Journal of Sound and Vibration (1997) **208**(5), 823–839

SYMBOLIC EQUATION PROCESSING UTILIZING VECTOR/DYAD NOTATION

A. A. Barhorst

Department of Mechanical Engineering, Texas Tech University, Lubbock, Texas 79409-1021, *U.S.A.*

(*Received* 11 *December* 1996, *and in final form* 14 *July* 1997)

In previous work the author has presented a methodology for obtaining closed form equations of motion for general non-holonomic hybrid parameter multiple body systems (HPMBS). It is well known that generating closed form symbolic equations of motion for HPMBS of few degrees of freedom rapidly degenerates to an algebraic soup unless computer tools are utilized. Methods dependent on Lagrange multipliers are virtually unable to provide the constraint free (minimal) symbolic equations for systems of moderate to high order due to the complication of eliminating the multipliers. In this paper, the intent is to provide the details, and example use of, a tool, based on commercial symbolic manipulation software, developed to assist the symbolic processing of the equations of motion as previously minimally formulated. The tool presented herein is novel in the way the vector notation of the formulation scheme is retained and the way that frame information is carried in the calculations. The symbolic tool provided in this paper allows the modelling methodology to be a practical tool for generating closed form equations of motion for extremely complicated HPMBS.

1. INTRODUCTION

The use of computer-based symbol manipulators in the derivation of equations of motion for multiple body systems has been in use for about two decades [1–3]. Work is ongoing and includes modelling and analysis applications [4–9] and control design applications [10–14]. Investigators use general purpose symbol manipulation software [15, 16] and application specific software [1–3, 15]. The advantages of using symbolic manipulation tools are apparent during the analysis of multiple body systems. These tools allow one to easily take advantage of the repetitive nature of multi-body dynamic modelling.

When the symbol manipulating tools are applied to flexible multi-body systems, a modelling finesse not readily available from brute force finite element discretization procedures is possible. This allows one to reduce the repetitive calculations associated to multi-body systems to memory look-up operations. Of course, by replacing multiplications by memory operations, the parallel structure of the matrix methods is often destroyed, thus precluding the efficient use of multi-processor computers for the simulation stage. Certain aspects of the above-mentioned problems are being addressed [14, 17–19].

The author of this work has been working in the area of hybrid parameter multiple body system (HPMBS) analysis. A method suitable for full motion regime non-holonomic motion analysis of HPMBS has been presented [20–22]. The tool is not restricted by topological, continuum or kinematic considerations. In this previous work, claims to the affability of the HPMBS modelling tool to symbolic manipulation and ultimately automation have been made. In this paper, the claims are addressed and work towards

0022–460X/97/500823 + 17 $25.00/0/sv971204

TABLE 1

Typical unit vectors

```
In[195]:=
          i=unitVector[N,''i'',1]
          n[1]=unitVector[N,n,1]
          b[1]=unitVector[B,b,1]
Out[195]=
          ^
          i
```

the automation of the HPMBS modelling method is presented. The fact of the matter is that obtaining explicit equations of motion for HPMBS of moderate complexity is cumbersome—for all practical purposes a computer algebra tool is needed. However, with the tool described herein and the methodology presented elsewhere, highly complicated systems can be model with explicit closed form equations.

Presented in this paper is a symbolic tool based on the commercial package *Mathematica* [23]. The pattern recognition utilities in *Mathematica* have been adjusted to allow a natural codification of the HPMBS modelling tool using vector/dyad notation, along with the spatial and temporal variables in a HPMBS. In this paper the associated partial differential equations are presented in weak form for numerical solution. The novelty of the described tool is due to the way in which the vector notation of the formulation scheme is retained and the way that co-ordinate frame information is carried in the calculations.

To demonstrate the tools, a double link flexible planar manipulator will be modelled and the response from a simple maneuver will be presented. A *Mathematica* notebook is used for equation generation and simulation result presentation. The simulation is done via a simple FORTRAN based code external to the notebook.

2. SYMBOLIC ESSENTIALS

In this section, the essential ideas underlying the use of *Mathematica* as the engine for the engineering vector notation based symbolic algorithms are presented. This package is well suited for the task of using vector notation because of the generality of its symbol matching algorithms. *Mathematica* allows the user to define symbols and operations through its symbol/function/head fundamental basis, and the ''underscore matches anything'' functionality.

