m/s}^2 \]

Activate Swing-Up = OFF (de-pressed)

6. Adjust the Angle/Energy (deg/mJ) scope scales to see between -250 and 250 (see QNET User Manual for help). Make sure the pendulum is hanging down motionless and the encoder cable is not interfering with the pendulum.

7. Set the Activate Swing-Up = ON (pressed) switch in the Swing-Up Control Parameters.

8. The pendulum should begin going back and forth. If not, click on the Disturbance button in the Signal Generator section to perturb the pendulum. Turn off the Active Swing-Up switch if the pendulum goes unstable or if the encoder cable interferes with the pendulum arm motion.

9. Gradually increase the reference energy \( E_r \) in the Control Parameters section to the energy read when the pendulum is vertically upwards. When that reference energy is reached, the pendulum should swing-up to the inverted position.

10. Click on the Stop button to stop running the VI.

Post Lab

a) What are hybrid systems?

b) What do you understand by Hybrid Swing Up control?
EXPERIMENT 13

Root Locus Analysis

Objective:

1. To understand the root locus thoroughly by implementing.
2. To analyze the root locus of the systems in MATLAB.

13.1 Design and analysis using Root Locus

The root locus of an (open-loop) transfer function $G(s)H(s)$ is a plot of the locations (locus) of all possible closed loop poles with proportional gain $K$ and unity feedback. The command used to plot root locus in MATLAB is

```matlab
sys = tf(num,den)
rlocus(sys)
```

We can use the following MATLAB command to find the location of closed loop poles for a specific gain $K$

```matlab
r=rlocus(num,den,K)
```

**Example 1:** Draw the root locus of the following unity feedback system using MATLAB

![Diagram of the system](image.png)

**Figure 13.1**

| $a = s(s + 1)$ | a. Determine the closed loop poles for $K=0$ and $K=\infty$? |
| $b = s^2 + 4s + 16$ | b. Find the break points? |
| $a=[1 1 0]$; | c. What is the location of closed loop poles at which system is marginally stable? |
| $b=[1 4 16]$; | d. Find the value of $K$ for which dominant closed-loop poles have damping ratio 0.5? |
| $c=conv(a,b)$ | e. Find the location of closed loop poles for $K = 10$. |
| $c=1 5 20 16 0$ | num=[0 0 1 3]; |
| num=[0 0 1 3]; | den=[1 5 20 16 0]; |
| rlocus(tf(num,den)) | |
Question 1: Plot the root loci for the system with

\[ G(s) = \frac{K}{s(s + 0.5)(s^2 + 0.6s + 10)} \quad H(s) = 1 \]

a. Determine the closed loop poles for \( K=0 \) and \( K=\infty \)?
b. Find the break points?
c. What is the location of closed loop poles at which system is marginally stable?
d. Find the value of \( K \) for which dominant closed-loop poles have damping ratio equal to one?

Question 2: Consider the system whose open-loop transfer function is given by

\[ G(s)H(s) = \frac{K(s - 0.66667)}{s^4 + 3.3401s^3 + 7.0325s^2} \]

Using MATLAB, Plot the root loci.

a. Determine the closed loop poles for \( K=0 \) and \( K=\infty \)?
b. Find the break points?
c. What is the location of closed loop poles at which system is marginally stable?
d. Find the value of \( K \) for which dominant closed-loop poles have damping ratio 0.7?
e. Find the location of closed loop poles for \( K = 100 \).

Question 3: Consider the system show in the figure below. Plot the root loci with MATLAB
Figure 13.2

a. Determine the closed loop poles for $K=0$ and $K=\infty$?
b. Find the break points?
c. What is the location of closed loop poles at which system is marginally stable?
d. Find the value of $K$ for which dominant closed-loop poles have damping ratio 0.7?
e. Find the location of closed loop poles for $K = 2$

**Question 4:**

Consider an open loop system which has a transfer function of

$$G(s)H(s) = \frac{K}{s(s + 3)(s^2 + 2s + 2)}$$

**Post Lab**

Draw the root locus for the following dynamic control system using MATLAB and perform complete analysis.

Figure 13.3
EXPERIMENT 14

Design of lead and lag compensator using Root Locus

Objective:

The experiment is designed to understand the Bode plot method that gives a graphical procedure for determining the stability of a control system based on sinusoidal frequency response. MATLAB Control system tool box contains two root locus design GUI, sisotool and rlttool. The main aim of this experiment is to use siso toolbox to understand features of the root locus design.

