CLOUD SIMULATION OF A FLEXIBLE MANIPULATOR SYSTEM

Prof. Abul K. M. Azad, Northern Illinois University

Abul K. M. Azad is a Professor in the Technology Department of Northern Illinois University. He has a Ph.D. in Control and Systems Engineering and M.Sc. and B.Sc. in Electronics Engineering. His research interests include remote laboratories, mechatronic systems, mobile robotics, and educational research. In these areas, Dr. Azad has over 100 refereed journal and conference papers, edited books, and book chapters. So far, he has attracted around $1.7 million in research and development grants from various national and international funding agencies. He serves on editorial boards of a number of professional journals and is Editor-in-Chief of the International Journal of Online Engineering. He is active with various professional organizations (IEEE, IET, ASEE, and ISA) as well as a member of board of Trustees of CLAWAR Association.

Dr. Reza Hashemian P.E., Northern Illinois University

Reza Hashemian: Currently at Northern Illinois University; received his B.S. from Tehran University, Iran in 1960, and his M.S. and Ph.D. from the University of Wisconsin, Madison, in 1965 and 1968, all in Electrical Engineering. He has served as the Director of Research and Industrial Cooperation at Sharif University of Technology (1982 – 84), and one of the founders, associate director and director of the Materials and Energy Research Center (MERC) in Tehran, Iran (1972 – 1979). He is a life member of the IEEE and a member of the first executive committee of the IEEE in Iran from1970 to1972.

Mr. Mohd Afwaan Ahmed Quadri,

Afwaan Ahmed is a Embedded engineer consultant. He is working on various embedded system platforms with an application to Internet of Things. He also works on various networking technologies involving embedded systems. His research and development interest also involves cloud computing, VLSI and signal processing. Afwaan received his Master of science degree in Electrical Engineering from Northern Illinois University. His Master's thesis was on cloud simulation of flexible manipulator systems and the outcome of his project has been published as a technical paper.

Cloud Simulation of a Flexible Manipulator System

Abstract: This paper reports the development of a cloud simulation environment for a single link flexible manipulator system, where users can perform a simulation exercise from a remote location via a graphical user interface (GUI). The cloud simulation is an arrangement where simulation runs on a server and can be accessed by the users from remote locations. Within the developed environment the user selects desired system specifications via the GUI and passes them to the hosting server for the simulation to be performed at the server. The simulation results are subsequently presented to the remote user via the GUI. This paper details the technical development process and highlights its advantages and shortcomings. A number of case studies are also provided to demonstrate the potential of this environment for educational activities.

1. Introduction

Simulation is a powerful method of studying the behavior and functionality of engineering systems. With the advancement of Internet and computing technology cloud simulation is becoming more popular. Cloud simulation is an arrangement in which the simulation environment is hosted on a remote server and users have access to the simulation environment over the web. A detailed review of web-based simulation is provided by Bryne, et.al. [1]. The difficulty of traditional stand-alone system simulation is that the simulation environment is hosted on a PC or workstation. Initial deployment and any new addition or update to the simulation software has to be implemented on each computer, which requires time and effort to keep the simulation system current. Cloud simulation is an approach to address this issue effectively and efficiently. As cloud simulation runs on a central server, all the updates can be implemented on the server itself, avoiding the time consuming and labor intensive process.

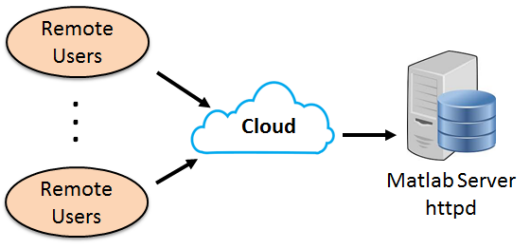


Figure 1: Basic web server configuration.

There are a number of software tools available for cloud simulation and they are described by many researchers, some of these are Mathematica, LabVIEW, and MATLAB [2, 3]. Mathematica is a computational software based on symbolic mathematics and is used in many scientific, engineering, mathematical, and computing fields [4]. LabVIEW is a system design platform and development environment for a visual programming language from National Instruments. LabVIEW is commonly used for data acquisition, instrument control, and industrial automation on a variety of platforms [5]. Support for web services was introduced in LabVIEW 8.6. The web services runtime engine is run by LabVIEW's built-in web server.

The application web server (AWS) is introduced in LabVIEW 2010. Unlike earlier versions of the LabVIEW web server, AWS does not require LabVIEW to be running on the server. The MATLAB web server enables creation of MATLAB applications that use the capabilities of the world wide web to send data to MATLAB for computation and to display the results through a web browser [6]. In the simplest configuration, a web browser runs on a client's workstation, while MATLAB, the MATLAB web server (MATLAB server), and the web server daemon (http) run on another machine. Figure 1 shows the basic configuration of web server application.

