Int J Adv Manuf Technol (1994) 9:343–350

The International Journal of
Advanced
Manufacturing
Technology

Application of Symbolic Manipulation to Inverse Dynamics and Kinematics of Elastic Robots

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

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

An inverse dynamics and kinematics of a flexible manipulator is derived in symbolic form based on the recursive Lagrangian assumed mode method. A PC-based program has implemented the algorithm to automatically generate the inverse dynamics and kinematics for an elastic robot in a symbolic form. A case study is given to illustrate how to use this program for inverse dynamic and kinematic generation. Simulation results for a case study by considering different mode shape are compared with the rigid case.

Keywords: Symbolic modelling; Elastic; Inverse dynamics

1. Introduction

The automatic equation derivation process is highly desirable for manipulators because the manual symbolic expansion of manipulator matrix equations is tedious, time-consuming, and error-prone. The symbolic manipulation programs provide a way to overcome the disadvantages of the method. Symbolic derivation allows one to expand the dynamic equations in a very short time, check the element of dynamic equations and manipulate them very conveniently. Although Newton–Euler formalism has been regarded as computationally efficient and that of the Lagrange as insightful in representing manipulator dynamics, if the vector/matrix equations were expanded symbolically to scalar form, the results from the Newton–Euler and Lagrange formalisms would be equivalent. The expanded scalar equations would not only provide insight into the system dynamics, but also result in faster computation than the numerical approach based on either formalism. Hence, there has been growing interest in the use of automatic symbolic generation for the production of computer code for the dynamic calculation. A fundamental benefit of symbolic generation is that this code can be "customised" to take advantage of the kinematic and dynamic structure of the manipulator (e.g. link lengths of zero, sparse inertia matrices to eliminate much unnecessary computation.

The inverse dynamics of manipulator arms have been studied extensively for rigid link models [1, 2]. Deformations of arm links, however, are not negligible as the speed of motion and the required accuracy increase. The design engineer faces having to predict dynamic responses and link deformations based on an appropriate model of the flexible arm. Among a number of modelling methods previously presented, Cannon and Schmitz [3] considered the end point feedback control of a one-link manipulator and achieved a fine positioning, despite the structural flexibility. Book and Majette [4] developed a formulation for a two-link flexible arm, using Lagrange's equations of motion. Other contributions have since been made [5–9]. Book [10] also developed an efficient formulation based on recursive Lagrangian dynamics.

A mathematical model of elastic robot dynamics can be summarised in the following form:

1. The joint equation j is given as

$$\frac{\mathrm{d}}{\mathrm{d}t}\left\{\frac{\partial K}{\partial \dot{q}_j}\right\} - \frac{\partial K}{\partial q_j} + \frac{\partial V}{\partial q_j} = F_j \tag{1}$$

2. The deflection equation jf is given as

$$\frac{\mathrm{d}}{\mathrm{d}t}\left\{\frac{\partial K}{\partial \dot{q}_{jf}}\right\} - \frac{\partial K}{\partial q_{jf}} + \frac{\partial V}{\partial q_{jf}} = 0 \tag{2}$$

The inverse dynamics problem involves the calculation of the joint torque or forces using equations (1) and (2) when the kinematic state of the manipulator is given. Algorithms perform this computation of complex but basically straightforward algebraic expressions.

This paper presents a method for deriving flexible manipulator inverse dynamic equations and kinematics using a PC-based symbolic language MATHEMATICA® [11]. This article is organised as follows. Section 2 presents mathematical modelling of a flexible arm and Section 3 describes an inverse kinematics algorithm coupled with the dynamic equation. In Section 4 a brief description of the inverse dynamics is given. Section 5 contains symbolic implementation based on

Correspondence and offprint requests to: M. H. Korayem, Locked Bag 8844, South Coast Mail Center, NSW 2521, Australia.

MATHEMATICA. Finally, an example involving a two-link flexible arm is considered, where different mode shapes are compared with the rigid case.

2. Mathematical Modelling of a Flexible Arm

For a flexible arm we use the Euler–Bernoulli model, for which rotary inertia and shear deformation effects are ignored. A solution of the flexible motion of the link can be obtained through modal analysis, under the assumption of small deflection of the link,

$$y(\mu,t) = \sum_{i=1}^{m_j} q_{il}(t)\, y_i(\mu) \tag{3}$$

where y_i is the eigenfunction expressing the displacement of assumed mode i of link deflection. For a clamp-free vibrating beam the orthonormal model eigenfunctions in (2) are given by

$$y_i(\mu) = \sin(\beta_i\mu) - \sinh(\beta_i\mu) + \gamma\,(\cos(\beta_i\mu) - \cosh(\beta_i\mu))$$

$$\text{where } \gamma = \frac{\sin(\beta_i\mu)+\sinh(\beta_i\mu)}{\cos(\beta_i\mu)+\cosh(\beta_i\mu)} \quad (i=1,2,\ldots,m)$$

$$\text{and } \beta^4 = \frac{\mu(\omega_i)^2}{EI} \tag{4}$$

For a simply supported case the general solution of Bernoulli beam theory is given by

$$y_i(\mu) = a_i \sin(\beta_i\mu) \tag{5}$$

where a_i are arbitrary constants.

3. Inverse Kinematics

Inverse kinematics provides information for the joint variables by giving the motion of the end effector. We first compute the manipulator Jacobian associated with the infinitesimal translation and rotation of the end effector and we denote the infinitesimal end effector translation and rotation by vectors $\mathbf{dX_e}$ and $\mathbf{d\Phi_e}$, respectively. For convenience, we combine the two vectors and define the following vector $\mathbf{dp}$ as

$$\mathbf{dp}=[\mathbf{dX_e d\Phi_e}]^{\mathrm{T}} \tag{6}$$

Dividing both sides by dt, we obtain the velocity and angular velocity of the end effector.

$$\dot{\mathbf{p}}=[\mathbf{V_e\dot{\omega}_e}]^{\mathrm{T}} \tag{7}$$

On the other hand, $\mathbf{dp}$ can be written, owing to the differential change of every joint variable as well as the differential change of displacements at the free end of the end link. Therefore,

$$\mathbf{dp}=\mathbf{J_r dq_r}+\mathbf{J_f dq_f} \tag{8}$$

Dividing both sides by dt, the end effector velocity and angular velocity can also be obtained by using the manipulator Jacobian:

$$\dot{\mathbf{p}}=\mathbf{J_r\dot{q}_r}+\mathbf{J_f\dot{q}_f} \tag{9}$$

By differentiating (6), one obtains

$$\ddot{\mathbf{p}}=\mathbf{J_r\ddot{q}_r}+\mathbf{J_f\ddot{q}_f}+\mathbf{\dot{J}_r\dot{q}_r}+\mathbf{\dot{J}_f\dot{q}_f} \tag{10}$$

or

$$\mathbf{J_r\ddot{q}_r}+\mathbf{J_f\ddot{q}_f}=\ddot{\mathbf{p}}-\mathbf{\dot{J}_r\dot{q}_r}-\mathbf{\dot{J}_f\dot{q}_f}=\mathbf{R_t} \tag{11}$$

The above expression in terms of generalised coordinates is essential for solving the inverse dynamic equation. A problem in solving the equations of motion is that the joint torques are unknown and we cannot solve the dynamic equation by itself. To obtain exact solutions for generalised coordinates, equation (11) must be solved simultaneously with deflection equation (2) for a given trajectory. The equations are highly coupled and nonlinear and by using the symbolic derivation package MATHEMATICA we can solve them by neglecting insignificant terms such as second-order deformations.

