Department of Electronic Engineering

NED University of Engineering & Technology

PRACTICAL WORKBOOK

For the course of

Instrumentation & Control (EL-305)

For T.E (ME & PE)

Instructor's Name:	
Student's Name:	
Roll No.:	Batch:
Semester:	Year:
Department:	

LABORATORY WORKBOOK FOR THE COURSE

Instrumentation & Control (EL-305)

Prepared By:

Miss Sidra Rahman (Lecturer)

Approved By:
The Board of Studies
Department of Electronic Engineering

CONTENTS

S. No	Date	Cognitive / Psychomotor Level	CLO	List of Experiments	Page No.	Signature
1.		P3	4	*Introduction* of MATLAB briefly including tutorial of polynomials, script writing and programming aspect of MATLAB from control systems viewpoint.	4	
2.		P3	4	*Implement* the designing and analysis of mathematical models of physical control systems	8	
3.		P3	4	*Build* the mathematical modeling of Multiple-ElementMechanical Translation System	13	
4.		P3	4	*Practice* mathematical modeling of Electrical System	17	
5.		P3	4	*Implement* the performance of First order and Second order systems and development of Time response specification’s function	21	
6.		P3	4	*Try* the process to generate the response of Control System to Ramp and Arbitrary Inputs	26	
7.		P3	4	Try to learn commands in MATLAB that would be used to reduce linear systems block diagram using series, parallel and feedback configuration.	29	
8.		P3	4	*Practice* to analyze the System Stability using MATLAB	34	
9.		P3	4	*Trace* the characteristics of the each of proportional (P), the integral (I), and the derivative (D) controls and obtain a desired response by using them.	37	
10.		P3	4	*Build* the designing of P, PD, PI, and PID controllers to meet closed-loop performance specifications including transientperformance and steady error.	42	
11.		P3	4	*Derive* the relationship between frequency response and step response characteristics using MATLAB	45	
12.		P3	4	*Do* the plotting of Root Loci with MATLAB	49	
13.		P3	4	Design control system for ball and beam system for the stated requirements. (Open ended lab)	54	

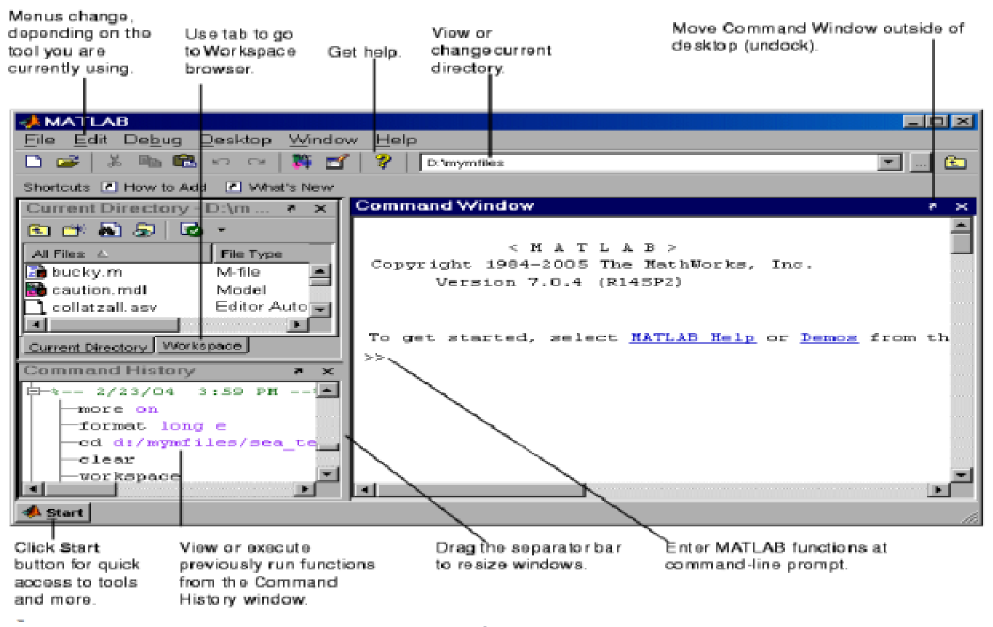
Lab Experiment # 01

Objective:
Introduction of MATLAB briefly including tutorial of polynomials, script writing and programming aspect of MATLAB from control systems viewpoint.

Equipment:
PCs with installed MATLAB

Theory:
The name MATLAB stands for MATrix LABoratory. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects.
MATLAB is a high-performance language for technical computing. It integrates Computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. These factors make MATLAB an excellent tool for teaching and research.
It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

Starting MATLAB:
After logging into account, double-click on the MATLAB shortcut icon (MATLAB) on Windows desktop. A special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible from the desktop are: The Command Window the Command History the Workspace the Current Directory the Help Browser the Start button


Menus change, depending on the tool you are currently using.
Use tab to go to Workspace browser.
Get help.
View or change current directory.
Move Command Window outside of desktop (undock).
MATLAB
File Edit Debug Desktop Window Help
D:\mymfiles
Shortcuts How to Add What's New
Current Directory - D:\m ...
All Files
File Type
bucky.m
M-file
caution.mdl
Model
collatzall.asv
Editor Auto
Current Directory
Workspace
Command History
Command Window
Start
Click Start button for quick access to tools and more.
View or execute previously run functions from the Command History window.
Drag the separator bar to resize windows.
Enter MATLAB functions at command-line prompt.

```
%-- 2/23/04   3:59 PM --%
more on
format long e
cd d:/mymfiles/sea_te
clear
workspace
```

```
              < M A T L A B >
  Copyright 1984-2005 The MathWorks, Inc.
         Version 7.0.4 (R14SP2)


 To get started, select MATLAB Help or Demos from th
>>
```

Figure 1. 1 the desktop window of MATLAB

A Matrix array is two-dimensional, having both multiple rows and multiple columns,like vector arrays:
It begins with [, and end with] spaces or commas are used to separate elements in a row.semicolon or enter is used to separate rows.
Example:

```
>> f = [ 1 2 3; 4 5 6]
        f =
                1    2    3
                4    5    6
```

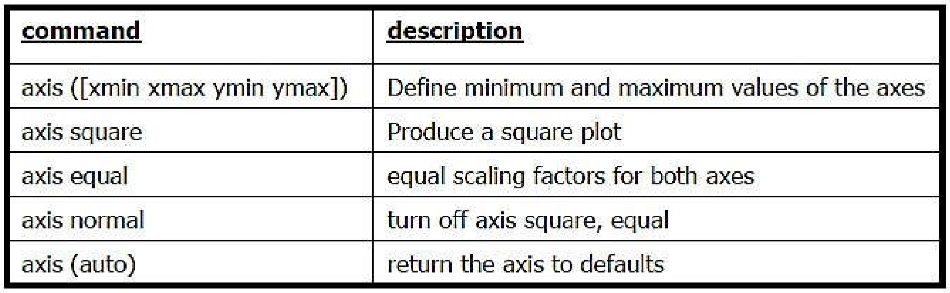
A system of 3 linear equations with 3 unknowns (x1, x2, x3):

$$3x1 + 2x2 - x3 = 10$$
$$x1 + 3x2 + 2x3 = 5$$
$$x1 - x2 - x3 = -1$$

$$A = \begin{bmatrix} 3 & 2 & 1 \\ -1 & 3 & 2 \\ 1 & -1 & -1 \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad b = \begin{bmatrix} 10 \\ 5 \\ -1 \end{bmatrix}$$

Some useful commands:

Table 1. 1 Some Useful Commands



command	description
axis ([xmin xmax ymin ymax])	Define minimum and maximum values of the axes
axis square	Produce a square plot
axis equal	equal scaling factors for both axes
axis normal	turn off axis square, equal
axis (auto)	return the axis to defaults

Plotting Curves:

plot (x,y)– generates a linear plot of the values of x (horizontal axis) and y (verticalaxis).
semilogx (x,y) – generate a plot of the values of x and y using a logarithmic scale forx and a linear scale for y
semilogy (x,y) – generate a plot of the values of x and y using a linear scale for x anda logarithmic scale for y.
loglog(x,y) – generate a plot of the values of x and y using logarithmic scales for both x and y

Adding new curves to the existing graph:
Use the hold command to add lines/points to an existing plot.
hold on – retain existing axes, add new curves to current axes. Axes are rescaledwhen necessary.
hold off – release the current figure window for new plot

Grids & Labels:

Table 1. 2 Grids & Labels

Command	Description
grid on	Adds dashes grid lines at the tick marks
grid off	Remove grid lines (default)
grid	Toggle grid status (off to on, or off to on)
title('text')	Labels top of plot with text in quotes
xlabel('text')	Labels horizontal (x) axis with text is quotes
ylabel('text')	Labels vertical (x) axis with text is quotes
text(x, y, 'text')	Adds text in quotes to location (x, y) on the current axes, where (x, y) is in units from the current plot.

Polynomial Evaluation:

Table 1. 3 Polynomial Functions

Function	Description
Conv	Multiply polynomials
Deconv	Divide polynomials
Poly	Polynomial with specified roots
Polyder	Polynomial derivative
Polyfit	Polynomial curve fitting
Polyval	Polynomial evaluation
Polyvalm	Matrix polynomial evaluation
Residue	Partial-fraction expansion (residues)
Roots	Find polynomial roots

Scripts:

Scripts do not accept input arguments or return output arguments. They operate on data in the workspace. MATLAB provides a full programming language that enables you to write a series of MATLAB statements into a file and then execute them with a single command. You write your program in an ordinary text file, giving the file a name of 'filename.m'. The term you use for 'filename' becomes the new command that MATLAB associates with the program. The file extension of .m makes this a MATLAB M-file.

Functions:

which can accept input arguments and return output arguments. Internal variables are local to the function. If you're a new MATLAB programmer, just create the M-files that you want to try out in the current directory. As you develop more of your own M-files, you will want to organize them into other directories and personal toolboxes that you can add to your MATLAB search path. If you duplicate function names, MATLAB executes the one that occurs first in the search path.

Task:

1. Consider the two polynomials $p(s) = s^2 + 2s + 1$ & $q(s) = s + 1$

   Use MATLAB to compute

   a. $p(s\ ) * q(s)$

   b. Roots of p(s) and q(s)

   c. *p(-1)* and *q(6)*

Use MATLAB command to find the partial fraction:

$$\frac{B(s)}{A(s)} = \frac{2s^3 + 5s^2 + 3s + 6}{s^3 + 6s^2 + 11s + 6}$$

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No.______________ Date: ______________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 02

Objective:
Implement the designing and analysis of mathematical models of physical control systems

Equipment:
PCs with installed MATLAB

Theory:

Mass-Spring System Model

Consider the following Mass-Spring system shown in the figure. Where $F_s(x)$ is the spring force $F_f()$ is the friction coefficient, x(t) is the displacement and $F_a(t)$ is the applied force:

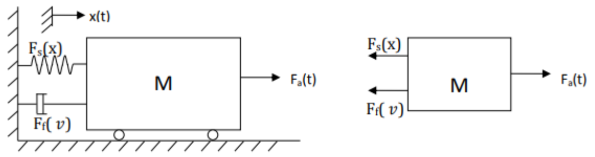


Figure 2. 1 Mass Spring System Model

$a = \frac{dv(t)}{dx(t)} = dx(t)$ is the accelerationdx(t) is the displacement
According to the laws of physics
$\mathrm{Ma} + F_f(\mathrm{v}) + F_s(\mathrm{x}) = F_a(\mathrm{t})$
The differential equation for the above Mass-Spring system can then be written asfollows

$$M\left(\frac{dx^2}{dt^2}\right) + B\left(\frac{dx(t)}{dt}\right) + Kx(t) = F_a(t)$$

B is called the friction coefficient and K is called the spring constant.

