Int. J. Engng Ed. Vol. 21, No. 5, pp. 896–905, 2005
Printed in Great Britain.

0949-149X/91 $3.00+0.00

MATLAB and Simulink in Mechatronics Education*

A. ALBAGUL, OTHMAN O. KHALIFA and WAHYUDI
Faculty of Engineering, IIUM, Jalan Gombak, 53100, Kuala Lumpur, Malaysia. E-mail: albagul@iiu.edu.my

MATLAB software package has been assisting engineers to design and test system models for different fields of engineering. It provides a deep understanding of system parameters and how they affect its performance. Since it is a programming package, it can interact with other programming languages to provide engineers who use those languages the flexibility to run their codes within a MATLAB environment. Engineering students can build their model and test it before implementing it in the real process. This paper presents and describes some dynamic models and control systems for mechatronics engineering students using MATLAB and Simulink. Those models will be used as laboratory experiments to expose students to different topics and techniques in control engineering. This will enhance students' knowledge and make them familiar with the MATLAB environment

INTRODUCTION

MATLAB is a high-performance computational and visualization software package and an interaction program, which has been developed through the years for scientific and engineering calculations. It also integrates mathematical computing, visualization, and a powerful language to provide a flexible environment for technical computing. It has many built-in functions that can be used to solve specific problems. This makes it easy for engineering students to solve many problems in just of writing one or two lines of program code rather writing a long code to solve the same problem with other programming languages. It is also possible to build and add any function to suit any scientific or engineering application. Once those functions are written, they can be used as built-in functions. Since MATLAB is a programming tool in nature, it is possible to interact with other programming languages such as C or FORTRAN by an external interface to run those programs within the MATLAB environment. It has the flexibility to accept data from outside to be analyzed and produce the required output. MATLAB have several toolboxes to support different applications. Besides MATLAB, Simulink is an interactive tool for modeling, simulating, and analyzing dynamic systems. It enables engineers to build graphical block diagrams, evaluate system performance, and refine their designs. Simulink integrates seamlessly with MATLAB and is tightly integrated with Stateflow for modeling event-driven behavior.

Simulink is the tool of choice for control system design, DSP design, communications system design and other simulation applications. In this paper, it has been decided to use MATLAB and Simulink as a teaching tool for laboratory work in mechatronics engineering. The examples will be devoted to control systems to explore different techniques based on control system theories. They begin with the basic function of MATLAB to familiarize students with the use of the MATLAB environment. Studying the behavior of a system is very important to be able to understand its dynamics and hence be able to design an appropriate controller to achieve the desired response. This will help the students to test their design before implementing it into the real system, through simulations. Engineers can explore how the system would perform under extreme conditions and evaluate component variations that are not easily duplicated in the product development lab. Some control design techniques will be presented to show students how to design a controller with different approaches and implement them in both MATLAB and Simulink.

BASIC MATLAB CONCEPTS AND INSTRUCTIONS

MATLAB has a variety of toolboxes for different applications but all of them share common functions, statements and commands which are computer platform independent. In this section, some of these objects are presented to utilize any typical session of MATLAB. These objects are statements, variables, matrices, graphics and scripts. Once the MATLAB prompt or workspace is invoked, it is possible to use any statement of the mentioned objects. The MATLAB statement has a special format that consists of variable, operator (assignment) and expression. The function of the

* Accepted 30 May 2005.

operator is to assign some value to a variable and the statement in MATLAB form will be:

```
>> A = 6;
```

This means that the variable A is equal to 6 through the whole program unless it is cleared or reassigned to another value. All mathematical operators can be used in expressions. MATLAB has many built-in functions ranging from a simple function such as absolute value of a variable, abs(x), to more complex ones such as finding the transfer function of a polynomial system, tf([numerator],[denominator]), yet the user can build his/her own functions to perform a special task and save it in the MATLAB library. MATLAB also offers the user the ability to analyze and deal with matrices easily. Entering a matrix is fairly easy. First assigning the matrix name, then use the equal sign and insert the elements of the matrix row by row enclosed in the square brackets. The column elements are separated by commas or blanks and the rows are separated by semicolons such as:

```
>> A = [1, 2; 3 4]
A =
1 2
3 4
>>
```

The user can perform all operations on matrices such as addition, subtraction and even finding the roots and rank of a matrix by a simple command. MATLAB has some functions for graphics which play a major role in both design and analysis of control systems. Graphs can be obtained in two-dimensions 2D or three-dimensions 3D according to the user need. It can produce a simple graph for two vectors with the same length such as plot(x,y) or semilogx(x,y) which is for a $\log_{10}$ scale vector x versus a linear vector y. 3D plots provide the user with extensive facilities for deep visualization of 3D data.

MATLAB FOR CONTROL SYSTEMS

MATLAB has a toolbox that can assist tremendously in the design and analysis of control systems based on the mathematical models of physical systems. Once the mathematical model of the system is obtained then the transfer function of this system can be easily determined.