4. Inverse Dynamics

This section follows generally reference [10] and is included for completeness of the paper. The resultant system of the dynamic equation of a flexible manipulator (equations (1) and (2)) can be organised in matrix form as

$$\mathbf{J\ddot{z}}=\mathbf{R} \tag{12}$$

where the elements of $\mathbf{J}$ matrices and the elements of $\mathbf{R}$ matrix are given below

$$J_{jh} = 2Tr\{\hat{\mathbf{W}}_{j-1}\,\mathbf{U}_j\,{}^{j}\bar{\mathbf{F}}_h\,\mathbf{U}_h^{\mathrm{T}}\,\hat{\mathbf{W}}_{h-1}^{\mathrm{T}}\} \tag{13}$$

$$J_{jhk} = 2Tr\{\hat{\mathbf{W}}_{j-1}\,\mathbf{U}_j\,{}^{j}\bar{\mathbf{W}}_n\,\mathbf{D}_{nk}\,\mathbf{W}_n^{\mathrm{T}}\}$$
$$(h=n;\quad j=1,\ldots,n) \tag{14}$$

$$J_{jhk} = 2Tr\{\hat{\mathbf{W}}_{j-1}\,\mathbf{U}_j[{}^{j}\mathbf{F}_h\,\mathbf{M}_{hk}^{\mathrm{T}} + {}^{j}\bar{\mathbf{W}}_h\,\mathbf{D}_{hk}]\,\mathbf{W}_h^{\mathrm{T}}\}$$
$$(h=j,\ldots,n-1;\quad j=1,\ldots,n-1) \tag{15}$$

$$J_{jhk} = 2Tr\{\hat{\mathbf{W}}_{j-1}\,\mathbf{U}_j[{}^{j}\mathbf{F}_h\,\mathbf{M}_{hk}^{\mathrm{T}}]\,\mathbf{W}_h^{\mathrm{T}}\}$$
$$(h=1,\ldots,j-1;\quad j=2,\ldots,n) \tag{16}$$

$$I_{nfnk} = 2Tr\{\mathbf{C}_{nkf}\}\quad (h=j=n) \tag{17}$$

$$I_{jfjk} = 2Tr\{\mathbf{M}_{jf}\,{}^{j}\mathbf{Ø}_j\,\mathbf{M}_{jk}^{\mathrm{T}} + \mathbf{C}_{jkf}\}$$
$$(h=j=1,\ldots,n-1) \tag{18}$$

$$I_{jfnk} = 2Tr\{\mathbf{W}_j\,\mathbf{M}_{jf}\,{}^{j}\mathbf{W}_n\,\mathbf{D}_{nk}\,\mathbf{W}_h^{\mathrm{T}}\}$$
$$(h=n;\quad j=1,\ldots,n-1) \tag{19}$$

$$I_{jfhk} = 2Tr\{\mathbf{W}_j\,\mathbf{M}_{jf}[{}^{j}\mathbf{Ø}_h\mathbf{M}_{hk}^{\mathrm{T}} + {}^{j}\mathbf{W}_h\,\mathbf{D}_{hk}]\,\mathbf{W}_h^{\mathrm{T}}$$
$$(h=1,\ldots,n-1;\quad j=j+1,\ldots,n-1) \tag{20}$$

$$R_1 = -2Tr\{\mathbf{U}_1\,\mathbf{Q}_1\} + \mathbf{g}^{\mathrm{T}}\mathbf{U}_1\mathbf{P}_1 + F_1 \tag{21}$$

$$R_j = -2Tr\{\hat{\mathbf{W}}_{j-1}\,\mathbf{U}_j\,\mathbf{Q}_j\} + \mathbf{g}^{\mathrm{T}}\,\hat{\mathbf{W}}_{j-1}\,\mathbf{U}_j\,\mathbf{P}_j + F_j \tag{22}$$

$$R_{nf} = -2Tr\left\{\left[\ddot{\mathbf{W}}_{vn}\,\mathbf{D}_{nf} + 2\dot{\mathbf{W}}_n \sum_{k=1}^{m_n} \dot{q}_{nk}\,\mathbf{C}_{nkf}\right]\mathbf{W}_n^{\mathrm{T}}\right\}$$

$$- \sum_{k=1}^{m_n} q_{nk}\,\mathbf{K}_{nkf} + \mathbf{g}^{\mathrm{T}}\,\mathbf{W}_n\,\Omega_{nf} \tag{23}$$

$$R_{jf} = -2Tr\Big\{\mathbf{W}_j\,\mathbf{M}_{jf}\,\mathbf{A}_{j+1}\,\mathbf{Q}_{j+1}$$

$$+ \left[\ddot{\mathbf{W}}_{vj}\,\mathbf{D}_{jf} + 2\dot{\mathbf{W}}_j \sum_{k=1}^{m_n} \dot{\mathbf{q}}_{jk}\,\mathbf{C}_{jkf}\right] \tag{24}$$

Required recursive expression and matrices for **J** and **R** are given in Appendix A.