- *B* is called the friction coefficient and
- *K* is called the spring constant.

The linear differential equation of second order describes the relationship between the displacement and the applied force. The differential equation can then be used to study the time behavior of x(t) under various changes of the applied force. The spring force and/or the friction force can have a more complicated expression or could be represented by a graph or data table

Solving the differential equation using MATLAB:
The objectives behind modeling the mass-damper system can be many and may include

- Understanding the dynamics of such system
- Studying the effect of each parameter on the system such as mass *M*, thefriction coefficient B, and the elastic characteristic $F_s(x)$.
- Designing a new component such as damper or spring.
- Reproducing a problem in order to suggest a solution.

MATLAB can help solve linear or nonlinear ordinary differential equations (ODE). To show how you can solve ODE using MATLAB we will proceed in two steps. We first see how we can solve first order and second

Procedure:

Speed Cruise Control example:
When $Fs(x) = 0$ which means that $K = 0$, Equation (1) becomes

$$M\left(\frac{dx^2}{dt^2}\right) + B\left(\frac{dx(t)}{dt}\right) = F_a(t)$$

or

$$M\left(\frac{dv(t)}{dt}\right) + Bv = F_a(t)$$

Using MATLAB solver, we can write the following:

<u>create a MATLAB-function cruise_speed.m</u>

```
function dvdt=cruise_speed(t, v)                %flow rate M=750; %(Kg)
B=30;                        %( Nsec/m)
Fa=300;                          %N
% dv/dt=Fa/M-B/Mv dvdt =Fa/M-B/M*v;
```

<u>create a new MATLAB m-file and write</u>

```
v0= 0; %(initial speed)
[t,v]=ode45('cruise_speed', [0 200],v0);
plot(t,v);
grid on;
title('cruise speed time response to a constant traction force Fa(t) ')
```

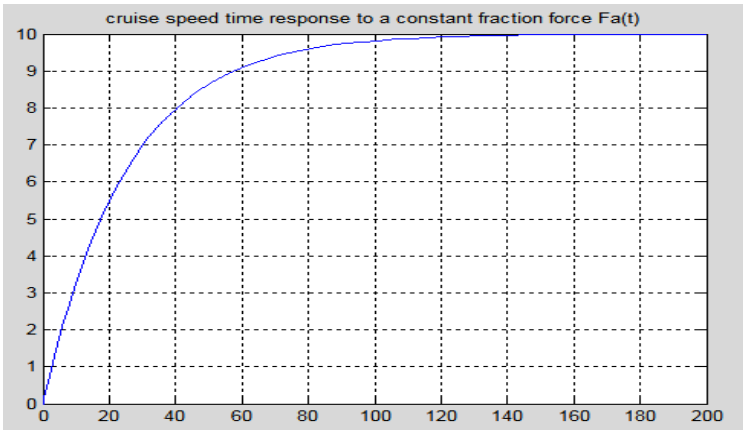


Figure 2. 2 behavior of a car speed

In the above program the behavior of a car speed is shown in which the car starts with rest position, after that it attains its maximum speed so that it reaches its maximum limit then after that its speed becomes constant throughout the time.

Mass-Spring System Example:

$$M\left(\frac{d^2x(t)}{dt^2}\right) + B\left(\frac{dx(t)}{dt}\right) + Kx^r(t) = F_a(t)$$

Table 2. 1 System equations

Variables	New Variables	Differential Equation
x(t)	X_1	$\frac{dX_1}{dt} = X_2$

dx(t)/dt	X_2	$\frac{dX_2}{dt} = -\frac{B}{M}X_2 - \frac{K}{M}X_1^r(t) + \frac{Fa(t)}{M}$

In vector form,

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}, \frac{dx}{dt} = \begin{bmatrix} \frac{dX_1}{dt} \\ \frac{dX_2}{dt} \end{bmatrix}$$

The system equations can be written as:

$$\frac{dX}{dt} = -\frac{B}{M}X^2 - \frac{K}{M}X_1^r(t) + \frac{Fa(t)}{M}$$

create a MATLAB-function mass_spring.mfunction

$$\frac{dX}{dt} = mass\ spring\ (t, X)$$

```
M=705;                                  % (Kg)
B=30;                                   % (Nsec/m)
Fa=300;                                 % (N)
K=15;                                   % (N/m)
r=1;                                    % dX/dt
dXdt(1,1) =X(2);
dXdt(2,1) =-B/M*X(2)-K/M*X(1)^r+Fa/M;
program of mass spring system with r=1
clear all
close all
clc
X0= [0;0];% (Initial speed and position)
[t,X]=ode45('mass_spring',[0 200],X0);figure;
plot(t,X(:,1));
xlabel('Time(t)');
ylabel('Position');
title ('Mass spring system');legend('Position ');
grid;
figure;
plot(t,X(:,2),'r'); xlabel('Time(t)');
label('Speed');
title ('Mass spring system');
legend ('Speed ');
grid;
```

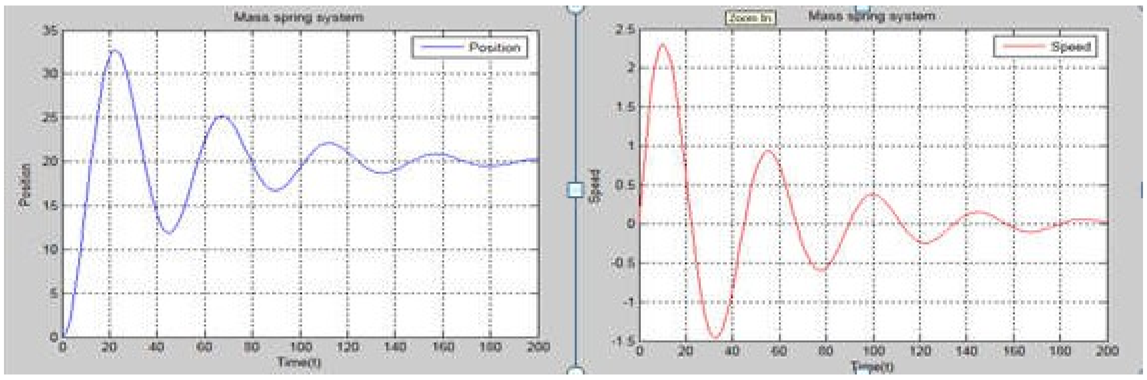


Figure 2. 3 Behavior of Mass Spring System

Observations:

Table 2. 2 Behavior of Mass Spring System Function

Parameter	Behavior of system
Mass	
Friction Coefficient	
Stiffness	
Applied Force	

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No._______________ Date: ________________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 03

Objective:
Build the mathematical modeling of Multiple-ElementMechanical Translation System

Equipment:
PCs with installed MATLAB

Theory:

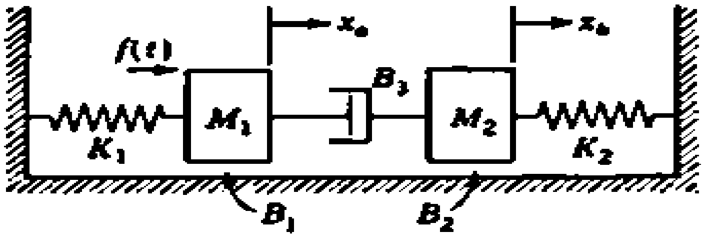


Figure 3. 1 Multiple Element Mechanical System

where,

- f(t) is applied force to the mass M1.
- B1 and B2 are the viscous friction coefficients indicating the slidingfriction between the masses M1 and M2 and the surface

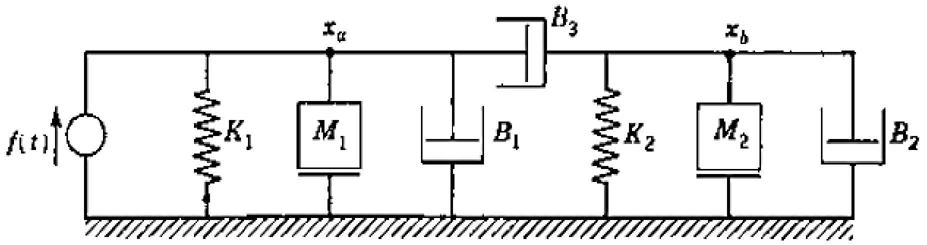


Figure 3. 2 Multiple Element Mechanical System

According to the rules for node equations:
For node a:

$$(M_1 D^2 + B_1 D + B_3 D + K_1)x_a - (B_3 D)x_b = f$$

For node b:

$$(B_3 D)x_a + (M_2 D^2 + B_2 D + B_3 D + K_2)x_b = 0$$

$X_1 = Xb$ for spring K2 $\qquad X_2 = X`_1 = Vb$
$X_3 = Xa$ for spring K1 $\qquad X_4 = X`_3 = Va$

The system equations are:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{K_2}{M_2} & \frac{B_2 + B_3}{M_2} & 0 & \frac{B_3}{M_2} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{B_3}{M_1} & -\frac{K_1}{M_1} & \frac{B_1 + B_3}{M_1} \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{M_1} \end{bmatrix}$$

Procedure:

1. create a MATLAB-function multiple_element_sys.m

```
function dXdt=multiple_element_sys (t,X)
        Fa=300;                 %(N)
        M1=750;                 %(Kg)
        M2=750;                 %(Kg)
        B1=20;                  %(Nsec/m)
        B2=20;                  %(Nsec/m)
        B3=30;                  %(Nsec/m)
        K1=15;                  %(N/m)
K2=15;
%(N/m)dXdt(1,1)=X(2);
dXdt(2,1)=-K2/M2*X(1)-((B1+B2)/M2)*X(2)+B3*X(4)/M2;dXdt(3,1)=X(4);
dXdt(4,1)=B3/M1*X(2)-K1/M1*X(3)-((B1+B3)/M1)*X(4)+Fa/M1;
```

Write another M. file to call the function:

```
clear all;
close all;
clc;
X0= [0;0;0;0]; % (Initial xb, Vb, xa, Va)
[t,X]=ode45('multiple_element_sys',[0 200],X0);figure;
Subplot (2,1,1);
plot(t,X(:,1));
plot(t,X(:,2),'r');
xlabel('Time(t)');
ylabel ('Position xb / Speed Vb');
title ('Mass spring system');
legend ('xb', 'Vb'); grid;
subplot (2,1,2);
plot(t,X(:,3)); hold;
plot(t,X(:,4),'r');
xlabel('Time(t)');
ylabel('Position xa / Speed Va');
title ('Mass spring system');
legend ('xa', 'Va');
grid;
```

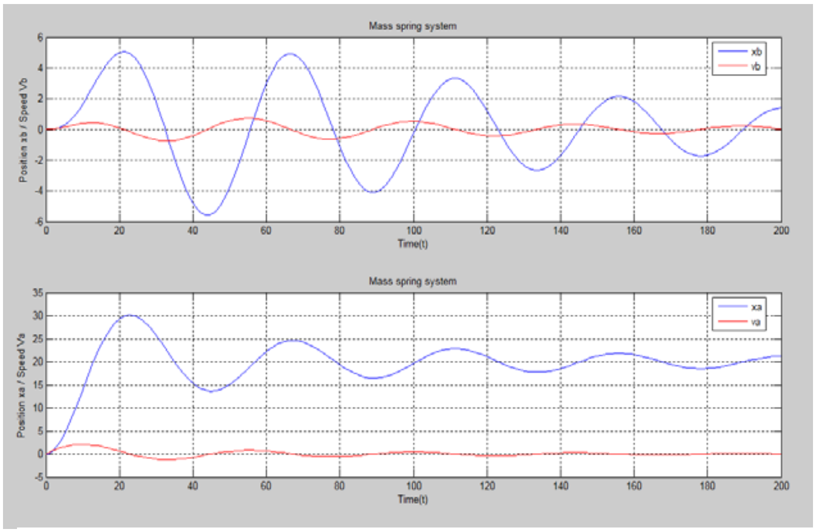


Figure 3. 3 Behavior of Multiple Element Mechanical System

Observations:

Table 3. 1 Behavior of Multiple Element Mechanical System

Parameter		Behavior of system
Mass	M1	
	M2	
Friction Coefficient	B1	
	B2	
	B3	
Stiffness	K1	
	K2	
Applied Force	Fa	

Task:

1. There is a mechanical rotational system consist of one flywheel J_1 is attached by a flexible shaft K_r to ground and has an applied torque τ_a. a second flywheel J_2 is driven by friction between the two flywheels B_{r1}. The second flywheel also has friction to the ground B_{r2}.

