White Rose Research Online

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk
Universities of Leeds, Sheffield and York

Deposited via The University of Sheffield.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/id/eprint/79841/

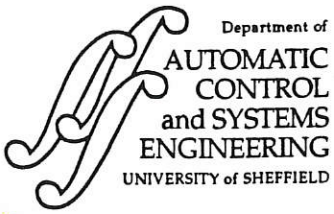
**Monograph:**
Tokhi, M.O. and Azad, A.K.M. (1994) Real-Time Finite Difference Simulation of a Single-Link Flexible Manipulator System. UNSPECIFIED. ACSE Research Report 541 . Department of Automatic Control and Systems Engineering

**Reuse**
Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

**Takedown**
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

UNIVERSITY OF LEEDS
University of Sheffield
UNIVERSITY of York

629.8 (S)




# REAL-TIME FINITE DIFFERENCE SIMULATION OF A SINGLE-LINK FLEXIBLE MANIPULATOR SYSTEM

**M O Tokhi and A K M Azad**

Department of Automatic Control and Systems Engineering, The University of Sheffield,
P O Box 600, Mappin Street, Sheffield, S1 4DU, UK.

Tel: (0742) 825136.
Fax: (0742) 731 729.
E-mail: O.Tokhi@sheffield.ac.uk.

Research Report No. 541.

October 1994.

## Abstract

This paper presents an investigation into the real-time simulation of a flexible manipulator system. A single-link flexible manipulator incorporating hub inertia and payload mass which can bend freely in the horizontal plane and is stiff in vertical bending and torsion is considered. A fourth-order partial differential equation (PDE) model of the system obtained through the utilisation of Lagrange equation and modal expansion method is considered. A finite-dimensional simulation of the flexible manipulator system is developed using a finite difference discretisation of the dynamic equation of motion of the manipulator. The algorithm proposed allows the inclusion of distributed actuator and sensor terms in the PDE and modification of boundary conditions. The real-time processing requirements of the algorithm are investigated by implementing the algorithm on several digital processing domains. Finally, simulation results verifying the performance of the algorithm in characterising the behaviour of the manipulator under various loading conditions are presented and discussed.

*Key words: Flexible manipulator, finite difference method, real-time simulation.*



SHEFFIELD UNIV. LIBRARY ST. GEORGE'S

# CONTENTS


| | | |
|---|---|---|
| Title | | i |
| Abstract | | ii |
| Contents | | iii |
| 1 | Introduction | 1 |
| 2 | The flexible manipulator system | 2 |
| 3 | The finite difference method | 4 |
| 4 | Algorithm development | 6 |
| | 4.1 The system without hub inertia and payload mass | 7 |
| | 4.2 The system with hub inertia and payload | 10 |
| 5 | Algorithm stability | 14 |
| 6 | Algorithm implementation | 16 |
| | 6.1 Hardware and software facilities | 17 |
| | 6.2 Real-time processing requirements | 19 |
| 7 | Simulation results | 21 |
| 8 | Conclusion | 23 |
| 9 | References | 24 |

# 1 Introduction

Flexible manipulator systems offer several advantages in contrast to the traditional rigid ones. These include faster system response, lower energy consumption, requiring relatively smaller actuators, reduced nonlinearity due to elimination of gearing, less overall mass and, in general, less overall cost. However, due to the distributed nature of the governing equations describing system dynamics, the control of flexible manipulators has traditionally involved complex processes (Aubrun, 1980; Balas, 1978; Omatu and Seinfeld, 1986). Moreover, to compensate for flexure effects and thus yield robust control the design focuses primarily on noncolocated controllers (Cannon and Schmitz, 1984; Harishima and Ueshiba, 1986). It is important initially to recognise the flexible nature of the manipulator and construct a mathematical model for the system that accounts for the interactions with actuators and payload. Such a model can be constructed using partial differential equations. A commonly used approach for solving a partial differential equation (PDE) representing the dynamics of a manipulator, sometimes referred to as the separation of variables method, is to utilise a representation of the PDE, obtained through a simplification process, by a finite set of ordinary differential equations. Such a model, however, does not always represent the fine details of the system (Hughes, 1987). A method in which the flexible manipulator is modelled as a massless spring with a lumped mass at one end and lumped rotary inertia at the other end has previously been proposed (Oosting and Dickerson, 1988; Feliu et al., 1992). Unfortunately, the solution obtained through this method is also not accurate and suffers from similar problems as in the case of the separation of variables method. The finite element (FE) method has also been previously utilised to describe the flexible behaviour of manipulators (Usoro et al., 1984; Dado and Soni, 1986). The computational complexity and consequent software coding involved in the FE method is a major disadvantage of this technique specially in real-time applications. However, as the FE method allows irregularities in the structure and mixed boundary conditions to be handled, the technique is found suitable in applications involving irregular structures. In applications involving uniform structures, such as the manipulator

system considered here, the finite difference (FD) method is found to be more appropriate. Previous simulation studies of flexible beam systems have demonstrated the relative simplicity of the FD method as compared to the FE method (Kourmoulis, 1990). Moreover, the relatively reduced amount of computation involved in the FD method makes the technique suitable in real-time applications. Real-time simulation is important from a system design and evaluation viewpoint. It provides a characterisation of the system in the real sense as well as allows on-line evaluation of controller designs in real-time. An investigation into the real-time simulation of a flexible manipulator system is thus presented in this paper on the basis of FD methods.

The FD method is used to obtain an efficient numerical method of solving the PDE by developing a finite-dimensional simulation of the flexible manipulator system through a discretisation, both, in time and space (distance) coordinates. The algorithm proposed here allows the inclusion of distributed actuator and sensor terms in the PDE and modification of boundary conditions. The algorithm thus developed is implemented digitally using several processing domains. A comparison of the results of the implementations is made and discussed on the basis of real-time performance of these processing domains. Finally, simulation results verifying the performance of the algorithm in characterising the behaviour of the system under various loading conditions are presented and discussed. The simulated system is constructed to provide a suitable platform for subsequent implementation and verification of various controller designs.

## 2 The flexible manipulator system

A schematic representation of a single link flexible manipulator system is shown in Figure 1, where, a manipulator of length $l$, linear mass density $\rho$, moment of inertia $I_b$, hub inertia $I_h$, carrying a payload of mass $M_p$ and inertia $I_p$ is considered. A control torque $\tau(t)$ is applied at the hub of the manipulator by an actuator motor. The angular displacement of the link in the (fixed) $POQ-$coordinates is denoted by $\theta(t)$. $u$ represents the elastic deflection of the manipulator at a distance $x$ from the hub, measured in the

moving coordinates $P'OQ'$. The height of the link is assumed to be much greater than its width, thus, allowing the manipulator to vibrate (be flexible) dominantly in the horizontal direction ($POQ-$plane) and be stiff in vertical bending and torsion.

The flexible manipulator system is modelled as a pinned-free flexible beam, incorporating an inertia at the hub and payload mass at the end-point. The model is developed through the utilisation of Lagrange equation and modal expansion method (Hastings and Book, 1987; Korolov and Chen, 1989). To avoid the difficulties arising due to time-varying length, the length of the manipulator is assumed to be constant. Moreover, shear deformation, rotary inertia and effect of axial force are neglected.

For an angular displacement $\theta$ and an elastic deflection $u$ the total displacement $y(x,t)$ of a point along the manipulator at a distance $x$ from the pinned end can be described as a function of both the rigid body motion $\theta(t)$ and elastic deflection $u(x,t)$;

$$y(x,t) = x\theta(t) + u(x,t)$$

Thus, the net deflection at $x$ is the sum of a rigid body deflection and an elastic deflection. Note that the elastic deflections of the manipulator are assumed to be confined to the horizontal plane only. This is achieved by allowing the manipulator to be dominantly flexible to horizontal deflection. In general, the motion of a manipulator will include elastic deflections in both the vertical and horizontal planes. Motion in the vertical plane due to gravity forces, for example, can cause permanent elastic deflections. This effect is neglected here as the manipulator is assumed to be dominantly flexible in the horizontal plane. This can be ensured by physical construction of the manipulator and by keeping the payload within a certain range.

The dynamic equations of motion of the manipulator can be obtained using the Hamilton's extended principle (Meirovitch, 1967) with the associated kinetic, potential and dissipated energies of the system. This yields the dynamic equation of motion of the manipulator as (Tokhi et al., 1994)

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

where, $E$ is Young's modulus, $I$ is the second (area) moment of inertia and $\tau(x,t)$ is the applied torque. The product $EI$ represents the flexural rigidity of the manipulator. The associated boundary conditions at the hub of the manipulator are given by

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

Similarly, the associated boundary conditions at the end-point of the manipulator are given by

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

The fourth order PDE in equation (1) with the associated boundary conditions in equations (2) and (3) describe the dynamic equations of motion of the flexible manipulator system.

## 3 The finite difference method

The PDE in equation (1) describing the dynamics of the flexible manipulator system is of a hyperbolic type and can be classified as a boundary value problem. This can be solved using an FD method as outlined here briefly. Figure 2 describes the discretisation of the FD method along distance $x$ and time $t$ in the $(x,t)$ co-ordinate system, where a manipulator of length $l$, located along $x$ from $a'$ to $b'$, is considered within a time interval $t$ of $(c',d')$. The initial conditions along the $t$ axis are described by

$$y(x,0)=0 \quad \text{and} \quad \frac{\partial y(x,0)}{\partial t}=0 \tag{4}$$

Partitioning the interval $(a',b')$ in Figure 2 into $n$ sections, each of length $\Delta x$ $(\Delta x=(b'-a')/n)$ and the interval $(c',d')$ into $m$ sections, each of length $\Delta t$ $(\Delta t=(d'-c')/m)$ results the grid of vertical and horizontal lines through points with co-ordinates $x_i,t_j$, where

$$x_i=i\Delta x; \quad i=0,1,\cdots,n$$
$$t_j=j\Delta t; \quad j=0,1,\cdots,m$$

The lines $x=x_i$ and $t=t_j$ are called grid lines and their intersections are called the mesh points of the grid or grid-points. For each mesh point in the interior of the grid $(x_i,t_j)$ $(i=0,1,\cdots,n$ and $j=0,1,\cdots,m)$ a Taylor series expansion is used to generate the central difference formulae for the partial derivative terms of the response $y(x,t)$ of the manipulator at points $x=i\Delta x$, $t=j\Delta t$ (Lapidus, 1982; Burden and Faires, 1989). This gives

$$\frac{\partial^2 y(x,t)}{\partial t^2}=\frac{y_{i,j+1}-2y_{i,j}+y_{i,j-1}}{\Delta t^2} \tag{5}$$

$$\frac{\partial^2 y(x,t)}{\partial x^2}=\frac{y_{i+1,j}-2y_{i,j}+y_{i-1,j}}{\Delta x^2} \tag{6}$$

$$\frac{\partial^4 y(x,t)}{\partial x^4}=\frac{y_{i+2,j}-4y_{i+1,j}+6y_{i,j}-4y_{i-1,j}+y_{i-2,j}}{\Delta x^4} \tag{7}$$

$$\frac{\partial^3 y(x,t)}{\partial x^3}=\frac{y_{i+2,j}-2y_{i+1,j}-2y_{i-1,j}+y_{i-2,j}}{2\Delta x^3} \tag{8}$$

$$\frac{\partial^3 y(x,t)}{\partial t^2\partial x}=\frac{y_{n-1,j+1}-2y_{n-1,j}+y_{n-1,j-1}-y_{n,j+1}+2y_{n,j}-y_{n,j-1}}{2\Delta x\Delta t^2} \tag{9}$$

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 discretisation is adopted in the evaluation of the response of the manipulator. A solution of the PDE in equation (1) can now be obtained using equations (5) to (9). This is described in the next section.