This paper describes the development of a cloud based simulation environment for a single link flexible manipulator system described by Tokhi et al. [7]. Main thrust of this paper is to present only the technical aspect of the development, rather than a classroom integration. However, the environment can easily be utilized for classroom use, homework, as well as a support tool for laboratory experiments. The next section of the paper will illustrate an outline of the flexible manipulator system and its representative mathematical equation along with its boundary conditions. Section 3 shows the Finite Difference (FD) simulation algorithm development process that will be implemented for cloud simulation. Section 4 will demonstrate the cloud simulation development using the MATLAB. Outcomes from some of the cloud simulation exercise will be presented in Section 5. Finally, the conclusion section summarizes the findings.

2. Flexible Manipulator System

A schematic model of the flexible manipulator system considered for the development is shown in Figure 2, where X_oOY_o and XOY represent the stationary and moving co-ordinate frames, respectively. The axis OX coincides with the neutral line of the link in its undeformed configuration and is tangent to it at the clamped end in a deformed configuration. τ represents the applied torque at the hub. E, I, ρ, A, I_h and M_p represent the Young's modulus, area moment of inertia, mass density per unit volume, cross sectional area, hub inertia and payload of the manipulator, respectively. $\theta(t)$ denotes an angular displacement (hub-angle) of the manipulator and $w(x,t)$ denotes an elastic deflection (deformation) of a point along the manipulator at a distance x from the hub of the manipulator.

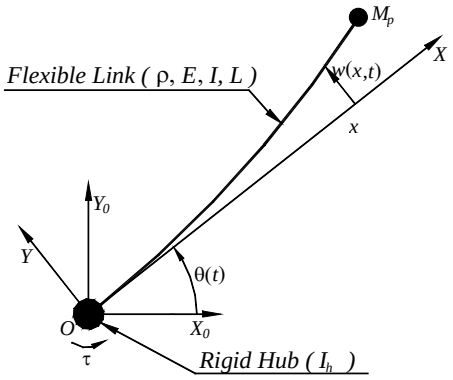


Figure 2: Schematic model of a flexible manipulator system.

In this work, the motion of the manipulator is confined to the X_oOY_o plane. The manipulator is assumed to be stiff in vertical bending and torsion, allowing it to vibrate dominantly in the horizontal direction, and thus, the gravity effects are neglected. Moreover, the manipulator is considered to have constant cross section and uniform material properties throughout.

The dynamic equation describing the motion of the flexible manipulator is given below. In this equation, the motion of the manipulator in presented in terms of $y(x,t)$.

$$EI\frac{\partial^4 y(x,t)}{\partial x^4}+\rho\frac{\partial^2 y(x,t)}{\partial t^2}-D_S\frac{\partial^3 y(x,t)}{\partial x^2\partial t}=\tau(t) \tag{1}$$

with the corresponding boundary conditions as

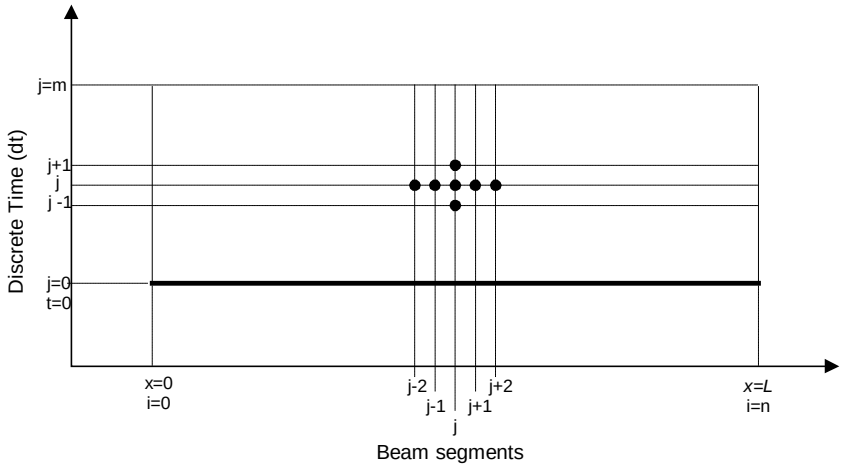
$$\begin{aligned} y(0,t) &= 0 \\ I_h\frac{\partial^3 y(0,t)}{\partial x\partial t^2}-EI\frac{\partial^2 y(0,t)}{\partial x^2} &= \tau(t) \\ M_p\frac{\partial^2 y(l,t)}{\partial t^2}-EI\frac{\partial^3 y(l,t)}{\partial x^3} &= 0 \\ EI\frac{\partial^2 y(l,t)}{\partial x^2} &= 0 \end{aligned} \tag{2}$$

and initial conditions as

$$y(x,0)=0,\quad \frac{\partial y(x,0)}{\partial x}=0 \tag{3}$$

3. Algorithm Development

The FD approach can be utilized to obtain a numerical method of solving Partial Differential Equations (PDE) by developing a finite-dimensional simulation of a single link flexible manipulator system through a discretization, both in time and space (distance) coordinates.