TABLE 2

Typical unit dyads

```
In[198]:=
          bn1[1,1]=unitDyad[b[1],n[1]]
Out[198]=
          ^ ^
          b n
           1 1
```

TABLE 3

Typical vector operations

```
In[241]:=
          bn1[1,1] >< b[1]
          b[1] >< n[1]
          (b[1] >< n[1]) . b[2]
Out[241]=
           ^  ^    ^
          b  n >< b
           1 1    1
Out[242]=
              ^    ^
             b >< n
              1    1
Out[243]=
                 ^     ^
          -1    b  .  n
                 3     1
```

2.1. BASIC SYMBOLS AND OPERATORS

The fundamental symbol for vector notation is the unit vector. To define a unit vector any symbol can be defined to have a head of **unitVector**. The author usually uses symbols such as $\hat{i}$ or $\hat{n}_1$ or $\hat{b}_1$ to represent unit vectors. In the symbolic algorithms this is defined as shown in the input/output (i/o) pair shown in Table 1. As seen in the table, the vector's frame is the first argument, its symbol is the second argument, and its direction is the third argument to **unitVector**, respectively. The output of `n[1]` and `b[1]` is suppressed for brevity. In order to get the unit vector output as shown, the output functions of *Mathematica* were adjusted.

The next symbol that is defined is the unit dyad. The unit dyad is a symbol with head **unitDyad** which has two unit vectors as arguments. An example is shown in Table 2.

The vector operations "Dot" and "Cross" are required. *Mathematica* has these functions defined in one of its packages related to tensor/matrix notation. These functions were modified to accept symbols with heads **unitVector** and **unitDyad**, with provisions to handle the base frame for each unit vector and the operator order. The symbol for the vector cross product is constructed from `>` and `<` as `><`. This allows easy typing of operations. In order to get the *Mathematica* parser to intercept this symbol and infix the function **Cross**, the built in **$PreRead** function was used. The symbol for dot multiplication remained the period. The functions **Dot** and **Cross** allow operations such as that shown

TABLE 4

Typical angular velocity

```
In[249]:=
          NwB=omega[N,B]=s1 n[1]+s2 n[2]+s3 n[3]
Out[249]=
             ^     ^     ^
          s1 n+s2 n+s3 n
             1     2     3
```

TABLE 5

Typical position vector

```
In[250]:=
          NorP= L1 n[1]+L2 n[2]+L3 n[3]+x1 b[1]+y1 b[2]+z1 b[3]
Out[250]=
                 ^     ^     ^     ^     ^     ^
          x1 b+y1 b+z1 b+L1 n+L2 n L3 n
             1    2    3    1    2    3
```

in Table 3. In the table, the vector `b[2]` is the two-vector and `b[3]` is the three-vector in the frame `B`.

Once these fundamental operations were defined, any vector could be constructed. To allow the operations **Dot** and **Cross** to distribute across these arbitrary vectors, many pattern matching scenarios were defined. Other functions that loosen the nesting of parenthesis, etc., also had to be defined.

2.2. DIFFERENTIATION, INTEGRATION AND OTHER OPERATIONS

In general, vectors are defined in multiple rotating co-ordinate frames, so time differentiation of the vector quantities is facilitated via the use of frame angular velocity. The angular velocity of a frame is a symbol with head **omega**, which has two arguments, a base frame (which can rotate) symbol and the rotating frame symbol. An example of an angular velocity i/o pair is shown in Table 4. With angular velocity defined, this allows a vector from point `No` to point `P`, shown in Table 5, to be differentiated to velocity as shown in Table 6. The operator **DvDt[a,b]** takes the differentiation frame and the vector as arguments. Implicit in the differentiation operator is that the angular velocity of the frame of reference, for each unit vector in the vector differentiated, is known relative to the differentiation frame. Multiple use of the **DvDt[a,b]** operator and operators defined to