## 4 Algorithm development

To solve the PDE in equation (1), a set of equivalent difference equations defined by the central FD quotients to replace the PDE are obtained. The manipulator is divided into a suitable number of sections of equal lengths and a difference equation, corresponding to each point of the grid is developed. The known boundary conditions are utilised to eliminate the displacements of the fictitious points outside the defined interval.

Substituting for $\frac{\partial^2 y}{\partial t^2}$ and $\frac{\partial^4 y}{\partial x^4}$ from equations (5) and (7) into equation (1) and simplifying yields the displacement $y_{i,j+1}$ as

$$y_{i,j+1} = -c\left[y_{i-2,j} + y_{i+2,j}\right] + b\left[y_{i-1,j} + y_{i+1,j}\right] + a y_{i,j} - y_{i,j-1} + \frac{\Delta t^2}{\rho}\tau(i,j) \qquad (10)$$

where, $a = 2 - \frac{6\Delta t^2 EI}{\rho \Delta x^4}$, $b = \frac{4\Delta t^2 EI}{\rho \Delta x^4}$, $c = \frac{\Delta t^2 EI}{\rho \Delta x^4}$ and $\tau(i,j) = \tau\left(x_i, t_j\right)$. Equation (10) gives the displacement of section $i$ of the manipulator at time step $j+1$. It follows from this equation that, to obtain the displacements $y_{1,j+1}$, $y_{n-1,j+1}$ and $y_{n,j+1}$, the displacements of the fictitious points $y_{-1,j}$, $y_{n+1,j}$ and $y_{n+2,j}$ are required. The evaluation of these displacements is based on the boundary conditions related to the dynamic equation of the flexible manipulator.

Using matrix notation, equation (10) can be written in a compact form as

$$\mathbf{Y}_{i,j+1} = \mathbf{A}\mathbf{Y}_{i,j} - \mathbf{Y}_{i,j-1} + \mathbf{B}\mathbf{F} \qquad (11)$$

where, $\mathbf{Y}_{i,j+1}$ represents the displacement of grid points $i = 1,2,\cdots,n$ of the manipulator at time step $j+1$, $\boldsymbol{Y}_{i,j}$ and $\boldsymbol{Y}_{i,j-1}$ represent the corresponding displacements at time steps $j$ and $j-1$ respectively, $\boldsymbol{A}$ is a constant $n \times n$ matrix entries of which depend on the flexible manipulator specification and the number of sections the manipulator is divided into, $\boldsymbol{F}$ is an $n \times 1$ matrix known as the forcing matrix and $\mathbf{B}$ is a scalar constant related to the time step $\Delta t$ and mass per unit length, $\rho$, of the flexible manipulator;

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

$$\mathbf{A} = \begin{bmatrix} m_1 & m_2 & m_3 & 0 & 0 & \cdots & 0 & 0 \\ b & a & b & -c & 0 & \cdots & 0 & 0 \\ -c & b & a & b & -c & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -c & b & a & b & -c \\ 0 & 0 & \cdots & 0 & m_{11} & m_{12} & m_{13} & m_{14} \\ 0 & 0 & \cdots & 0 & m_{21} & m_{22} & m_{23} & m_{24} \end{bmatrix},$$

$$\mathbf{B} = \frac{\Delta t^2}{\rho},$$

$$\mathbf{F}^T = \left[\tau(1,j), \quad \tau(2,j), \quad \cdots, \quad \tau(n,j)\right].$$

The element values $m_1$, $m_2$, $m_3$, $m_{11},\ldots,m_{24}$ in matrix **A** depend on the boundary conditions of the manipulator. Equation (11) gives, as a general solution of the PDE, the displacement of section $i$ of the manipulator at time step $j+1$. Note in equation (11) that the displacement of the hub-end of the manipulator ($i=0$) has not been included, as this is already known from the boundary conditions in equation (2). In the following sections, equivalent relations for special cases of the system with or without the hub inertia and payload mass are developed.

### *4.1 The system without hub inertia and payload mass*

In this section a solution of the PDE is presented assuming no hub inertia and end-point payload. That is, the flexible manipulator is considered simply as pinned at the motor end

and carrying no payload. Thus, assuming $I_h = I_p = 0$ and $M_p = 0$ in equations (2) and (3) yield the boundary conditions for this situation as

$$y(0,t) \quad , \quad EI\frac{\partial^2 y(0,t)}{\partial x^2} + \tau(0,t) = 0$$
$$\frac{\partial^2 y(l,t)}{\partial x^2} = 0 \quad , \quad \frac{\partial^3 y(l,t)}{\partial x^3} = 0 \tag{12}$$

Discretising the system in time and space (length) and using the boundary conditions in equation (12) and the central FD formulae in equations (5) to (9) expressions for the displacements $y_{0,j}$, $y_{1,j}$, $y_{n-1,j}$ and $y_{n,j}$ can be obtained.

It follows from equations (6) and (12) that,

$$y(0,j) = 0 \quad \text{for} \quad j = 0,1,\cdots,m$$
$$\frac{\partial^2 y(0,j)}{\partial x^2} = \frac{y_{1,j} - 2y_{0,j} + y_{-1,j}}{\Delta x^2} = \frac{y_{i,j} + y_{-1,j}}{\Delta x^2} \tag{13}$$

Substituting for $\frac{\partial^2 y(0,j)}{\partial x^2}$ from equation (13) into equation (12) (with $t = j$), simplifying and rearranging yields

$$y_{-1,j} = \frac{\Delta x^2 \tau(0,j)}{EI} - y_{1,j} \tag{14}$$

Using equation (10) and noting that $\tau(i,j) = 0$ for $i \geq 1$ as the torque is applied at the hub of the flexible manipulator, the displacement $y_{1,j+1}$ can be written as

$$y_{1,j+1} = -c\left[y_{3,j} + y_{-1,j}\right] + b\left[y_{2,j} + y_{0,j}\right] + ay_{1,j} - y_{1,j-1} \tag{15}$$

Substituting for $y_{-1,j}$ from equation (14) into equation (15), using equation (13) and simplifying yields

$$y_{1,j+1} = -cy_{3,j} + by_{2,j} + (2-5c)y_{1,j} - c\frac{\Delta x^2 \tau(0,j)}{EI} \tag{16}$$

To obtain the displacements $y_{n-1,j+1}$ and $y_{n,j+1}$ the boundary conditions in equation (12) can be used. Using equation (10) and noting that $\tau(n-1,j)=0$, the displacement $y_{n-1,j+1}$ can be written as

$$y_{n-1,j+1} = -c\left[y_{n+1,j} + y_{n-3,j}\right] + b\left[y_{n,j} + y_{n-2,j}\right] + a y_{n-1,j} - y_{n-1,j-1} \tag{17}$$

Substituting for $\partial^2 y(x,t)/\partial x^2$ from equation (12) for $x=l$ $(i=n)$ into equation (6) and simplifying yields

$$y_{n+1,j} = 2y_{n,j} - y_{n-1,j} \tag{18}$$

Substituting for $y_{n+1,j}$ from equation (18) into equation (17) and simplifying yields

$$y_{n-1,j+1} = -cy_{n-3,j} + by_{n-2,j} + (2-5c)y_{n-1,j} + 2cy_{n,j} - y_{n-1,j-1} \tag{19}$$

Using equation (10) and noting that $\tau(n,j+1)=0$, the displacement $y_{n,j+1}$ can be written as

$$y_{n,j+1} = -c\left[y_{n+2,j} + y_{n-2,j}\right] + b\left[y_{n+1,j} + y_{n-1,j}\right] + a y_{n,j} - y_{n,j-1} \tag{20}$$

Substituting for $\partial^3 y(x,t)/\partial x^3$ from equation (12) for $x=l$ $(i=n)$ into equation (8), simplifying and using equation (18) yields

$$y_{n+2,j} = y_{n-2,j} - 4y_{n-1,j} + 4y_{n,j} \tag{21}$$

Substituting for $y_{n+1,j}$ and $y_{n+2,j}$ from equations (18) and (21) into equation (20) and simplifying yields

$$y_{n,j+1} = -2cy_{n-2,j} + 4cy_{n-1,j} + (2-2c)y_{n,j} - y_{n,j-1} \tag{22}$$

Equations (16), (19) and (22) give the displacements at grid points $(1,j+1)$, $(n-1,j+1)$ and $(n,j+1)$ of the manipulator in the absence of hub inertia and payload. The

element values $m_1$ to $m_3$ and $m_{11}$ to $m_{24}$ of the matrix $A$ in equation (11), as obtained from these three equations are thus given by

$$\begin{aligned} &m_1 = 2-5c, \quad m_2 = b, \quad m_3 = -c, \\ &m_{11} = -c, \quad m_{12} = b, \quad m_{13} = 2-5c, \quad m_{14} = 2c, \\ &m_{21} = 0, \quad m_{22} = 2c, \quad m_{23} = 4c, \quad m_{24} = 2-2c. \end{aligned}$$

Therefore, incorporating the expressions for hub and end-point displacements, as obtained above, into equation (11) and using the initial conditions in equation (4) yields a simulation algorithm for the manipulator in the absence of hub inertia and payload.

### *4.2 The system with hub inertia and payload mass*

In the previous section, a solution for the PDE was developed on the basis of assuming no lumped inertia at the hub and no additional mass at the end-point of the manipulator. In practice there will be some hub inertia due to clamping of the manipulator with the motor shaft and inertia of the rotating part of the motor, tachometer and shaft encoder. Moreover, the flexible manipulator will be carrying some payload from one point to another. Therefore, it is required that the inclusion of the hub inertia and payload are accounted for within the simulation algorithm. With a payload mass of $M_p$ at the end-point of the manipulator, inertia $I_p$ around the mass $M_p$ and hub inertia $I_h$, equations (2) and (3) yield the boundary conditions for this situation as

$$\begin{aligned} &y(0,t) = 0 \quad , \quad I_h \frac{\partial^3 y(0,t)}{\partial t^2 \partial x} - EI \frac{\partial^2 y(0,t)}{\partial x^2} = \tau(0,t) \\ &M_p \frac{\partial^2 y(l,t)}{\partial t^2} - EI \frac{\partial^3 y(l,t)}{\partial x^3} = 0 \quad , \quad I_p \frac{\partial^2}{\partial t^2} \frac{\partial y(l,t)}{\partial x} + EI \frac{\partial^2 y(l,t)}{\partial x^2} = 0 \end{aligned} \tag{23}$$

A discretisation of the boundary conditions is obtained by substituting for the partial derivative terms from equations (5) to (9) into equation (23) accordingly and simplifying to yield

$$y_{0,j} = 0 \tag{24}$$

$$y_{1,j+1} = \frac{EI\Delta t^2}{\Delta x I_h}\left[y_{1,j} + y_{-1,j}\right] + 2y_{1,j} - y_{1,j-1} + \tau(0,j)\frac{\Delta x \Delta t^2}{I_h} \tag{25}$$

$$y_{n,j+1} = -\frac{EI\Delta t^2}{2M_p\Delta x^3}\left[y_{n-2,j} + 2y_{n-2,j} + 2y_{n+1,j} - y_{n+2,j}\right] + 2y_{n,j} - y_{n,j-1} \tag{26}$$

$$y_{n,j+1} = -\frac{2EI\Delta t^2}{I_p\Delta x}\left[y_{n-1,j} - 2y_{n,j} + y_{n+1,j}\right] - 2y_{n-1,j} + 2y_{n,j} - y_{n,j-1} + y_{n-1,j-1} + y_{n-1,j+1} \tag{27}$$

Equations (24) to (27) can further be manipulated to yield the displacements at grid points $(1, j+1)$, $(n-1, j+1)$ and $(n, j+1)$ of the manipulator.

