SICICI '92

Singapore International Conference on Intelligent Control and Instrumentation

PROCEEDINGS

17 - 21 FEBRUARY 1992
MARINA MANDARIN HOTEL SINGAPORE

Organised by:

IEEE Singapore Section

Co-organisers:

Instrumentation and
Control Society, (NMO, IFAC)

The Institution of Engineers,
Singapore

SYMBOLIC DERIVATION AND DYNAMIC SIMULATION OF FLEXIBLE MANIPULATORS

M.H. Korayem*, Dr. Y. Yao† and Dr. A. Basu*

*Department of Mechanical Engineering, University of Wollongong, NSW,2500 Australia
†School of Mechanical and Manufacturing Engineering,The University of New South Wales, Kensington, NSW,2033 Australia

ABSTRACT

This paper presents development of a symbolic derivation and dynamic simulation package for flexible manipulators using a PC-based symbolic language MATHEMATICA. The package, which takes the full advantages of the symbolic language, is meant to be versatile and applicable to multi-link flexible manipulators. A case study involving a two-link flexible manipulator by using recursive Lagrangian assumed mode method is presented. The advantages of expanding the dynamic equations into symbolic form and simulation results are discussed. Techniques for overcoming computer memory limitation, simplifying intermediate derivation, and improving efficiency of equation generation are also discussed.

NOMENCLATURE

K	kinetic energy of the system
V_g	gravitational potential energy
V_e	elastic potential energy
F_i	generalized forces
g	the gravity vector expressed at the base coordinate frame
M_i	a mass concentrated at the joint i
J	inertia matrix $\begin{bmatrix} [J_{jh}] & [J_{jhk}] \\ [J_{hjk}] & [I_{jfhk}] \end{bmatrix}$
z	the vector of generalized coordinates $[q_1,q_2,\ldots,q_h\ldots,q_n,q_{11},q_{12}\ldots,q_{1m_1},q_{21}\ldots, q_{2m_2}\ldots,q_{h1}\ldots,q_{hm_n}\ldots,q_{nm_n}]^T$
q_h	the joint variable of the h-th joint
q_{hk}	the deflection variable (amplitude) of the k-th model of link h
R	vector of remaining dynamics and external forcing terms $[R_1,R_2,\ldots,R_h\ldots,R_n,R_{11},R_{12}\ldots ,R_{1m_1},R_{21}\ldots,R_{2m_2}\ldots,R_{h1}\ldots,R_{hm_n}\ldots,R_{nm_n}]^T$
R_j	dynamics from the joint equation j
R_{jf}	dynamics from the deflection equation jf
r_i	the vector locating the centre of mass of link i
x_{ij}, y_{ij}, z_{ij}	the displacement component of mode j of link i's deflection along x_i, y_i and z_i
$\theta_{xij},\theta_{yij},\theta_{zjj}$	the x_i, y_i and z_i rotation components of link i, along x_i, y_i and z_i axes the length of link i
n	number of degrees of freedom
nm	number of degrees of mode
A_i	joint transformation relates system i the point before deflection to system i-1
E_i	link transformation relates the deflection of system i to system $\hat{i}$
W_i	the transformation from the base to the i-th link
h_i	a vector from the base origin to a point fixed in link i
μ_i	spatial variable along element i
$\hat{W}_i$	the transformation from the base to the system $\hat{i}$

1. INTRODUCTION

The advantages of symbolic derivation of dynamic equations of motion for robotic manipulators have been recognized in relation to the needs for insight to understanding of the system dynamics and for computational efficiency. The influence of various parameters, such as masses, lengths, different modes, flexural rigidities, etc., can be examined with relative ease. Expanding the vector/matrix equations of motion results equations which are even more computationally efficient than the efficient recursive Newton-Euler formulation in vector/matrix form [1].

Manual symbolic expansion of manipulator matrix equations is tedious, time-consuming, and error-prone, because of the significant complexity of intermediate steps. Automated derivation of the equations using a suitable symbolic language is desirable. Symbolic derivation of dynamic equations of manipulators has been reported and various computer programs have been written. For instance, Leu and Hemati [1] have presented a general computer procedure using the symbolic language MACSYMA. These programs are applicable only to rigid manipulators.

For flexible manipulators, it is nearly impossible to expand the equations symbolically by hand. The much greater complexity of flexible manipulator dynamics literally forbids any practical manual symbolic derivations. Therefore, the advantages promised by symbolic manipulation programs are even desirable for flexible manipulators. The symbolic derivation of flexible manipulator dynamic is a relatively new area. Cetinkunt and Book [2-3] have written a symbolic manipulation program based on SMP and simulated with a VAX-11/750 micro computer.

This paper presents a method for deriving flexible manipulator dynamic equations using a PC-based symbolic language

MATHEMATICA. MATHEMATICA was used mainly due to its versatile symbolic manipulation capabilities, such as, symbolic simplification of polynomials and rational expressions, linearization of trigonometric functions, automated evaluation of the relative significance of terms and subsequently, neglecting the less significant terms, and symbolic integration and differentiation [4]. It was used also because of the PC platform it runs on, its user friendliness and its integrated graphics environment. It can also communicate at a high level with other programs using the MathLink communication standard.