5. Symbolic Implementation Based on Mathematica

The matrix form of the manipulator inverse dynamic equation was expanded symbolically for any desired manipulator using symbolic manipulation programs MATHEMATICA [15]. Based on the formulations described in the previous section, inputs to the program from the user are the following:

1. n total number of links
2. m_i number of modes used to describe the deflection of link i
3. l_i length of link i
4. $\mathbf{g}$ gravity vector
5. $\mathbf{r}_i$ link mass centre position vector
6. μ link density, and
7. $(EI)_i$ flexural rigidity of link i

The codes for generating dynamic equations are listed in Appendix B. The program uses the following mathematical operators of MATHEMATICA for symbolic manipulation:

1. To define a matrix one writes the elements one by one. For example, to generate $A=[B_{ij}]_{4\times 4}$ write

   A = {{B$_{11}$,B$_{12}$,B$_{13}$,B$_{14}$}, {B$_{21}$,B$_{22}$,B$_{23}$,B$_{24}$},
   {B$_{31}$,B$_{32}$,B$_{33}$,B$_{34}$}, {B$_{41}$,B$_{42}$,B$_{43}$,B$_{44}$}}
2. To multiply two matrices D and C one simply writes D.C
3. To differentiate and to integrate a matrix F by a variable x one writes

   D[F,x] ,Integrate[F,{x,0,x}]
4. To simplify the expression of a function one uses Simplify for algebraic simplification and TrigLinear for trigonometric simplification.
5. To replace approximate real numbers in expression that are close to zero by the exact integer 0 one simply writes

   Chop[*expr*]
6. To eliminate insignificant terms (x^2) one simply writes

   xfree[e_]: = FreeQ[e, x^2], Select[*expr*, x^2]

With users inputs, substituting ${}^j\mathbf{F}_h, {}^j\bar{\mathbf{F}}_h, {}^j\mathbf{Ø}_h, \mathbf{Q}_j$ and $\mathbf{P}_j$ in equation (12) results in the dynamic model which usually takes a very complicated form. 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.

The computational procedure is summarised below:

Step 1: Select mode shapes
Step 2: Derive recursive expressions ${}^j\mathbf{F}_h, {}^j\bar{\mathbf{F}}_h, {}^j\mathbf{Ø}_h, \mathbf{Q}_j$ and $\mathbf{P}_j$
Step 3: Derive **J** and **R**
Step 4: Assemble recursive Lagrangian dynamic equation (12)
Step 5: Specify dynamic trajectory
Step 6: Compute Jacobian matrix and equation (11)
Step 7: Solve coupled nonlinear differential equations (11) and (2)
Step 8: Calculate joint torques from joint equation (1)

The derivation of dynamic equations of motion for a two-link flexible manipulator is presented as an illustrative example. Simplified forms are compared with the rigid case and the simulation results are shown.

Example. Two Link Flexible Manipulator

6.1 Expansion of Dynamic Equation

By using the computational procedure as mentioned above the actuator torque can be obtained automatically. The equations of motion which are extremely lengthy have been derived successfully without any simplifications first. After mathematical simplification process, substituting $m_1 = m_2 = m$, $l_1 = l_2 = L$, $m_i = 1$, and $(EI)_1 = (EI)_2 = 2.04$, the equations become much simpler as shown in the following

$$F_1 = 1.5\,L\,g\,m\cos[q_1] + 0.5\,L\,g\,m\cos[q_1 + q_2]$$

$$-L^3\,\mu\sin[q_2](q_1)'(q_2)' - 0.5\,L^3\,\mu\sin[q_2](q_2)'^2$$

$$+3.146\,L^2\mu\sin[q_2](q_1)'(q_{11})' + 3.146\,L^2\mu\sin[q_2](q_2)'(q_{11})'$$

$$-1.274\,L^2\mu\,sin[q_2](q_1)'(q_{21})' - 1.274\,L^2\mu\sin[q_2](q_2)'(q_{21})'$$

$$+4\,L\,\mu\sin[q_2](q_{11})'(q_{21})' - 0.3186\,L\,g\,\mu\sin[q_1]q_{11}$$

$$+1.571\,g\,m\sin[q_1 + q_2]q_{11} + 3.142\,L^2\,\mu\cos[q_2](q_1)'(q_2)'q_{11}$$

$$+1.571\,L^2\mu\cos[q_2](q_2)'^2q_{11} + 7.578\,L\,\mu\,(q_1)'(q_{11})'q_{11}$$

$$+6.578\,L\,\mu\,(q_2)'(q_{11})'q_{11} + 4\,L\,\mu\cos[q_2](q_1)'(q_{21})'q_{11}$$

$$+4\,L\,\mu\,cos[q_2](q_2)'(q_{21})'q_{11} + 6.284\,\mu\,(q_{11})'(q_{21})'q_{11}$$

$$+(q_{21})''\,(0.318\,L^2\mu + 0.637\,L^2\mu\cos[q_2]$$

$$+2\,L\,\mu\sin[q_2]q_{11}) - 0.318\,L\,g\,\mu\sin[q_1 + q_2]\,q_{21}$$

$$-1.273\,L^2\mu\cos[q_2](q_1)'(q_2)'q_{21} - 0.637\,L^2\,\mu\cos[q^2](q^2)'^2q_{21}$$

$$+4\,L\,\mu\,cos[q_2](q_1)'(q_{11})'q_{21} + 4\,L\,\mu\cos[q_2](q_2)'(q_{11})'q_{21}$$

$$+L\,\mu\,(q_1)'(q_{21})'q_{21} + L\,\mu\,(q_2)'(q_{21})'q_{21}$$

$$-3.142\,\mu\,(q_{11})'(q_{21})'q_{21} + (q_{11})''(-0.729\,L^2\mu - 1.571\,L^2\mu\cos[q_2]$$

$$+2\,L\,\mu\sin[q_2]q_{21}) + (q_1)''(1.665\,L^3\,\mu + L^3\,\mu\cos[q_2]$$

$$+3.142\,L^2\,\mu\sin[q_2]q_{11} - 1.273\,L^2\,\mu\sin[q_2]q_{21})$$
$$+(q_2)''(0.333\,L^3\mu + 0.5\,L^3\mu\cos[q_2]$$
$$+1.571\,L^2\mu\sin[q_2]q_{11} - 0.637\,L^2\mu\sin[q_2]q_{21})$$
$$F_2 = 0.5\,L^3\,g\,m\cos[q_1 + q_2] + 0.5\,L^2\mu\sin[q_2](q_1)'$$
$$-1.047\,L^2\mu\,(q_{11})'' + 0.318\,L^2\mu\,(q_{21})''$$
$$+1.571\,g\,m\sin[q_1 + q_2]q_{11} - 1.571\,L^2\mu\cos[q_2](q_1)'^2q_{11}$$
$$+6.58\,L\,\mu\,(q_1)'(q_{11})'q_{11} + 6.578\,L\,\mu\,(q_2)'(q_{11})'q_{11}$$
$$+6.283\,\mu\,(q_{11})'(q_{21})'q_{11} + (q_2)''\,(0.333\,L^3\mu + 3.29\,L^2\mu\,q_{11})$$
$$-0.318\,L\,g\,\mu\sin[q_1 + q_2]q_{21} + 0.637\,L^2\mu\cos[q_2](q_1)'^2q_{21}$$
$$+L\,\mu\,(q_1)'(q_{21})'q_{21} + L\,\mu\,(q_2)'(q_{21})'q_{21}$$
$$-3.142\,\mu\,(q_{11})'(q_{21})'q_{21} + (q_1)''(0.333\,L^3\mu + 0.5\,L^3\mu\cos[q_2]$$
$$+1.571\,L^2\mu\sin[q_2]q_{11} - 0.637\,L^2\mu\sin[q_2]q_{21})$$