It follows from equation (24) that

$$y(0,j) = 0 \quad \text{for} \quad j = 0,1,\cdots,m \tag{28}$$

Using equations (10) and (28) and noting that $\tau(1,j) = 0$, $y_{1,j+1}$ can be expressed as

$$y_{1,j+1} = -c\left[y_{-1,j} + y_{3,j}\right] + by_{2,j} + ay_{1,j} - y_{1,j-1} \tag{29}$$

Solving equation (25) for $y_{-1,j}$ , substituting into equation (29) and simplifying yields $y_{1,j+1}$ as

$$y_{1,j+1} = Q_1 y_{1,j} + Q_2 y_{2,j} + Q_3 y_{3,j} - y_{1,j-1} + Q_4\left[\tau(0,j)\right] \tag{30}$$

where,

$$Q_1 = \frac{2\rho\Delta x^4 + 2I_h\Delta x - 5\Delta t^2 EI}{\Delta x\left(\rho\Delta x^3 + I_h\right)} \quad , \quad Q_2 = \frac{4\Delta t^2 EI}{\Delta x\left(\rho\Delta x^3 + I_h\right)}$$

$$Q_3 = -\frac{\Delta t^2 EI}{\Delta x\left(\rho\Delta x^3 + I_h\right)} \quad , \quad Q_4 = \frac{\Delta t^2}{\Delta x\left(\rho\Delta x^3 + I_h\right)}$$

Equations (28) and (30) give the displacements at the hub of the manipulator.

Using equation (10) and noting that $\tau(n-1,j)=0$, $y_{n-1,j+1}$ can be expressed as

$$y_{n-1,j+1} = -c\left[y_{n-3,j} + y_{n+1,j}\right] + b\left[y_{n,j} + y_{n-2,j}\right] + a y_{n-1,j} - y_{n-1,j-1} \tag{31}$$

Substituting for $y_{n-1,j+1}$ from equation (31) into equation (27) and simplifying yields $y_{n+1,j}$ as

$$y_{n+1,j} = -\frac{1}{c+F_1}\left[c y_{n-3,j} - b y_{n-2,j} + (F_1+2-a) y_{n-1,j} + (-2-b-2F_1) y_{n,j} + y_{n,j-1} + y_{n,j+1}\right] \tag{32}$$

where, $F_1 = \left[2EI\Delta t^2 / I_p \Delta x\right]$.

Using equation (10) and noting that $\tau(n,j)=0$, $y_{n,j+1}$ can be expressed as

$$y_{n,j+1} = -c\left[y_{n-2,j} + y_{n+2,j}\right] + b\left[y_{n+1,j} + y_{n-1,j}\right] + a y_{n,j} - y_{n,j-1} \tag{33}$$

Substituting for $y_{n,j+1}$ from equation (33) into equation (26) and simplifying yields

$$y_{n+1,j}(-b-2F_2) + y_{n+2,j}(c+F_2) = y_{n-2,j}(F_2-c) + y_{n-1,j}(b-2F_2) + y_{n,j}(a-2) \tag{34}$$

where, $F_2 = \left[EI\Delta t^2 / 2M_p \Delta x^3\right]$. Similarly, substituting for $y_{n,j+1}$ from equation (33) into equation (32) and simplifying yields

$$\begin{aligned} &y_{n+1,j}\left[1 + \frac{b}{c+F_1}\right] - y_{n+2,j}\left[\frac{c}{c+F_1}\right] = \\ &-\frac{1}{c+F_1}\left[c y_{n-3,j} + y_{n-2,j}(-b-c) + y_{n-1,j}(F_1+2-a+b) + y_{n,j}(a-b-2-2F_1)\right] \end{aligned} \tag{35}$$

Thus, solving equations (34) and (35) simultaneously for the fictitious displacements $y_{n+2,j}$ and $y_{n+1,j}$ yields

$$y_{n+2,j} = P_1 y_{n-3,j} + P_2 y_{n-2,j} + P_3 y_{n-1,j} + P_4 y_{n,j} \tag{36}$$

and

$$y_{n+1,j} = T_1 y_{n-3,j} + T_2 y_{n-2,j} + T_3 y_{n-1,j} + T_4 y_{n,j} \tag{37}$$

where,

$$P_1 = \frac{-b-2F_2}{F_2(F_1+b-1)+F_1c}$$

$$P_2 = \frac{(b+c+F_1)(-b-2F_2)}{(b+c+F_1)(c+F_2)+c(-b-2F_2)}\left[\frac{b(3F_2+b_2)+c(3F_2-c)-F_1(c-F_2)}{(b+c+F_1)(-b-2F_2)}\right]$$

$$P_3 = \frac{(b+c+F_1)(-b-2F_2)}{(b+c+F_1)(c+F_2)+c(-b-2F_2)}\left[\frac{b^2-9bc-2F_2(b+11c+2F_2)}{(b+c+F_1)(-b-2F_2)}\right]$$

$$P_4 = \frac{F_1(a-2+2b+4F_2)+(a-2)(b+c)+10c(b+2F_2)}{(b+c+F_1)(-b-2F_2)}\left[\frac{(b+c+F_1)(-b-2F_2)}{(b+c+F_1)(c+F_2)+c(-b-2F_2)}\right]$$

and

$$T_1 = -\frac{F_2(b-c)+F_1(c+F_2)+c^2}{c+F_2}$$

$$T_2 = \frac{F_2(b-c)+F_1(c+F_2)+c^2}{c(c+F_2)^2}\left[c(b+c)+(c+F_2)(c-F_2)\right]$$

$$T_3 = \frac{F_2(b-c)+F_1(c+F_2)+c^2}{c(c+F_2)^2}\left[F_1(c-b)+2F_2(c+F_2)+c(10c-b)\right]$$

$$T_4 = \frac{F_2(b-c)+F_1(c+F_2)+c^2}{c(c+F_2)^2}\left[(2-a)+(F_2+c)-c(2F_2+10c)\right]$$

Substituting for $y_{n+1,j}$ from equation (37) into equation (10), for $i=n-1$, and simplifying yields

$$y_{n-1,j+1} = M_1 y_{n-3,j} + M_2 y_{n-2,j} + M_3 y_{n-1,j} + M_4 y_{n,j} - y_{n-1,j-1} \tag{38}$$

where,

$$M_1 = -c - cT_1 \quad , \quad M_2 = b - cT_2$$

$$M_3 = a - cT_3 \quad , \quad M_4 = b - cT_4$$

Similarly, substituting for $y_{n+1,j}$ and $y_{n+2,j}$ from equations (37) and (36) into equation (10), for $i = n$, and simplifying yields

$$y_{n,j+1} = N_1 y_{n-3,j} + N_2 y_{n-2,j} + N_3 y_{n-1,j} + N_4 y_{n,j} - y_{n,j-1} \tag{39}$$

where,

$$N_1 = bT_1 - cP_1 \quad , \quad N_2 = bT_2 - cP_2 - c$$

$$N_3 = bT_3 - cP_3 + b \quad , \quad N_4 = bT_4 - cP_4 + a$$

Equations (30), (38) and (39), thus, give the displacements at grid points $(1, j+1)$, $(n-1, j+1)$ and $(n, j+1)$ of the manipulator in the presence of hub inertia and payload mass. From these three equations the values of the elements $m_1$ to $m_3$ and $m_{11}$ to $m_{24}$ of the matrix $\boldsymbol{A}$ in equation (11) corresponding to this situation can be obtained as

$$\begin{aligned} &m_1 = Q_1, \quad m_2 = Q_2, \quad m_3 = Q_3, \\ &m_{11} = M_1, \quad m_{12} = M_2, \quad m_{13} = M_3, \quad m_{14} = M_4, \\ &m_{21} = N_1, \quad m_{22} = N_2, \quad m_{23} = N_3, \quad m_{24} = N_4. \end{aligned}$$

Therefore, importing the equations for hub and end-point displacements, as obtained above, into equation (11) and utilising the initial conditions in equation (4) yields a simulation algorithm for the manipulator system in the presence of hub inertia and payload.

## 5 Algorithm stability

The stability of the algorithm can be examined by ensuring that the iterative scheme described in equation (10) converges to a solution. Discretising equation (1), using equations (5) and (7), yields

$$\left[\frac{EI\Delta t^2}{\rho \Delta x^4}\right]\left(y_{i+2,j} - 4y_{i+1,j} + 6y_{i,j} - 4y_{i-1,j} + y_{i-2,j}\right) = -y_{i,j-1} + 2y_{i,j} - y_{i,j-1} \tag{40}$$

Since $y_{i,j}$ can be expressed as $y_{i,j} = \phi(t)e^{\gamma\alpha x}$, where $\alpha$ represents spatial frequency and $\gamma = \sqrt{-1}$, equation (40) can be written as

$$c\phi(t)\left[e^{\gamma\alpha(x+2\Delta x)} - 4e^{\gamma\alpha(x+\Delta x)} + 6e^{\gamma\alpha(x)} - 4e^{\gamma\alpha(x-\Delta x)} + e^{\gamma\alpha(x-2\Delta x)}\right] = e^{\gamma\alpha x}\left[-\phi(t+\Delta t) + 2\phi(t) - \phi(t-\Delta t)\right]$$

Simplifying the above, using the Euler's identity; $e^{\gamma\alpha\Delta x} = \cos(\alpha\Delta x) + \gamma\sin(\alpha\Delta x)$, yields

$$\left[2 - 4c\left(\cos^2(\alpha\Delta x) + 1 - 2\cos(\alpha\Delta x)\right)\right] = \frac{\phi(t+\Delta t) + \phi(t-\Delta t)}{\phi(t)} \qquad (41)$$

In, general, all multiples of the spatial frequency $\alpha$ will be present in $y(x,t) = \phi(t)e^{\gamma\alpha x}$. Even if these are not present in the initial conditions, or not brought in by the boundary conditions, they are likely to be introduced by round off errors. This means that unbounded amplification of $\phi(t)$ must be guard against for all $\alpha$. The original error component will not grow in time, provided that

$$\left|e^{\gamma t}\right| \le 1$$

This is the well known Von Neumann stability condition (Mitchel an Griffiths, 1980; Press et al., 1988; Vemuri and Karplus, 1981). Thus, denoting $\phi(t) = e^{\gamma t}$, the right-hand-side of equation (41) can be simplified as

$$\frac{\phi(t+\Delta t) + \phi(t-\Delta t)}{\phi(t)} = \frac{e^{\gamma(t+\Delta t)} + e^{\gamma(t-\Delta t)}}{e^{\gamma t}} = e^{\gamma\Delta t} + e^{-\gamma\Delta t} = 2\cos\Delta t$$

Since $\left|2\cos(\Delta t)\right|$ can only take a maximum value of 2, the above yields

$$\frac{\phi(t+\Delta t)\phi(t-\Delta t)}{\phi(t)} \le 2$$

Using the above in equation (41) and simplifying yields

$$\left|1 - 2c\left(\cos(\alpha\Delta x) - 1\right)^2\right| \le 1$$

The maximum value $\left[\cos(\alpha\Delta x) - 1\right]^2$ can have is 4. Therefore, the above equation can be written as

$$|1-8c| \leq 1$$

or

$$0 \leq c \leq \frac{1}{4} \tag{42}$$

Equation (42) is the necessary and sufficient condition for stability of the simulation algorithm.