The first step in improving the performance of flexible manipulators is the development of a mathematical frame work for the modeling of these arms. Several recent works have addressed the general modeling problem. Book, and Majette [5] developed a formulation for a two beam component flexible arm, using Lagrange's equations of motion. Book [6] recently develop a recursive Lagrangian formulation for flexible manipulator. Interaction between gross motion and elastic deformation response is considered in the formulation system and inertia matrices are recursively calculated. However no simulation results are reported. The method employed here follows closely to that of [6], with slight modifications. Finally, a case study involving a two-link flexible arm is considered, where different number of modes are assumed and results are compared with the rigid case.

2. DYNAMIC MODELING OF A MULTI-LINK FLEXIBLE MANIPULATOR

To derive equations of motion for the manipulator, we describe the position of a point on the beam with a combination of a rigid body motion and flexible deflection using a Bernoulli-Euler beam model. A point along the link is described in a fixed reference coordinate system by two transformations between the coordinate systems. The joint transformation A_i relates system i, the point before deflection, to system i-1. The link transformation E_i relates the deflection of system i to system $\hat{i}$. Let ${}^i h_i$ be a vector from a point fixed in link i with respect to O_{ixyz}, then h_i a vector from the base origin to a point fixed in link i is given by

$$h_i = W_j {}^i h_i \tag{1}$$

where W_j is the transformation from the base to the i-th link

$$W_j = W_{j-1} E_{j-1} A_j = \hat{W}_{j-1} A_j \tag{2}$$

$$E_i = H_i + \sum_{j=1}^{m_i} q_{ij} M_{ij} \tag{3}$$

$$H_j = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_i & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4a}$$

$$M_{ij} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ x_{ij} & 0 & -q_{zij} & q_{yij} \\ y_{ij} & q_{zij} & 0 & -q_{xij} \\ z_{ij} & -q_{yij} & q_{xij} & 0 \end{bmatrix} \tag{4b}$$

To find the velocity of a point on link i take the time derivative of the position:

$$\dot{h}_i = \dot{W}_i {}^i h_i + W_i {}^i \dot{h}_i \tag{5}$$

By differentiating (2), one obtains

$$\dot{W}_j = \dot{\hat{W}}_{j-1} A_j + \hat{W}_{j-1} \dot{A}_j \tag{6}$$

$$\ddot{W}_j = \ddot{\hat{W}}_{j-1} A_j + 2 \dot{\hat{W}}_{j-1} \dot{A}_j + \hat{W}_{j-1} \ddot{A}_j \tag{7}$$

where $\dot{A}_j = U_j \dot{q}_j$ (8)

$$\ddot{A}_j = U_{2j} \dot{q}_j^2 + U_j \ddot{q}_j \tag{9}$$

$$U_j = \partial A_j / \partial q_j$$

$$U_{2j} = \partial^2 A_j / \partial \dot{q}_j^2$$

We also need $\dot{\hat{W}}_j$, $\dot{W}_j$ and $\ddot{\hat{W}}_j$. These can be computed:

$$\dot{\hat{W}}_j = W_j E_j \tag{10}$$

$$\dot{W}_j = \dot{W}_j E_j + W_j \dot{E}_j \tag{11}$$

$$\ddot{W}_j = \ddot{W}_j E_j + 2 \dot{W}_j \dot{E}_j + W_j \ddot{E}_j \tag{12}$$

Once the kinematic of the system is set up, by using Lagrange's equations of motion, the dynamic equation of a flexible manipulator is obtained with generalized coordinates.

1. The joint equation j is given as

$$\frac{d}{dt}\left\{\frac{\partial K}{\partial \dot{q}_j}\right\} - \frac{\partial K}{\partial q_j} + \frac{\partial V_e}{\partial q_j} + \frac{\partial V_g}{\partial q_j} = F_j \tag{13}$$

2. The deflection equation jf is given as

$$\frac{d}{dt}\left\{\frac{\partial K}{\partial \dot{q}_{jf}}\right\} - \frac{\partial K}{\partial q_{jf}} + \frac{\partial V_e}{\partial q_{jf}} + \frac{\partial V_g}{\partial q_{jf}} = 0 \tag{14}$$

The resultant system of equations can be organised in matrix form as

$$\sum_{h=1}^{m_n} J_{jh} \ddot{q}_h + \sum_{h=k=1}^{m_n m_n} J_{jhk} \ddot{q}_{hk} = R_j \tag{15}$$

$$\sum_{h=1}^{m_n} J_{jhk} \ddot{q}_h + \sum_{h=k=1}^{m_n m_n} I_{jfhk} \ddot{q}_{hk} = R_{jf} \tag{16}$$

or $J\ddot{z} = R$ (17)

The elements of J and R are given in the appendix A and B.

3. SYMBOLIC IMPLEMENTATION OF THE ALGORITHEM

The matrix form of manipulator dynamic equation was expanded symbolically for any desired manipulator using symbolic manipulation programs Mathematica [2]. Mathematica was selected to implement this algorithm mainly for simplification and linearization of trigonometric function, and production of Fortran code ready for computation in solving dynamic equation and implementation of result in graphics form in reduced time. Based on the formulations described in the previous section, a user-friendly computer program has been developed in MAC-II to symbolically derive the dynamic equations of the flexible manipulator using MATHEMATICA.