Boundary conditions at the hub

$$y(0,t)=0$$
$$I_h\frac{\partial^4 y(0,t)}{\partial x\partial t^2}-EI\frac{\partial^2 y(0,t)}{\partial x^2}=\tau(t)$$

Initial conditions

$$y(x,0)=0$$
$$\frac{\partial y(x,0)}{\partial x}=0$$

Boundary conditions at the end-point

$$M_p\frac{\partial^2 y(l,t)}{\partial t^2}-EI\frac{\partial^3 y(l,t)}{\partial x^3}=0$$
$$EI\frac{\partial^2 y(l,t)}{\partial x^2}=0$$

Figure 3: Finite difference discretization in time and space variables.

The algorithm allows the inclusion of distributed actuator and sensor terms in the PDE and modification of boundary conditions. The development of such an algorithm for a system with no damping has been previously reported [8].

The dynamic equation describing the motion of the flexible manipulator can be presented by a fourth-order PDE as shown in equation (1). The FD method is used to solve this equation and to develop a suitable simulation environment characterizing the behavior of the system. To start, a set of equivalent difference equations defined by the central finite difference quotients of the FD method are obtained by discretizing the PDE (1) along with its associated boundary and initial conditions in equations (2) and (3). As shown in Figure 3, the process involves dividing the manipulator into *n* sections, each of length Δx, and considering the deflection of each section at sample times, Δt. In this manner, a solution of the PDE is obtained by generating the central difference formulae for the partial derivative terms of the response $y(x,t)$ of the manipulator at points $x = i.\Delta x$, $t = j.\Delta t$ [8]. The difference equations are provided in equation 4.

$$
\begin{aligned}
\frac{\partial^2 y(x,t)}{\partial t^2} &= \frac{y_{i,j+1} - 2y_{i,j} + y_{i,j-1}}{\Delta t^2} \\
\frac{\partial^2 y(x,t)}{\partial x^2} &= \frac{y_{i+1,j} - 2y_{i,j} + y_{i-1,j}}{\Delta x^2} \\
\frac{\partial^3 y(x,t)}{\partial x^3} &= \frac{y_{i+2,j} - 2y_{i+1,j} + 2y_{i-1,j} - y_{i-2,j}}{2\Delta x^3} \\
\frac{\partial^4 y(x,t)}{\partial x^4} &= \frac{y_{i+2,j} - 4y_{i+1,j} + 6y_{i,j} - 4y_{i-1,j} + y_{i-2,j}}{\Delta x^4} \\
\frac{\partial^3 y(x,t)}{\partial t^2 \partial x} &= \frac{y_{i,j+1} - 2y_{i,j} + y_{i,j-1} - y_{i-1,j+1} + 2y_{i-1,j} - y_{i-1,j-1}}{\Delta x \Delta t^2} \\
\frac{\partial^3 y(x,t)}{\partial x^2 \partial t} &= \frac{y_{i+1,j} - 2y_{i,j} + y_{i-1,j} - y_{i+1,j-1} + 2y_{i,j-1} - y_{i-1,j-1}}{\Delta t \Delta x^2}
\end{aligned}
\qquad (4)
$$

where, $y_{i,j}$ represents the response $y(x,t)$ at $x = i\Delta x$ and $t = j\Delta t$ or $y(x_i, t_j)$. Note that a time-space discretization is adopted in the evaluation of the response of the manipulator.

A solution of the PDE in equation (1) can be obtained by substituting for $\frac{\partial^2 y}{\partial t^2}, \frac{\partial^4 y}{\partial x^4}$ and $\frac{\partial^3 y}{\partial x^2 \partial t}$ from equation (4) and simplifying to yield

$$
\begin{aligned}
&\frac{EI}{\Delta x^4}\left[y_{i+2,j} - 4y_{i+1,j} + 6y_{i,j} - 4y_{i-1,j} + y_{i-2,j}\right] + \frac{\rho}{\Delta t^2}\left[y_{i,j+1} - 2y_{i,j} + y_{i,j-1}\right] \\
&\quad - \frac{D_S}{\Delta x^2 \Delta t}\left[y_{i+1,j} - 2y_{i,j} + y_{i-1,j} - y_{i+1,j-1} + 2y_{i,j-1} - y_{i-1,j-1}\right] = 0
\end{aligned}
$$

or

$$
\begin{aligned}
y_{i,j+1} = &- c\left[y_{i+2,j} + y_{i-2,j}\right] + b\left[y_{i+1,j} + y_{i-1,j}\right] + a\, y_{i,j} - y_{i,j-1} \\
&+ d\left[y_{i+1,j} - 2\, y_{i,j} + y_{i-1,j} - y_{i+1,j-1} + 2\, y_{i,j-1} - y_{i-1,j-1}\right]
\end{aligned}
\qquad (5)
$$

$$
\text{Where, } a = 2 - \frac{6EI\Delta t^2}{\rho \Delta x^4};\ b = \frac{4EI\Delta t^2}{\rho \Delta x^4};\ c = \frac{EI\Delta t^2}{\rho \Delta x^4};\ d = \frac{D_S \Delta t}{\rho \Delta x^2}.
$$