TABLE 6

Typical velocity vector

```
In[251]:=
          NovP=DvDt[N,NorP]
Out[251]=
                    ^    ^     ^    ^     ^    ^
          x1 (s1 n >< b+s2 n >< b+s3 n >< b)+
                 1   1    2   1    3   1
                   ^    ^     ^    ^     ^    ^
            y1 (s1 n >< b+s2 n >< b+s3 n >< b)+
                   1    2    2    2    3    2
                   ^    ^     ^    ^     ^    ^
            z1 (s1 n >< b+s2 n >< b+s3 n >< b)+
                   1    3    2    3    3    3
              ^    ^     ^    ^     ^    ^
          ẋ1 b+ẏ1 b+ż1 b+L̇1 n+L̇2 n+L̇3 n
             1    2    3    1    2    3
```

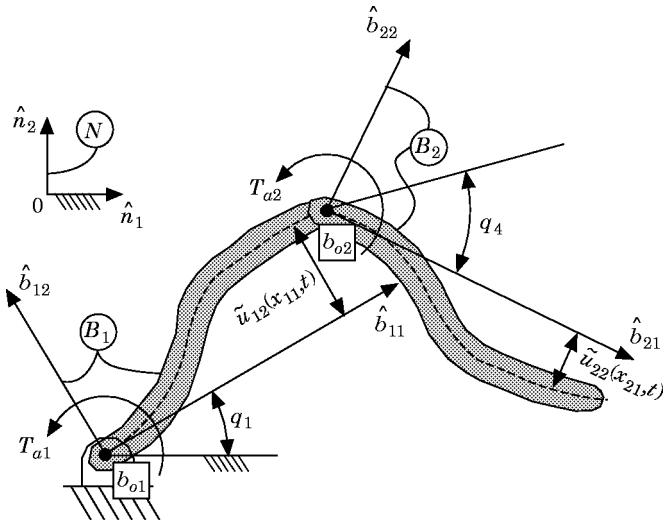

Figure 1. A two-link flexible manipulator.

spread out the symbols (some intrinsic others created) allow the kinematic analysis to be executed.

In some instances, special integration operations had to be defined to assist in the integration of complicated expressions, but in general, the intrinsic integration function worked well, provided that the expression was sufficiently simple, which is a vague statement, but does describe the situation.

Other functions were defined to "loosen dots", "distribute scalars", "un-square", "gather divisors", "symplify right-hand sides", etc. These functions and their brethren were used to get the symbols into a suitable form. The equations of motion are derived by implementing the modelling algorithm, previously mentioned, with these symbolic tools. This is demonstrated below.

TABLE 7

Angular velocity and acceleration

```
In[211]:=
          NwB[1]=omega[N,B[1]]=s[1][t] b[1,3]
          NwB[2]=omega[N,B[2]]=(s[1][t]+sp[2][t]+s[4][t])b[2,3]
          NaB[1]=DvDt[N,NwB[1]]//Simplify
          NaB[2]=DvDt[N,NwB[2]]//Simplify
Out[212]=
                                                    ^
          (s [t]+s [t]+sp [t]) b
            1     4       2     23
Out[214]=
                                                    ^
          (ṡ [t]+ṡ [t]+ṡp [t]) b
            1     4       2     23
```

2.3. ADVANTAGES OF RETAINING VECTOR NOTATION

The advantages of using engineering vector notation are manifold. The first advantage is that the notation is second nature for engineers and is readily related to the geometry of the problem at hand. The second advantage is that co-ordinate transformations are not needed until all the manipulations are completed. Also, the transformations have been reduced to minimal operations between the respective vectors via vector identities where applicable. The third advantage is that the terms of the equations are easily identified with respect to normal naming conventions, such as a Coriolis term, and the direction in which they act is apparent. The fourth advantage is that recursive algebraic operations and repetitive operations can be readily identified and replaced with memory operations. The fifth advantage is that non-holonomic equations of motion for HPMBS are rigorously and simply implemented with the vector-based method utilized herein.