Inputs to the program from the user are the following;

1) n — number of degrees of freedom
2) nm — number of degrees of mode
3) A_i — joint transformation
4) E_i — link transformation
5) g — the gravity vector
6) r_i — link mass centre position vector

Given these parameter and the development of the previous section, the algorithm implemented in Mathematica is as follows;

1) Select mode shape
2) Compute the matrices C_i, C_{ij}, C_{ijk}, K_{ijk}

3) Compute the transformation matrices $\mathbf{W}_i$ from Eq. 2
4) Compute derivative of $\mathbf{W}_i$ from Eqs. 6 and 9
5) Compute the transformation matrices $\hat{\mathbf{W}}_j$ from Eq. 10
6) Compute derivative of $\hat{\mathbf{W}}_j$ from Eqs. 11 and 12
7) Compute recursive expression from Eqs. A9 and A13
8) Compute R vector from Eqs. B1 and B13
9) Compute J vector according to Eqs. A1 and A8
9) Compute Eq. 17

With user inputs, such as the number of links and modes, the centre of mass and the gravitational field vectors, transformation matrices and its derivatives are generated and so are all coefficient matrices based on selected mode shapes. The recursive expressions ${}^{j}\mathbf{F}_h$, ${}^{j}\tilde{\mathbf{F}}_h$, ${}^{j}\emptyset_h$, $\mathbf{Q}_j$ and $\mathbf{P}_j$ required for R and J matrices are then derived. Substituting them in Eq. 17 results in the dynamic model which usually takes a very complicated form.