Equation (5) provides the displacement of section ***i*** of the manipulator at time step $j+1$. It follows from this equation that, to obtain the displacements $y_{n-1,j+1}$ and $y_{n,j+1}$, displacements of

the fictitious points $y_{n+2,j}$, $y_{n+1,j}$ and $y_{n+1,j-1}$ are required. These can be obtained using the boundary conditions related to the dynamic equation of the flexible manipulator. Using a similar manner, the discrete form of the corresponding boundary conditions can be obtained:

$$y_{0,j} = 0 \tag{6}$$

$$y_{-1,j} = y_{1,j} + \frac{\Delta x I_h}{EI\Delta t^2}\left[y_{1,j+1} - 2y_{1,j} + y_{1,j-1}\right] + \frac{\Delta x^2}{EI}\tau(j) \tag{7}$$

$$y_{n+2,j} = 2y_{n+1,j} - 2y_{n-1,j} + y_{n-2,j} + \frac{2\Delta x^3 M_p}{\Delta t^2 EI}\left[y_{n,j+1} - 2y_{n,j} + y_{n,j-1}\right] \tag{8}$$

$$y_{n+1,j} = 2y_{n,j} - y_{n-1,j} \tag{9}$$

3.1 Matrix Formulation

Using matrix notation, equation (5) can be written in a compact form as

$$\mathbf{Y}_{i,j+1} = \mathbf{A}\mathbf{Y}_{i,j} + \mathbf{B}\mathbf{Y}_{i,j-1} + \mathbf{C}\mathbf{F} \tag{10}$$

Where $\mathbf{Y}_{i,j+1}$ is the displacement of grid points $i = 1,2,\cdots,n$ of the manipulator at time step $j+1$, $\mathbf{Y}_{i,j}$ and $\mathbf{Y}_{i,j-1}$ is the corresponding displacement at time steps j and $j-1$, respectively. **A** and **B** are constant $n \times n$ matrices whose entries depend on the flexible manipulator specification and the number of sections the manipulator is divided into, **C** is a constant matrix related to the given input torque, and **F** is an $n \times 1$ matrix related to the time step Δt and mass per unit length of the flexible manipulator;

$$\mathbf{Y}_{i,j+1} = \begin{bmatrix} y_{1,j+1} \\ y_{2,j+1} \\ \vdots \\ y_{n,j+1} \end{bmatrix}, \quad \mathbf{Y}_{i,j} = \begin{bmatrix} y_{1,j} \\ y_{2,j} \\ \vdots \\ y_{n,j} \end{bmatrix}, \quad \mathbf{Y}_{i,j-1} = \begin{bmatrix} y_{1,j-1} \\ y_{2,j-1} \\ \vdots \\ y_{n,j-1} \end{bmatrix} \tag{11}$$

$$\mathbf{A} = \begin{bmatrix} K_1 & K_2 & K_3 & 0 & 0 & \cdots & 0 & 0 \\ (b+d) & (a-2d) & (b+d) & -c & 0 & \cdots & 0 & 0 \\ -c & (b+d) & (a-2d) & (b+d) & -c & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -c & b+d & a-2d & b+d & -c \\ 0 & 0 & \cdots & 0 & K_7 & K_8 & K_9 & K_{10} \\ 0 & 0 & \cdots & 0 & 0 & K_{14} & K_{15} & K_{16} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} K_4 & K_5 & 0 & 0 & 0 & \cdots & 0 & 0 \\ -d & 2d-1 & -d & 0 & 0 & \cdots & 0 & 0 \\ 0 & -d & 2d-1 & -d & 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & -d & 2d-1 & -d & 0 \\ 0 & 0 & \cdots & 0 & 0 & K_{11} & K_{12} & K_{13} \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & K_{17} \end{bmatrix}$$

$$\mathbf{C} = \tau(j), \qquad \mathbf{F} = \begin{bmatrix} K_6 & 0 & \cdots & 0 \end{bmatrix}^T$$

3.2 State-Space Formulation

A state-space formulation of the dynamic equation of the manipulator can be constructed by referring to the matrix formulation. Using the notation for simulation of discrete-time linear systems, the dynamic equations of the flexible manipulator can be written as:

$$\mathbf{P}=\left[\begin{array}{c|c} \mathbf{A} & \mathbf{B} \\ \hline I_{\mathrm{NxN}} & 0_{\mathrm{NxN}} \end{array}\right],\quad \mathbf{Q}=\left[\begin{array}{c} \mathbf{C} \\ \hline 0_{\mathrm{N\times 1}} \end{array}\right],\quad \mathbf{R}=\left[I_{\mathrm{N}} \quad 0_{\mathrm{N}}\right],\quad \mathbf{S}=\left[0_{2\mathrm{N}}\right] \tag{12}$$

$$u=\left[\tau \quad 0 \quad \cdots \quad 0\right]^{\mathrm{T}},\quad y(n)=\left[x(1,n)\cdots x(N,n),\, x(1,n-1)\cdots x(N,n-1)\right]$$

Note that *N* represents the number of sections.

4. Simulation

The aim of this project is to develop a cloud environment for simulation of a single link flexible manipulator system.