## 6 Algorithm implementation

In the simulation results presented here, an aluminium type flexible manipulator system characterised by the parameters given in Table 1 is considered. The first and second modes of vibration of the manipulator are, thus, theoretically at 12.034 and 36.43 Hz.

For implementation of the simulation algorithm using the FD method, the manipulator is divided into nineteen equal sections, with the spacing between successive grid points being 0.051 *mm*. The value of $c$, see equation (42), is chosen as 0.2 which satisfies the stability requirement of the algorithm. This value of $c$ yields the required sampling time interval of $21\mu$ sec, which is sufficient enough to cover the high frequency behaviour of the flexible manipulator. The displacement at each grid point is calculated for an applied torque at the hub of the flexible manipulator. Figure 3 shows a bang-bang torque with an amplitude of 0.1 *Nm* and a duration of 0.6 sec used as an input throughout the simulation exercises. With this torque the manipulator initially accelerates when the input torque is positive and decelerates when the torque is negative, thus, causing it to stop at a certain position.

As it was shown through the development of the simulation algorithm, the fictitious displacement values $y_{-1,j}$, $y_{n+1,j}$ and $y_{n+2,j}$ are calculated in terms of $y_{1,j}$, $y_{n-1,j}$ and $y_{n-2,j}$ using the initial and boundary conditions, which in turn modify the constant matrix **A** in equation (11). Thus, any change in a boundary or an initial condition will affect the element values of matrix **A**. Therefore, the development of relations to account for such

variations and subsequent calculation of element values of matrix **A** will be tedious and cumbersome.

Proper initialisation of the algorithm is important to avoid any instabilities. This requires discretisation of the initial conditions in the form of $y_{i,0} = y_{0,j}$ and $y_{i,1} = y_{i,0} + \tau \dot{y}_{i,0}$. Moreover, during the calculation process, it is important to provide a means of checking the boundary conditions at each time step so that the matrix **A** be updated, upon detecting any change. In some cases all the calculations for $y_{-1}$, $y_{n+1}$ and $y_{n+2}$ are included within the calculation loop during each iteration. This increases the computational burden on the processor. To overcome this problem, two assumptions are made: (a) once the system is in operation there will be no change in the boundary conditions at the hub of the manipulator; (b) during operation a change in matrix **A** will occur only when there is a change in the payload. This implies that a change will affect only eight elements of the matrix **A**. Therefore, a checking facility for payload $M_p$ is incorporated within the program, so that upon detection of a change in $M_p$ a recalculation of the required elements of **A** is carried out.

The simulation program thus developed is divided into two parts. The first part involves the calculation of initial values based on the system parameters and boundary conditions and the second part involves the iterative procedure for calculation of displacement of the flexible manipulator at successive time steps. In the case of a multi-link manipulator system, the program can be extended by modifying the equations characterising the system so that to include the nonlinear terms accounting for the interactions between the links. This includes modification of the boundary conditions to involve each link and the effect of mass and dynamics of each link on one another. The general philosophy thus is the same as far as the program structure is concerned.

## *6.1 Hardware and software facilities*

Three processing domains, namely the SUN4 ELC workstation, a 486DX based IBM compatible PC and an integrated DSP-transputer system are utilised for the

implementation of the algorithm. The SUN4 ELC workstation under consideration uses the SPARC processor, operating at a speed of 33 MHz. The 486DX based PC implementation utilises a Viglen PC compatible with the IBM PC(AT). This incorporates a 32-bit 80486 processor and a 40487 co-processor with a 50 MHz clock speed.

The integrated DSP-transputer system utilised in this investigation incorporates the Intel's i860 64-bit microprocessor and a 32 bit T805 transputer node. The i860 is designed for numerically and vector intensive applications. It can deliver a peak arithmetic performance of 80 MFLOPS (single precision) and 60 MFLOPS (double precision) and is capable of executing up to three operations each clock cycle (25 ns at 40 MHz). The T805 transputer is a high performance microprocessor used to facilitate interprocess and interprocessor communication and is targeted at the efficient exploitation of VLSI technology. The most important feature of the transputer is its external links which enable it to be used as a building block in the construction of low cost, high performance multiprocessing systems (INMOS Ltd, 1988). The integrated DSP-transputer system is aimed at utilising and exploiting the extensive signal processing power of the DSP with the high communication facility including processing capability of the transputer.

The C programming language is utilised for the algorithm implementation on the three processors. This is usually considered as a medium level language. Various types of C compilers suitable to the three domains are used. The UNIX-C which is very similar to ANSI C is used for the SPARC processor. The Borland $C^{++}$ is used for implementations in the 80486 based PC. Programs for the i860 are compiled using the C compilers from the Portland Group in addition with the i860 Transtech toolset (Portland Group Inc., 1992).

The Transtech toolset provides a development and runtime environment. It supports the execution of standard programs on the i860 and provides the full range of standard I/O facilities including code optimisation, linking and vectorisation. The 3L Parallel C compiler is used for implementations in the T805 transputer. This compiler is intended to program a transputer system, whether writing a conventional sequential program or using a full support for concurrency which the transputer processor has to offer (3L Ltd, 1989). The MATLAB package is also used, independently, for the implementation of the simulation

algorithm within the SPARC processor. MATLAB is an interactive software package for scientific and engineering numeric computation, signal processing and graphics. It provides an environment where problems and solutions are expressed as they would be written mathematically (The Mathworks Inc., 1991). The package is widely used in control, signal processing and other engineering applications.

## *6.2 Real-time processing requirements*

In this section a comparative investigation into the processing times in implementing the algorithm on the three types of processing domains is presented. Table 2 shows the simulation times using the SUN4 ELC workstation with C programming language and MATLAB, the 486 PC, a single T805 transputer and a single i860 processor, where actual simulation time is the time duration for which the manipulator is allowed to move, representing the real processing time required, and network processing time represents the time taken by a processor to simulate the manipulator movement. As shown, for an average actual simulation time of 3.33 sec the i860 takes 1.598 sec, the shortest processing time, while the SUN4 ELC workstation with MATLAB requires the longest time of 370.626 sec. It appears from Table 2 that on average MATLAB takes about 30 times longer processing time than the UNIX-C both operating under the same SPARC processor. However, MATLAB is widely used for matrix operations and numeric calculation and is not recommended for calculating large iteration loops. It follows from Table 2 that among the processors used, only the i860 performs within the real-time requirements of the simulation algorithm.

A graphical representation of the timing performances of these processors using C programming language is shown in Figure 4. It is noted that the transputer performs marginally faster than the SPARC processor for short simulation times. However, as the simulation time increases the transputer performs relatively slower than the SPARC processor. The processing time using the 486 based processor is lower than both the transputer and the SPARC processor.

The relative timing performances of the various types of processor under consideration in relation to the i860 processor is given in Table 3. It is noted that the i860 is performing at the order of about 9 times faster than the T805 transputer, 8 times faster than the SPARC processor and 6 times faster than the 486 based PC. The performance of these processors within each domain can be enhanced by utilising several of these within suitable parallel architectures. It has been reported that several transputer nodes operating in a parallel structure will perform faster than a single node. Although, in principle increasing the number of processors will decrease the net processing time for a given task. However, there is an optimum limit to the number of processors to which a reduction in the net processing time could be achieved by increasing the number of processors. Beyond this limit, due to the algorithm structure and interprocessor communication overheads an increase in the number of processors will not decrease the net processing time considerably (Chalmers and Gregory, 1993). An algorithm similar to the one described here has been considered for parallel implementation of cantilever beam simulation using T800 transputers ( Kourmoulis, 1990). Results of this study show that employing more than three processors in a parallel structure will actually increase the total processing time, thus limiting the timing performance obtainable using a network comprising of only transputer nodes. Further investigations into the real-time control of the cantilever beam have also been carried out involving up to five transputer nodes in a parallel structure, although timing requirements have not been mentioned (Virk, 1992). It has been reported, in another study, that although increased number of processing elements in a single instruction multiple data (SIMD) stream computer leads to a speedup in the execution of the program, but due to the communication overheads this architecture is not suitable for simulation of flexible manipulators (Tzes, et al., 1989).

Previous investigations have demonstrated the limitations of finer-grain DSP devices in applications involving irregular tasks. Similar investigations at utilising transputers have revealed the limitations of such processors in carrying out regular signal processing tasks. The results presented here demonstrate that the i860 DSP device meets the real-time processing requirements of the developed simulation algorithm. It has been reported,

however, that to achieve real-time performance in applications involving complex algorithms with extensive processing, suitable alternative solutions with enhanced capabilities could be obtained by devising parallel architectures incorporating DSP devices and transputer nodes (Tokhi and Hossain, 1993; Tokhi, et al., 1992).

The stability of the simulation algorithm is an important consideration in such a study. It is noted that the stability of the algorithm is dependent on the non-negative parameter $c$; for the algorithm to be stable the value of $c$ must be less then 0.25. The sampling time $\Delta t$ for the displacement calculation is also determined by the value of $c$. The sampling time in turn determines the upper frequency limit that can be captured within the flexible modes of vibration of the manipulator. For a set of the higher modes of vibration of the manipulator to be included, the value of $\Delta t$ must be sufficiently small. It has been demonstrated that, for a given value of c, the sampling time $\Delta t$ is affected by the number of grid points along the length of the manipulator used in the FD discretisation method. An increase in the number of grid points leads to a decrease in the value of $\Delta t$. Increasing the number of grid points will also lead to obtaining finer details of the manipulator movement. However, an increase in the number of grid points increases the computation burden on the processor. Therefore, for a given value of c, the choice of number of grid points should be a trade off between the required details of manipulator deflection and the processing time. This is an important consideration, specially, in real-time applications.

## 7 Simulation results

The set of results to follow describe the response of the manipulator to the excitation in Figure 3 under various conditions over a duration of 1.2 seconds. In developing the simulation algorithm two sets of relations were obtained. The first set of relations given by equations (16), (19) and (22) are obtained on the basis of ignoring hub inertia and payload, whereas, the second set given by equations (30), (38) and (39) are obtained by considering the hub inertia and a payload at the end-point of the manipulator.

To study the performance of the simulation program in characterising the behaviour of the manipulator, the system was excited once without the effect of hub inertia and then with the effect of the hub inertia. Figure 5(a) shows the end-point angular displacement (movement) of the manipulator with the hub inertia ignored, while Figure 5(b) shows the corresponding end-point angular displacement of the manipulator when the effect of hub inertia is taken into account. No payload mass is included in either case. It is noted that the overall displacement is relatively larger when the hub inertia is ignored as compared to the situation when the effect of the hub inertia is included. Moreover, a steady state situation is reached in each case within similar timescales (0.6 sec) with inherent residual fluctuation (vibration) throughout the movement. The time history of movement of the flexible manipulator for all grid points with the effect of the hub inertia included is shown in Figure 6. This shows the manipulator behaviour during its movement from one position to another.