Take as a well known counter-example the usual method for implementing the vector cross product in the matrix based formulation schemes. A 3×3 skew-symmetric angular velocity matrix is multiplied into a 3×1 vector of position or velocity components. In order for the multiplication to be defined, the vectors must be transformed into a common co-ordinate basis, which may require the additional multiplication of the vectors by 3×3 transformation matrices. In the case of the transformation and angular velocity cross product operator matrices, there are chances for many zero elements to exist. If the matrix multiplication algorithm is not sufficiently frugal, many unneeded operations will be performed. In any case, the co-ordinate transformation multiplications will be applied. It also appears that identifying recurring operations, operands and results will be difficult, if not impossible. The vector method lends a clear advantage in this situation because the transformations are completed via vector operations which are exact (in the symbolic implementation) and the co-ordinate frame information is imbedded, which allows for the clear delineation of recurring and unneeded operations. However, the parallel nature of the matrix methods is lost.

As another counter-example, consider a scalar-based Lagrangian formulation of a problem with flexible bodies and multiple rotating reference frames. In this case it is usually decided to carry the transcendental terms of the transformations form the outset. In many cases the identities and reductions that are readily identified in the vector notation go unnoticed, because they are so deeply imbedded in the terms of the equations. The computer-based symbolic tools often miss the reductions because they are so deep in the expressions and the time allocated for simplifications is usually not extensive. Therefore, many terms that are identically zero or constant are calculated at each time step in a simulation. The vector-based method keeps the transformations distinctly identifiable, which allows for greater simplification, resulting in fewer operations per execution step.

Other advantages become apparent as experience with the modelling technique evolves. These advantages will be highlighted as the example presented below unfolds.

3. FLEXIBLE TWO-LINK PLANAR MANIPULATOR

In this section a flexible two-link planar manipulator will be modelled and a simulated maneuver presented. The system is a simplified version of reality and is presented only as a demonstration of the tools herein. The links are taken to be Euler–Bernoulli beams and it is assumed that modelling deflection is sufficient. The low angular speeds expected are not of sufficient magnitude to expect instabilities due to the lack of geometric stiffness terms in the models for the beams. The links are assumed to be driven by massless motors and are connected via massless hubs. Proportional damping is present in the joints and

TABLE 8

Position vectors

```
In[215]:=
          NorBo[1]=0
          BorIo[1]=x[1,1] b[1,1]+u[1,2)[x[1,1], t] b[1,2]
          BorBo[1,2]=L[1] b[1,1]+qp[3][t] b[1,2]
          BorIo[2]=x[2,1] b[2,1]+u[2,2][x[2,1], t] b[2,2]
```

structural damping proportional to the bending rate of change is assumed to be present in both beams.

The typeset equations will be supplemented with the corresponding input and output (where appropriate) from the symbolic algorithms as the model is developed. A discussion of tools used, but not mentioned in the previous section, will be provided in the body of the model development.

3.1. SYSTEM MODEL

The system is shown in Figure 1. The domain of each beam is one-dimensional, the independent co-ordinates being x_{11} and x_{21} measured from the root of beam B_1 and B_2, respectively, along the undeformed neutral axis of each beam. The "special" point of beam B_1 is labelled b_{o1} and b_{o2} for beam B_2. The co-ordinate frames, denoted by B_1 and B_2, are attached as shown in Figure 1. At the root of each beam there is a massless hub to which torques T_{a1} and T_{a2} are applied to the hubs of B_1 and B_2. The angular positions of frames B_1 and B_2 are q_1 and q_4. Beam deflection is measured with $\tilde{u}_{12}(x_{11}, t)\hat{b}_{12}$ and $\tilde{u}_{22}(x_{21}, t)\hat{b}_{22}$ (simple flexure) as shown in Figure 1. The beams have mass per unit length ρ, total lengths are L_1 and L_2, cross-sectional area A, area moment of inertia I, and Young's modulus E. Structural damping is assumed to be present in proportion to the rate of bending for each beam. The details of the formulas used below can be found in the work mentioned above.