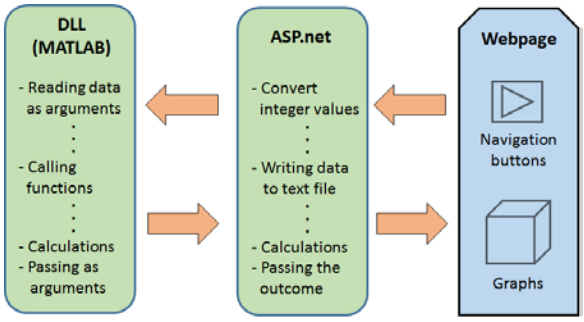


Figure 4: Implementation block diagram.

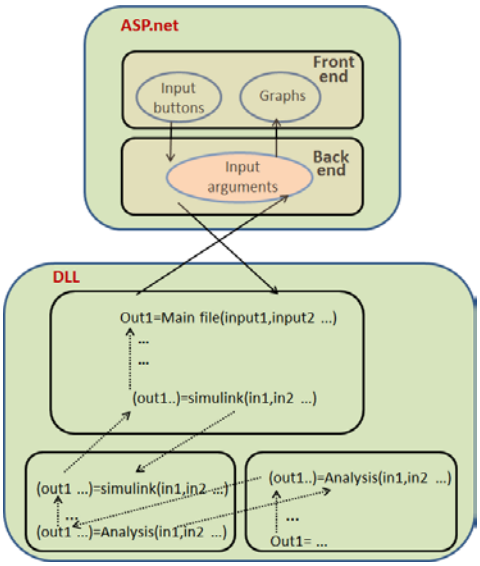


Figure 5: Interaction between the ASP.net and DLL.

The process involves preparing MATLAB m-files for algorithm implementation and converting the m-files to a dynamic link library (DLL) using MATLAB Compiler SDK, ASP.net application to visualize the results. A block diagram showing the process is provided in Figure 4, and Figure 5 shows the interaction between the ASP.net and DLL.

4.1 MATLAB Development

The developed FD algorithm for a single link flexible manipulator system is coded in MATLAB as m-files. A diagram showing the structure of the MATLAB programming is shown in Figure 6. All the user defined specifications, like material properties and simulation parameters, are provided as inputs in the form of arguments to the MATLAB main function. This main function calls a number of sub-functions at various stages. The sub-functions are: Simulink block coded function, generation of output function, and data scaling function. MATLAB Compiler SDK is used to convert the developed m-files to DLL files.

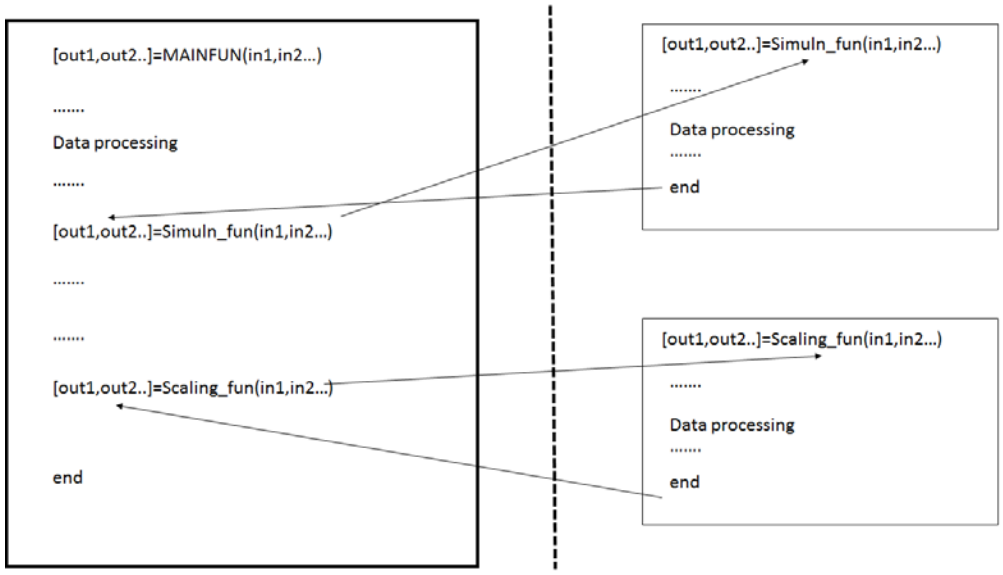


Figure 6: MATLAB program structure.

4.2 Microsoft Visual Studio .net Assembly (ASP.net)

MATLAB Compiler SDK is used for converting the MATLAB library to a .net Assembly and C# library. The data are passed directly from MATLAB to a .net framework. When a client interacts with the web page, the requests are passed to the MATLAB (server end) via the ASP.net. MATLAB then performs the calculations as requested and passes the output to the ASP.net. The ASP.net in turn plots the output graphs for the remote users within the webpage. Figure 7 illustrates the implementation process.

The first and foremost step in a .net framework is to reference the DLL file from the redistribution folder, which is created by the MATLAB Compiler SDK in the previous step. The next important step is to add another reference called MW Array that is required by Visual Studio to interpret the output data from MATLAB.