Mathematical model and transfer function

In this section, the typical and popular spring-mass-damper shown in Fig. 1 is considered for analysis with the assistance of MATLAB. The mathematical equation describing the motion of the mass is:

$$M\ \ddot{x}(t) + f_v\ \dot{x}(t) + k\ x(t) = f(t) \qquad (1)$$

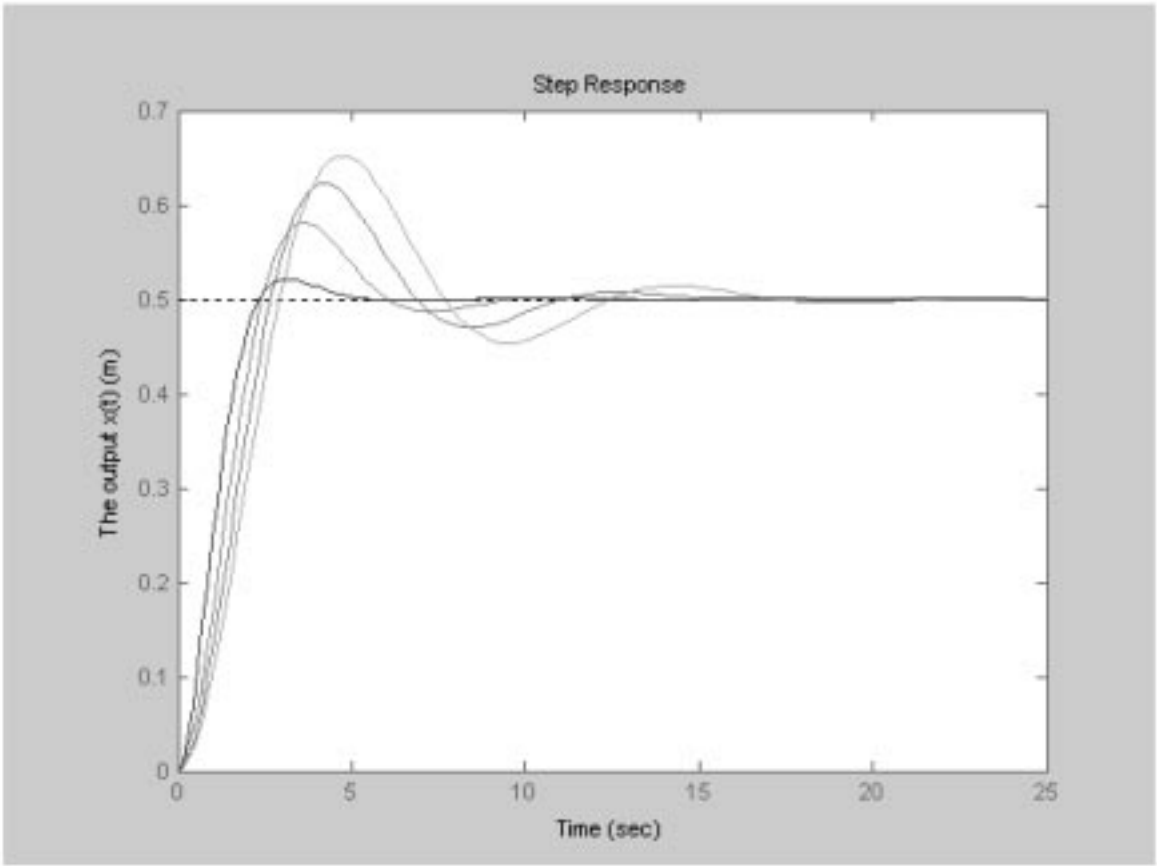

Fig. 1. The spring-mass-damper system.

where, M is the mass, $\ddot{x}(t)$ is the linear acceleration of the mass, f_v is the coefficient of viscous friction, $\dot{x}(t)$ is the linear velocity, k is the spring constant, $x(t)$ is the displacement and $f(t)$ the applied force.

Assuming all initial conditions are zeros and taking the Laplace transform of Equation (1) yields:

$$Ms^2\ Xx(s) + f_v s\ X(s) + ks\ X(s) = F(s) \qquad (2)$$

The transfer function of the system can be obtained:

$$G(s) = \frac{X(s)}{F(s)} = \frac{1}{Ms^2 + f_v s + k} \qquad (3)$$

Once the transfer function is obtained, the analysis of the system can be carried out using MATLAB. First step is to assign the values of the parameters M, f_v and k then the transfer function can be entered into MATLAB as polynomial of the numerator and denominator and hence the transfer function can be produced as follows:

```
>; M = 1, fv = 2, k = 2;
>> num = [1]; den = [M fv k];
>> TF = tf (num,den)
```

Transfer function:

```
1
---------
s^2 + 2 s + 2
```

The poles and zeros of the system can be obtained separately or in a column vector:

```
>> p = pole (TF); z = zero (TF)
p =
-1.0000 + 1.0000i
-1.0000 - 1.0000i
z =
Empty matrix: 0-by-1
```

It can be seen that the z is an empty matrix because the system only has two poles but no zeros.

The output response of the system for a step input can be determined and plotted as shown in Fig. 2 using the following commands:

```
>> step (TF)
>> xlabel ('Time (sec)')
>> ylabel ('The output x(t) (m)')
>> title ('The step response of S-M-D
   system')
```

The output response can be obtained for different values of any of the parameter and produce multiple graphs using the for-end loop in the MATLAB with the same statements.