Table 1 MATHEMATICA program for generating dynamic equations

```
"/This program derives the equation of motion of a manipulator with flexible
link using the Lagrangian formulation./"
        For[j=1, j <=n,j++,
T[C][ß[j]]=µ/2 Integrate[{{1},{x},{0},{0}}.{{1,x,0,0}},{x,0,T[i][ß[j]]}];
        For[k=1, k <=nm,k++,
T[øz][ß[j],ß[k]];T[C][ß[j],ß[k]]; T[Ω][ß[j],ß[k]]
        For[f=1, f <=nm,f++,
T[øz][ß[j],ß[f]]=D[T[y][ß[j],ß[f]],x];T[K][ß[j],ß[k],ß[f]]=EI
Integrate[D[T[øz][ß[j],ß[f]],x]D[T[øz][ß[j],ß[k]],x]];T[C][ß[j],ß[k],ß[f]];T[D][ß
[j],ß[k]]=T[C][ß[j],ß[k]]+Sum[T[C][ß[j],ß[k],ß[f1]]T[q][ß[j],ß[f1]][t],{f1,1,nm
}]];T[r][ß[j]]=T[M][ß[j]]T[r][ß[r],ß[j]]+T[Me][ß[j]]T[r][ß[e],ß[j]]+Sum[T[Ω][ß[j
],ß[k2]]T[q][ß[j],ß[k2]][t],{k2,1,nm}];T[G][ß[j]]=T[C][ß[j]]+Sum[(T[C][ß[j],ß[
k1]]+Transpose[T[C][ß[j],ß[k1]]])T[q][ß[j],ß[k1]][t],{k1,1,nm}]]
"************** Transformation Matrix ***************************"
        For[j=1, j <=n,j++,
T[A][ß[j]];T[H][ß[j]];T[U][ß[j]]=D[T[A][ß[j]],T[q][ß[j]][t]];T[U2][ß[j]]=D[T[U][
ß[j]],T[q][ß[j]][t]];T[E][ß[j]]=T[H][ß[j]]+Sum[T[M][ß[j],ß[k]]T[q][ß[j],ß[k]][t],{
k,1,nm}];T[E][ß[j],π[T]]=Transpose[T[E][ß[j]]];T[DE][ß[j]]=D[T[E][ß[j]],t] ] };
        For[j=1, j <=n,j++,
T[W][ß[j]]   =T[WB][ß[j-1]].T[A][ß[j]];T[W][ß[j],π[T]]=Transpose[T[W][ß[j]]];
T[WB][ß[j]] =T[W][ß[j]].T[E][ß[j]];T[WB][ß[j],π[T]]=Transpose[T[WB][ß[j]] ];
T[DW][ß[j]]   =T[DWB][ß[j-1]].T[A][ß[j]]+T[WB][ß[j-1]].T[DA][ß[j]];
T[DWB][ß[j]]   =T[DW][ß[j]].T[E][ß[j]]+T[W][ß[j]].T[DE][ß[j]];
T[DDW][ß[v],ß[j]]=T[DDWB][ß[v],ß[j-1]].T[A][ß[j]]+2T[DWB][ß[j-
1]].T[DA][ß[j]]+T[WB][ß[j-1]].T[U2][ß[j]]Derivative[1][T[q][ß[j]]][t]^2;
T[DDWB][ß[v],ß[j]]=T[DDW][ß[v],ß[j]].T[E][ß[j]]+2T[DW][ß[j]].T[DE][ß[j] ];
"********** Recursive Expression jFBh,jFh and jØh ******************"
T[FB][ß[n],π[n]]=T[G][ß[n]];
        For[j=n-1,  j >=1,  j--,
            For[h=n,   h >j,   h--,
T[FB][ß[h],π[j]]=T[G][ß[h]]+T[E][ß[j]].T[A][ß[j+1]].T[FB][ß[h],π[j+1]] ] };
        For[j=n-1,  j >=1,  j--, h=j;
T[FB][ß[h],π[h]]=T[FB][ß[h+1],π[h]].T[A][ß[j+1],π[T]].T[E][ß[j],π[T]] ] };
        For[h=n-1,  h >=1,  h--,
            For[j=n,    j >h,   j--,
T[FB][ß[h],π[j]]=T[FB][ß[h+1],π[j]].T[A][ß[h+1],π[T]].T[E][ß[h],π[T]]] ];
        For[j=1, j <=n,j++,
            For[h=1, h <n,h++,
T[F][ß[h],π[j]]=T[G][ß[h]]+T[FB][ß[h+1],π[j]].T[A][ß[h+1],π[T]] ];
        For[j=1,  j <n,  j++,

            For[h=1,   h <n,  h++,
T[Ø][ß[h],π[j]]=T[A][ß[j+1]].T[FB][ß[h+1],π[j+1]].T[A][ß[j+1],π[T]]] ]
"**Vector of remaining dynamics and external forcing terms R ************"
        For[j=n, j <=n,j++,
T[Q][ß[n]]=T[DDW][ß[v],ß[n],π[T]]+2Sum[Derivative[1][T[q][ß[n],ß[kk]]][t]
T[D][ß[n],ß[kk]],{kk,1,nm}].T[DW][ß[n],π[T]];T[P][ß[n]]=T[r][ß[j]];
        For[f=1, f <=nm,f++,
T[R][ß[n],ß[f]]=2Trace[(T[DDW][ß[v],ß[n]].T[D][ß[n],ß[f]]+2T[DW][ß[n]],Sum
[Derivative[1][T[q][ß[n],ß[kk]]][t]T[C][ß[n],ß[kk],ß[f]],{kk,1,nm}]).
T[W][ß[n],π[T]]]-Sum[T[q][ß[n],ß[jj]][t]T[K][ß[n],ß[jj],ß[f]],{jj,1,nm}]+]]];
        For[j=1, j <n,j++,
T[Q][ß[j]]=T[DDW][ß[v],ß[j],π[T]]+2Sum[Derivative[1][T[q][ß[j],ß[kk]]][t]
T[D][ß[j],ß[kk]],{kk,1,nm}].T[DW][ß[j],π[T]]+T[E][ß[j]].T[Q][ß[j+1]];
T[P][ß[j]]=T[r][ß[j]]+T[E][ß[j]].T[A][ß[j+1]].T[P][ß[j+1]]
        For[f=1, f <=nm,f++,
T[R][ß[j],ß[f]]:=-2Trace[T[W][ß[j]].T[M][ß[j],ß[f]].T[A][ß[j+1]].T[Q][ß[j+1]]+
(T[DDW][ß[v],ß[j]].T[D][ß[j],ß[f]]+2T[DW][ß[j]].Sum[Derivative[1][T[q][ß[j],
ß[kk]]][t]T[C][ß[j],ß[f],ß[n],ß[kk]]).T[W][ß[j],π[T]]+T[E][ß[j]].T[R][ß[j+1]]];
T[J][ß[j],ß[h]]=2Trace[T[WB][ß[j1]].T[U][ß[j]].T[FB][ß[h],π[j]].T[U][ß[h],π[T]
].T[WB][ß[h-1],π[T]] ];T[J][ß[h],ß[j]]=T[J][ß[j],ß[h]] ] } ]
        For[k=1, k <=nm,k++,
            For[j=1, j <=n,j++,
T[J][ß[j],ß[n],ß[k]]= 2Trace[T[WB][ß[j-1]].T[U][ß[j]].T[WA][ß[n],π[j]].
T[D][ß[n],ß[k]].T[W][ß[n],π[T]] ];T[I][ß[j],ß[n],ß[k]]=T[J][ß[j],ß[n],ß[k]] ]]];
        For[k=1, k <=nm,k++,
            For[j=1, j <=n-1,j++,
                For[h=j, h <=n-1,h++,
T[J][ß[j],ß[h],ß[k]]=2Trace[T[WB][ß[j1]].T[U][ß[j]].(T[F][ß[h],π[j]].T[M][ß[h]
,ß[k],π[T]]+T[WA][ß[h],π[j]].T[D][ß[h],ß[k]]).T[W][ß[h],π[T]]] ]];
        For[k=1, k <=nm,k++,
            For[j=2, j <=n,j++,
                For[h=1, h <=j-1,h++,
T[J][ß[j],ß[h],ß[k]]=2Trace[T[WB][ß[j1]].T[F][ß[h],π[j]].T[M][ß[h],ß[k],π[T]].
T[W][ß[h],π[T]]];T[I][ß[j],ß[h],ß[k]]=T[J][ß[j],ß[h],ß[k]]]]]]
        For[k=1,k <=nm,k++,
            For[f=1, f <=nm,f++,
T[I][ß[n],ß[f],ß[n],ß[k]]=2Trace[T[S][ß[n],ß[k],ß[f]]+T[C][ß[n],ß[k],ß[f]] ]]];
        For[k=1,k <=nm,k++,
            For[f=1, f <=nm,f++,
                For[j=1, j <=n-1,j++,
T[I][ß[j],ß[f],ß[j],ß[k]]=2Trace[T[Je][ß[j]]T[S][ß[j],ß[k],ß[f]]+
T[M][ß[j],ß[f]].T[Ø][ß[j],π[j]].T[M][ß[j],ß[k],π[T]]+T[C][ß[j],ß[k],ß[f]]];]];
        For[k=1,k <=nm,k++,
            For[f=1, f <=nm,f++,
                . For[j=1, j <=n-1,j++,
T[I][ß[j],ß[f],ß[n],ß[k]]=2Trace[T[Je][ß[j]]T[S][ß[j],ß[k],ß[f]]+
T[W][ß[j]].T[M][ß[j],ß[f]].T[W][ß[n],π[j]].T[D][ß[n],ß[k]].T[W][ß[n],π[T]]];
T[I][ß[n],ß[k],ß[j],ß[f]]=T[I][ß[j],ß[f],ß[n],ß[k]]]]];
        For[j=1, j <=n-1,j++,
            For[h=j+1, h <=n-1,h++,
T[I][ß[j],ß[f],ß[h],ß[k]]=2Trace[T[Je][ß[j]]T[S][ß[j],ß[k],ß[f]]+
T[W][ß[j]].T[M][ß[j],ß[f]].(T[Ø][ß[h],π[j]].T[M][ß[h],ß[k],π[T]]+
T[W][ß[h],π[j]].D][ß[h],ß[k]]).T[W][ß[h],π[T]] ];
```