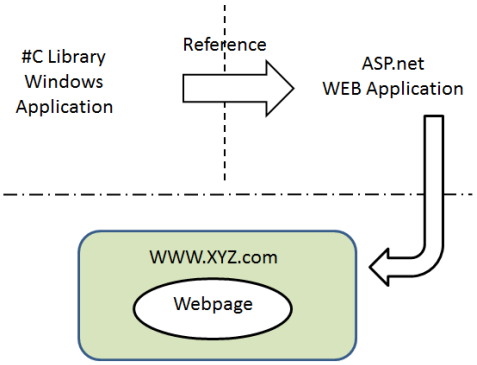


Figure 7: Microsoft Visual Studio implementation.

Figure 8: Home page for the simulation environment.

5. Results

This section demonstrates the simulation results in the form of web pages, images, and graphs that remote users will visualize. The strategy described in this paper is used for implementing the FD and finite element (FE) simulation algorithms as well as the open loop response behavior of a single link flexible manipulator system with various loading conditions. Figure 8 shows the home page for the environment in which options are provided for various simulation choices. The choices are 'Finite Difference Simulation', Finite Element Simulation', and 'Open Loop Control.

In this section, only the FD simulation outcomes are provided. Figure 9 shows the landing page for FD simulation. In this interactive page, users provide a desired system specification in terms of physical size, material properties, excitation input type, and desired output as well as a few timing inputs.

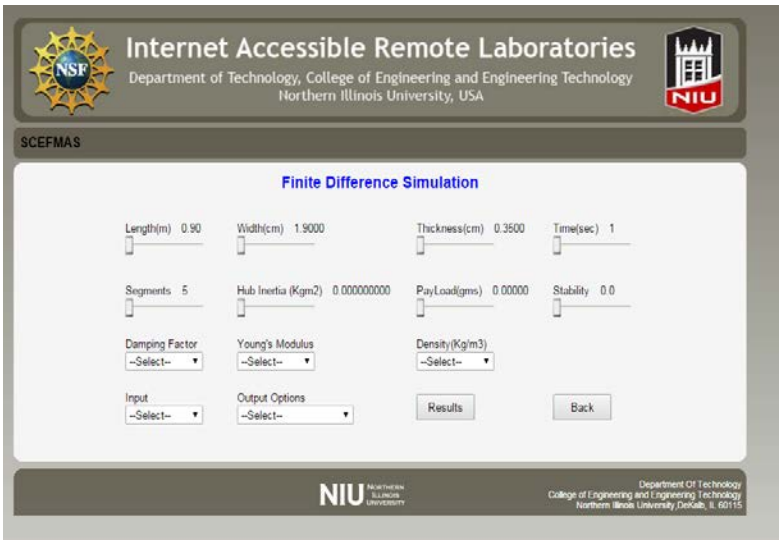
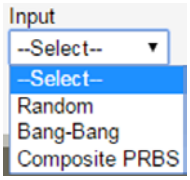


Figure 9: User input for system configuration page.

There are three input types and seven output choices (Figure 10 and Figure 11). Once finished with entering all of the choice options, the user clicks on the 'Results' button. This in turn passes the provided data to the server over the web. The MATLAB script on the server processes these and passes the output results to the website for user visualization.



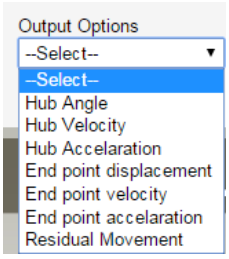


Figure 10: Input options.

Figure 11: Output choices.

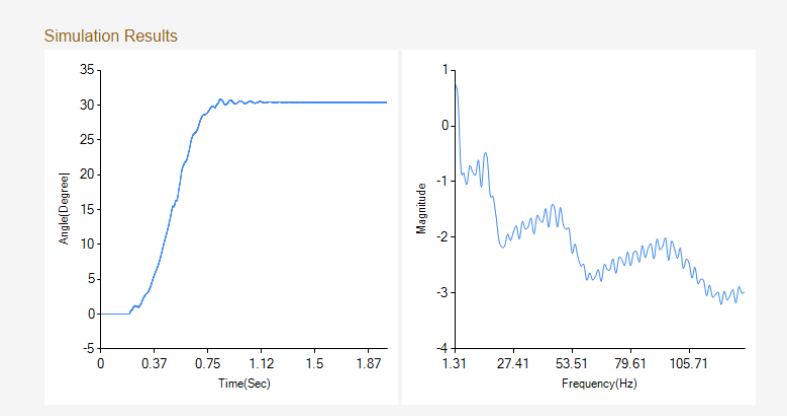


Figure 12: Hub angle without payload.

The simulation results are presented for the hub angle, hub velocity, end-point displacement, and end-point velocity, both in the time and frequency domains. Figures 12 through 15 show the output graphs without any payload, while Figures 16 through 19 show the output graphs with a 20gm of payload.