Figure 7(a) shows the flexible modes of vibration of the manipulator as extracted from the end-point movement. This is done by using a highpass filter to separate the flexible modes from the rigid body motion. The second and higher modes of vibration as separated from the rigid body motion, in a similar manner, are shown in Figure 7(b). The first and second flexible modes of vibration of the manipulator, as filtered out from the others using suitable band-pass filters, are shown in Figures 8(a) and 8(b) respectively. It follows from Figures 7 and 8 that the first two modes are the dominant ones, characterising the flexible behaviour of the manipulator. These are measured, from the simulation results in Figures 7 and 8, as 12.016 and 35.397 Hz which are reasonably close to the corresponding theoretical values, thus validating the accuracy of the simulation program.

To study the effect of payload on the overall movement of the flexible manipulator, the response of the system was monitored with two different payloads at the end-point. Figures 9(a) and 9(b) show the end-point displacement of the flexible manipulator with payloads of 10 grams and 20 grams respectively. It is noted that the overall angular displacement decreases with increasing payloads at the expense of increased levels of vibration (amplitude of elastic deflection). This is further evidenced in Figures 10 and 11

which show the overall time histories of movement of the flexible manipulator for all grid points with payloads of 10 grams and 20 grams respectively.

## 8 Conclusion

A numerical method of solution of the governing PDE describing the characteristic behaviour of a flexible manipulator system incorporating the effects of hub inertia and payload has been presented and discussed. A simulation algorithm characterising the behaviour of the system has initially been developed without the hub inertia and payload. This has then been extended to include the effect of the hub inertia and payload. Simulation results with and without the effects of the hub inertia show that the overall angular displacement of the flexible manipulator when hub inertia is ignored is about three times more than that when the effect of the hub inertia is accounted for. The effect of the payload is such that the angular displacement decreases at the expense of increased levels of vibration for increased payloads.

The hub inertia has been considered to remain fixed throughout an assumed movement of the manipulator. Such an assumption is valid in practice. The payload mass on the other hand, could change at any stage of the movement cycle. As a result of this variation a payload checking facility for each iteration of displacement calculation has been incorporated into the algorithm so that the system matrix **A** is updated in case of a payload change. With such a provision the repetition of unnecessary calculations for the same payload is avoided which in turn makes the processing time relatively shorter.

Noticeable levels of vibration are evident in the simulated response of the manipulator throughout its movement in the presence of hub inertia and/or payload. An investigation into studying the end-point vibration of the flexible manipulator for a given input torque is carried out through an analysis process to extract the various flexible modes of vibration. It is shown through such an investigation that the frequency of the first two flexible modes correspond to the theoretically obtained frequencies. The overall elastic fluctuations of the

manipulator with various payloads reveal that an increase in the payload leads to an increase in the level of vibrations (elastic movements).

The real-time processing requirements of the developed simulation algorithm has been assessed by implementing the process on various types of processor. It has been demonstrated that high speed processing capability is required in the real-time simulation of flexible manipulator systems.

## 9 References

AUBRUN, J.-N. (1980). Theory of the structures by low-authority controllers, *Journal of Guidance and Control*, **3**, pp. 441-451.

BALAS, M. J. (1978). Feedback control of flexible systems, *IEEE Transaction on Automatic Control*, **AC-23**, pp. 673-679.

BURDEN, R. L. and FAIRES, J. D. (1989). *Numerical analysis*, PWS-KENT Publishing Company, Boston.

CANNON, R. H. and SCHMITZ, E. (1984). Initial experiments on the end-point control of a flexible one-link robot, *International Journal of Robotic Research*, **3**, pp. 62-75.

CHALMERS, A. G. and GREGORY, S. (1993). Constructing minimum path configuration for multiprocessor systems, *Parallel Computing*, **19**, pp. 343-355.

DADO, M. and SONI, A. H. (1986). A Generalized approach for forward and inverse dynamics of elastic manipulators, *Proceedings of IEEE Conference on Robotics and Automation*, pp. 359-364.

FELIU, V., RATTAN, K. S. and BROWN, H. B., Jr. (1992). Modelling and control of single link flexible arms with lumped masses, *Transaction of ASME Journal of Dynamic Systems, Measurement, and Control*, **114**, pp. 59-69.

HARISHIMA, F. and UESHIBA, T. (1986). Adaptive control of flexible arm using end-point position sensing, *Proceedings of Japan-USA Symposium of Flexible Automation*, 14-18 July, Osaka, pp. 225-229.

HASTINGS, G. G. and BOOK, W. J. (1987). A linear dynamic model for flexible robotic manipulator, *IEEE Control Systems Magazine*, **7**, pp. 61-64.

HUGHES, P. C. (1987). Space structure vibration modes: How many exists? Which are important, *IEEE Control Systems Magazine*, **7**, pp. 22-28.

INMOS Ltd. (1988). *Transputer reference manual*, Prentice Hall, Englewood Cliffs.

KOROLOV, V. V. and CHEN, Y. H. (1989). Controller design robust to frequency variation in a one-link flexible robot arm, *Journal of Dynamic Systems, Measurement and Control*, **111**, pp. 9-14.

KOURMOULIS, P. K. (1990). *Parallel processing in the simulation and control of flexible beam structures*, PhD Thesis, Department of Automatic Control and Systems Engineering, The University of Sheffield, UK.

LAPIDUS, L. (1982). *Numerical solution of partial differential equations in science and engineering*, John Wiley and Sons, New York.

MEIROVITCH, L. (1967). *Analytical methods in vibrations*, Macmillan, New York.

MITCHEL, A. R. and GRIFFITHS, D. F. (1980). *The finite difference method in partial differential equations*, John Wiley and Sons, New York.

OMATU, S. and SEINFELD, J. H. (1986). Optimal sensor actuator locations for linear distributed parameter systems, *Proceedings of 4th IFAC Symposium on Control of Distributed Parameter Systems*, Los Angeles, pp. 215-220.

OOSTING, K. and DICKERSON, S. L. (1988). Simulation of a high-speed lightweight arm, *Proceedings of the IEEE International Conference on Robotics and Automation*, Philadelphia.

PORTLAND GROUP INC. (1992). *i860 Program development manual*, Wilsonville, Oregon.

PRESS, W. H., FLANNERY, B. P., TEUKOLSKY, S. A. and VETTERLING, W. T. (1988). *Numerical recipes: in C: The art of scientific computing*, Cambridge University Press, Cambridge.

THE MATHWORKS INC. (1991). *MATLAB for Sun workstations, users guide*, The Mathworks Inc, Natick.

TOKHI, M. O., AZAD, A. K. M., MORRIS, A. S. and HOSSAIN, M. A. (1994). Modelling and simulation of a flexible manipulator system, *IEE Control-94 Conference*, 21-24 March 1994, Coventry, **2**, pp. 1400-1405.

TOKHI, M. O. and HOSSAIN, M. A. (1993). Signal processing for parallel processing for active noise control, *IEE Digest No. 1993/087: Colloquium on Signal Processing for Control*, 21 April 1993, London, pp. 5/1-5/4.

TOKHI, M. O., VIRK, G. S. and HOSSAIN, M. A. (1992). Integrated $DSP^3$ strategy for adaptive active control, *IEE Digest No. 1992/185: Colloquium on Active Techniques for Vibration Control- Sources, Isolation and Damping*, 28 October 1992, London, pp. 6/1-6/4.

TZES, A. P., YURKOVICH, S. and LANGER, F. D. (1989). A method for simulation of the Euler-Bernoulli beam equation in flexible link robotic systems, *Proceedings of International Conference on System Engineering*, pp. 557-560.

USORO, P. B., NADIRA, R. and MAHIL, S. S. (1984). A finite element/Lagrange approach to modelling lightweight flexible manipulator, *Transaction of ASME Journal of Dynamic Systems Measurement and Control*, **108**, pp. 198-205.

VEMURI, V. and KARPLUS, W. J. (1981). *Digital computer treatment for partial differential equations*, Prentice-Hall, Englewood Cliffs.

VIRK, G. S. (1992). Active vibration suppression in flexible structures, *IEE Digest No. 1992/185: Colloquium on Active Techniques for Vibration Control Sources, Isolation and Damping*, 28 October 1992, London, pp. 8/1-8/4.

3L LTD. (1989). *3L Parallel C users guide*, 3L Ltd, Livingston.

Table 1: Prameters of the flexible manipulator.

| Parameter | value |
|---|---|
| Length | $960mm$ |
| Thickness | $3.2004mm$ |
| Width | $19.230mm$ |
| Area of cross section | $6.1544\times10^{-5}m^2$ |
| Mass density per volume | $2710\,kg/m^3$ |
| Young's Modulus | $7.11\times10^{10}$ |
| Area moment of inertia | $5.1924\times10^{-11}m^2$ |
| Hub inertia | $5.86\times10^{-4}kgm^2$ |
| Manipulator inertia | $0.0495kgm^2$ |

Table 2: Processing times using different types of processor.

| Actual time of simulation (sec) | Network processing time (sec) | | | | |
|---|---|---|---|---|---|
| | SUN4-ELC MATLAB | SUN4-ELC Unix-c | 486DX | T805 | i860 |
| 0.8 | 20.399 | 3.378 | 2.152 | 3.092 | 0.381 |
| 1.2 | 38.738 | 4.846 | 3.227 | 4.679 | 0.571 |
| 1.6 | 63.954 | 6.315 | 4.303 | 6.209 | 0.752 |
| 2.0 | 97.074 | 7.784 | 5.360 | 8.214 | 0.942 |
| 2.4 | 133.529 | 9.253 | 6.456 | 10.772 | 1.132 |
| 3.0 | 198.162 | 11.455 | 8.069 | 12.451 | 1.413 |
| 4.0 | 343.762 | 15.127 | 10.759 | 16.779 | 1.934 |
| 5.0 | 520.314 | 18.798 | 13.448 | 21.187 | 2.415 |
| 10.0 | 1919.70 | 37.156 | 26.896 | 43.833 | 4.84 |
| Average: 3.33 | 370.626 | 12.679 | 8.963 | 14.135 | 1.598 |

Table 3: Processing times relative to the i860 processor.

| | T805 | SUN4-ELC Unix-c | 486DX |
|---|---|---|---|
| $T_1/T_2^*$ | 8.845 | 7.934 | 5.61 |

*$T_1$ is the time taken by a processor under consideration.

*$T_2$ is the time taken by the i860 processor.