Representing the control system

Control systems are usually represented by a block diagram consisting of the process or plant denoted by G(s), the controller $G_c(s)$ and some other components such as sensors and actuators. MATLAB has some functions that can be used to interconnect the components of the block diagram of control systems to determine the overall transfer function T(s). These components can be interconnected in series, parallel or in feedback as shown in Fig. 3.

```
>> numg1 = [1 1]; deng1= [1 5 6]; TF1 = tf
   (numg1,deng1);
>> numg2 = [1]; deng2= [1 1]; TF2 = tf
   (numg2,deng2);
>> numg3 = [1 1]; deng3 = [1 2]; TF3= tf
   (numg3,deng3);
>> numg4 = [1]; deng4= [1 1]; TF4= tf
   (numg4,deng4);
>> TFA = parallel (TF1,TF2);
>> TFB = series (TF3,TFA);
>> TFOA = feedback (TFB, TF4)
```

Transfer function:

```
2 s^4 + 11 s^3 + 23 s^2 + 21 s + 7
---------------------------
s^5 + 9 s^4 + 33 s^3 + 60 s^2 + 54 s + 19
```

The analysis and performance of the S-M-D control system

Most of the successful control systems are feedback systems which are found in many applications everywhere. This success is due to certain reasons such as:

- They reject the effect of the disturbance in the system.
- They reduce and in some cases eliminate the steady-state errors.
- They allow the adjustment of the transient response of the system.
- They decrease the sensitivity of the system to the variation in the process parameters.

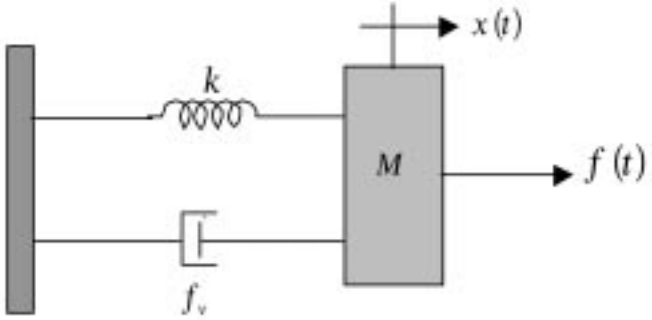

Fig. 2. The output response of the S-M-D system.

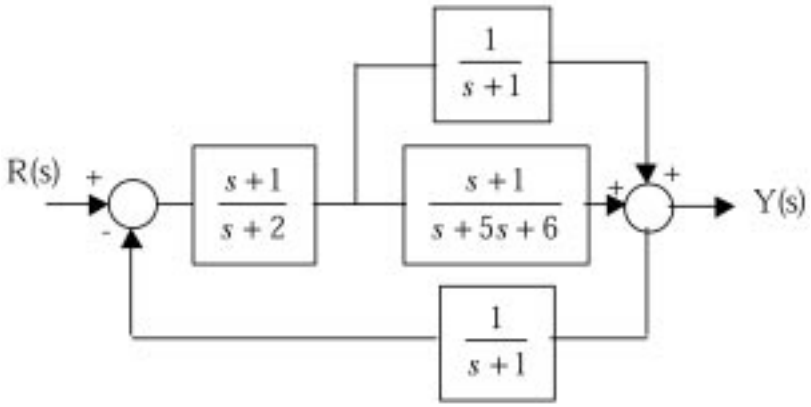

Fig. 3. Block diagram of a control system.