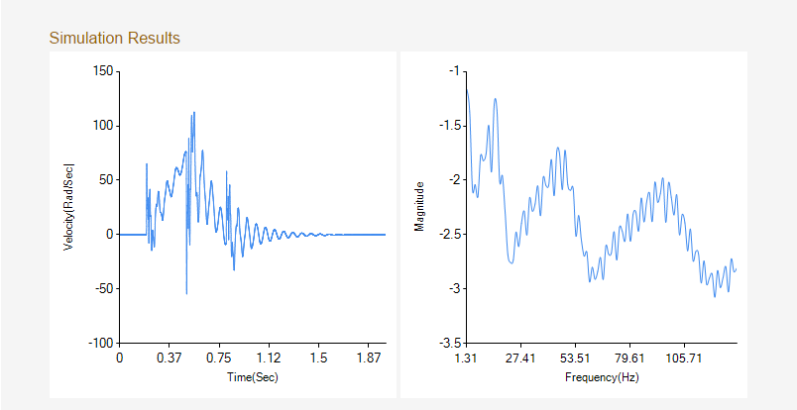


Figure 13: Hub velocity without payload.

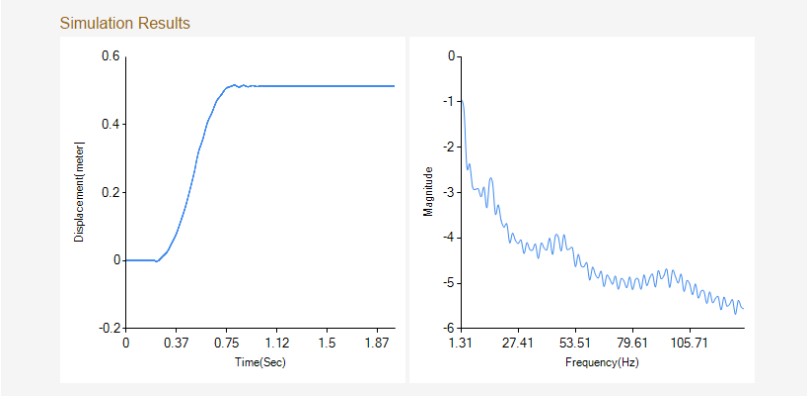


Figure 14: End-point displacement without payload.

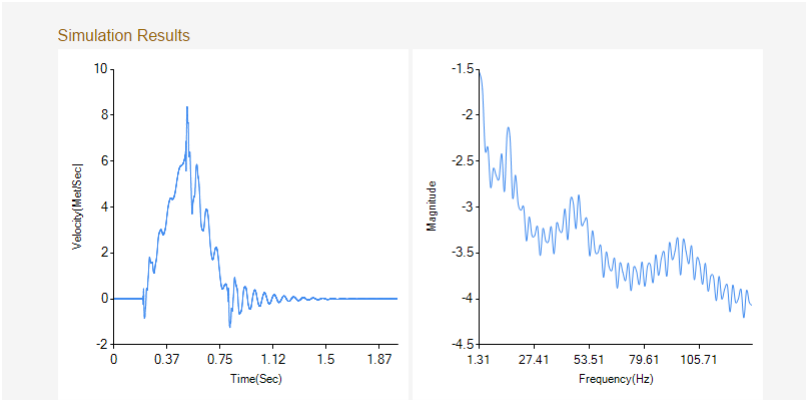


Figure 15: End-point velocity without payload.

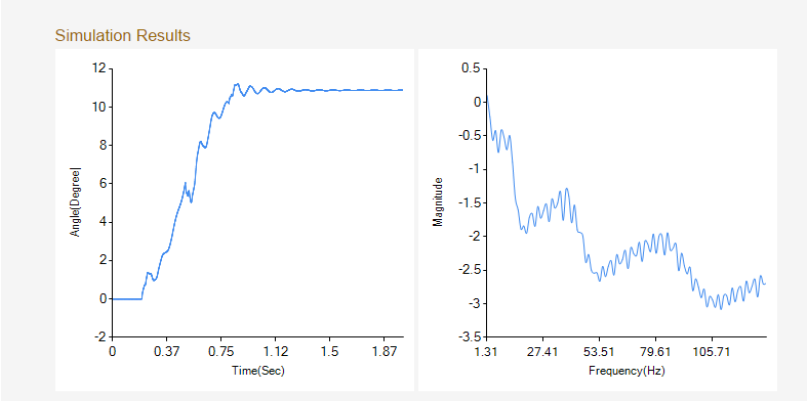


Figure 16: Hub angle with payload (20g).

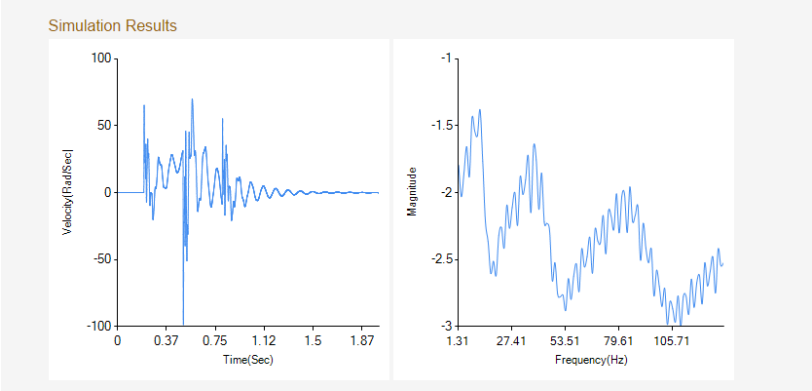


Figure 17: Hub velocity with payload (20g).