By using the available mathematical simplification processes, the model is reduced to a much simpler form and the relatively less significant terms, such as the second order terms of deflection are further neglected automatically.

4. EXAMPLES

Two examples are presented for a flexible manipulator. To compute dynamic equation, parameters such as density and Young's module etc must be provided. The bending deflection of links are approximated with two assumed mode shapes. Mode shapes are chosen from analytical solution of a Euler-Bernoulli beam eigenfunction analysis. The selection of best mode shapes for a given flexible beam has not been a clearly answered in problem [4]. One should be able to simulate the effect of different mode shapes on the system behaviour easily. For simplicity, the terms including the squares of deflections are neglected, since they are considerably smaller, compared with other terms. The effect of different mode shapes on the system are considered. To verify the model, results are compared with the same system with rigid arm.

4.1 Example 1: One Link Flexible Manipulator.

In this example, for the sake of simplicity, we deal with a single link, and planer manipulator arm. Gravity effect was ignored in this case study in order to isolate the dynamic flexibility effects. The flexible link is a 1 m steel beam, whose cross section is 5 mm by 100 mm, mass density is μ= 7.86 x 10^{-6} kg/mm^3, and Young's module is E=2.1x10^{11} pa. Figure 1a shows the torque pattern to be exerted by the actuator. The responses were computed by solving dynamic equation, where the arm is assumed to be completely rigid (Fig. 1b). Figure 2 shows the response of the flexible arm when the torque

commend in Fig. 1a is applied.

4.2 Example 2: Two Link Flexible Manipulator.

In this example, only the link flexibilities are considered and the joint flexibilities are not included. The bending deflections of links are approximated with simply supported mode shape for each link. Now, let us consider the case that one would like to use different set of mode shapes. The necessary change required in the model is to re-evaluate C_i, C_{ij}, C_{ijk}, K_{ijk} terms with new mode shapes. For the case where the robot becomes more rigid, EI becomes larger, joint variable response of the system converges to the rigid arm. The responses were computed by solving dynamic equation, where comparison of the flexible responses of the system with rigid response are shown in Fig. 3. Comparison of the flexible torques of the system with load and without payload are shown in Fig. 4.

5. DISCUSSION

The automatic equation derivation process is highly desirable while manual symbolic expansion of manipulator matrix equations is tedious, time-consuming, and error-prone. The derivation of dynamic equations involves a large number of symbolic operations. Deriving the dynamic equations symbolically, insight on the dynamics of a manipulator can be generated in two ways; one is examining the terms of the dynamic equations and the other is using the dynamic equations to simulate individual force components.

Another merit of expanding the vector/matrix equations of motion is that, if most manipulator links are symmetric in geometry, the resultant equations are more computationally efficient than even the efficient recursive Newton-Euler formulation in vector/matrix form which is in a numerical approach. Efficiency in deriving the dynamic equations is another factor to be considered. Improvement to the efficiency can be achieved by exploiting the symmetry of the coefficients and only operating on the diagonal elements of a product of diagonal matrices, instead of multiplying them explicitly.