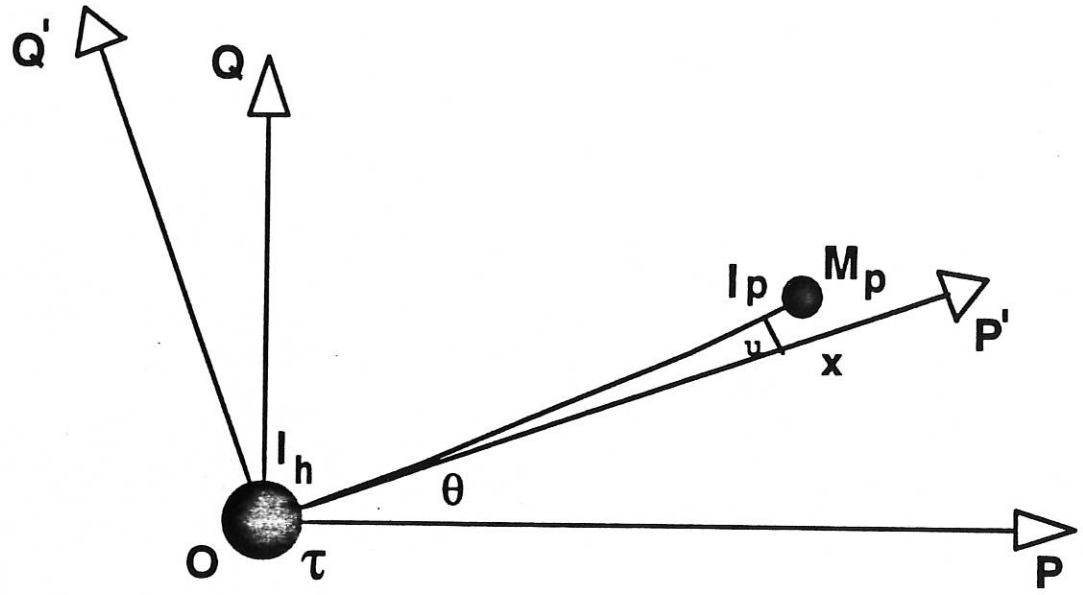



Figure 1: Schematic representation of the flexible manipulator system.

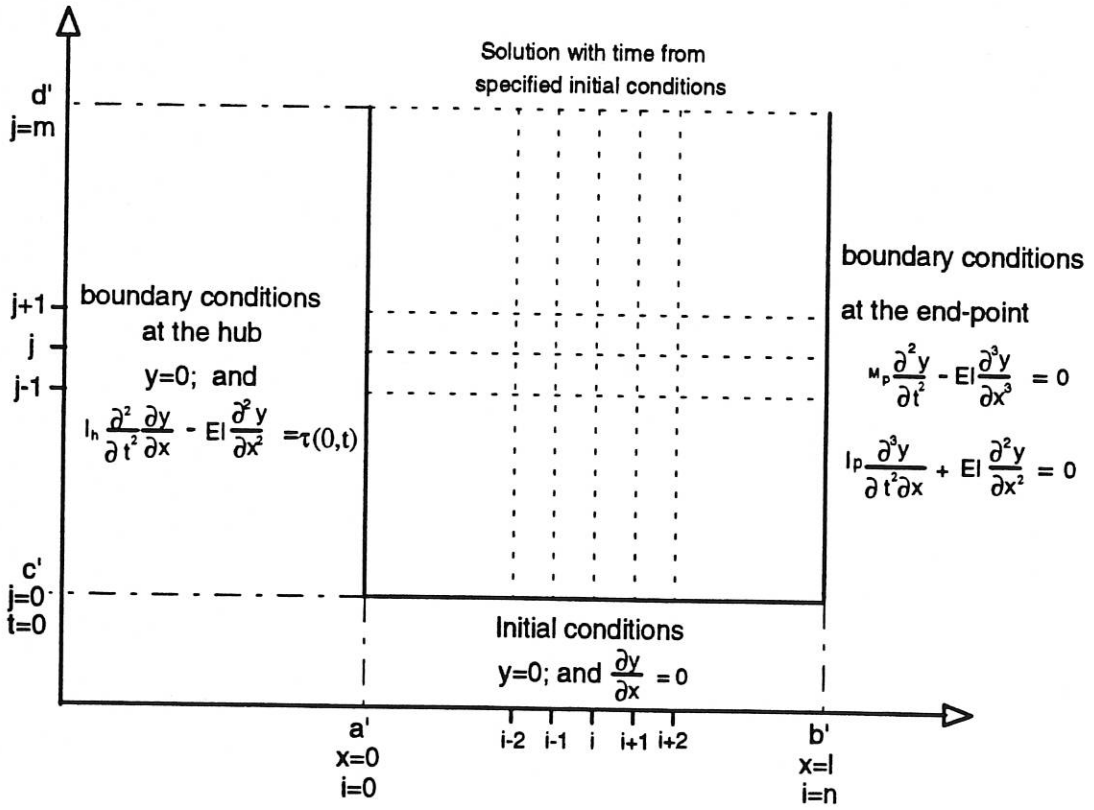



Figure 2: Discretisation of time and space variable for FD method.

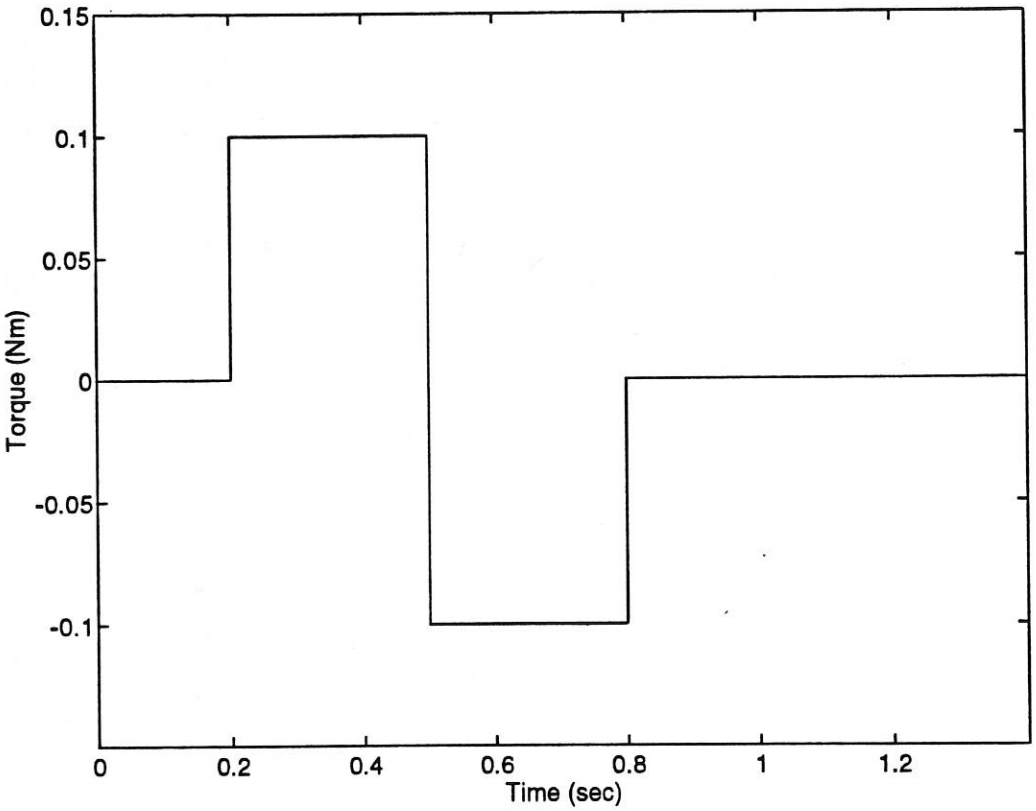



Figure 3: Input torque applied at the hub of the manipulator.

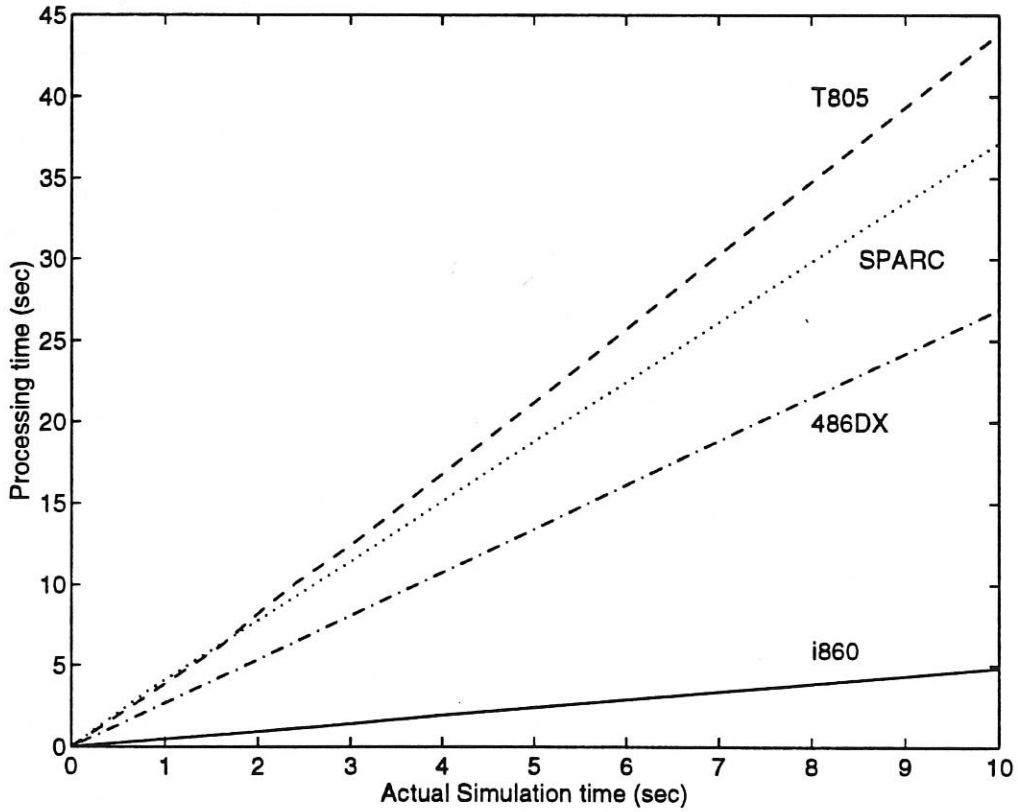
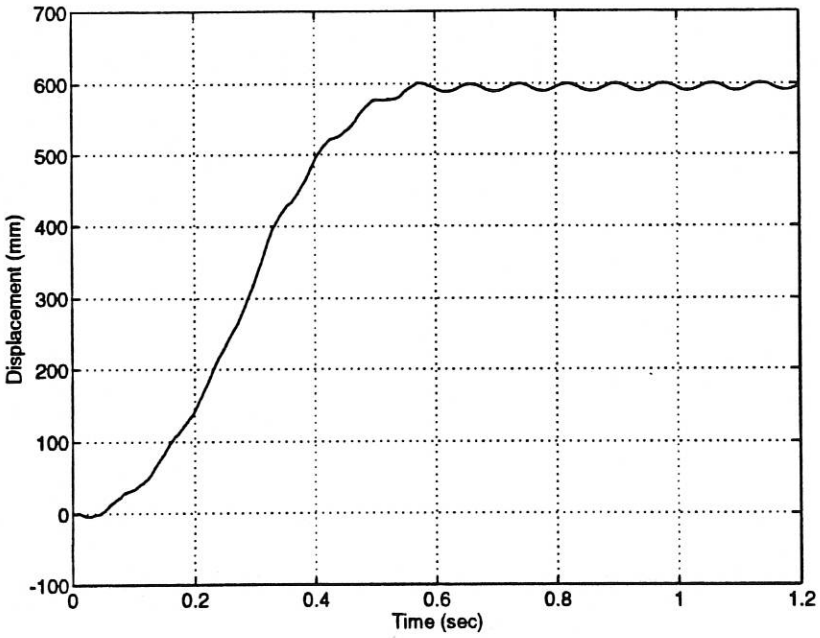
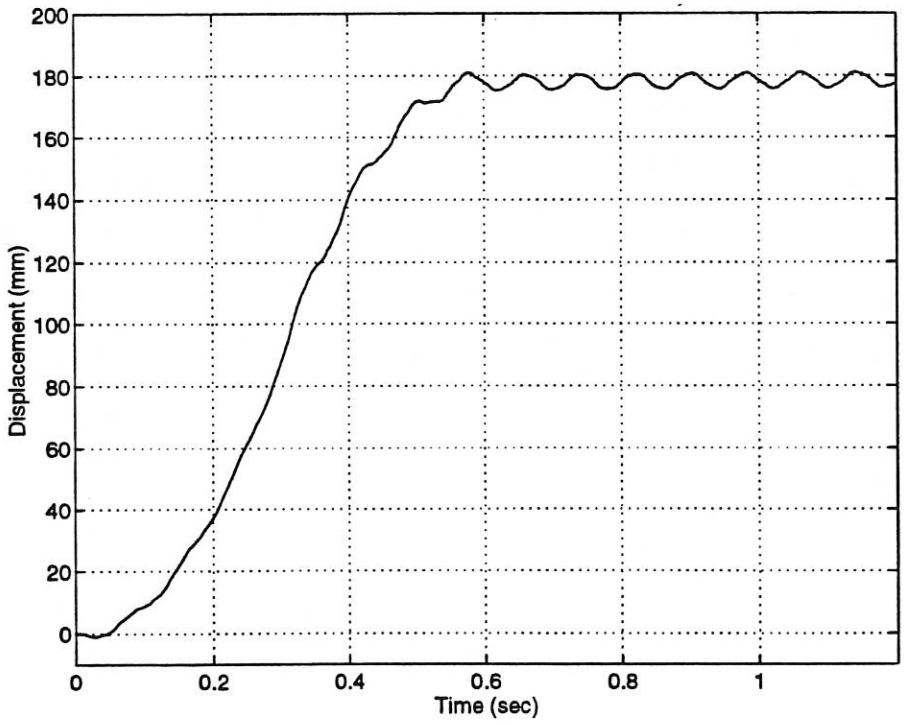