If we assume that there is no deformation and both links are rigid, these equations will be the same as those which would be obtained from a manual derivation process or Asada [13] as follows:

$$F_1 = 1.5\,L\,g\,m\cos[q_1] + 0.5\,L\,g\,m\cos[q_1 + q_2]$$
$$-L^3\mu\sin[q_2](q_1)'(q_2)' - 0.5\,L^3\mu\sin[q_2](q_2)'^2$$
$$+(q_1)''(1.665\,L^3\mu + L^3\mu\cos[q_2])$$
$$+(q_2)''(0.333\,L^3\mu + 0.5\,L^3\mu\cos[q_2])$$
$$F_2 = 0.5\,L^3\,g\,m\cos[q_1 + q_2] + 0.5\,L^2\mu\sin[q_2](q_1)'$$
$$(q_2)''(0.333\,L^3\mu) + (q_1)''(0.333\,L^3\mu + 0.5\,L^3\mu\cos[q_2])$$

6.2 Simulation Study

The derived inverse dynamics equation and inverse kinematics equation are applied to a two-link planer flexible manipulator (Fig. 1*a*) and the effect of gravity was ignored in these case studies in order to isolate the dynamic flexibility effects. The desired trajectory is given by (Fig. 1*b*)

$$x_d(t) = L - a\,t^2, \quad y_d(t) = L + b\,t$$

where a=0.1, b=0.01, and t ranges from 0 to 3 s. By differentiating the desired trajectory, velocities and acceleration are obtained. Next, we couple the inverse kinematic and deflection equations in matrix form.

$$\begin{bmatrix} [J_r] & [J_f] \\ [J_{hjk}] & [I_{jfhk}] \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_r \\ \ddot{\mathbf{q}}_f \end{bmatrix} = \begin{bmatrix} \mathbf{R}_r \\ \mathbf{R}_{ij} \end{bmatrix} \quad (j=1,2) \tag{25}$$

To compute equation (25), parameters such as density and Young's modulus etc. must be provided. These are listed in Appendix C. The bending deflection of links are approximated with assumed mode shapes. Mode shapes are chosen from the analytical solution of an Euler–Bernoulli beam eigenfunction analysis. For simplicity, the terms including the squares of deflections are neglected, since they are considerably smaller than the other terms. To verify the model, the results are compared with the simulation results of the same system with a rigid arm. The algorithm is shown in the flow-chart (Fig. 2). For the case where the robot becomes more rigid, *EI* becomes larger, and the joint variable response of the system converges to the rigid arm. The resonses were computed by solving the dynamic equation. The responses of the flexible system for a given trajectory are shown in Fig. 3. For comparison, the responses derived from a rigid model are also shown in Fig. 3. Both clamp and simply supported mode shapes are shown in Fig. 4. Joint torques can be computed by substituting generalised coordinates into joint equations. Comparisons of the flexible torques of the system with rigid torques are shown in Fig. 5. These results are required for determining the dynamic load carrying capacity [14].

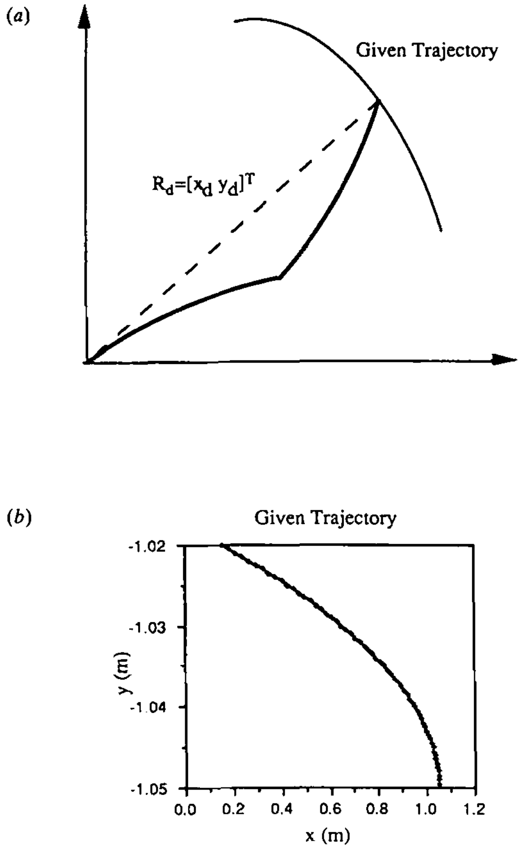

Fig. 1. (*a*) Inverse dynamics of two-link arm. (*b*) A typical given trajectory.

7. Applications and Discussions

Given a trajectory (positions, velocities and accelerations), the dynamic load carrying capacity (DLCC) of a flexible manipulator is defined as the maximum load (mass and moment of inertia) that the manipulator can carry in executing the trajectory with an acceptable tracking accuracy. For manipulators under the rigid-body assumption, the major limiting factor in determining the maximum allowable load