This paper presents a technique for deriving the scalar form of flexible manipulator dynamic equations by symbolically expanding the Lagrange's equations using the symbolic language Mathematica. The derivation processes are shown in Table 1. In the present method, other than the multiplication of each of the two matrices explicitly, the unnecessary operation parts which quickly reach a zero value are eliminated. Especially, in the computation of ${}^{j}F_h$, ${}^{j}\tilde{F}_h$, ${}^{j}\emptyset_h$, Q_j and P_j using matrices, many terms are computed which eventually are dropped (since at the end of chain of matrix multiplications they are high order of deformation). Moreover, the other contribution of the paper is to simplify trigonometric function by using TrigLinear in each necessary step. Furthermore, it is noted that the computer system applied in this paper is a MAC-II only, and the present method becomes a very easy and powerful tool for deriving the dynamic equations of motion of flexible multi-link.

6. SUMMARY AND CONCLUSION

This paper presents a technique for deriving the scalar form of flexible manipulator dynamic equations by symbolically expanding Lagrange matrix equations using MATHEMATICA. The algebraic dynamic robot modeling program has been implemented to enable the control engineer to formulate and gain physical insight into Lagrangian dynamic robot models for the systematic design. The advantages of the method presented in this paper include less significant terms can be examined and neglected in different phases of the symbolic derivation, number of modes and mode shapes can be easily varied and evaluated, and insights to influences of various terms on the manipulator dynamics can be achieved. Simulation results are discussed and shown that the method worked very well for this example case.

REFERENCES

1. Leu, M.C., and Hemati, N, "Automated symbolic derivation of dynamic equations of motion for robotic manipulators, " *J. Dyn. Syst. Msmt Control.* 108: 1986, 172-170.
2. Cetinkunt, S., and Book, W.J., "Symbolic modeling of flexible manipulators,"*Proceedings of 1987 IEEE International Conference on Robotics and Automation*, March 31-April 3 1987, Raleigh, NC, Vol. 3, pp. 2074-2080.
3. Cetinkunt, S., and Book, W.J., "Symbolic modeling and dynamic simulation of robotic manipulators with compliant links and joints," *Robotics & Computer-Integrated Manufacturing*, Vol. 5, No. 4, 1989, pp. 301-310.
4. Wolfram, S., Mathematica, Addison-Wesley, 1991.
5. Book, W.J., Majette, M, " Controller Design for Flexible manipulator, Distributed Parameter Mechanical Arms Via Combined State Space and Frequency Domain Technique", Journal of Dynamic Systems, Measurement, and Control, Vol. 105, PP 245-254, Dec, 1983.
6. Book, W.J., "Recursive lagrangian dynamics of flexible manipulator arm," *International Journal of Robotics Research*, Vol. 3 No. 3, 1984, pp. 87-101.

Appendix A

The elements of J matrices are given below;

$$J_{jh}=2Tr\{\hat{W}_{j-1}U_j{}^{j}\tilde{F}_h U_h^T \hat{W}_{h-1}^T\} \quad (A.1)$$

For h=n ; j=1,...n

$$J_{jhk}=2Tr\{\hat{W}_{j-1}U_j{}^{j}\tilde{W}_n D_{nk} W_n^T\} \quad (A.2)$$

For h=j,...n-1 ; j=1,...n-1

$$J_{jhk}=2Tr\{\hat{W}_{j-1}U_j[{}^{j}F_h M_{hk}^T+{}^{j}\tilde{W}_h D_{hk}]W_h^T\} \quad (A.3)$$

For h=1...j-1 ; j=2,...n

$$J_{jhk}=2Tr\{\hat{W}_{j-1}U_j[{}^{j}F_h M_{hk}^T]W_h^T\} \quad (A.4)$$

For h=j=n

$$I_{nfnk}=2Tr\{C_{nkf}\} \quad (A.5)$$

For h=j=1,...n-1

$$I_{jfjk}=2Tr\{M_{jf}{}^{j}\emptyset_j M_{jk}^T+C_{jkf}\} \quad (A.6)$$

For h=n ; j=1,...n-1

$$I_{jfnk}=2Tr\{W_j M_{jf}{}^{j}W_n D_{nk} W_h^T\} \quad (A.7)$$

For h=1,...n-1 ; j=j+1,...n-1

$$I_{jfhk}=2Tr\{W_j M_{jf}[{}^{j}\emptyset_h M_{hk}^T+{}^{j}W_h D_{hk}]W_h^T \quad (A.8)$$

For h=j=n

$${}^{n}\tilde{F}_n=G_n \quad (A.9)$$

For j<h<=n

$${}^{j}\tilde{F}_h=E_j A_{j+1}{}^{j}\tilde{F}_h \quad (A.10)$$

For j=h<n

$${}^{h}\tilde{F}_n=G_n+{}^{h}\tilde{F}_{h+1}(E_j A_{j+1})^T \quad (A.11)$$

For h =1,.....n-1 ; j =1,.....n

$${}^{j}F_h={}^{h}\tilde{F}_{h+1}A_{h+1}^T \quad (A.12)$$

For j=1,...n-1 ; h=1,...n

$${}^{j}\emptyset_h=A_{j+1}{}^{h}\tilde{F}_{h+1}(A_{h+1})^T \quad (A.13)$$

Appendix B

In this appendix the elements of R matrix are given below

$$R_1=-2Tr\{U_1Q_1\}+g^TU_1P_1+F_1, \quad (B.1)$$

$$R_j=-2Tr\{\hat{W}_{j-1}U_jQ_j\}+g^T\hat{W}_{j-1}U_jP_j+F_j, \quad (B.2)$$

$$R_{nf}=-2Tr\{[\ddot{W}_{vn}D_{nf}+2\dot{W}_n\sum_{k=1}^{m_n}\dot{q}_{nk}C_{nkf}]W_n^T\}$$