Figure 4: Processing time for different types of processor as related to actual simulation time.




(a)




(b)

Figure 5: End-point movement of the flexible manipulator;
(a) Without hub inertia and payload.
(b) With hub inertia and no payload.

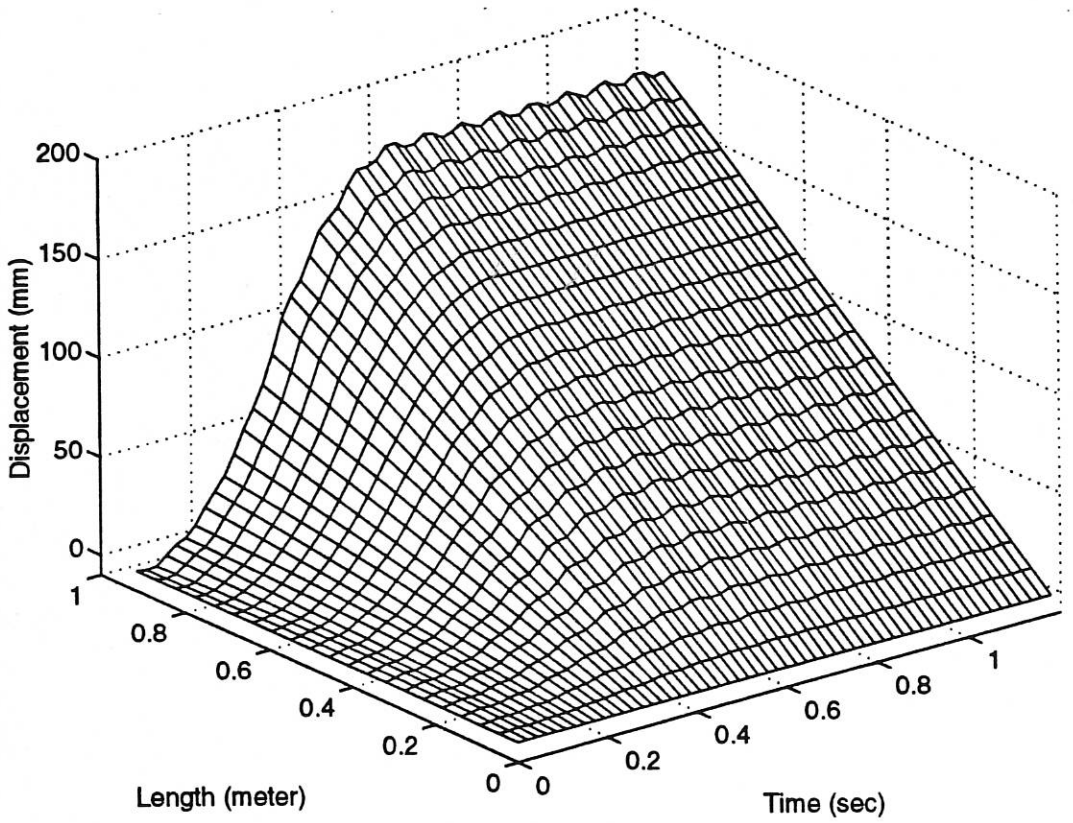
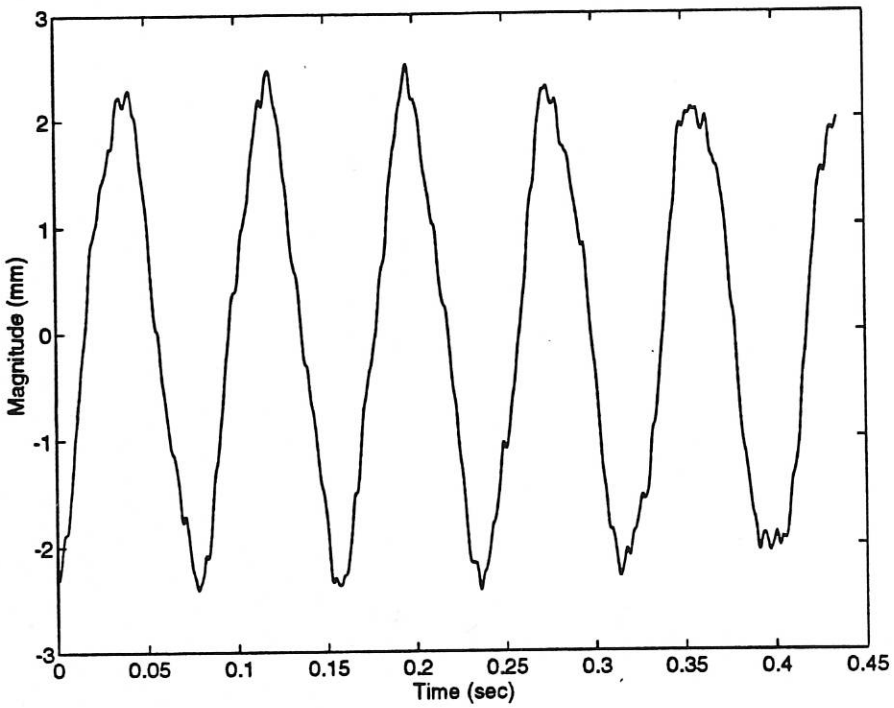
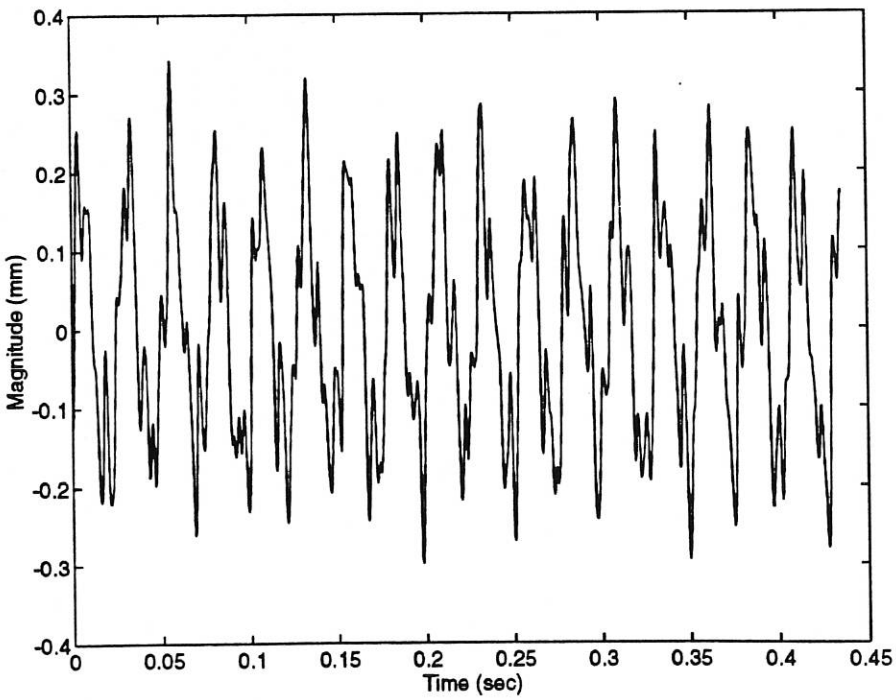



Figure 6: Movement of flexible manipulator including hub inertia.

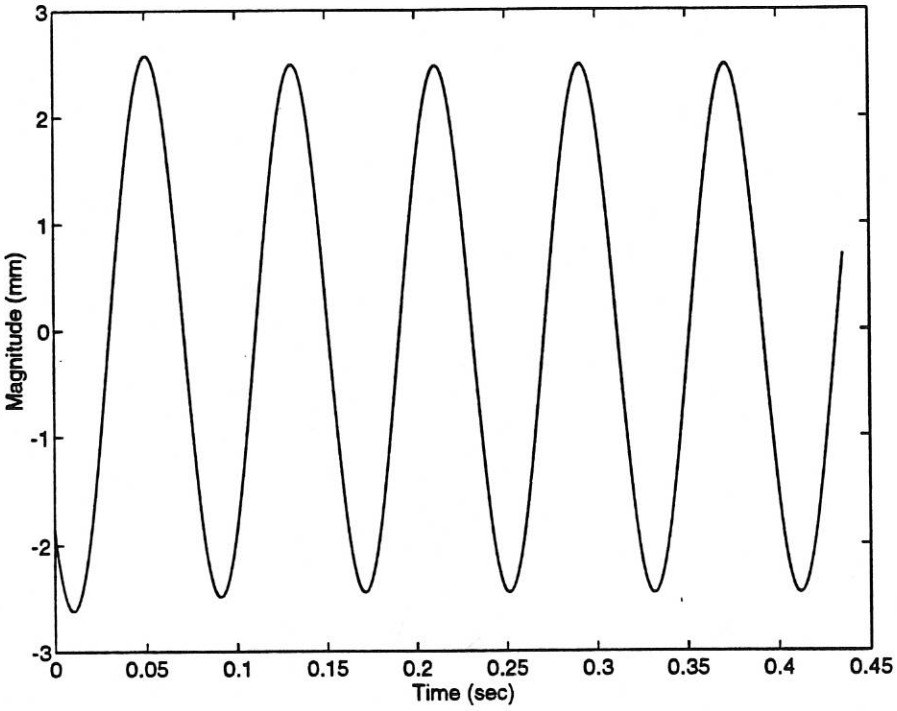



(a)