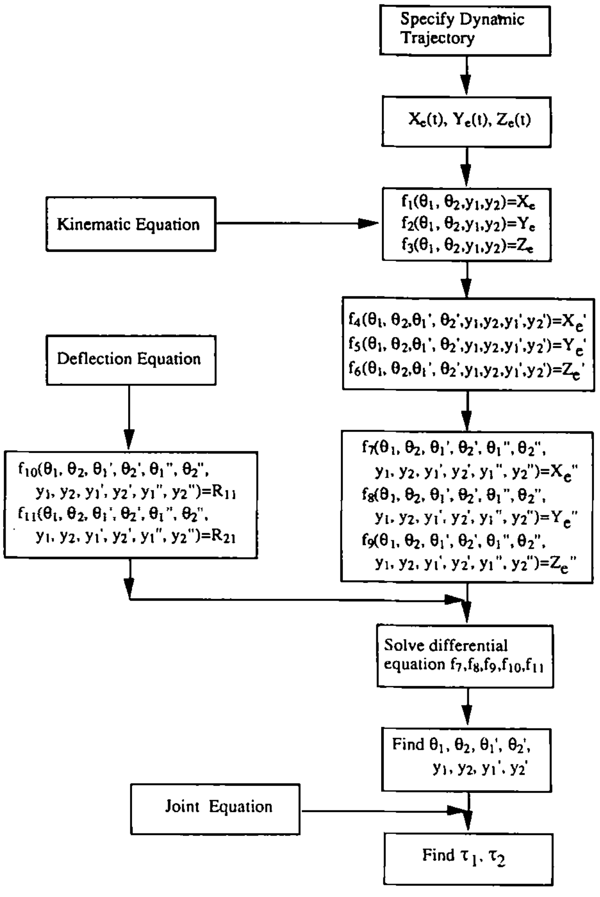
Specify Dynamic Trajectory
Xe(t), Ye(t), Ze(t)
Kinematic Equation
f1(θ1, θ2,y1,y2)=Xe
f2(θ1, θ2,y1,y2)=Ye
f3(θ1, θ2,y1,y2)=Ze
Deflection Equation
f4(θ1, θ2,θ1', θ2',y1,y2,y1',y2')=Xe'
f5(θ1, θ2,θ1', θ2',y1,y2,y1',y2')=Ye'
f6(θ1, θ2,θ1', θ2',y1,y2,y1',y2')=Ze'
f10(θ1, θ2, θ1', θ2', θ1'', θ2'', y1, y2, y1', y2', y1'', y2'')=R11
f11(θ1, θ2, θ1', θ2', θ1'', θ2'', y1, y2, y1', y2', y1'', y2'')=R21
f7(θ1, θ2, θ1', θ2', θ1'', θ2'', y1, y2, y1', y2', y1'', y2'')=Xe''
f8(θ1, θ2, θ1', θ2', θ1'', θ2'', y1, y2, y1', y2', y1'', y2'')=Ye''
f9(θ1, θ2, θ1', θ2', θ1'', θ2'', y1, y2, y1', y2', y1'', y2'')=Ze''
Solve differential equation f7,f8,f9,f10,f11
Find θ1, θ2, θ1', θ2', y1, y2, y1', y2'
Joint Equation
Find τ1, τ2

Fig. 2. Computational algorithm.

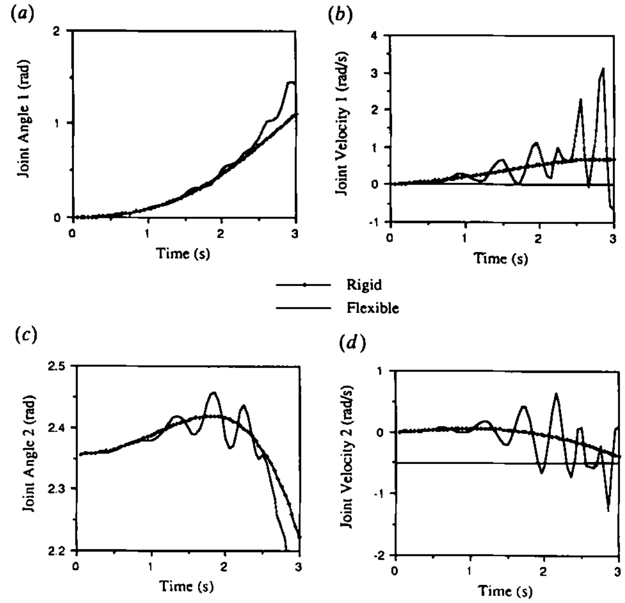

Fig. 3. Comparison of the flexible responses of the system with rigid link.

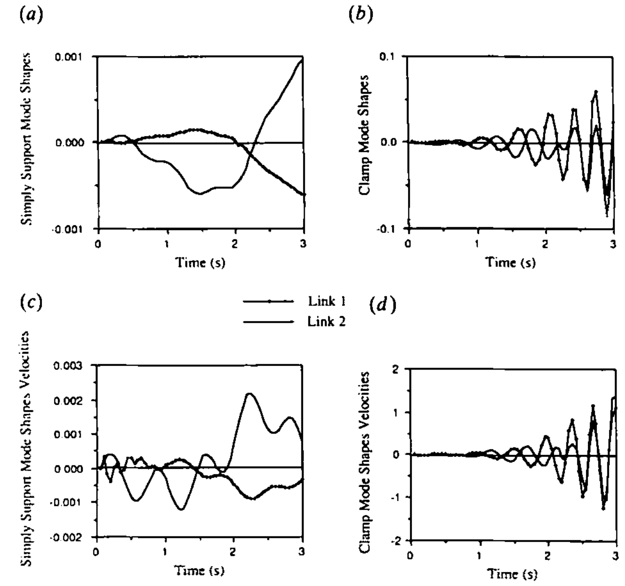

Fig. 4. Comparison of the flexible responses (clamp and simply supported mode shape).

(mass and mass moment of inertia) for a prescribed dynamic trajectory is the joint actuator capacity, while the flexibility inevitably exhibited by relatively lightweight robots or by robots operating at a higher speed dictates the need for an additional constraint to be imposed for tasks requiring precision tracking, that is, the allowable deformation at end effector. Deflection at end effector could be attributed to both static and dynamic factors, such as, link flexibility, joint clearance, manipulator and load inertia. These factors are configuration dependent or motion dependent, therefore, DLCC varies from place to place on a given trajectory. A strategy to determine the DLCC subject to both constraints mentioned above is formulated where the end effector deflection constraint is specified in terms of a series of spherical bounds with a radius equal to the allowable deformation while a typical DC motor speed-torque characteristics curve is used in the actuator constraint.

This was achieved by subjecting the manipulator to dual constraints, that is, actuator capacity and end effector deformation constraints, when the maximum load is determined. Both constraints were imposed in determining DLCC and a load m_{load}=0.341 kg was found to be the maximum load that the given actuators can carry in executing the trajectory while the load moment of inertia I_{load} was not presented for simplicity. Fig. 6 shows the time varying torques required to execute the trajectory against the upper and lower bounds of the available torques which depend on the joint velocities. It is seen that the load so determined uses the joint 1 to its maximum extent at about 1.2 s. The magnitude of the end effector deflection with such a load compared to the imposed upper and lower bounds is shown in Fig. 7. It is seen that all the magnitudes remain within the bounds because the load was determined subject to both constraints. The actual trajectory is further plotted in terms of the base coordinates

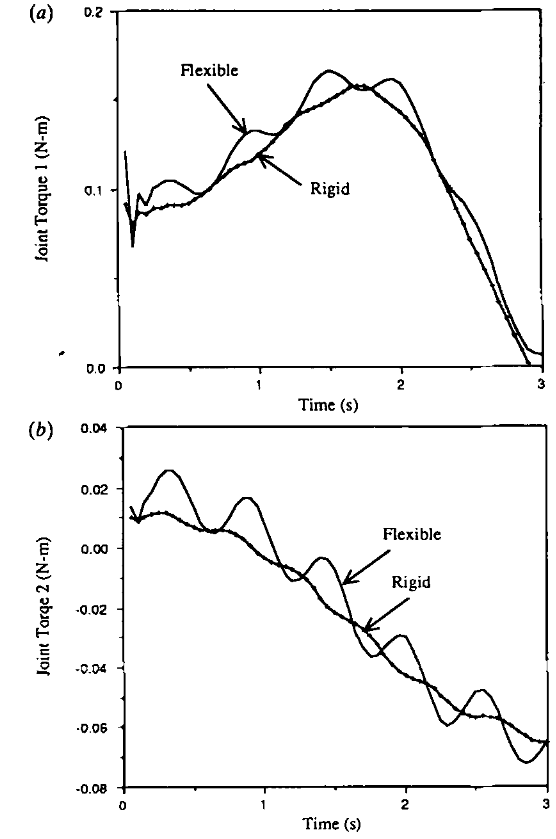

Fig. 5. Comparison of the flexible torques of the system with rigid link.