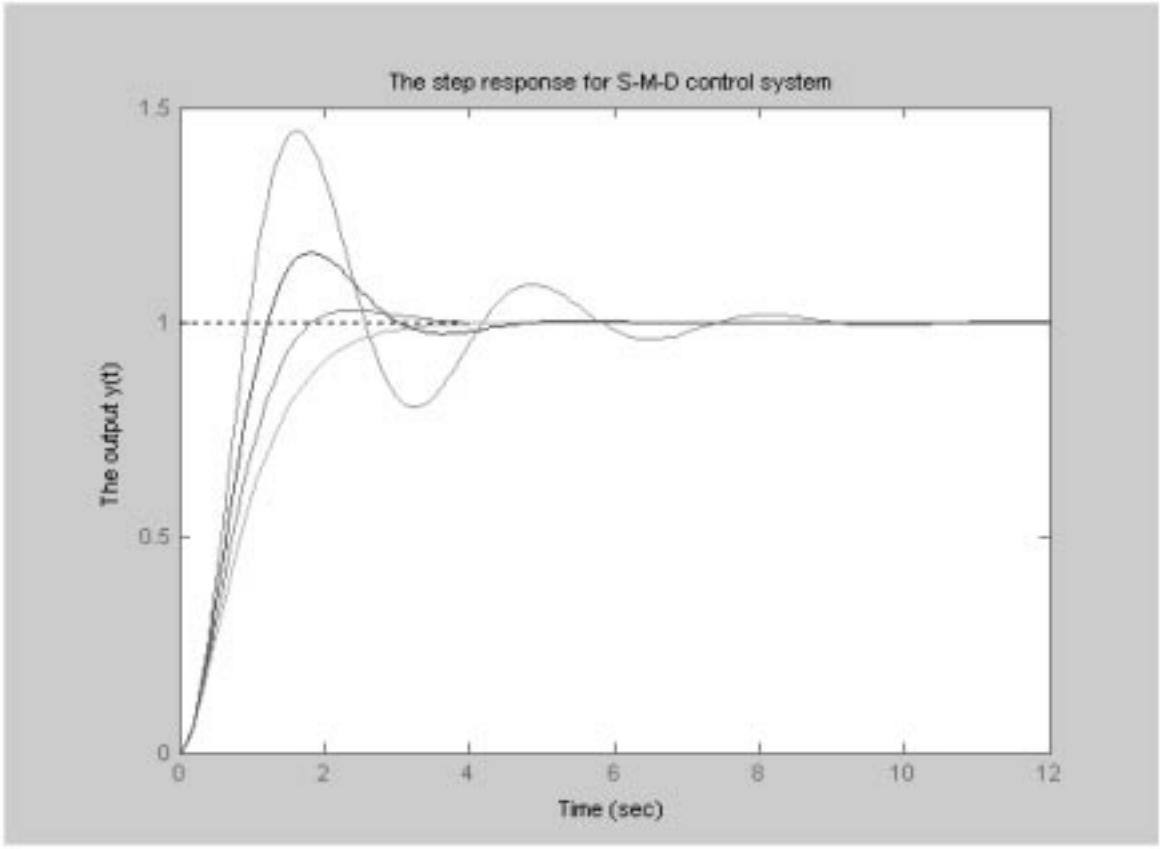

Fig. 4. The step response of the S-M-D system.

In this section, the transfer function of the S-M-D system obtained above will be considered to study and analyze the performance as well as the characteristics of the system. The transfer function can be rewritten in terms of the characteristic equation of the second order system:

$$X(s) = \frac{1/M}{s(s^2 + (f_v/M)s + k/M)} = \frac{\omega_n}{s(s^2 + 2\zeta\omega_n s + \omega_n^2)} \quad (4)$$

The characteristic parameters of the second-order system such as the steady-state error, e_{ss}, maximum overshoot, pos, the rise time, T_r, the peak time, T_p, the settling time, T_s, and the damping ratio, ζ, can be obtained by executing a set of MATLAB functions stored in a MATLAB file called `gsos.m`. By executing the `gsos.m`, the user will be asked to enter the system polynomial (i.e. the numerator and denominator of the system) and then it will produce the values of all the mentioned parameters as well as a plot of the output response as shown in Fig. 4.

The stability of a control system in MATLAB

Stability is a fundamental issue and very important in control engineering. An unstable system is

neither useful nor practical when a precise response is needed. A stable system is a system which produces a bounded output for every bounded input. Furthermore, all poles of the transfer function of a stable system must have a negative real part (i.e. all poles must lay on the left-half of the plane).

One of the most simple and easy tests to determine the stability of a system is the Routh-Hurwitz method. This method based on analyzing the characteristic equation of the transfer function to determine the stability of the system as well as how many poles are in the unstable region as (i.e. the right-half plane). MATLAB has the ability to assist the user by providing a simple and accurate method to determine the stability as well as the range of the gain that makes the system stable. Here, the user will execute a MATLAB file called `stability.m` and will be asked to enter the range of the gain k and then the polynomial of the transfer function. This program based on a single parameter in the characteristic equation but can be modified easily according to the user needs. Once the program is executed, it will produce the transfer function, the roots of the transfer function and the value of the gain that makes the system stable.

Most common techniques in control using MATLAB

This section is devoted to some common techniques to study and analyze the relative stability of control systems. These techniques are the Root Locus and Bode Diagram. The root locus is a method that provides the designer with a graphical representation of the closed-loop poles as one parameter is varied. Meanwhile, a Bode diagram is a graphical method based on the frequency response of the system. The Bode diagram consists of two graphs, one is for the magnitude and the other is for the phase angle of the system. The system is tested by a sinusoidal input signal where the frequency varies. MATLAB has the ability to provide the user with graphical output of both methods by entering simple commands in MATLAB prompt. Considering a simple second-order system which is represented by a transfer function of:

$$G(s) = \frac{s+1}{s^2+3s+6} \tag{5}$$

The root locus of the system can be obtained from MATLAB as shown in Fig. 5 and the Bode diagram of the same system is shown in Fig. 6.

```
>> num = [1 2];
>> den = [1 3 6];
>> TF = tf(num,den)
>> rlocus (TF), grid, figure;
>> bode (TF)
>> grid
```

These figures will give the designer the opportunity to analyze the system and check for the relative stability as well as obtaining the parameters associated with each technique. In the case of the root locus, by clicking on the root locus graph produced by MATLAB, the designer will be able to determine the gain of the system, the position of the