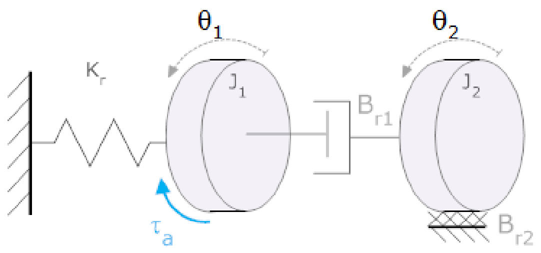


Figure 3. 4 Multiple Element Mechanical Rotational System

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No.______________ Date: ______________________

Software Use Rubric

Criterion	Level of Attainment				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 04

Objective:
Practice mathematical modeling of Electrical System

Equipment:
PCs with installed MATLAB

Theory:

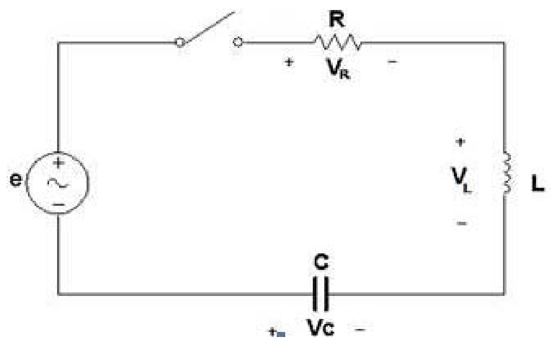


Figure 4. 1 Electrical System

e is applied potential.
i is the mesh current.
The differential equations for the given figure.
According to Mesh Analysis:

$$e(t) = V_L + V_C + V_R$$

$$e(t) = LD_i + \frac{1}{CD}i + iR$$

The state equations for the given figure.
This circuit contains two energy-storage elements, inductor, and capacitor
Let state variables are
$X_1 = V_C$ the voltage across the capacitor and $X2 = I$ the current in the inductor.
State Equation:

$$\begin{bmatrix} x'1 \\ x'2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{C} \\ -\frac{1}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} x1 \\ x2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} [u]$$

Procedure:

1. create a MATLAB-function RLC.m

```
function dXdt=RLC(t,X)
e=60;          % (V)
R=10;          % (Ohm)
L=1;           % (H)
C=10;          % (F)
%dX/dt dXdt(1,1) =(1/C)*X(2)
dXdt(2,1) =(-1/L)*X(1)-(R/L)*X(2)+(1/L)*e;
```

Write an other M. file to call the function:

```
clear all;
close all;
clc;
X0= [0 0];
[t,X]=ode45('RLC',[0 500],X0);
Subplot (2,1,1;plot(t,X(:,1));
legend('Vc'); gridon;
title('Vc');
subplot (2,1,2); plot(t,X(:,2),'r');
legend('i');
grid on;title('i');
```

<u>Graph:</u>

Time constant = RC = 10*10= 100 sec

For first time constant :

Vc=63.2% * e = 0.632*60 =37.92 V

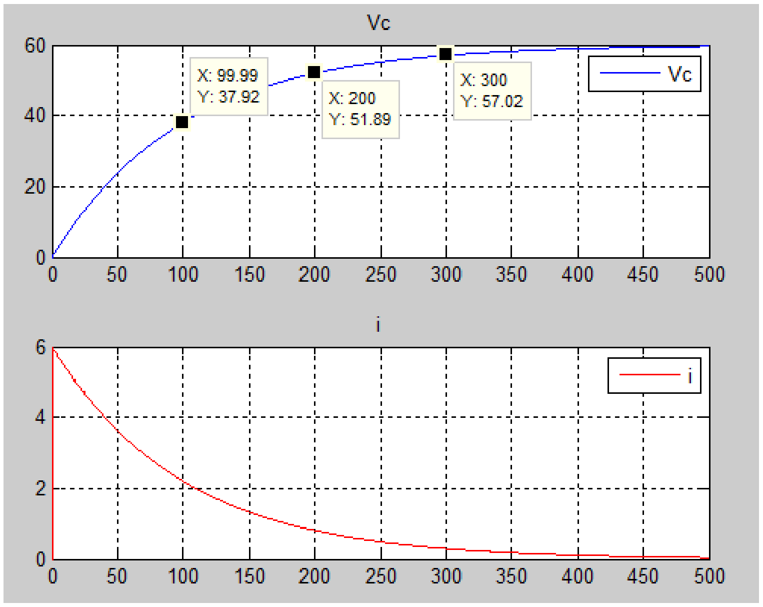


Figure 4. 2 RC constant curve

Observations:

Table 4. 1 Behavior of Electrical System

Parameter	Behavior of system
Voltage source(e)	
Resistance(R)	
Inductance(L)	
Capacitance(C)	

Task:
Write the function and program of the following circuit diagram. Also explain the plotsof the respective state variables.

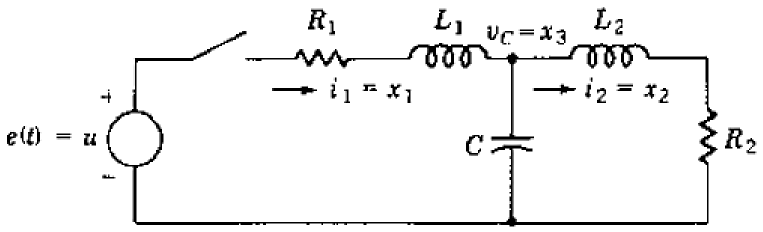


Figure 4. 3 Electrical System

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No.______________ Date: ______________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 05

Objective:
Implement the performance of First order and Second order systems and development of Time response specification's function

Equipment:
PCs with installed MATLAB

Theory:
An electrical RC-circuit is the simplest example of a first order system. It comprises of a resistor and capacitor connected in series to a voltage supply as shown below on Figure

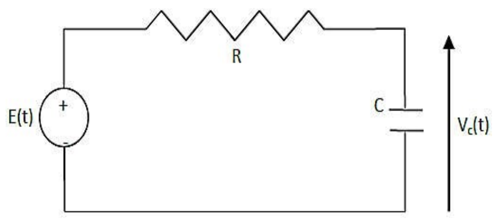


Figure 5. 1 First Order Electrical System

where,

- Vc(t) is the voltage across the capacitor,
- R is the resistance and
- C is the capacitance.

Obtain the transfer function of the above electrical circuit. (Take V_c as output and $Vc(0) = Vo$)
For the RC-circuit as shown in Fig. 1, the equation governing its behavior is given by :

$$\frac{dV_C(t)}{dt} + \frac{1}{RC}V_C(t) = \frac{1}{RC}\ E\ where\ V_C(0) = \ V_o$$

The constant $\tau = RC$ is the time constant of the system and is defined as the time required by the system output i.e. Vc(t) to rise to 63% of its final value (which is E).Hence the above equation

$$\tau\frac{dv_C(t)}{dt} + \ V_C(t) = E$$

Transfer Function:
Obtaining the transfer function of the above differential equation, we get

$$\frac{V_C\ (s)}{E(s)} = \frac{1}{\tau s + 1}$$

The above system is known as the first order system.
The performance measures of a first order system are its time constant and its steadystate.

Second Order System:
Consider the following Mass spring system shown

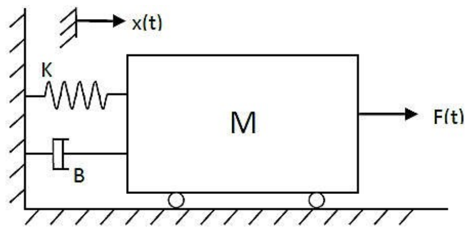


Figure 5. 2 Second Order Mechanical System

where,

- *K* is the spring constant,
- *B* is the friction coefficient,
- *x(t)* is the displacement and
- *F(t)* is the applied force:

The differential equation for the above Mass-Spring system can be derived as follows:

$$M\left(\frac{dx^2}{dt^2}\right)+B\left(\frac{dx(t)}{dt}\right)+Kx(t)=F(t)$$

Transfer Function

Applying the Laplace transformation, we get

$$(Ms^2+Bs+K)*X(s)=F(s)$$

Provided that, all the initial conditions are zero. Then the transfer function representationof the system is given by:

$$TF=\frac{Output}{Input}=\frac{F(s)}{X(s)}=\frac{1}{(Ms^2+Bs+K)}$$

The above system is known as a second order system.

The generalized notation for a second order system described above can be written as:

$$Y(s)=\frac{\omega_{n^2}}{s^2+2\zeta\omega_n+\ \omega_{n^2}}\ R(s)$$

With the step input applied to the system, we obtain

$$Y(s)=\frac{\omega_{n^2}}{s(s^2+2\zeta\omega_n s+\ \omega_{n^2}\ )}$$

For which the transient output, as obtained from the Laplace transform table

$$y(t)=1-\frac{1}{\sqrt{1-\ \zeta^2}}\ e^{-\zeta\omega_n t}\ \sin(\omega_n\sqrt{1-\ \zeta^2}t+\ \cos^{-1}(\zeta))$$

- where $0 < \zeta < 1$.
- The transient response of the system changes for different values of dampingratio, ζ.

Standard performance measures for a second order feedback system are defined interms of step response of a system.

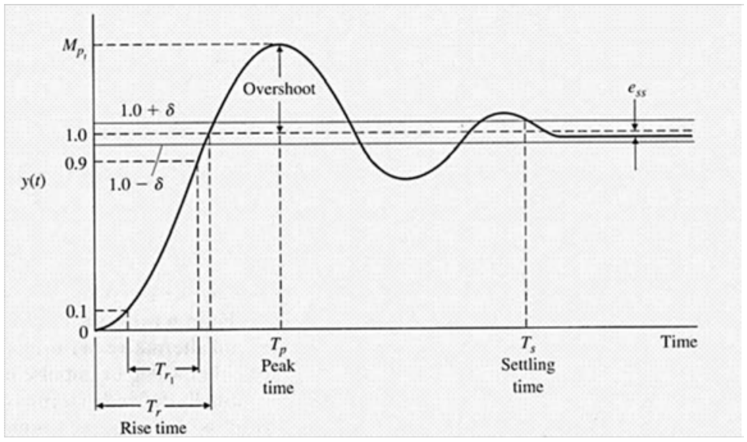


Figure 5. 3 Response of Mechanical System

The performance measures could be described as follows

- Rise Time 'T_r':

measures the time from 10% to 90% of the response to the step input.