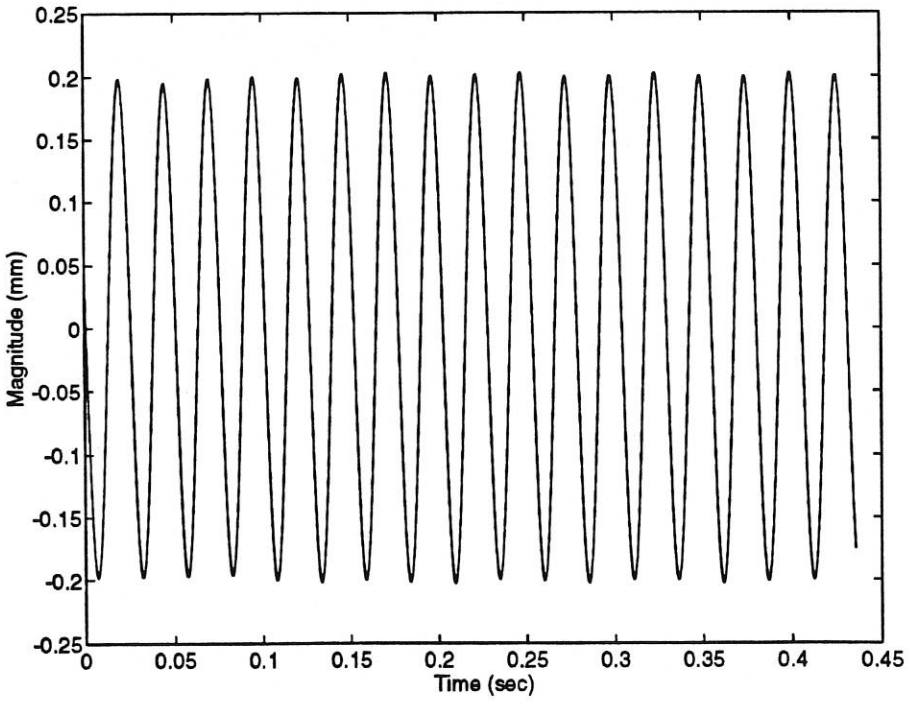



(b)

Figure 7: Flexible motion of the manipulator;
(a) All modes of vibration.
(b) Second and higher modes of vibration.

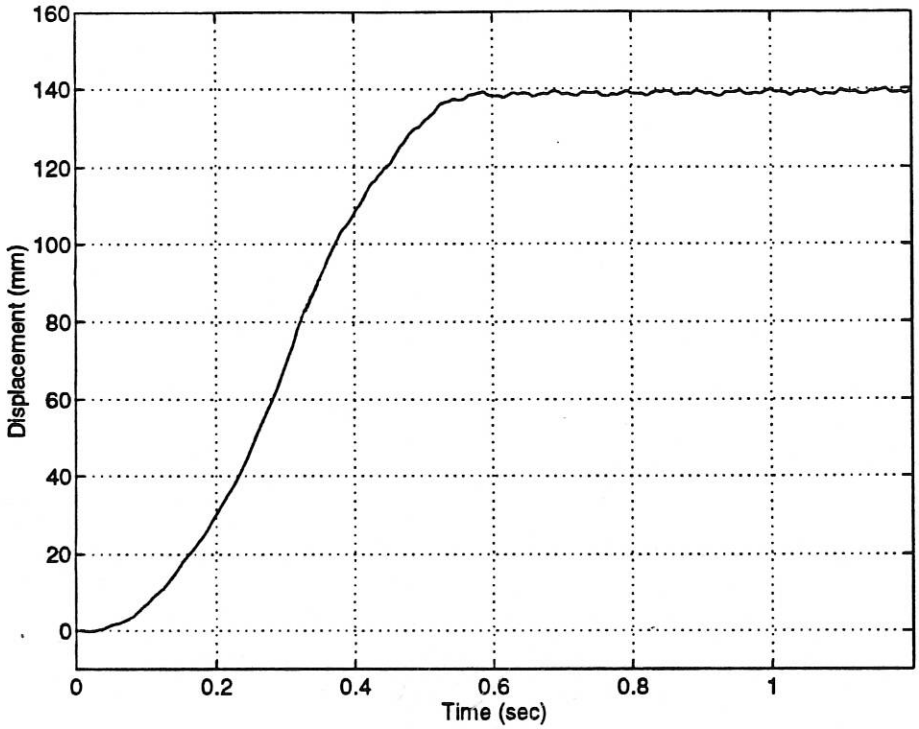



(a)

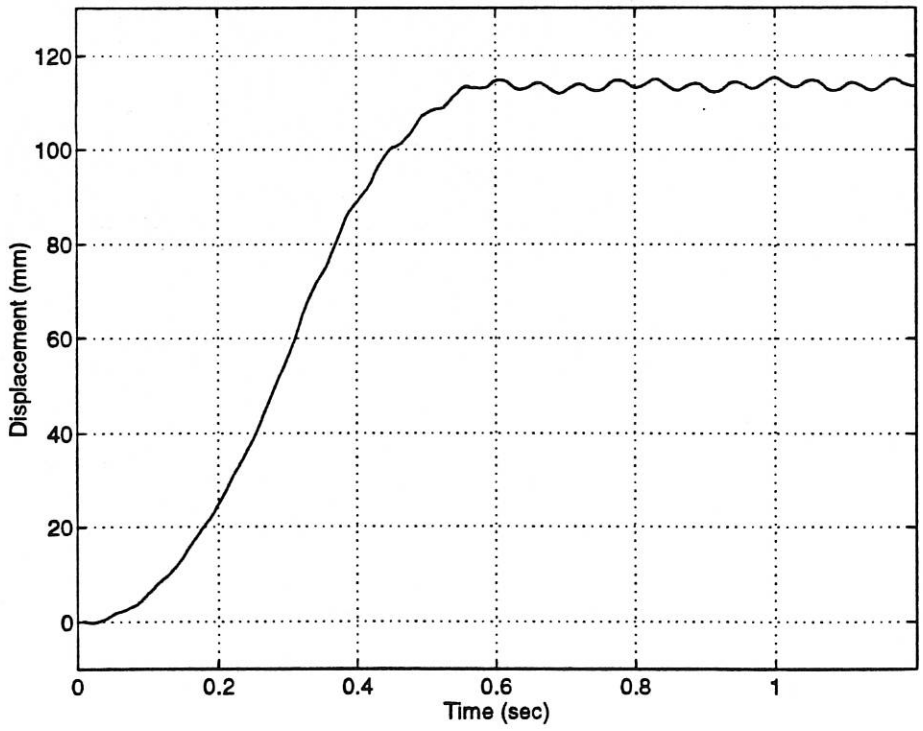



(b)

Figure 8: Flexible motion of the manipulator;
(a) First mode of vibration.
(b) Second mode of vibration.




(a)




(b)

Figure 9: End-point movement of the manipulator;
(a) With hub inertia and a payload of 10 grams.
(b) With hub inertia and a payload of 20 grams.

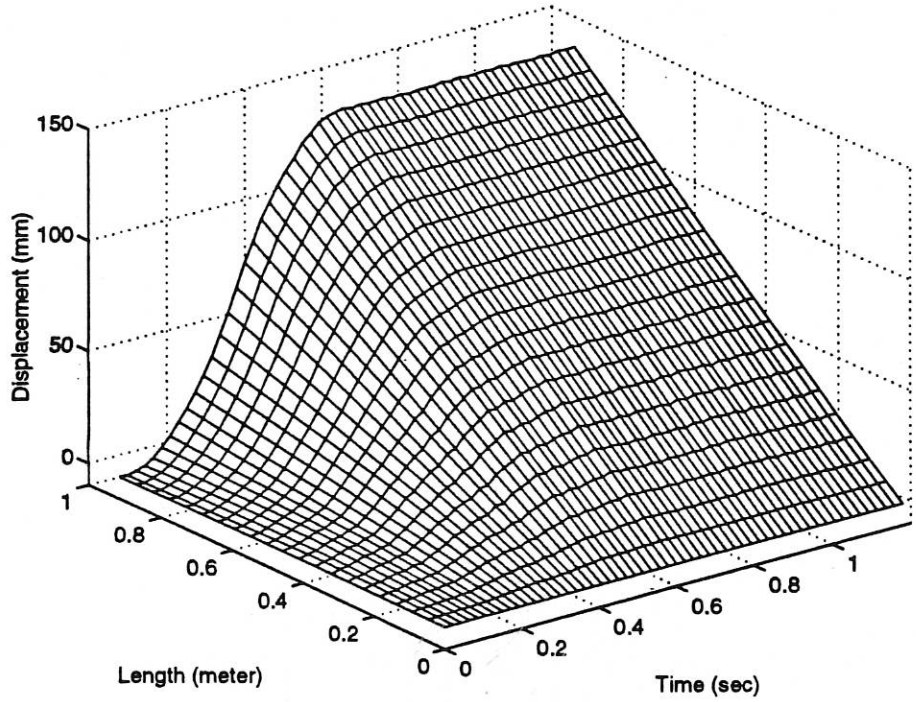



Figure 10: Movement of flexible manipulator with hub inertia and a payload of 10 grams.

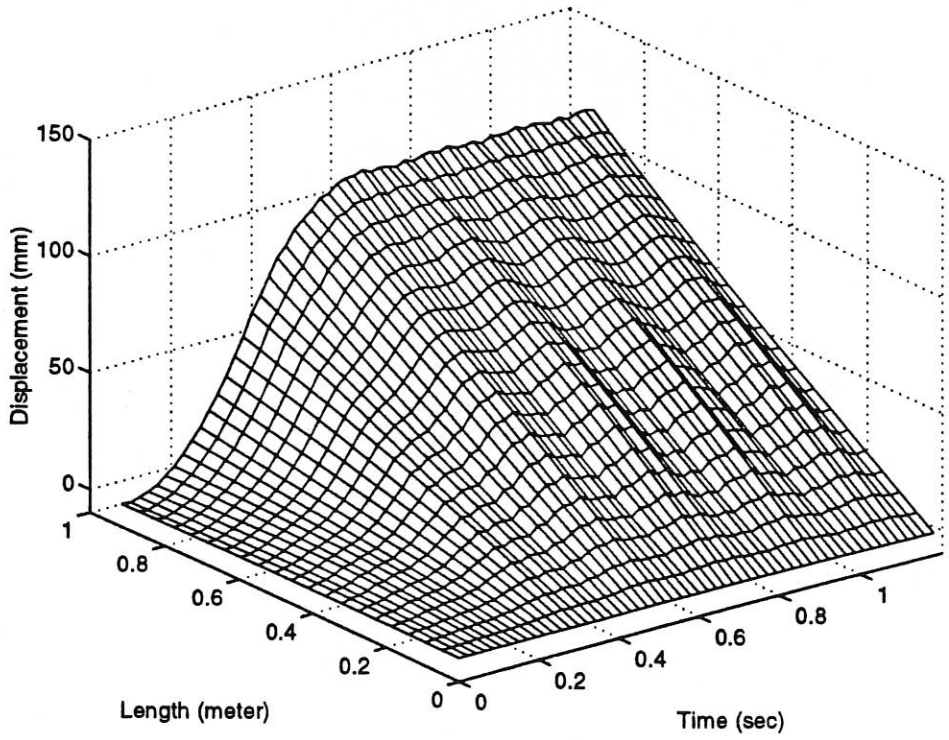



Figure 11: Movement of flexible manipulator with hub inertia and a payload of 20 grams.

SHEFFIELD UNIV
LIBRARY
ST. GEORGE'S