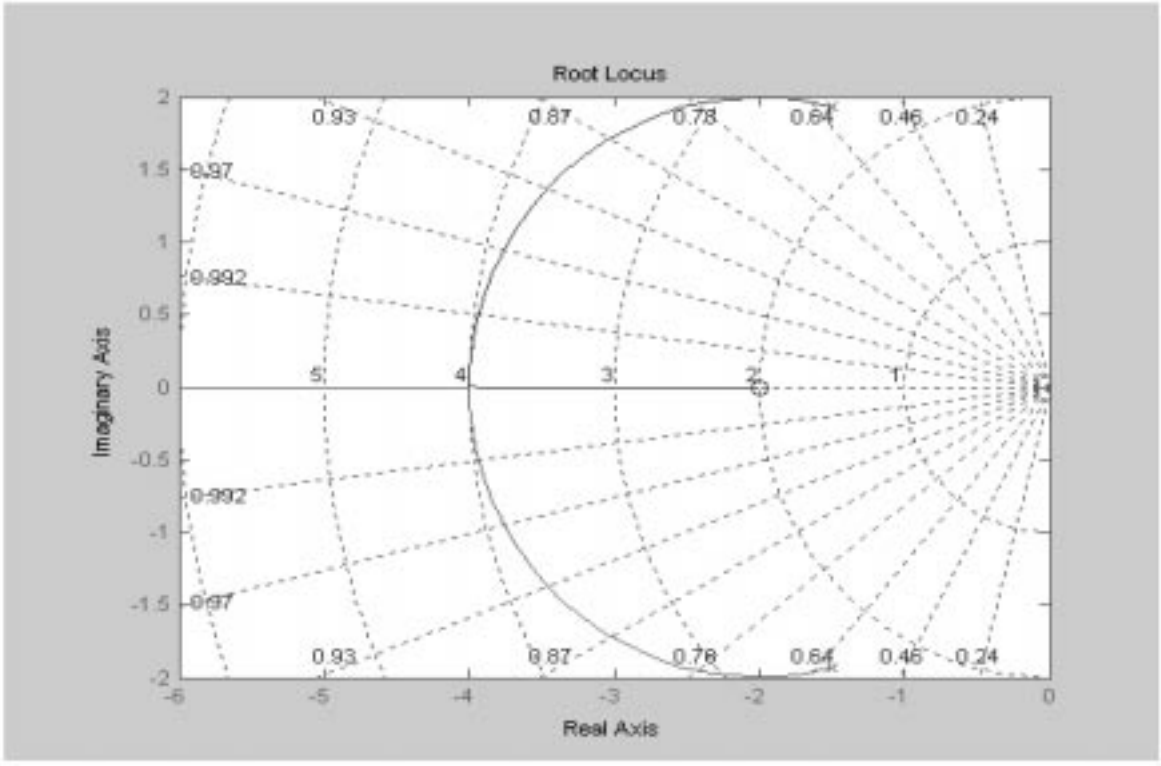

Fig. 5: The root locus using MATLAB.

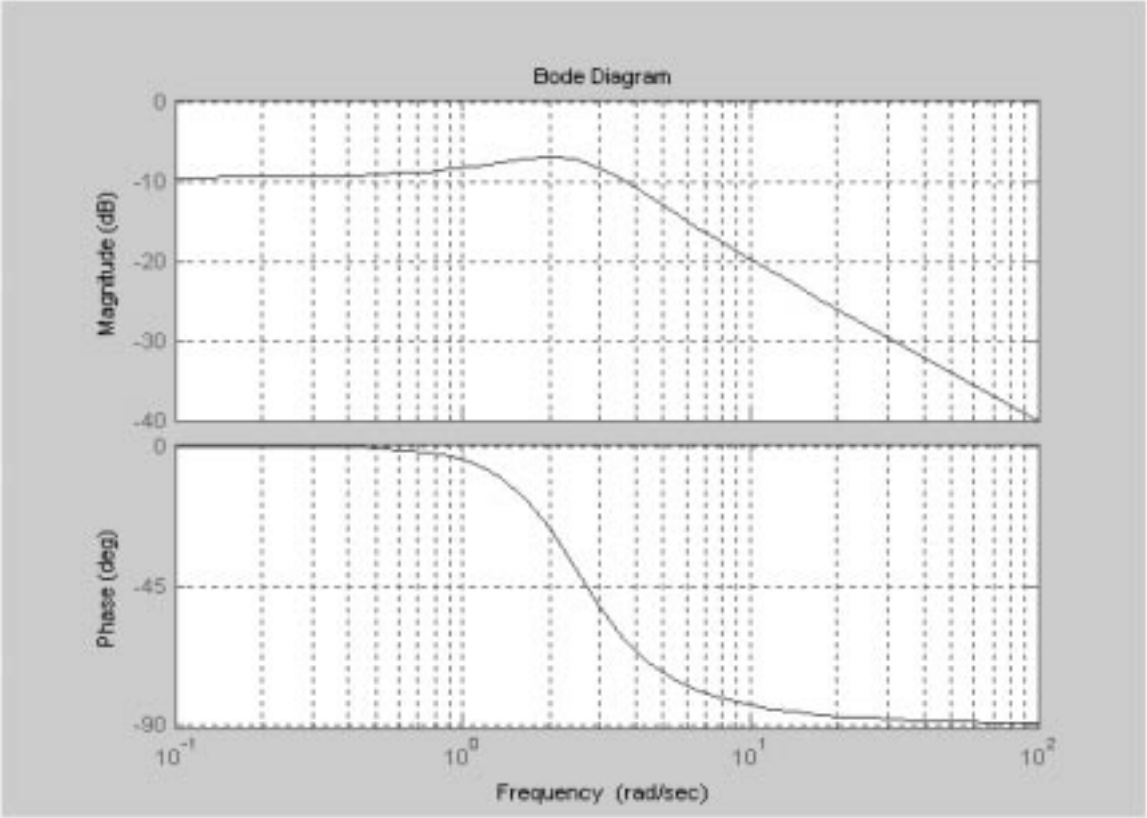

Fig. 6: The Bode diagram using MATLAB.

poles, the damping ratio, the percentage overshoot and the frequency of the system. In the case of a Bode diagram, the designer will be able to determine the gain margin and phase margin of the system as well as the gain and phase crossover frequencies.