- Peak Time 'T_p':

The time for a system to respond to a step input and rise to peak response.

Overshoot:

The amount by which the system output response proceeds beyond the desired response.
It is calculated as

$$P.O. = \frac{M_{pt} - fv}{fv} * 100\%$$

where MPt is the peak value of the time response, and *fv* is the final value of the response.

Settling Time 'T_s':

The time required for the system's output to settle within a certain percentage of the inputamplitude (which is usually taken as 2%). Then, settling time, Ts, is calculated as

$$T_s = \frac{4}{\zeta} \omega_n$$

Delay Time 'T_d':

It is the time required for the response to reach 50% of the final value the very first time.

Observations:

1. Effect of damping ratio 'ζ' on performance measures of the second order system. Find the step response of the system for values of ω_n= 1 and ζ = 0.1, 0.4, 0.7, 1.0 and
2. Use Matlab code, plot all the results in the same figure window and fill the following table

Table 5. 1 Result of First Order Mechanical System

ζ	Rise time	Peak Time	% Overshoot	Settling time	Steady state value
0.1					
0.4					
0.7					
1.0					
2.0					

Task:

1. Given the values of .3re 5R and C, obtain the unit step response of the first order system.
   a) R=2KΩ and C=0.01F
   b) R=2.5KΩ and C=0.003F

   Verify in each case that the calculated time constant ($\tau = RC$) and the one measured from thefigure as 63% of the final value are same. Obtain the steady state value of the system.

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No.______________ Date: ______________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 06

Objective:

Try the process to generate the response of Control System to Ramp and Arbitrary Inputs

Equipment:

PCs with installed MATLAB

Procedure:

The ramp response of the following transfer function

$$\frac{30}{s^2 + 5s + 6}$$

There is no ramp command in MATLAB. However, ramp signal is one order higher thanstep signal. The step input signal can be used to obtain the ramp response by dividing thetransfer function by s and then evaluating it using the step command. The following program can be used

```
close all;clear all;clc;
n= [0 0 30];
d= [1 5 6];
% The ramp response can be obtained by using step command for transfer
% function divided by s.The transfer function G1(s)=G(s)/s.n1=[0 0 0 30];
d1= [1 5 6 0];
[y,x,t]=step(n1,d1);
% To plot output y vs time t and t vs t i.e ramp signal on same graph window.
v=[0 10 0 10];
plot(t,y);
axis(v);
hold on;
plot(t,t);
grid;
title ('Plot of unit ramp response of G(s)=[30]/[s^2+5s+6]');
xlabel('Time');
ylabel ('Amplitude';
```

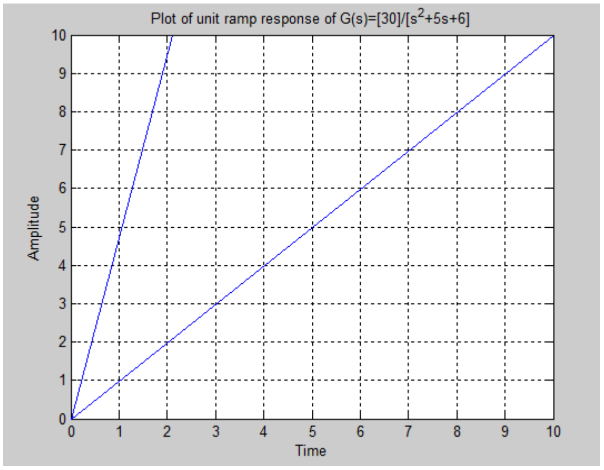


Figure 6. 1 Plot of unit ramp response

A closed-loop control system has a transfer function

$$\frac{s+5}{s^3+\ 2s^2+3s+5}$$

Obtain the response of the system for an input r (t) =e^-0.2t, for t=0 to 9 sec, insteps of 0.15 sec
In case the input signal is not a standard signal, MATLAB command lsim can be used toobtain the response of the system.

The syntax of the command islsim(n,d,u,t)

Transfer Function:

u is the arbitrary input signal and t defines time for which response of the system isrequired.
Another form of this command, which gives the output **y** in vector form without responseplot, is;

```
[y,t]=lsim(n,d,u,t)
```

The following program can be used:

```
n=[1 5];
d=[1 2 3 5];
t=0:0.15:9;
r=exp(-0.2*t);
y=lsim(n,d,r,t);
plot(t,r,'-',t,y,'o');grid;
title('plot of the sysytem for arbitrary input r(t)=e-0.2t');
xlabel('Time');
ylabel('Amplitude')
```

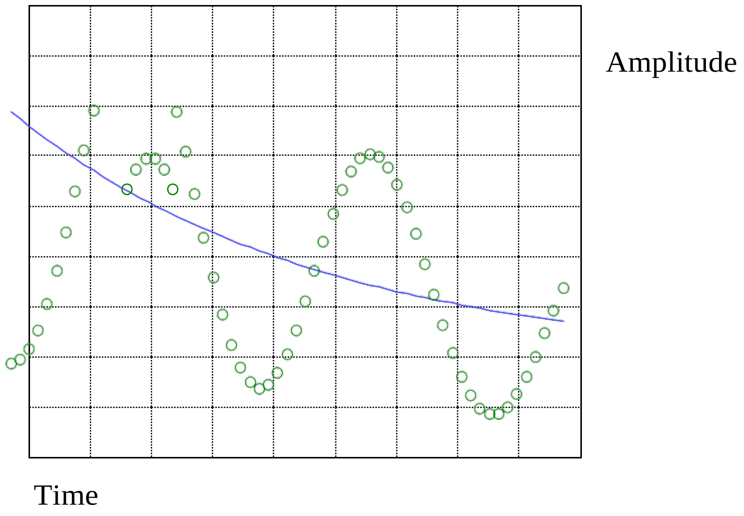


Figure 6. 2 Plot of the system for Arbitrary input

Task:

1. Obtain the ramp response of the following transfer function.

$$\frac{30}{s^2+5s+6}$$

Obtain the response of the system for an input $r(t) =\ \sin t +\ e^{-0.2t}\ for\ t = 0\ to$ 15 sec,in steps of 0.001 sec and comment on the result.

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No._______________ Date: ________________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 07

Objective:
Try to learn commands in MATLAB that would be used to reduce linear systems block diagram using series, parallel and feedback configuration.

Equipment:
PCs with installed MATLAB

Theory:

Series Configuration

If the two blocks are connected as shown below then the blocks are said to bein series

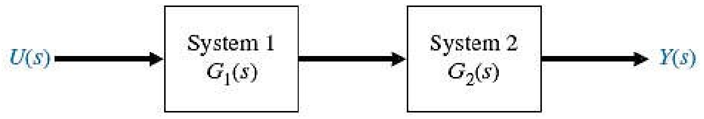
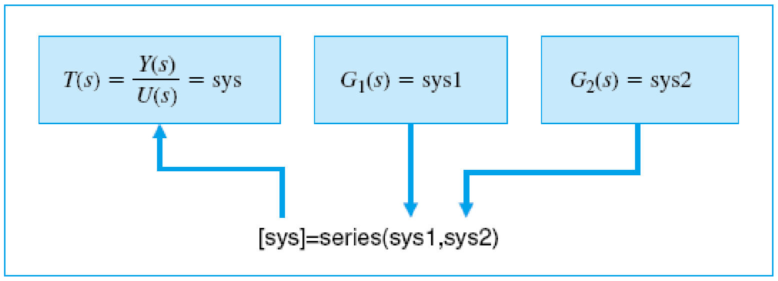


Figure 7. 1 Linear Systems connected in series

It would like **to multiply** two transfer functions.
The MATLAB command for such configuration is "**series**".
The series command is implemented as shown below:



```
[sys]=series(sys1,sys2)
```

Figure 7. 2 Command implementation on series connected linear system

Parallel configuration: If the two blocks are connected as shown below then the blocksare said to be in parallel. It would like to add two transfer functions

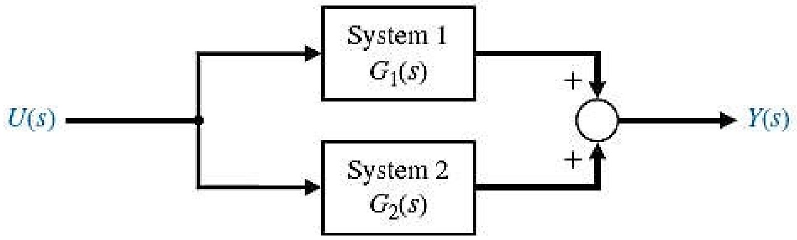


Figure 7. 3 Linear Systems connected in parallel

It would like **to add** two transfer functions.
The MATLAB command for such configuration is "***parallel***".
The *parallel* command is implemented as shown below

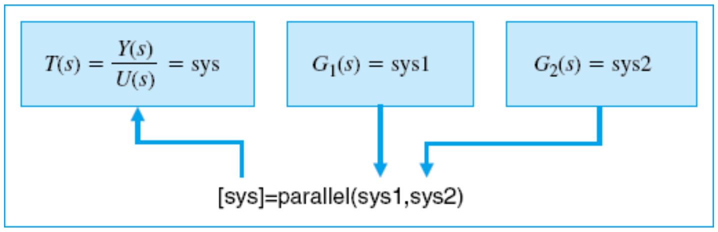

$T(s) = \frac{Y(s)}{U(s)} = \text{sys}$
$G_1(s) = \text{sys1}$
$G_2(s) = \text{sys2}$
[sys]=parallel(sys1,sys2)

Figure 7. 4 Command implementation on parallel connected linear system

Feedback Configuration:

If the blocks are connected as shown below then the blocks are said to be in ***feedback***

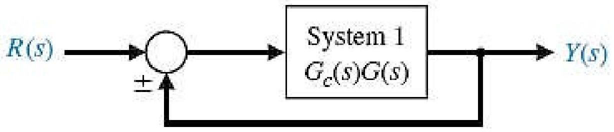


Figure 7. 5 Linear System with unity Feedback

Notice that in the feedback there is *no transfer function H(s)* defined.Such a system is said to be a *unity feedback system*
The MATLAB command for implementing a feedback system is “feedback”.
The feedback command is implemented as shown below:

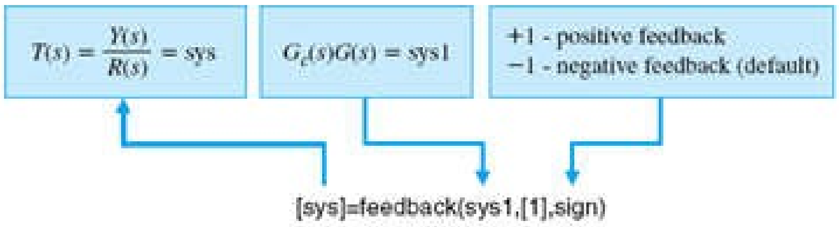

$T(s) = \frac{Y(s)}{R(s)} = \text{sys}$
$G_c(s)G(s) = \text{sys1}$
+1 - positive feedback
−1 - negative feedback (default)
[sys]=feedback(sys1,[1],sign)

Figure 7. 6 Command implementation on linear system with feedback

When H(s) is non-unity or specified, such a system is said to be a non-unity feedback system asshown below