14.1 SISO Toolbox Tutorial

SISOTOOL opens a SISO Design GUI for interactive compensator design. This GUI allows you to design a single-input/single-output (SISO) compensator using root locus, Bode diagram. By default, the SISO Design Tool:

- Opens the Control and Estimation Tools Manager with a default SISO Design Task node.
- Opens the Graphical Tuning editor with root locus and open-loop Bode diagrams.
- Places the compensator, C, in the forward path in series with the plant, G.
- Assumes the prefilter, F, and the sensor, H, are unity gains. Once you specify G and H, they are fixed in the feedback structure.

The default control architecture is shown in the figure below.

![Figure 14.1](image-url)

Figure 14.1
Example 1:

In the Control System shown in the figure below, \( G(s) \) is a proportional controller, \( K \).

\[
G_p(s) = \frac{1}{s(s + 2)(s + 5)}
\]

![Figure 14.2](image)

Using sisotool determine the following:

a) Range of \( K \) for system stability

b) Value of \( K \) for the complex dominant poles damping ratio 0.6. For this value of \( K \) obtain the frequency response gain margin and phase margin. Also obtain the step response and time domain specifications.

To make the plant model and start the SISO Design Tool, at the MATLAB prompt type

\[
>> Gp = \text{tf}(1, \begin{bmatrix} 1 & 7 & 10 \end{bmatrix})
\]

\[
>> \text{sisotool}
\]

An empty SISO Design Tool opens as shown.

![Figure 14.3](image)
The SISO Design Tool by default assumes that the compensator is in the forward path. Select Import Model under the File menu. This opens the Import System Data dialog box, which is shown below. All the available models will appear in the Model Listbox: The Blocks are coded as F (A preamplifier), G (Plant model) H (Sensor), K (Compensator) with default values of 1. Select Gp and click on the arrow button to place it in the G field.

The other button toggles between the two configurations. Click Ok. The root locus and open-loop Bode diagrams are displayed. The red squares on the loci represent the closed loop poles corresponding to gain set point. Also the gain margin and phase margin are shown on the bode plot.
From Tools Menu select Loop responses/Close-Loop Step to obtain the step response. Right-click on the plot, and select Characteristics and then Rise Time and Peak Overshoot. Left click on the blue dots, this will display the time-domain specifications shown in the Figure below:

**Figure 14.5**
Example 2:

For the unity feedback system

\[ G(s) = \frac{1}{s(s+2)(s+5)}, \]

Design a phase lead compensator for the following time domain specifications:

- Dominant poles damping ratio=0.707
- Dominant poles settling time=2sec

On the MATLAB prompt type

```
>> Gp=tf([1],[1 7 10 0])
>> sisotool
```

An empty SISO design tool opens. Select Import Model under the File menu. This opens the Import System Data dialog box. Select Gp and place it in the G field. Click OK. The root locus and open loop bode diagrams are displayed in the plot regions.
To place a pair of complex poles on your diagram at a damping ratio of 0.707, select **Design Constraints** and then New from the right-click menu in the root locus. This opens the Design Constraints editor. In the damping ratio field enter 0.707. This results in a pair of shaded rays at the desired slope. To add a settling time constraint, reopen the **New Constraint** window and choose settling time from the pull-down menu set the file to 2. Place the cursor next to the controller C(s) and click to open its dialog box. In the **Edit Compensator** C type -1.75 for a Real Zero. Also, add a real pole, turn on the pole × on the toolbar and place the pole ⬇️ on the real axis to the right of the pole with the largest magnitude. Move the closed-loop pole up the loci towards the intersection of ζ and τ constraints. You can turn on the Mouse zoom and place a square in the area containing the controller zero, pole and the closed loop pole. Adjust the pole position on the real axis and at the same time hold the left mouse and drag the closed-loop pole until the pole is placed at the desired location as appears on the bottom panel.

![Figure 14.7](image)

To save the compensator design, select **Export** from the **File menu**. This opens the **SISO Tool Export**. Select **Compensator C** in the **Component** column. To change the export name, double-
click in the cell for Compensator C and change the name to say PhLead. Click on the Export to Workspace button.