SIMULINK FOR CONTROL SYSTEMS

Simulink is a dynamic system which is used in conjunction with MATLAB environment to simulate systems using the built-in blocks rather than writing MATLAB codes. It is user-friendly due to the block-driven interface and can interact with the MATLAB workspace. Once the user is in the MATLAB prompt, the Simulink can be invoked by entering its name as a MATLAB command:

```
>> simulink
```

This will open the Simulink library browser and provide the user with various toolboxes for specific subjects as well as the common and generally used block for all topics. The user can open a new file to construct his/her system and then chose the right block from any of the toolbox's library as well as the Simulink Extra library to build the required model for the application. The user needs only to click on the block and then drag it into the window used to build up the system. Once the model is constructed, the user is required to enter the parameters for each block. These parameters can be changed interactively during a simulation. The user is also required to enter the simulation parameters in order to be able to run the system. Simulation results can be observed during the simulation process and then exported to the MATLAB workspace for further subsequent off-line analysis.

Building a simple system using Simulink

In this section, the S-M-D control system is considered in order to demonstrate the use of Simulink in the control system. The system parameters are set as follows:

```
M = 1kg
Fv = 10 N.sec/m
k = 20 N/m
```

$$G(s) = \frac{X(s)}{F(s)} = \frac{1}{s^2 + 10s + 20} \tag{6}$$

The system can be represented using Simulink as shown in Fig. 7 and the parameters entered for the system and then run the model. The simulation results for the open-loop response for different values of $F(s)$ are obtained and recorded a show in Fig. 8.

PID controller using Simulink

The term PID stands for Proportional, Integral and Derivative controller. This type of controller is very popular and widely used in many applica-

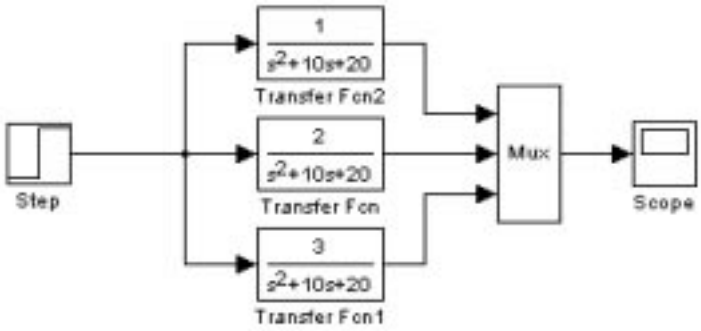

Fig. 7. The open-loop S-M-D system using Simulink.

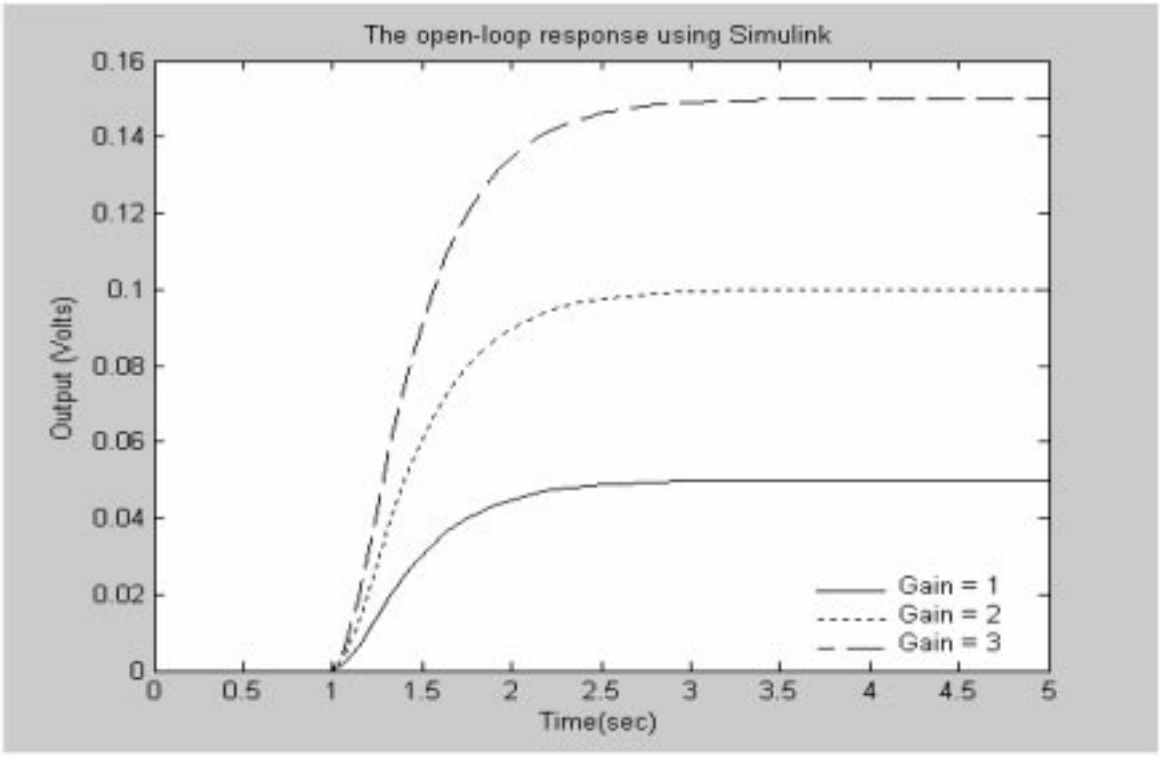

Fig. 8: The open-loop response using Simulink.