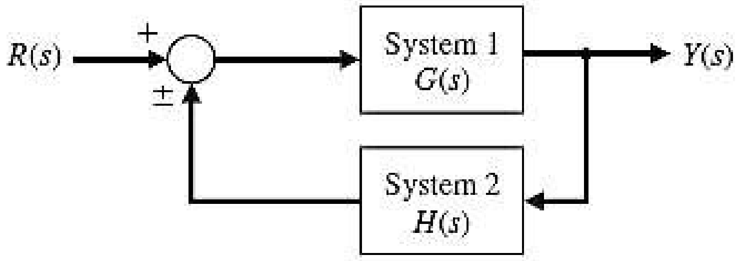


Figure 7. 7 Linear System with non-unity Feedback

Procedure:

Example 1:

Given the transfer functions of individual blocks generate the system transfer function of theblock combination

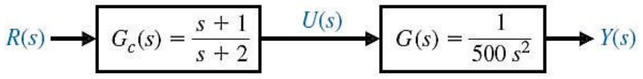


Figure 7. 8 Linear System with series configuration

```
>>numg=[1]; deng=[500 0 0]; sysg=tf(numg,deng);
```

```
>>numh=[1 1]; denh=[1 2]; sysh=tf(numh,denh);
>>sys=series(sysg,sysh);
>>sys
```
Transfer Function:

$$\frac{s+1}{500\,s^3+1000\,s^2}$$

Example 2:

Given a unity feedback system as shown in the figure, obtain the overall transfer function usingMATLAB

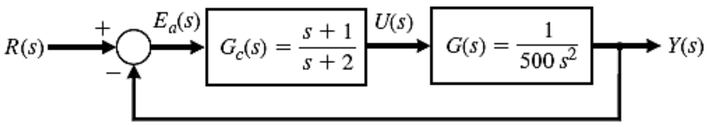


Figure 7. 9 Linear System with series configuration

```
>>numg=[1]; deng=[500 0 0]; sys1=tf(numg,deng);
>>numc=[1 1]; denc=[1 2]; sys2=tf(numc,denc);
>>sys3=series(sys1,sys2);
>>sys=feedback(sys3,[1])
```
Transfer Function:

$$\frac{s+1}{500s^3+1000s^2+s+1}$$

$$\frac{Y(s)}{R(s)}=\frac{G_c(s)G(s)}{1+\ G_c(s)G(s)}$$

Example 3:

Given a non-unity feedback system as shown in the figure, obtain the overall transfer function using MATLAB

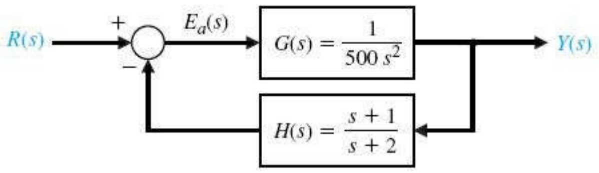


Figure 7. 10 Linear System with parallel configuration

```
>>numg=[1]; deng=[500 0 0]; sys1=tf(numg,deng);
>>numh=[1 1]; denh=[1 2]; sys2=tf(numh,denh);
>>sys = feedback(sys1,sys2);
>>sys;
```
Transfer Function:

$$\frac{s+1}{500s^3+1000s^2+s+1}$$

$$\frac{Y(s)}{R(s)}=\frac{G_c(s)G(s)}{1+\ G_c(s)G(s)}$$

Task:

1. Obtain the overall transfer function using MATLAB

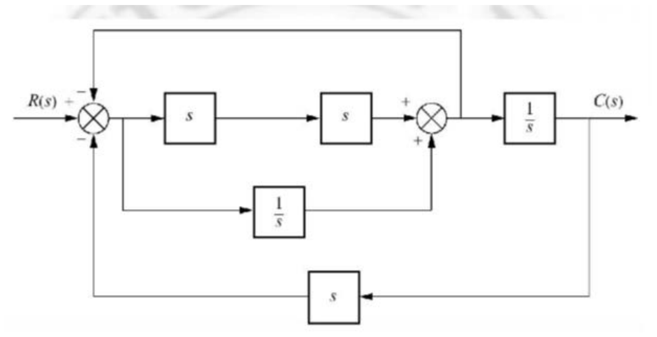


Figure 7. 11 Linear System with Multiple Feedback Blocks

2. Obtain the overall transfer function using MATLAB

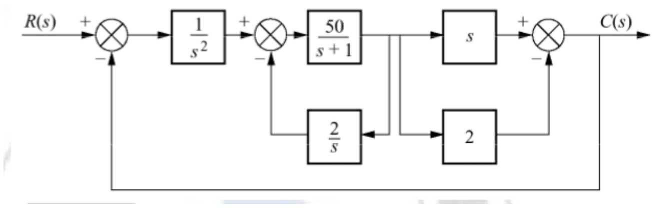


Figure 7. 12 Linear System with Feedback Blocks

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No.______________ Date: ______________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 08

Objective:

Practice to analyze the System Stability using MATLAB

Equipment:

PCs with installed MATLAB

Theory:

This section begins with a discussion of the Routh-Hurwitz stability method. We will see howthe computer can assist us in the stability analysis by providing an easy and accurate method for computing the poles of the characteristic equation.

For the case of the characteristic equation as a function of a single parameter, it will be possible to generate a plot displaying the movement of the poles as the parameter varies.

Routh Stability Analysis

As stated earlier, the Routh criterion is a necessary and sufficient criterion for stability. Given a characteristic equation with fixed coefficients, we can use Routh-Hurwitz to determine thenumber of roots in the right half-plane.

Stability Design via Routh Criteria

Whenever the characteristic equation is a function of a single parameter, the Routh-Hurwitz method can be utilized to determine the range of values that the parameter may take while maintaining stability.

Example:

Consider the closed-loop feedback system shown in Figure 1 with transfer function associated with the forward path as:

$$G(s) = \frac{1}{s^3 + 2s^2 + 4s + K}$$

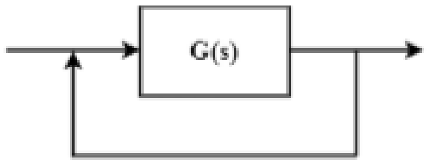


Figure 8. 1 Close Loop Control System

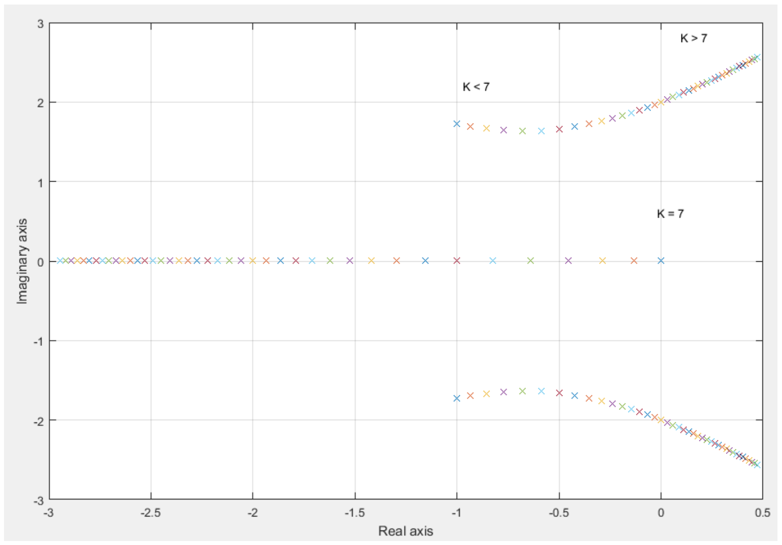


Figure 8. 2 Poles of Close loop system for the range of K<20

Using a Routh-Hurwitz approach, we find that we require $0 < K < 7$ for stability. We can verify this result graphically. As shown in script file, we establish a vector of values for K at which we wish to compute the roots of the characteristic equation. Then using the roots' function, we calculate and plot the roots of the characteristic equation, as shown in Figure. It can be seen that as K increases, the roots of the characteristic equation move toward the right half-plane asthe gain tends toward $K = 7$, and eventually into the right half-plane when $K > 7$
The MATLAB script contains for loop. This function provides a mechanism for repeatedly executing a series of statements a given number of times. The for loop connected to an end statement sets up a repeating calculation loop

```
% This script computes the roots of the characteristic% equation
G(s) = s^3 + 2 s^2 + 4 s + K for 0 < K < 20K= [0:0.5:20];        % 0 < K < 20
for i=1: length(K)                                           % for loop
q= [1 2 4 K(i)];
p(:,i)=roots(q)end
figure (1) plot(real(p),imag(p),'x'),grid
xlabel('Real axis') ylabel('lmaginary axis')
gtext('K < 7')                  % Writing text on graphicgtext('K = 7')
gtext('K > 7')
num=[1]; den=[1 2 4 8];
sysg=tf(num,den);
sys=feedback(sysg,[1]);
pole(sys)                                 % poles of closed loop system
step(sys,100);
```

Task:

1. Consider the closed loop system shown in figure 3. Find the range of gain *K* that willcause system to be stable, unstable and marginally stable. Plot step response for three values of *K*

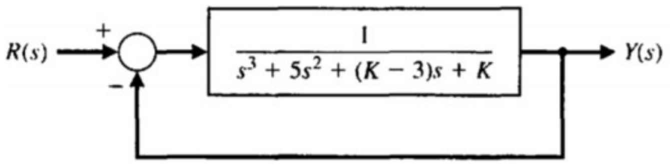


Figure 8. 3 unity feedback close loop system

2. Given the forward-path transfer function of unity-feedback control system

$$G\ (s) = \frac{K(s+10)+(s+20)}{s^2\ (s+2)}$$

Apply the Routh-Hurwitz criterion to determine the stability of the closed-loop system asa function of *K*. Determine the value of *K* that will cause sustained constant-amplitude oscillations in the system. Determine the frequency of oscillations.

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No._______________ Date: ________________________

Software Use Rubric

Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 09

Objective:
Trace the characteristics of the each of proportional (P), the integral (I), and the derivative (D) controls and obtain a desired response by using them.

Equipment:
PCs with installed MATLAB

Theory:
Consider the following unity feedback system:

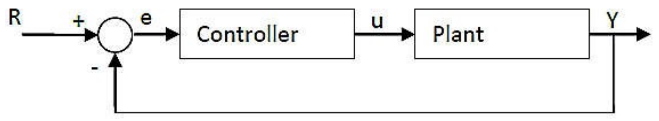


Figure 9. 1 unity feedback system

Plant: A system to be controlled.
Controller: Provides excitation for the plant; Designed to control the overall system behavior
The three-term controller: The transfer function of the PID controller looks like the following:

$$K_P + \frac{K_I}{s} + K_D s = \frac{K_D s^2 + K_P s + K_I}{s}$$

K_P = Proportional gain
K_I = Integral gain
K_D = Derivative gain
First, let's take a look at how the PID controller works in a closed-loop system using the schematic shown.