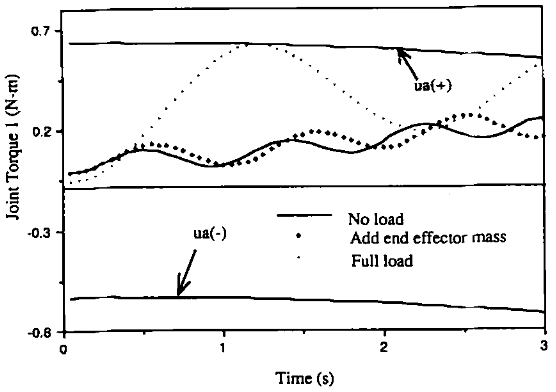

Fig. 6. Joint torques against torque bounds.

in Fig. 8, which again shows it is within the bounds. The further view of the DLCC of flexible manipulator reader is referred to [15]. The work also shows that in dealing with flexible manipulator dynamics and determining their DLCC in particular great benefit was obtained from using a symbolic derivation language.

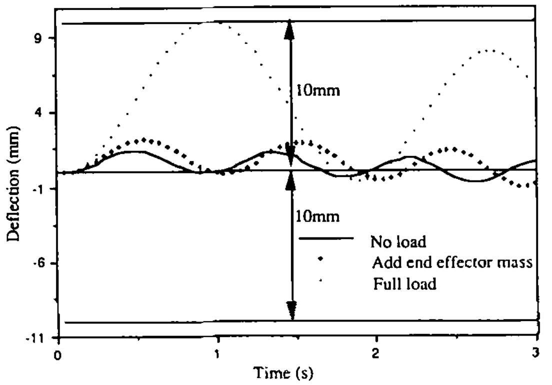

Fig. 7. Deflection against its bounds (m_{load} = 0.341 kg under both constraints.

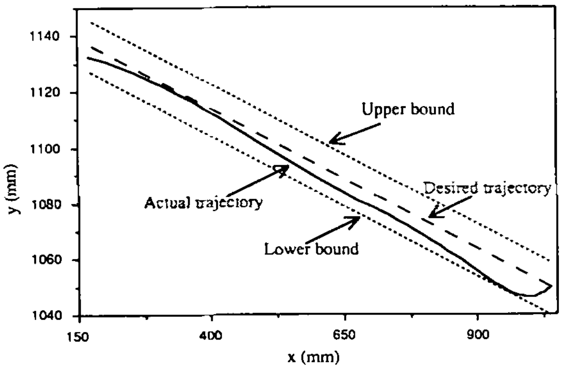

Fig. 8. The desired and actual trajectory (m_{load} = 0.341 kg, under both constraints).

8. Conclusion

This paper presents a technique for deriving the scalar form of flexible manipulator inverse dynamic equations using a PC-based symbolic language MATHEMATICA. The algebraic dynamic robot modelling program has been implemented to enable the control engineer to formulate a Lagrangian assumed mode method and to gain physical insight into dynamic equations for the systematic design as well as for determining the dynamic load carrying capacity for a given trajectory. To validate symbolic derivation and simulation results, a flexible model is compared with rigid links in a case study.

References

1. J. Y. S. Luh, M. H. Walker and R. P. C. Paul, "On-line computational scheme for mechanical manipulators", *Transactions of ASME, Journal of Dynamic System, Measurement and Control*, **102**, pp. 69–76, 1980.
2. J. M. Hollerbach, "A recursive formulation of Lagrangian manipulator dynamics", *IEEE, J. Trans. System, Man and Cybernetics*, SMC-10-11, pp. 730–736, 1980.
3. R. H. Cannon and E. Schmitz, "Initial experiments on the end-point control of a flexible one-link robot", *International Journal of Robotics Research*, **3**(3), pp. 62–75, 1984.

4. W. J. Book and M. Majette, "Controller design for flexible manipulator, distributed parameter mechanical arms via combined state space and frequency domain technique", *Transactions of ASME, Journal of Dynamic Systems, Measurement, and Control,* **105**, pp. 245–254, 1983.
5. S. Cetinkunt and W. J. Book, "Symbolic modeling and dynamic simulation of robotic manipulators with compliant links and joints", *Robotics and Computer-Integrated Manufacturing,* **5**(4), pp. 301–310, 1989.
6. L. W. Chang and J. F. Hamilton, "Dynamics of robotic manipulators with flexible links", *Transactions of ASME, Journal of Dynamic System, Measurement and Control,* **113**, pp. 54–59, 1991.
7. J. J. Murray and C. P. Neuman, "ARM: An algebraic robot dynamic modeling program", *Transactions of ASME, Journal of Dynamic System, Measurement and Control,* **108**, pp. 172–179, 1984.
8. G. Naganathan and A. H. Soni, "Nonlinear flexibility studies for spatial manipulators", *Proceedings of the International IEEE Conference Robotics and Automation*, pp. 373–378, 1986.
9. A. A. Shabana, "Dynamics of flexible bodies using generalized Newton–Euler equation", *Transactions of ASME, Journal of Dynamic System, Measurement and Control,* **112**, pp. 496–503, 1990.
10. W. J. Book, "Recursive Lagrangian dynamics of flexible manipulator arm", *International Journal of Robotics Research,* **3**(3), pp. 87–101, 1984.
11. S. Wolfram, *Mathematica*, Addison-Wesley, 1991.
12. R. E. Maeder, *Programming in Mathematica*, Addison-Wesley, 1990.
13. H. Asada and J.-J. E. Slotine, *Robot Analysis and Control*, John Wiley & Sons, Inc, pp. 97–99, 1986.
14. M. H. Korayem, Y. Yao and A. Basu, "Load carrying capacity for a two-link planer flexible arm", *Proceedings of the Thirteenth Canadian Congress of Applied Mechanics*, vol. 2, pp. 664–665, June 1991.
15. M. H. Korayem, Y. Yao and A. Basu, "Dynamic load carrying capacity for a multi-link flexible arm", *Proceedings of the International Conference Control and Robotics*, pp. 79–82, 1992.