$$-\sum_{k=1}^{m_n}q_{nk}K_{nkf}+g^TW_n\Omega_{nf} \quad (B.3)$$

$$R_{jf}=-2Tr\{W_jM_{jf}A_{j+1}Q_{j+1}+$$

$$[\ddot{W}_{vj}D_{jf}+2\dot{W}_j\sum_{k=1}^{m_n}\dot{q}_{jk}C_{jkf}]W_j^T\}-$$

$$\sum_{k=1}^{m_n}q_{jk}K_{jkf}+g^TW_jM_{jf}A_{j+1}P_{j+1}+g^TW_j\Omega_{jf} \quad (B.4)$$

$$Q_n=G_n\dot{W}_{nv}^T+2(\sum_{k=1}^{m_n}\dot{q}_{nk}D_{nk})\dot{W}_n^T \quad (B.5)$$

$$Q_j=G_j\ddot{W}_{nj}^T+2(\sum_{k=i}^{m_n}\dot{q}_{jk}D_{jk})\dot{W}_j^T+E_jA_{j+1}Q_{j+1} \quad (B.6)$$

$$P_n=M_nr_{rn}+\sum_{k=1}^{m_n}q_{nk}\Omega_{nf} \quad (B.7)$$

$$P_n=M_jr_{rj}+\sum_{k=1}^{m_n}q_{jk}\Omega_{jk}+E_jA_{j+1}P_{j+1} \quad (B.8)$$

$$\ddot{W}_{vj}=\ddot{\hat{W}}_{v,j-1}A_j+2\dot{\hat{W}}_{j-1}\dot{A}_j+\hat{W}_{j-1}U_{2j}\dot{q}_j^2 \quad (B.9)$$

$$\dot{\hat{W}}_{v,j}=\ddot{W}_{v,j}E_j+2\dot{W}_j\dot{E}_j \quad (B.10)$$

$$D_{jk}=C_{jk}+\sum_{k=1}^{m_n}q_{il}C_{ilk} \quad (B.11)$$

$$G_j=C_j+\sum_{k=1}^{m_n}q_{il}((C_{ik}+C_{ik}^T)+\sum_{k=1}^{m_n}q_{il}C_{ilk}) \quad (B.12)$$

$$C_i=1/2\int_0^{l_i}[1,\mu_i,0,0]^T[1,\mu_i,0,0]dm_i \quad (B.13)$$

$$C_{ij}=1/2\int_0^{l_i}[1,\mu_i,0,0]^T[0,x_{ij},y_{ij},z_{ij}]dm_i \quad (B.14)$$

$$C_{ijk}=1/2\int_0^{l_i}[0,x_{ij},y_{ij},z_{ij}]^T[0,x_{ij},y_{ij},z_{ij}]dm_i \quad (B.15)$$

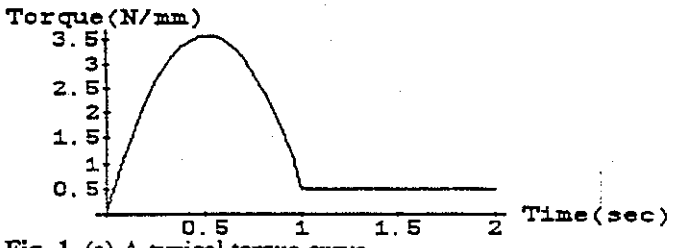
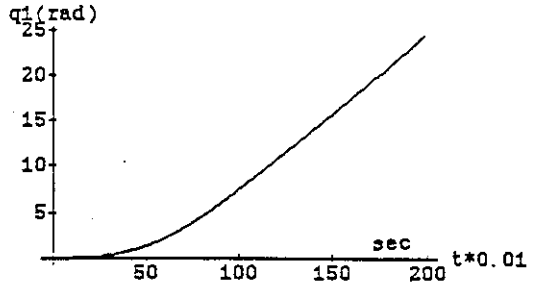