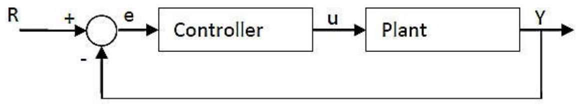


Figure 9. 2 PID Controller with close loop control system

The variable (e) represents the tracking error, the difference between the desired input value (R) and the actual output (Y).
This error signal (e) will be sent to the PID controller, and
The controller computes both the derivative and the integral of this error signal.
The signal (u) just past the controller is now equal to the *proportional gain (KP) times themagnitude of the error plus the integral gain (KI) times the integral of the error plus thederivative gain (KD) times the derivative of the error*
This *signal (u)* will be sent to the *plant*, and the new output (Y) will be obtained.

$$u = K_P e(t) + K_I \int e(t)dt = K_D \frac{de(t)}{dt}$$

This new output (Y) will be sent back to the sensor again to find the new error signal (e). The controller takes this new error signal and computes its derivatives and its internal again.The process goes on and on

Procedure:

For a simple mass, spring, and damper problem

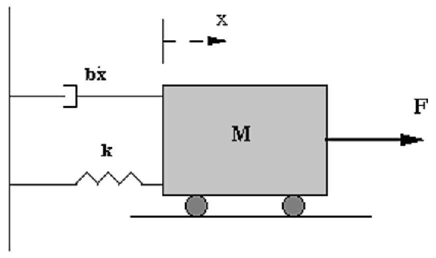


Figure 9. 3 Mass Spring System

The transfer function between the displacement *X(s)* and the input *F(s)* then becomes:

$$\frac{X(s)}{F(s)} = \frac{1}{Ms^2 + Bs + K}$$

Let

- M = 1kg
- b = 10 N.s/m
- k = 20 N/m
- F(s) = 1
- Plug these values into the above transfer function

$$\frac{X(s)}{F(s)} = \frac{1}{s^2 + 10s + 20}$$

The goal of this problem is to show you how each of Kp, Ki and Kd contribute to obtain

- Fast rise time
- Minimum overshoot
- No steady-state error

Open-loop step response:

Let's first view the open-loop step response.

MATLAB command window should give you the plot shown below:

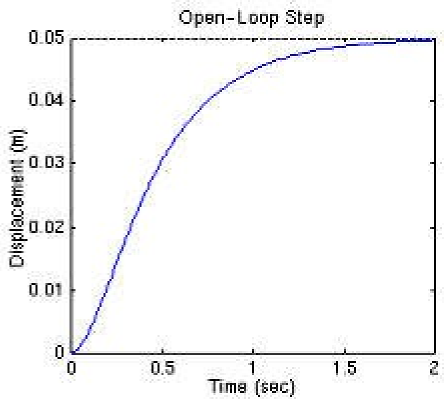


Figure 9. 4 Open loop step response

```
>>num=1;
>>den=[1 10 20];
>>plant= tf(num,den);
```

```
>>step(plant)
```

- 0.05 is the final value of the output to a unit step input.
- This corresponds to the steady-state error of 0.95, quite large indeed.
- Furthermore, the rise time is about one second, and the settling time is about 1.5seconds.
- Let's design a controller that will reduce the rise time, reduce the settling time, andeliminates the steady-state error

Proportional Control:

The closed-loop transfer function of the above system with a proportional controller is:

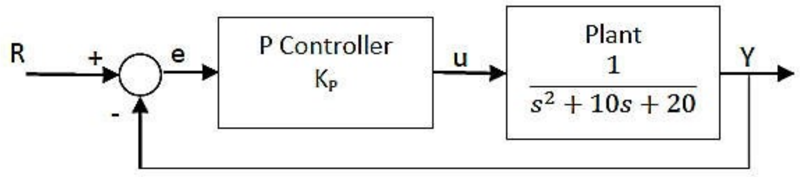


Figure 9. 5 Proportional Controller with control System

$$\frac{X(s)}{F(s)} = \frac{K_P}{s^2 + 10s + (20 + K_P)}$$

Let the proportional gain (KP) equal **300**:

MATLAB PROGRAM:

```
Kp=300;
contr=Kp;
sys_cl=feedback(contr*plant,1);                    %by default –ve feedbackt=0:0.01:2;
step(sys_cl,t)
```

Proportinal Derivative Control:

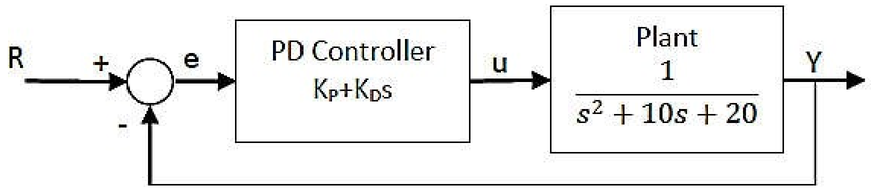


Figure 9. 6 Proportional and Derivative controller with control system

The closed-loop transfer function of the given system with a PD controller is:

$$\frac{X(s)}{F(s)} = \frac{K_D s + K_P}{s^2 + (10 + K_D)s + (20 + K_P)}$$

Let $KP = 300$ as before and let $KD = 10$

Proportional-Derivative control:

```
Kp=300;
Kd=10;
contr=tf([Kd Kp],1);
sys_cl=feedback(contr*plant,1);
t=0:0.01:2;
step(sys_cl,t)
```

Proportional-Integral control:

The closed-loop transfer function of the given system with a PI controller is:

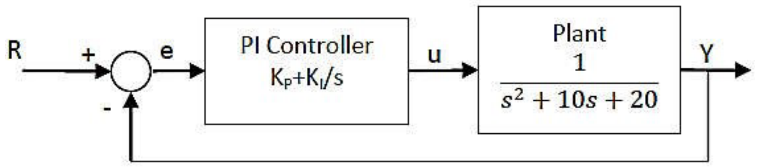


Figure 9. 7 Proportional and Integral controller with control system

$$\frac{X(s)}{F(s)} = \frac{K_P s + K_I}{s^3 + 10s^2 + (20 + K_P)s + K_I}$$

Let K_P equal 30 and let K_I equal 70

```
Kp=30;
Ki=70;
contr=tf([Kp Ki],[1 0]);
sys_cl=feedback(contr*plant,1);
t=0:0.01:2;
step(sys_cl,t)
```

Proportional-Integral-Derivative control:

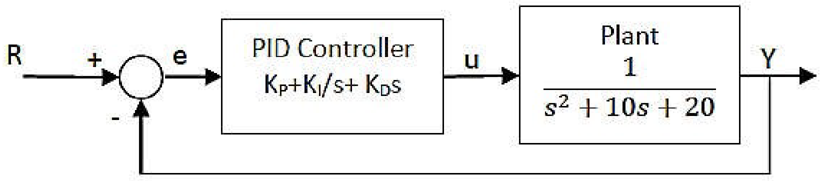


Figure 9. 8 Proportional Integral and Derivative controller with control system

Now, let's take a look at a PID controller. The closed-loop transfer function of the givensystem with a PID controller is;

$$\frac{X(s)}{F(s)} = \frac{K_D s^2 + K_P s + K_I}{s^3 + (10+K_D)s^2 + (20 + K_P)s + K_I}$$

After several trial-and-error runs, the gains K_p=350, K_i=300, and K_d=50 provided the desiredresponse

```
Kp=350;
KI=300;
D=50;
contr=tf([Kd Kp Ki],[1 0]);
sys_cl=feedback(contr*plant,1);
t=0:0.01:2;
step(sys_cl,t)
```

Observations:

CL Response	Rise time	Overshoot time	Settling time	S-S error
K_P				
K_P and K_I				
K_P and K_D				
K_P, K_I and K_D				

Table 9. 1 Result of Proportional, Integral & Derivative Controller

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No.________________ Date: __________________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 10

Objective:
Build the designing of P, PD, PI, and PID controllers to meet closed-loop performance specifications including transientperformance and steady error.

Equipment:
PCs with installed MATLAB

Theory:

For this lab, we will assume a unity feedback controller of the form shown in Figure 10.1, were *C(s)*the controller transfer is function and *P(s)* is the plant transfer function

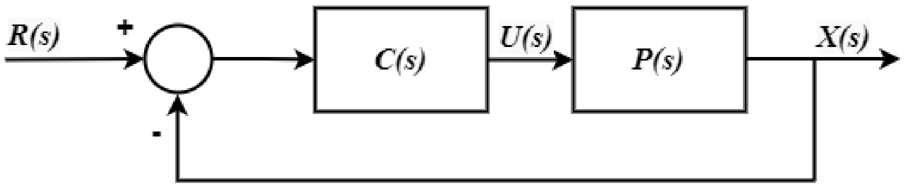


Figure 10. 1 Unity Feedback Control System

Recall that the first step to any root locus controller design is computing the defect angle, which can be interpreted as how far a set of desired second order closed-loop poles are from being on the root locus. The larger the magnitude of the defect angle, the "further" those desired closed loop poles are from being on the root locus. Once you know the defect angle, you can make an informed decision for controller structure. With controller structure specified, the defect angle prescribes the relative locations of the poles and zeros.

Note, in controller design there are multiple possible solutions, some better than others. It is possible to have multiple designs that satisfy the given performance constraints, but practical implementation issues and cost could be prohibitive for some designs. As a general rule, it is a good idea to keep your controller as simple as possible while meeting the prescribed performance criteria. In this lab we will be investigating several controller structures on individual plants and comparing the design process and performance. The common controller structures we will be using in this lab are listed in Table 1 along with their respective transfer functions.

Table 10. 1 Common Controller Types

Controller Type	**Controller Structure**
Proportional (P)	$C(s) = k_p$
Integral (I)	$C(s) = \frac{k_i}{s}$
Proportional + Integral (PI)	$C(s) = k_p + \frac{k_i}{s} = \frac{k(s+z)}{s}$
Lag Controller	$C(s) = \frac{k_c(s+z)}{(s+p)}$ $, where\ \lvert z\rvert > \lvert p\rvert(s$
Proportional + Derivative (PD)	$C(s) = k_p + k_d s = k(s+z)$
Lead Controller	$C(s) = \frac{k(s+z)}{(s+p)}$ $, where\ \lvert p\rvert > \lvert z\rvert(s$
Proportional + Integral + Derivative (PID) (Real zeros)	$C(s) = k_p + \frac{k_i}{s} + k_d s = \frac{k(s+z_1)(s+z_2)}{s}$

Proportional + Integral + Derivative (PID) (Complex Conjugate zeros)	$C(s) = k_p + \frac{k_i}{s} + k_d s = \frac{k(s+z)(s+z^*)}{s}$

Introduction to MATLAB ‘’sisotool:

A. Getting Started

1. Enter the transfer function for the plant, in your workspace (i.e., from theMATLAB command prompt).
2. Type „sisotool‟ at the command prompt.
3. Click „close‟ when the help window comes up.

B. Loading the Transfer Function

1. We will usually be assigning to block „G‟ (the plant). Under Control System Tab, click on Edit Architecture. Against block name G type **P** in the value space. SelectOK.

C. Generating the Step Response and the Control Effort Plot

1. You can now click on the pink boxes on the root locus (the current closed-loop poles for the given gain) and move them along the root locus. Essentially, you are exploring different controller gain values by doing this. Note how the step response changes as you move the closed-loop pole locations.
2. The values of the closed-loop poles will appear at the bottom of the root locus window as you click and hold the mouse on the pink boxes representing them. This only gives you the value of the closed loop pole you are clicking on. If you need the other closed-loop pole locations, you will have to click on them on each of the other branches.

D. Adding Design Constraints

1. Right-click on the root locus plot. From the menu that pops-up, select Design Requirements New… to add constraints or Design Requirements Edit… to edit existing constraints

1. At this point, point you can choose from settling time, percent overshoot, damping ratio,and natural frequency constraints.

Task:

1. Use the plant given

$$P(s) = \frac{30}{s^2 + 11s + 30} = \frac{30}{(s+5)(s+6)}$$

This is a second order system with two real poles located at -5 and -6. Our goal is to speed up theclosed-loop system response so that the two percent settling time is less than 1 second, produce aposition error of 0.1 or less, and keep percent overshoot less than 10%. To keep things reasonable, keep the gain, k less than 10 for all designs.