Appendix A

Required recursive expression and matrices for **J** and **R** are given below:

$$^{n}\bar{\mathbf{F}}_n = \mathbf{G}_n \quad (h = j = n) \tag{A1}$$

$$^{j}\bar{\mathbf{F}}_h = \mathbf{E}_j\,\mathbf{A}_{j+1}\,{}^{j}\bar{\mathbf{F}}_h \quad (j < h \leq n) \tag{A2}$$

$$^{h}\bar{\mathbf{F}}_n = \mathbf{G}_n + {}^{h}\bar{\mathbf{F}}_{h+1}\,(\mathbf{E}_j\,\mathbf{A}_{j+1})^{\mathrm{T}} \quad (j = h < n) \tag{A3}$$

$$^{j}\mathbf{F}_h = {}^{h}\bar{\mathbf{F}}_{h+1}\mathbf{A}^{\mathrm{T}}_{h+1} \quad (h = 1,\ldots, n-1;\quad j = 1,\ldots,n) \tag{A4}$$

$$^{j}\boldsymbol{\emptyset}_h = \mathbf{A}_{j+1}{}^{h}\bar{\mathbf{F}}_{h+1}\,(\mathbf{A}_{h+1})^{\mathrm{T}} \quad (j = 1,\ldots,n-1;\quad h = 1,\ldots,n) \tag{A5}$$

$$\mathbf{Q}_n = \mathbf{G}_n\dot{\mathbf{W}}^{\mathrm{T}}_{nv} + 2\left(\sum_{k=1}^{m_n} \dot{q}_{nk}\,\mathbf{D}_{nk}\right)\dot{\mathbf{W}}^{\mathrm{T}}_n \tag{A6}$$

$$\mathbf{Q}_j = \mathbf{G}_j\bar{\mathbf{W}}^{\mathrm{T}}_{nj} + 2\left(\sum_{k=1}^{m_n} \dot{q}_{jk}\,\mathbf{D}_{jk}\right)\dot{\mathbf{W}}^{\mathrm{T}}_j + \mathbf{E}_j\,\mathbf{A}_{j+1}\,\mathbf{Q}_{j+1} \tag{A7}$$

$$\mathbf{P}_n = \mathbf{M}_n\,\mathbf{r}_{rn} + \sum_{k=1}^{m_n} q_{nk}\,\boldsymbol{\Omega}_{nf} \tag{A8}$$

$$\mathbf{P}_n = \mathbf{M}_j\,\mathbf{r}_{rj} + \sum_{k=1}^{m_n} q_{jk}\,\boldsymbol{\Omega}_{jk} + \mathbf{E}_j\,\mathbf{A}_{j+1}\,\mathbf{P}_{j+1} \tag{A9}$$

$$\tilde{\mathbf{W}}_{vj} = \ddot{\hat{\mathbf{W}}}_{v,j-1}\,\mathbf{A}_j + 2\,\dot{\hat{\mathbf{W}}}_{j-1}\dot{\mathbf{A}}_j + \hat{\mathbf{W}}_{j-1}\,\bar{\mathbf{U}}_{2j}\,\dot{q}^2_j \tag{A10}$$

$$\ddot{\hat{\mathbf{W}}}_{v,j} = \tilde{\mathbf{W}}_{v,j}\,\mathbf{E}_j + 2\,\dot{\mathbf{W}}_j\,\dot{\mathbf{E}}_j \tag{A11}$$

$$\mathbf{D}_{jk} = \mathbf{C}_{jk} + \sum_{k=1}^{m_n} q_{il}\,\mathbf{C}_{ilk} \tag{A12}$$

$$\mathbf{G}_j = \mathbf{C}_j + \sum_{k=1}^{m_n} q_{il}\left((\mathbf{C}_{ik} + \mathbf{C}^{\mathrm{T}}_{ik}) + \sum_{k=1}^{m_n} q_{il}\,\mathbf{C}_{ilk}\right) \tag{A13}$$

$$\mathbf{C}_i = \frac{1}{2}\int_0^{1_i} [1,\mu_i,0,0]^{\mathrm{T}}\,[1,\mu_i,0,0]\,\mathrm{d}m \tag{A14}$$

$$\mathbf{C}_{ij} = \frac{1}{2}\int_0^{1_i} [1,\mu_i,0,0]^{\mathrm{T}}\,[0,x_{ij},y_{ij},z_{ij}]\,\mathrm{d}m \tag{A15}$$

$$\mathbf{C}_{ijk} = \frac{1}{2}\int_0^{1_i} [0,x_{ij},y_{ij},z_{ij}]^{\mathrm{T}}\,[0,x_{ij},y_{ij},z_{ij}]\,\mathrm{d}m \tag{A16}$$