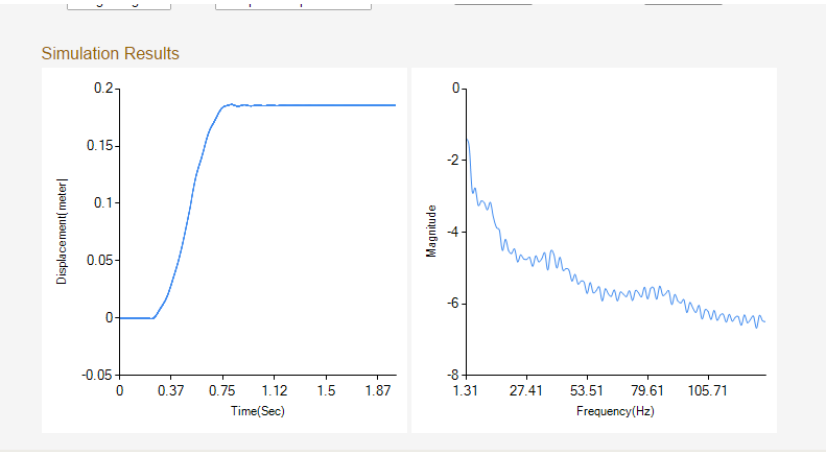


Figure 18: End-point displacement with payload (20g).

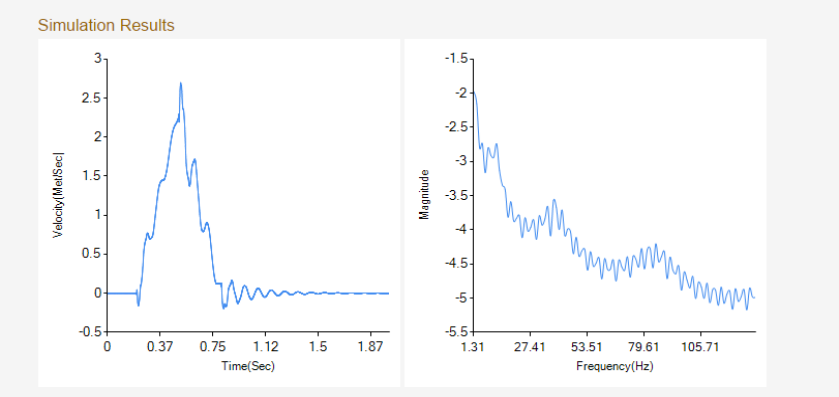


Figure 19: End-point velocity with payload (20g).

6. Conclusions

This paper illustrates the development of a cloud based simulation environment for a single link flexible manipulator system that can be used for educational and research purposes. The first part of the paper provides the mathematical description of the system and then uses a FD approach for simulation algorithm development. MATLAB is used for core simulation work, while MATLAB Compiler SDK is used for producing DLL files for interacting with the ASP.net. Finally the ASP.net is used for interaction and visualization over the web. Main thrust of this paper is to present only the technical aspect of the development, rather than a classroom integration. However, the environment can easily be utilized for classroom use, homework, as well as a support tool for laboratory experiments.

Acknowledgements: The authors would like to thank the NSF for its support for the reported work [NSF TUES project, award number DUE-1140502].

References

1. Heavey, C. and Byrne, P.J. (2010). A review of Web-based simulation and supporting tools, *Simulation Modelling Practice and Theory*, **18**, pp. 253-276.
2. Fortmann-Roe, S. (2014). Insight Maker: A general-purpose tool for web-based modeling & simulation, *Simulation Modelling Practice and Theory*, **47**, pp. 28–45.
3. Casalicchio, E., Lancellotti, R., and Poleggi, M.E. (2007). A Simulation Framework for Cluster-Based Web Services, *International Journal of Simulation*, **8**(4), pp. 21-33.
4. Wolfram Mathematica, http://www.wolfram.com/mathematica/
5. LabVIEW Web Services White Paper, http://www.ni.com/white-paper/7747/en/
6. Cheng, Y. P. and Brutzman, D. (2015). Matlab and Simulink creation and animation of X3D in web-based simulation, *Proceeding of the 20th International Conference on 3D Web Technology*, pp. 169-169, New York, US.
7. Tokhi, M. O. and Azad, A.K.M. (1995). Real-time finite difference simulation of a single-link flexible manipulator system incorporating hub inertia and payload, *Proc Institution of Mechanical Engineers, Part 1: Journal of Systems and Control Engineering*, **209**(I1), pp. 21-33.
8. Azad, A.K.M. (1994). *Analysis and design of control mechanism for flexible manipulator systems*, PhD. Thesis, Department of Automatic Control and Systems Engineering, The University of Sheffield, UK.