Now make a PID controller with real zeros at -7 and -8. Determine the rootlocus for this system. Find a value of on this root locus so that percent overshoot is less than 2% and the settling time is less than 0.02 seconds. (Remember tokeep $k < 10$.) Save the step response and the controller that produced it.

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No._______________ Date: ________________________

Software Use Rubric

Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 11

Objective:
Derive the relationship between frequency response and step response characteristics using MATLAB

Equipment:
PCs with installed MATLAB

Theory:
The frequency response is the steady state response of a system to a sinusoidal input; the category is the plot of the magnitude of the output in dB versus frequency using logarithmic scale, and the phase shift versus frequency as shown in figure below:

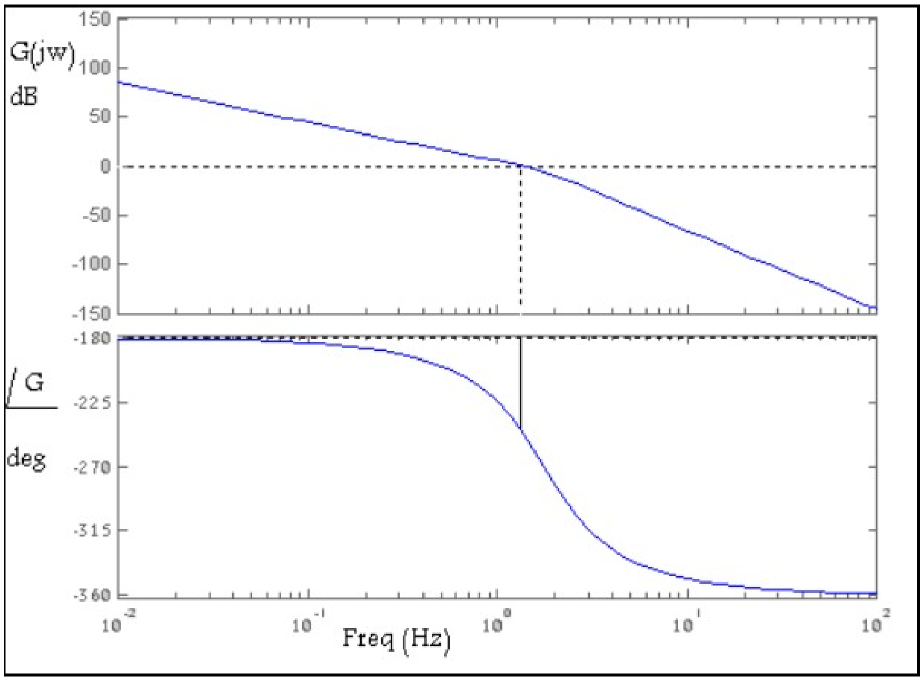


Figure 11. 1 Frequency response of a control system

The frequency response characteristics of the system can be obtained directly from the sinusoidaltransfer function in which "*s*" is replaced by "*jω*", where "*ω*" is the frequency. Consider the linear time invariant system shown in figure below where the transfer function is *G(s)* and the input is a sinusoidal and it is given by *X (t)* and the output is *Y (t):*

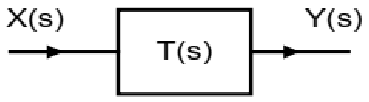


Figure 11. 2 Linear time invariant system

To completely characterize a linear system in the frequency domain we must specify both the amplitude ratio and the phase angle as function of frequency

$$|G(jw)| = \left|\frac{Y(jw)}{X(jw)}\right|$$

$$< G(jw) \leq \frac{Y(jw)}{X(jw)}$$

There are some terms associated with Bode Plot

Resonant Peak M_r

The resonant peak M_r is the maximum value of$|M(j\omega)|$

$$M_r = \frac{1}{2\zeta\sqrt{1-\zeta^2}}$$

$$M_r = 1, \zeta = 0.707$$

Resonant Frequency ω_r

The resonant frequency ω_r is the frequency at which the peak resonance M_r occurs

$$w_r = w_n\sqrt{1-2\zeta^2}$$

Bandwidth BW:

The bandwidth BW is the frequency at which $|M(j\omega)|$ drops to 0.707 percent of, or 3dB downfrom, its zero-frequency value.

$$BW = w_n\left[(1-2\zeta^2) = \sqrt{4\zeta^4 - 4\,\zeta^2 + 2}\right]½$$

Example:

Find the steady state response for the system where the open loop transfer function is given by:

$$G(s) = \frac{64}{s^2+8s+64}$$

```
  n=64;
  d=[1 8 6 4];
  bode(n,d)
  grid on
  w_n= input('Enter natural frequency: ');
  zeta = input('Enter damping ratio: ');
  if zeta >=0.707M_r=1;
        w_r = 0;
else
        M_r=1/(2*zeta*sqrt(1-zeta^2));
        w_r= w_n* sqrt(1-2*zeta*zeta);
  end
  BW=w_n*sqrt((1-2*zeta*zeta) +sqrt(4*zeta^4-4*zeta^2+2));
  figure
  sys=tf(n,d);
  step(sys)
  grid on
```

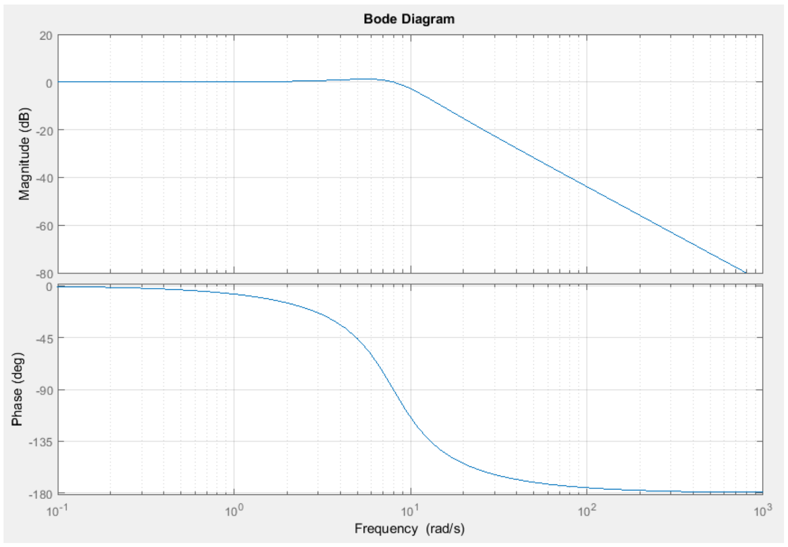


Figure 11. 3 Close-loop frequency response

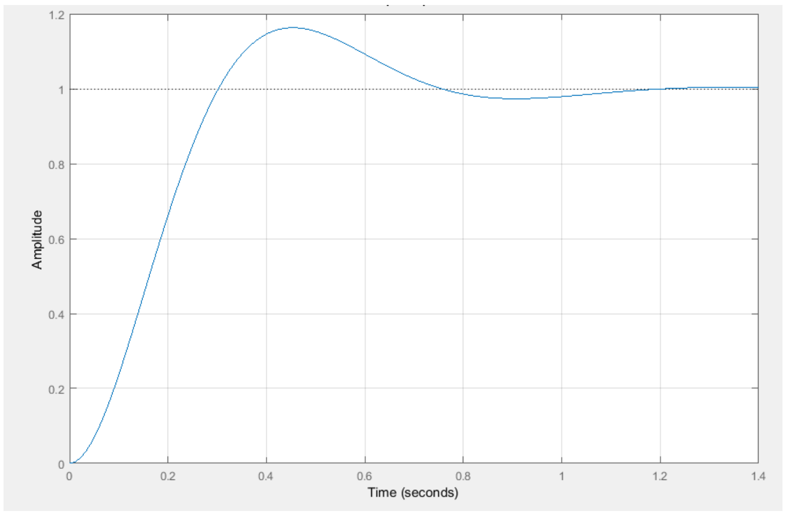


Table 11. 4 Step Response of the System

Task:

1. Consider the forward-path transfer functions of unity-feedback control system as:

$$G(s) = \frac{4}{s^2 + 3s + 4}$$

- Using MATLAB, find frequency response and step response of the system.
- With the help of MATLAB, determine bandwidth and rise time of the system.
- For the stated values of zeta and natural frequency in the table below, findbandwidth and rise time.
- Analyze the obtained results and find a relationship between time-domain andfrequency domain parameters.
- Discuss their effect on pole-zero map

Table 9. 1 $\omega n = 2$

ζ	**BW**	**t_r**
0.1		
0.2		
0.4		
0.7		
1		

Table 9. 2 $\zeta = 0.75$

ω_n	**BW**	**t_r**
1		
2		
5		
10		
20		

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No.______________ Date: ______________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 12

Objective:
Do the plotting of Root Loci with MATLAB

Equipment:
PCs with installed MATLAB

Theory:
Root locus analysis is a graphical method for examining how the roots of a system change with variation of a certain system parameter, commonly gain within a feedback system. The root locus plots the poles of the closed loop transfer function in the complex s plane as a function of a gain parameter. The path that closed loop poles attain when the value of k changes is called Root Locus. The path of root locus can be of any shape. It starts with the open loop pole and ends at the open loop zero. Root locus effects on stability, steady state error. Here k changes from 0 to ∞. At k=0 the position of poles of open loop system and closed loop system is same but by increasing k, the position of closed loop pole will change only. The rules of the root locus give us a clear and precise understanding of the endless patterns that can be created by an infinite set of characteristic equations. The command rlocus (GH, K) allows us to specify the range of gain K for plotting the root locus. Also study the commands [p,K]=rlocus(GH) and [p]=rlocus(GH,K) using MATLAB.

Procedure:
Example 1:
Consider the system shown in the block diagram of Fig. 1

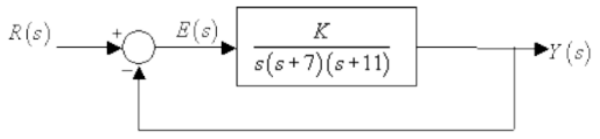


Figure 12. 1 Close Loop system

The characteristic equation of the system is with

$$G(s) = \frac{K}{s(s+7)(s+11)}$$

The following MATLAB script plots the root loci for $0 < K < \infty$

```
s = tf('s');
G = 1/(s*(s+7)*(s+11));
rlocus(G);
axis equal;
```

Clicking at the point of intersection of the root locus with the imaginary axis gives the data shown in Fig. 12.2. We find that the closed-loop system is stable for $K < 1360$; and unstable for $K > 1360$.