tions. The combination of the three terms gives the controller the advantages over other conventional controllers. Each term has its function and effect on the system. The proportional term, K_p, will have the effect to reduce the rise time of the output response. The integral term, K_i, is used to eliminate the steady-state error but it may affect the transient response. The derivative term, K_d, has the effect of improving the stability of the system, reducing the overshoot and improve the transient response. The transfer function of the PID controllers is:

$$G_{PID}(s) = K_P + \frac{K_I}{s} + K_D s = \frac{K_D s^2 + K_P s + K_I}{s} \quad (7)$$

The closed-loop transfer function of the S-M-D system with the as PID controller can be obtained as:

$$G_C(s) = \frac{X(s)}{F(s)} = \frac{K_D s^2 + K_P s + K_I}{s^3 + (10 + K_P)s^2 + (20 + K_P)s + K_I} \quad (8)$$

The closed-loop system with PID controller is constructed in Simulink as shown in Fig. 9. The user can set any combination of the three terms of the PID controller by setting the unwanted term equal to zero. It is important to mention that it is not necessary to implement all three terms if any simple combination provides a good and accepted output response. Simulation results for a selected setting of the PID is shown in Fig. 10.

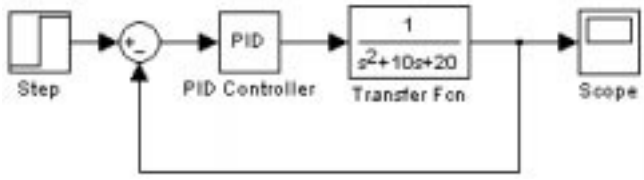

Fig. 9: PID controller for S-M-D system using Simulink

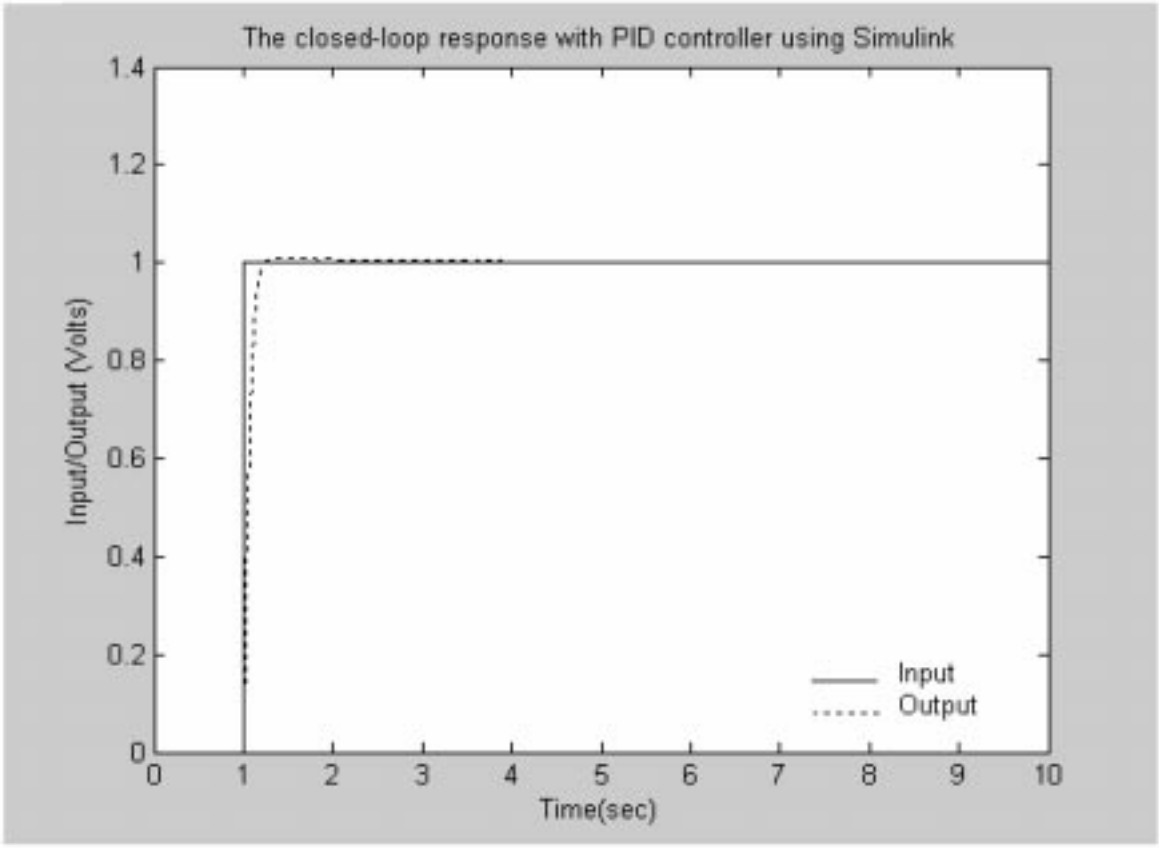

Fig. 10: The closed-loop response with PID controller using Simulink

CONCLUSION

In this paper the use of MATLAB and Simulink software package is presented to familiarize the mechatronics engineering student with the advancement of the software in studying and analyzing control systems. Firstly, some basic and introductory topics about MATLAB have been introduced. Then the mathematical modeling of a physical system is obtained for the S-M-D system using MATLAB and some output response with different system parameters for a step input have been obtained and plotted.