![Step Response Graph](image)

**Figure 14.8**

Exercise: A unity feedback system has open loop transfer function

\[ G(s) = \frac{4}{s(s + 2)} \]

It is desired that dominant closed loop poles provide damping ratio=0.5 and have an un-damped natural frequency=4rad/sec. Velocity error constant is required to be greater than 4.

a) Verify that only gain adjustment cannot meet these objectives.

b) Design a lead compensator using SISO design tool to meet the objectives.

c) Using GUI, determine the peak overshoot and settling time of the lead-compensated system.

<table>
<thead>
<tr>
<th>Peak Overshoot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Settling Time</th>
</tr>
</thead>
</table>
POST LAB

Use the following plant

\[ P(s) = \frac{8.96}{0.00147s^2 + 0.01455s + 1} \]

Simulate its step response. Using a PD or PI controller, try to get the settling time to less than 0.5 sec (approx. 0.1 sec) while keeping the %OS lesser than 10%. Justify your design in the report.
Appendix A: Lab Evaluation Criteria

Labs with projects
1. Experiments and their report 50%
   a. Experiment 60%
   b. Lab report 40%
2. Quizzes (3-4) 15%
3. Final evaluation 35%
   a. Project Implementation 60%
   b. Project report and quiz 40%

Labs without projects
1. Experiments and their report 50%
   a. Experiment 60%
   b. Lab report 40%
2. Quizzes (3-4) 20%
3. Final Evaluation 30%
   i. Experiment 60%
   ii. Lab report, pre and post experiment quiz 40%

Notice:
Copying and plagiarism of lab reports is a serious academic misconduct. First instance of copying may entail ZERO in that experiment. Second instance of copying may be reported to DC. This may result in awarding FAIL in the lab course.
Appendix B: Safety around Electricity

In all the Electrical Engineering (EE) labs, with an aim to prevent any unforeseen accidents during conduct of lab experiments, following preventive measures and safe practices shall be adopted:

- Remember that the voltage of the electricity and the available electrical current in EE labs has enough power to cause death/injury by electrocution. It is around 50V/10 mA that the “cannot let go” level is reached. “The key to survival is to decrease our exposure to energized circuits.”
- If a person touches an energized bare wire or faulty equipment while grounded, electricity will instantly pass through the body to the ground, causing a harmful, potentially fatal, shock.
- Each circuit must be protected by a fuse or circuit breaker that will blow or “trip” when its safe carrying capacity is surpassed. If a fuse blows or circuit breaker trips repeatedly while in normal use (not overloaded), check for shorts and other faults in the line or devices. Do not resume use until the trouble is fixed.
- It is hazardous to overload electrical circuits by using extension cords and multi-plug outlets. Use extension cords only when necessary and make sure they are heavy enough for the job. Avoid creating an “octopus” by inserting several plugs into a multi-plug outlet connected to a single wall outlet. Extension cords should ONLY be used on a temporary basis in situations where fixed wiring is not feasible.
- Dimmed lights, reduced output from heaters and poor monitor pictures are all symptoms of an overloaded circuit. Keep the total load at any one time safely below maximum capacity.
- If wires are exposed, they may cause a shock to a person who comes into contact with them. Cords should not be hung on nails, run over or wrapped around objects, knotted or twisted. This may break the wire or insulation. Short circuits are usually caused by bare wires touching due to breakdown of insulation. Electrical tape or any other kind of tape is not adequate for insulation!
- Electrical cords should be examined visually before use for external defects such as: Fraying (worn out) and exposed wiring, loose parts, deformed or missing parts, damage to outer jacket or insulation, evidence of internal damage such as pinched or crushed outer jacket. If any defects are found the electric cords should be removed from service immediately.
- Pull the plug not the cord. Pulling the cord could break a wire, causing a short circuit.
- Plug your heavy current consuming or any other large appliances into an outlet that is not shared with other appliances. Do not tamper with fuses as this is a potential fire hazard. Do not overload circuits as this may cause the wires to heat and ignite insulation or other combustibles.
- Keep lab equipment properly cleaned and maintained.
- Ensure lamps are free from contact with flammable material. Always use lights bulbs with the recommended wattage for your lamp and equipment.
• Be aware of the odor of burning plastic or wire.
• ALWAYS follow the manufacturer recommendations when using or installing new lab equipment. Wiring installations should always be made by a licensed electrician or other qualified person. All electrical lab equipment should have the label of a testing laboratory.
• Be aware of missing ground prong and outlet cover, pinched wires, damaged casings on electrical outlets.
• Inform Lab engineer / Lab assistant of any failure of safety preventive measures and safe practices as soon you notice it. Be alert and proceed with caution at all times in the laboratory.
• Conduct yourself in a responsible manner at all times in the EE Labs.
• Follow all written and verbal instructions carefully. If you do not understand a direction or part of a procedure, ASK YOUR LAB ENGINEER / LAB ASSISTANT BEFORE PROCEEDING WITH THE ACTIVITY.
• Never work alone in the laboratory. No student may work in EE Labs without the presence of the Lab engineer / Lab assistant.
• Perform only those experiments authorized by your teacher. Carefully follow all instructions, both written and oral. Unauthorized experiments are not allowed.
• Be prepared for your work in the EE Labs. Read all procedures thoroughly before entering the laboratory. Never fool around in the laboratory. Horseplay, practical jokes, and pranks are dangerous and prohibited.
• Always work in a well-ventilated area.
• Observe good housekeeping practices. Work areas should be kept clean and tidy at all times.
• Experiments must be personally monitored at all times. Do not wander around the room, distract other students, startle other students or interfere with the laboratory experiments of others.
• Dress properly during a laboratory activity. Long hair, dangling jewelry, and loose or baggy clothing are a hazard in the laboratory. Long hair must be tied back, and dangling jewelry and baggy clothing must be secured. Shoes must completely cover the foot.
• Know the locations and operating procedures of all safety equipment including fire extinguisher. Know what to do if there is a fire during a lab period; “Turn off equipment, if possible and exit EE lab immediately.”
Appendix C: Guidelines on Preparing Lab Reports