Fig. 1. (a) A typical torque curve



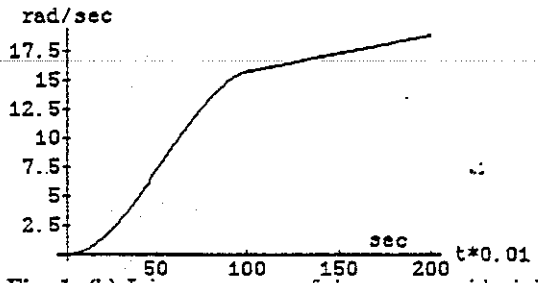
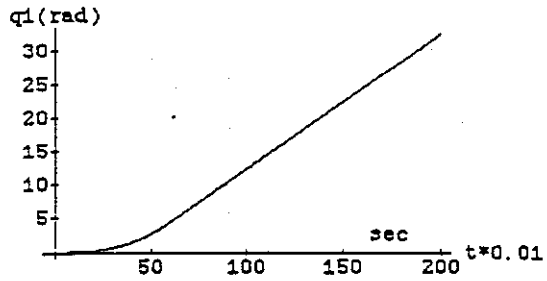
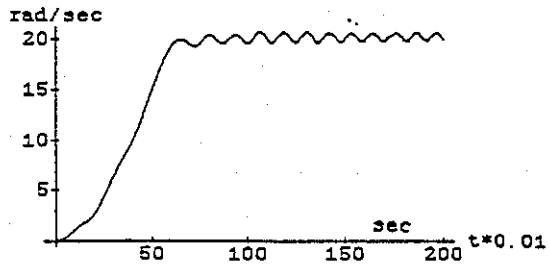
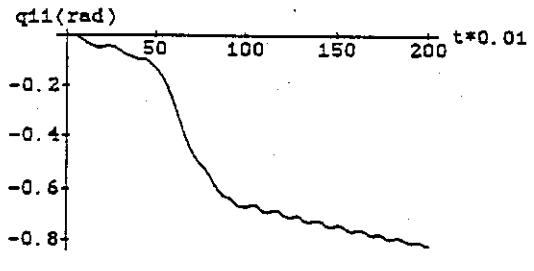


Fig. 1. (b) Joint responses of the system with rigid link.







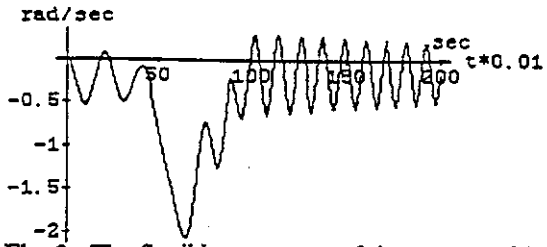


Fig. 2. The flexible responses of the system with simply supported mode shape.

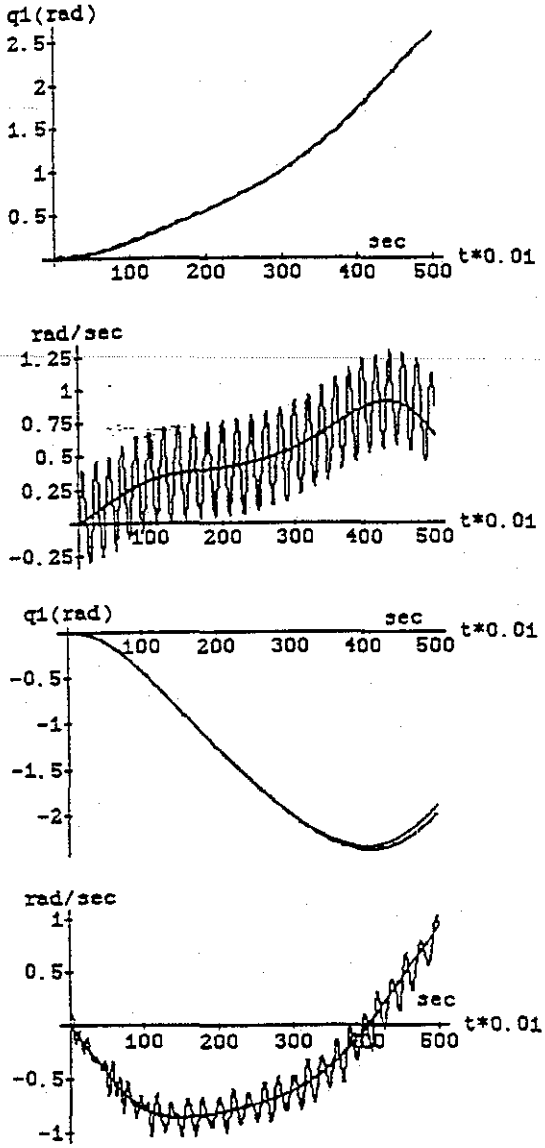


Fig. 3. Comparison of the flexible responses of the system with rigid link.

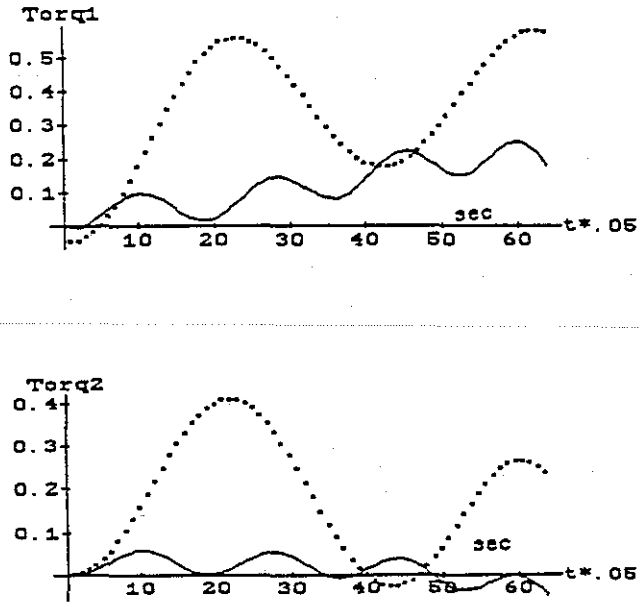


Fig. 4. Comparison of the torques of the system with load and without load at joint 1 and 2.