Appendix B. MATHEMATICA Program for Generating Dynamic Equations

```
"/This program derives the equation of motion of a manipulator with flexible link using the
Lagrangian formulation./"
("**************** Specified Trajectory ****************";
a=.1;b=.01;T[l][ß[1]]=1.05;gx=0;T[F][ß[1]]=0;T[F][ß[2]]=0;
Xe ;Ye;Ze;n=2;nm=1;
Derivative[1][Xe]; Derivative[1][Ye]; Derivative[1][Ze];
Derivative[2][Xe] ;Derivative[2][Ye] ;Derivative[2][Ze];
GeCon;
"*******************Jacobian Matrix*********************";
        For[j=1, j <=2,j += 1,
                For[k=1, k <=2,k += 1,
T[JR][ß[j],ß[k]] =D[GeCon[[j]],T[q][ß[k]][t]];
T[JF][ß[j],ß[k]] =D[GeCon[[j]],T[y][ß[k]][t]];
T[DJR][ß[j],ß[k]]=D[T[JR][ß[j],ß[k]],t];
T[DJF][ß[j],ß[k]]=D[T[JF][ß[j],ß[k]],t]  ] ]
"*********** Velosity & Acceleration ******************";
Jr= { {T[JR][ß[1],ß[1]],T[JR][ß[1],ß[2]]},{T[JR][ß[2],ß[1]],T[JR][ß[2],ß[2]]} };
Jf= { {T[JF][ß[1],ß[1]],T[JF][ß[1],ß[2]]},{T[JF][ß[2],ß[1]],T[JF][ß[2],ß[2]]} };
DJr={ {T[DJR][ß[1],ß[1]],T[DJR][ß[1],ß[2]]},{T[DJR][ß[2],ß[1]],T[DJR][ß[2],ß[2]]} };
DJf={ {T[DJF][ß[1],ß[1]],T[DJF][ß[1],ß[2]]}, {T[DJF][ß[2],ß[1]],T[DJF][ß[2],ß[2]]} };
V= { {Derivative[1][Xe]},{Derivative[1][Ye]},{Derivative[1][Ze]} };
DTetar ={Derivative[1][T[q][ß[1]][t]],Derivative[1][T[q][ß[2]][t]]};
DTetaf ={Derivative[1][T[y][ß[1]][t]],Derivative[1][T[y][ß[2]][t]]};
DV    ={Derivative[2][Xe],Derivative[2][Ye]};
DDTetaf={Derivative[2][T[y][ß[1]][t]],Derivative[2][T[y][ß[2]][t]]};
Rtraj=DV-DJr.DTetar-DJf.DTetaf;
"*******************Specified Deflection***************";
        For[j=1, j <=n,j++,
                For[k=1, k <=nm,k++,
T[l][ß[j]]=1.05;T[r][ß[x],ß[j]]=T[l][ß[j]]/2;
T[B][ß[1]]=N[Pi]/1.05;T[B][ß[2]]=2 N[Pi]/1.05;T[M][ß[j]]=m;
T[y][ß[j],ß[k]]=T[y][ß[j],ß[k]]=Sin[T[B][ß[k]]x];
T[øz][ß[j],ß[k]]=D[T[y][ß[j],ß[k]],x] ] ]
For[j=1, j <=n,j++,
T[y][ß[j]][t]=Sum[T[y][ß[j],ß[f1]]T[q][ß[j],ß[f1]][t],{f1,1,nm}];
T[y][ß[j+5]][t]=Sum[T[y][ß[j],ß[f1]]
              Derivative[1][T[q][ß[j],ß[f1]]][t],{f1,1,nm}];
T[øz][ß[j]][t]=Sum[T[øz][ß[j],ß[f2]]T[q][ß[j],ß[f2]][t],{f2,1,nm}];
T[øz][ß[j+5]][t]=Sum[T[øz][ß[j],ß[f2]]
Derivative[1][T[q][ß[j],ß[f2]]][t],{f2,1,nm}];]Derivative[1][T[y][ß[1]][t]]=T[y][ß[6]][t];
Derivative[1][T[y][ß[2]][t]]=T[y][ß[7]][t];<<siml.m
"****************** R matrix ******************************";
R21={RLSim21[[3]],RLSim21[[4]],Rtraj[[1]],Rtraj[[2]]};
J21={ {JLSim21[[3]][[1]],JLSim21[[3]][[2]],JLSim21[[3]][[3]],JLSim21[[3]][[4]]},
    {JLSim21[[4]][[1]],JLSim21[[4]][[2]],JLSim21[[4]][[3]],JLSim21[[4]][[4]]},
  {T[JR][ß[1],ß[1]],T[JR][ß[1],ß[2]],T[JF][ß[1],ß[1]],T[JF][ß[1],ß[2]]},
  {T[JR][ß[2],ß[1]],T[JR][ß[2],ß[2]],T[JF][ß[2],ß[1]],T[JF][ß[2],ß[2]]} };
J21=Triglinear[J21];J21=Chop[J21]; InJ21=Inverse[J21];InJ21=Chop[InJ21];
GENE21=InJ21.R21;GENE21=Chop[GENE21;Tf=3;nn=Tf/.05;
        For[j=1, j <=1,j++,
µ =.405;m= .425; m2 = 0.00; j2 = 0;
N2NM1L=RungeKutta[
        {T[q][ß[6]][t]        ,GENE21[[1]]        ,T[q][ß[7]][t]        ,GENE21[[2]],
        T[q][ß[6],ß[6]][t]    ,GENE21[[3]]        ,T[q][ß[7],ß[6]][t]   ,GENE21[[4]]},
        {T[q][ß[1]][t]        ,T[q][ß[6]][t]      ,T[q][ß[2]][t]        ,T[q][ß[7]][t],
        T[q][ß[1],ß[1]][t],T[q][ß[6],ß[6]][t],T[q][ß[2],ß[1]][t],T[q][ß[7],ß[6]][t]},
        {0.00,0.00,1.5708,0.00,0.00,0.00,0.00,0.00},{t,0,Tf,.05}];]
```

Appendix C. Numerical Values for Simulation

Parameter	Value	Unit
Young's modulus	$E_1 = E_2 = 2.06 \times 10^{11}$	N/m²
Area moment of inertia	$I_1 = I_2 = 9.9 \times 10^{-12}$	m⁴
Link length	$l_1 = l_2 = 1.05$	m
Link linear mass density	$\mu_1 = \mu_2 = 0.405$	kg/m

Nomenclature

$\mathbf{A}_i$ joint transformation relates system i to system $i-1$

$\mathbf{E}_i$ link transformation relates the deflection of system i to system $\hat{i}$

F_i joint torque acting on joint i

$\mathbf{g}$ gravity vector expressed at the base coordinates

$\mathbf{J}$ inertia matrix $= \begin{bmatrix} [J_{jh}] & [J_{jhk}] \\ [J_{hjk}] & [I_{jfhk}] \end{bmatrix}$

K kinetic energy of the system

l_i length of link i

M_i a mass concentrated at the joint i

m_i number of modes used to describe the deflection of link i

n number of links

q_h joint variable of the hth joint

q_{hk} time-varying amplitude of mode k of link h

$\mathbf{R}$ vector of remaining dynamics and external forcing terms $= [R_1, R_2, \ldots, R_h \ldots. R_n, R_{11}, R_{12} \ldots, R_{1m_1}, R_{21} \ldots, R_{2m_2} \ldots, R_{h1} \ldots, R_{hm_n} \ldots, R_{nm_n}]^T$

$\mathbf{r}_i$ vector locating the centre of mass of link i

R_j dynamics from the joint equation j, excluding second derivatives of the generalized coordinates

R_{jf} dynamics from the deflection equation jf, excluding second derivatives of the generalized coordinates

V potential energy

$\mathbf{W}_i$ transformation from the base to the ith link

$\hat{\mathbf{W}}_i$ transformation from the base to the system $\hat{i}$

$\mathbf{z}$ the vector of generalised coordinates $= [q_1, q_2, \ldots, q_h \ldots, q_n, q_{11}, q_{12} \ldots, q_{1m_1}, q_{21} \ldots, q_{2m_2} \ldots, q_{h1} \ldots, q_{hm_n} \ldots, q_{nm_n}]^T$

μ link density