A format has been developed for writing the lab reports. The format for lab report will include:

1. **Introduction**: Introduce area explored in the experiment.
2. **Objective**: What are the learning goals of the experiment?
3. **Measurements**: In your own words write how the experiment is performed (Do not copy/paste the procedure).
   a. **Issues**: Which technical issues were faced during the performance of the experiment and how they were resolved?
   b. **Graphs**, if any
4. **Conclusions**: What conclusions can be drawn from the measurements?
5. **Applications**: Suggest a real world application where this experiment may apply.
6. Answers to post lab questions (if any).

**Sample Lab Report:**

**Introduction:**

The DC Motor control system consists of a DC motor with an encoder and an inertia wheel on the motor shaft. The motor is driven using a pulse width modulation (PWM) power amplifier. The speed of the DC motor is controlled using proportional integral control system. The PI control also includes set point weight.

**Objective:**

The main objective of this experiment is to get full understanding of PID controllers and to give the knowledge to simulate and validate the controllers.

**Experimental Procedure:**

The DCMCT Modeling VU, runs the DC motor in open-loop and plots the corresponding speed and input voltage responses. The transfer function representing the DC-motor speed voltage relation is used to design the PI controller. The table below describes the main elements of the QNET_DCMCT Speed virtual instrument front panel:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed $w_m$</td>
<td>Motor output speed numeric display rad/sec</td>
</tr>
<tr>
<td>Current $I_m$</td>
<td>Motor armature current numeric display A</td>
</tr>
<tr>
<td>Voltage $V_m$</td>
<td>Motor input voltage numeric display</td>
</tr>
<tr>
<td>Disturbance $V_{sd}$</td>
<td>Apply simulated disturbance voltage</td>
</tr>
<tr>
<td>Kp $K_p$</td>
<td>Controller proportional gain input box</td>
</tr>
</tbody>
</table>
The following parameters are set in QNET_DCMCT_Speed_Control.vi. before running the QNET_DCMCT_Speed_Control vi. simulation.

In the Signal Generator section set:
- Signal type = ‘square wave’
- Amplitude = 25.0 rad/s
- Frequency = 0.40 Hz
- Offset = 100.0 rad/s

In the Control Parameters section set:
- $k_p = 0.050 \text{ V.s/rad}$
- $k_i = 1.00 \text{ V/rad}$
- $bsp = 0.00$

### Measurements and Results:

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
</table>

### Graphical Analysis

The behavior of the measured speed with respect to the reference speed is examined in the Speed (rad/s) scope and a certain overshoot is observed on voltage change.

### Conclusion

The value of $k_p$ parameter affects the speed of motor.