Furthermore, the block diagram representation of control systems has been investigated and then the characteristic and stability issues were presented. Some control techniques to study and analyze control systems such as root locus and Bode diagrams have been introduced and obtained for the S-M-D system.

Finally, the Simulink is introduced to give the user the ability to explore its advantages and construct systems with just simple knowledge about control systems. Simulink provides the user with the ability to investigate control systems and design controllers even if he/she does not command much knowledge about control systems design techniques just by playing around with the controller parameters and adjust them to achieve the desired performance.

In general, this paper presents the use of MATLAB and Simulink in a simple and direct way to study, analyze and design control systems.

REFERENCES

1. *Using MATLAB*, Mathworks, Inc., MA (1999).
2. *Using SIMULINK*, Mathworks, Inc., MA (1999).
3. R. Pratap, *Getting Started with MATLAB,* Oxford University Press (2002).
4. R. D. Dorf, and R. H. Bishop, *Modern Control Systems,* Prentice-Hall (2001).
5. Norman S. Nise, *Control Systems Engineering,* John Wiley & Sons (2000).
6. K. Ogata, *Solving Control Engineering Problems with MATLAB,* Prentice-Hall (1994).
7. R. H. Bishop, *Modern Control Systems Analysis and Design Using MATLAB,* Addison-Wesley (1993).
8. Amos Gilat, *MATLAB: An Introduction with Applications,* John Wiley & Sons (2004).

APPENDIX

The MATLAB code for the general second-order system performance `gsos.m`

```
num=input('The numerator=');% This is to provide the numerator of transfer function.
den=input('The denominator=');% This is to provide the denominator of transfer function.
Tf = tf (num,den)% This is to obtain the transfer function.
Omega = sqrt (den(3));% This is to obtain the natural frequency w_n.
Zeta = den(2) / (2*Omega);%This is to obtain the damping ratio.
Tr = (1.76*Zeta^3---0.417 * Zeta^2 + 1.039*Zeta + 1)/Omega;% The rise time.
Tp = pi/(Omega*sqrt(1---Zeta^2)); % The peak time.
Ts = 4/(Zeta*Omega); % The settling time.
Os = 100*exp(-Zeta*pi/sqrt(1-Zeta^2)); The overshoot.
sprintf('The natural frequency wn = %d',Omega) % To display the variables.
sprintf('The dampimg ratio zeta = %d',Zeta)
sprintf('The percentage overshoot = %d',Os)
sprintf('The rise time = %d',Tr)
sprintf('The peak time = %d',Tp)
sprintf('The settling time = %d',Ts)
step(Tf)% To obtain the step response of the transfer function
xlabel('Time')
ylabel ('The output y(t)')
title ('The step response for S-M-D control system')
```

The MATLAB code for the stability analysis `stability.m`

```
k = input ('The range of gain k ='); % To provide the gain of the system.
num=input('The numerator=');%This is to provide the numerator of transfer function.
den=input(The denominator='); This is to provide the denominator of transfer function.
for i= 1: length(k) % To determine the roots of the system and the range of k for stability.
 num1= num;
 den1 = den;
 det= num1+den1;
 Roots= roots (det);
 A = real (Roots);
 B = max (A);
 if B < 0
  g = k(i)
  break
 end
end
```

Abdulgani Albagul was born in Baniwalid. Libya. in 1968. He received his B.Sc. degree in Electronic Engineering from the Higher Institute of Electronics, Baniwalid in 1989, M.Sc. in Control Engineering from University of Bradford in 1993 and Ph.D. in Electrical Engineering from University of Newcastle upon Tyne in 2001. He is currently an Assistant Professor at the Department of Mechatronics Engineering, Faculty of Engineering, International Islamic University, Malaysia. His research interests are control systems, system dynamics and modeling, smart sensors and instrumentation, robotics and automation. He is a member of IEEE and has several publications.

Wahyudi was born in Indonesia in July 1970. He received the B.Eng. and M.Sc. degrees in mechanical engineering from Institute of Technology Bandung, Indonesia, in 1994 and 1997, respectively and the Ph.D. degree in precision machinery systems from Tokyo Institute of Technology, Tokyo, Japan, in 2002. He is currently an assistant professor at the Mechatronics Engineering Department of the International Islamic University, Malaysia. His research fields are motion control, mechatronics and intelligent systems. Dr. Wahyudi is a member of IEEE.

Othman Omran Khalifa received his Bachelor's degree in Electronic Engineering from the Garyounis University, Libya, in 1986. He obtained his Master degree in Electronics Science Engineering and Ph.D. in Digital Image Processing from Newcastle University, UK, in 1996 and 2000 respectively. He worked in industry for eight years and he is currently an Assistant Professor at the department of Electrical and Computer Engineering, International Islamic University, Malaysia. His area of research is communications, information theory and coding, digital image/video processing, coding and compression, wavelets, fractal and pattern recognition. He is an IEEE member, IEEE Computer, Image Processing and Communication Society member.