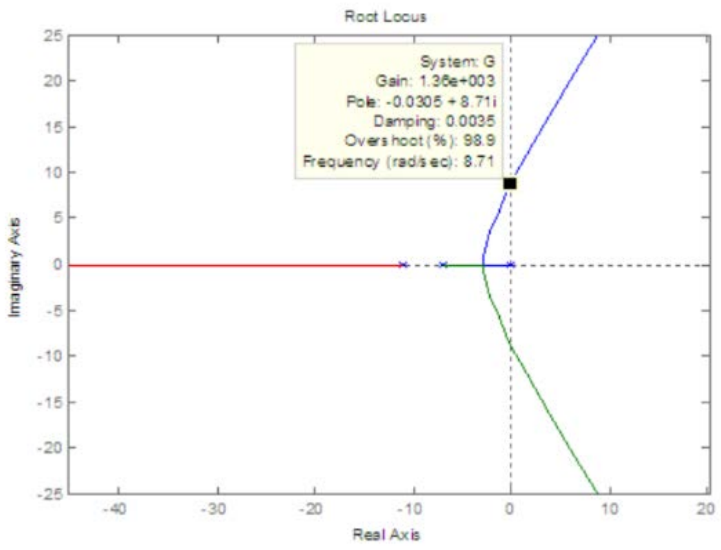


Figure 12. 2 Root Locus Plot

Figure 3 shows step responses for two values of K.

```
>> K = 860;
>> step(feedback(K*G,1),'b',5)
>> hold;                % Current plot held
>> K = 1460;
>> step(feedback(K*G,1),'g',5)
```

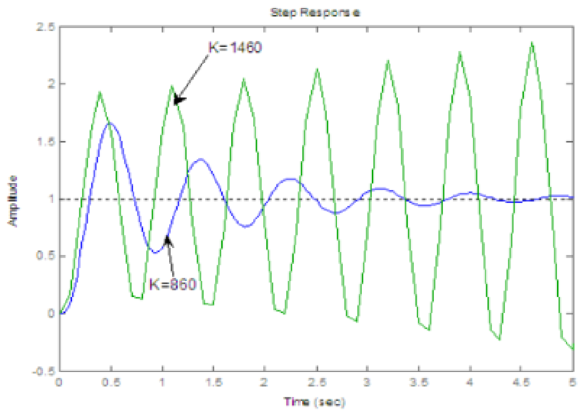


Figure 12. 3 Step responses for two values of K.

Example 2:
Consider the system shown in Figure 4.

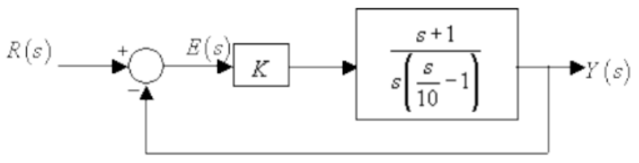


Figure 12. 4 Close Loop system

The plant transfer function G(s) is given as

$$G(s) = \frac{s+1}{s(0.1s-1)}$$

The following MATLAB script plots the root locus for the closed-loop system.

```
clear all;
close all;
s = tf('s');
G = (s+1)/(s*(0.1*s-1));
rlocus(G);
axis equal;
sgrid;
title('Root locus for (s+1)/s(0.1s-1)');
[K,p]=rlocfind(G)
```

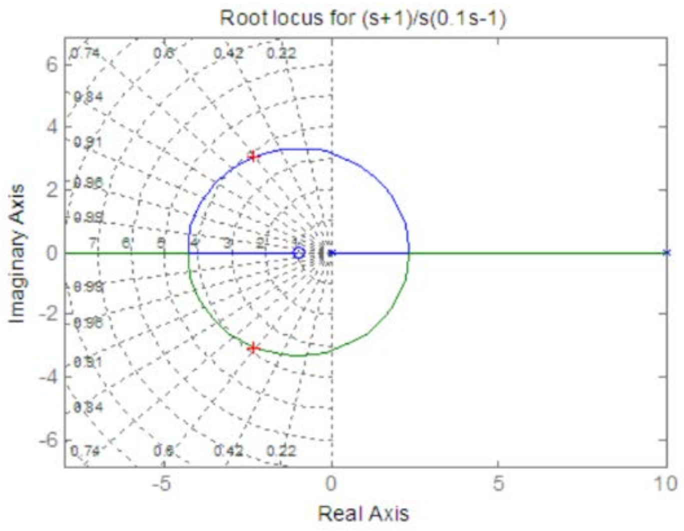


Figure 12. 5 Root Locus plot

```
selected_point = -2.2204 + 3.0099i
K =
1.4494
p =
-2.2468 + 3.0734i
-2.2468 - 3.0734i
```

Example 3

For a unity feedback system with open-loop transfer function

$$G(s) = \frac{K(s^2 - 4s = 20)}{(s+2)(s+4)}$$

a root locus plot shown in Fig. 6 has been generated using the following MATLAB code.

```
s = tf('s');
G = (s^2-4*s+20)/ ((s+2) *(s+4));
rlocus(G);
zeta = 0.45;
wn = 0;
sgrid(zeta,wn);
```

Properly redefine the axes of the root locus using Right click --> Properties --> Limits.

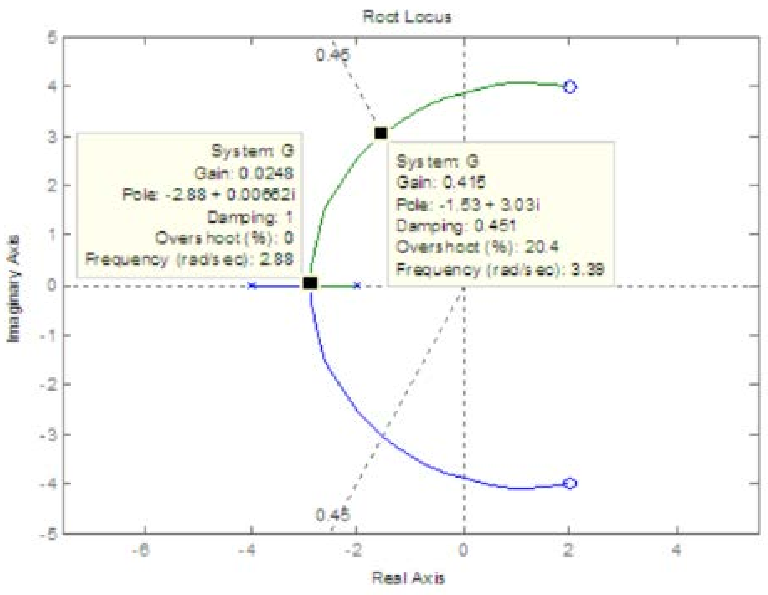


Figure 12. 6 Root Locus plot

Clicking on the intersection of the root locus with $zeta = 0.45$ line gives the system gain $K = 0.415$ that corresponds to closed-loop poles with Clicking on the intersection of the root locus with the real axis gives the breakaway point and the gain at that point

Task

1. Sketch the root loci for the system shown in Figure 10.7. (The gain K is assumed to be positive.) Observe that for small or large values of K the system is overdamped and for medium values of K it is underdamped.

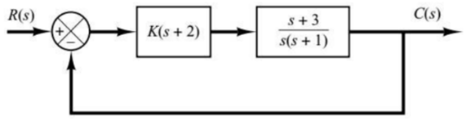


Figure 12. 7 Close Loop Control System

2. Consider the system shown in Figure 8. Plot the root loci with MATLAB. Locate the closed-loop poles when the gain K is set equal to 2

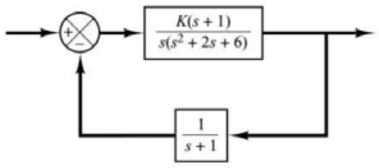


Figure 12. 8 Close Loop Control System

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No.________________ Date: __________________________

Software Use Rubric

Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	

Lab Experiment # 13

Objective:
Design control system for ball and beam system for the stated requirements

Equipment:
PCs with installed MATLAB

Theory:
The open-loop transfer function of the plant for the ball and beam example is given below

$$\frac{R(s)}{\theta(S)} = \frac{mgd}{L(\frac{1}{R^2} + m^2)} \frac{1}{s^2}$$

Let, m = 0.111; R = 0.015; g = -9.8; L = 1.0; d = 0.03; J = 9.99e-6

The design criteria for this problem are:

- Settling time less than <last digit of your NED seat no.> seconds
- Overshoot less than <third last and second last digit of your NED seat no.>%

For example, if your seat no. is 12345, then the design criteria will be

- Settling time less than 5 seconds
- Overshoot less than 34%

A schematic of the closed loop system with a controller is given below:

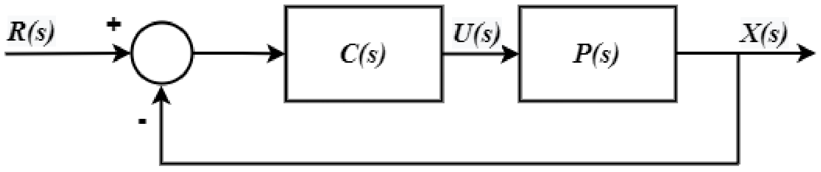


Figure 13. 1 Generic Unity Feedback Control System

The common controller structures are listed in Table 1 along with their respective transfer functions. You can choose a controller from the table below for meeting the design requirements. Discuss the corresponding reason for selection.

Table 13. 1 Common Controller Types

Controller Type	Controller Structure
Proportional (P)	$C(s) = k_p$
Integral (I)	$C(s) = \frac{k_i}{s}$
Proportional + Integral (PI)	$C(s) = kp + \frac{ki}{s} = \frac{k(s+z)}{s}$
Lag Controller	$C(s) = \frac{k_c(s+z)}{(s+p)}, where \lvert z \rvert > \lvert p \rvert$

Proportional + Derivative (PD)	$C(s) = k_p + k_d s = k(s + z)$
Lead Controller	$C(s) = \frac{k(s+z)}{(s+p)}, where\lvert p\rvert > \lvert z\rvert$
Proportional + Integral + Derivative (PID) (Real zeros)	$C(s) = kp + \frac{ki}{s} + k_d s = \frac{k(s+z_1)(s+z_2)}{s}$
Proportional + Integral + Derivative (PID) (Complex Conjugate zeros)	$C(s) = kp + \frac{ki}{s} + k_d s = \frac{k(s+z)(s+z^*)}{s}$

Deliverables

A complete lab report including the following:

- Figures with plots of open loop and closed-loop step responses.
- Controller parameters, gain, pole(s), and zero(s), for each of the controller design alongwith their response and compare.
- Report properly with MATLAB codes and respective plots based on root locus and bodemethods.

NED University of Engineering & Technology
Department of Electronic Engineering

Course Code and Title: EL-305 Instrumentation & Control

Laboratory Session No._______________ Date: ________________________

Software Use Rubric					
Criterion	**Level of Attainment**				
	Below Average (1)	**Average (2)**	**Good (3)**	**Very Good (4)**	**Excellent (5)**
Identification of software menu (syntax, components, commands, tools, layout etc.).	Can't identify software menus.	Rarely identifies software menus.	Occasionally identifies software menus.	Able to identify software menus.	Perfectly able to identify software menus.
Skills to use software (schematic, syntax, commands, tools, layout) efficiently.	Can't use software efficiently.	Rarely uses software efficiently.	Occasionally uses software efficiently.	Often uses software efficiently.	Efficiently uses software (syntax, commands, tools, layout)
Adherence to safety procedures and handling of equipment (computing unit, peripheral devices, and other equipment in lab).	Doesn't handle equipment with required care and safety.	Rarely handles equipment with required care and safety.	Occasionally handles equipment with required care and safety.	Often handles equipment with required care and safety.	Handles equipment with required care and safety.
Ability to troubleshoot software errors (detection and debugging).	Not able to troubleshoot the errors	Rarely able to troubleshoot the errors	Occasionally able to troubleshoot the errors	Often able to troubleshoot the errors	Fully able to troubleshoot the errors
Analysis and interpretation of results/outputs.	Not able to analyze and interpret results/outputs.	Rarely able to perform the analysis and interpretation.	Occasionally able to perform the analysis and interpretation.	Often able to perform the analysis and interpretation.	Perfectly able to perform the analysis and interpretation.

Weighted CLO (Score)	
Remarks	
Instructor's Signature with Date	