3.2. KINEMATICS

The first vectors needed are the angular velocities of frame B_1 and B_2. They are

$$^{\mathcal{N}}\vec{\omega}^{B_1} = s_1\,\hat{b}_{13} \tag{1}$$

and

$$^{\mathcal{N}}\vec{\omega}^{B_2} = (s_1 + s_2' + s_4)\hat{b}_{13}. \tag{2}$$

The angular accelerations of frames B_1 and B_2 are

$$^{\mathcal{N}}\vec{\alpha}^{B_1} = \dot{s}_1\,\hat{b}_{13} \tag{3}$$

and

$$^{\mathcal{N}}\vec{\alpha}^{B_2} = (\dot{s}_1 + \dot{s}_2' + \dot{s}_4)\hat{b}_{13}. \tag{4}$$

The generalized and pseudo-generalized speeds are defined as

$$s_1 = \dot{q}_1, \qquad s_2' = \dot{q}_2' = \frac{\partial^2 \tilde{u}_{12}(L_1, t)}{\partial x_{11}\,\partial t}, \qquad s_4 = \dot{q}_4, \tag{5}$$

where $q_2' = \partial \tilde{u}_{12}(L_1, t)/\partial x_{11}$. The corresponding computer input and output is shown in Table 7. In the table, the complete output is not shown, but can be inferred from the information provided.

TABLE 9

Derivatives of position vectors

```
In[220]:=
          NovBo[1]=DvDt[N,NorBo[1]]
          BovIo[1]=DvDt[N,BorIo[1]]//gatherVectors
          BovBo[1,2]=DvDt[N,BorBo[1,2]]/.qp[3]'[t] -> sp[3][t]//
          distributeScalars//gatherVectors
          BovIo[2]=DvDt[N,BorIo[2]]//distributeScalars//gatherVectors
Out[223]=
          (-(s [t] u [x,t])-s[t] u [x,t]-
              1     22 21    4    22 21
                                 ^
          sp [t] u[x,t]) b  +
            2     22 21   21
                                                          ^
          (x  s [t]+x s [t]+x sp [t]+u̇ [x, t]) b
           21 1     21 4     21 2     22 21     22
```

The position vectors of interest are

$$ {}^{o}\vec{r}^{b_{o1}} = 0, \qquad {}^{b_{o1}}\vec{r}^{I_{o1}} = x_{11}\,\hat{b}_{11} + \tilde{u}_{12}\,\hat{b}_{12}, \qquad {}^{b_{o1}}\vec{r}^{b_{o2}} = L_1\,\hat{b}_{11} + q_3'\,\hat{b}_{12}, $$
$$ {}^{b_{o2}}\vec{r}^{I_{o2}} = x_{21}\,\hat{b}_{21} + \tilde{u}_{22}\,\hat{b}_{22}, \tag{6} $$

where ${}^{b_{o1}}\vec{r}^{I_{o1}}$ and ${}^{b_{o2}}\vec{r}^{I_{o2}}$ are vectors to the differential mass of each beam. The separation of the overall position vectors, shown above, is provided to take advantage of recurring quantities as these vectors and their derivatives are used in the analysis. The corresponding computer generated symbols are shown in Table 8. Output has been suppressed.

The absolute velocities of the points b_{o1} and b_{o2} are required for partial velocity calculations and for acceleration calculations. Therefore,

$$ {}^{o}_{\mathcal{N}}\vec{v}^{b_{o1}} = 0, \qquad {}^{o}_{\mathcal{N}}\vec{v}^{b_{o2}} = -s_1\,q_3'\,\hat{b}_{11} + (s_3' + s_1\,L_1)\hat{b}_{12}, \tag{7} $$

where $q_3' = \tilde{u}_{12}(L_1, t)$ and $s_3' = \dot{q}_3' = \partial\tilde{u}_{12}(L_1, t)/\partial t$. The corresponding computer i/o is shown in Table 9. In the table, only the output from the last input is shown.

The absolute accelerations of the differential beam elements can be written as

$$ {}^{o}_{\mathcal{N}}\vec{a}^{I_{o1}} = -\left(2s_1\frac{\partial\tilde{u}_{12}}{\partial t} + \dot{s}_1\,\tilde{u}_{12} + x_{11}\,s_1^2\right)\hat{b}_{11} + \left(\frac{\partial^2\tilde{u}_{12}}{\partial t^2} + x_{11}\,\dot{s}_1 - \tilde{u}_{12}\,s_1^2\right)\hat{b}_{12}, $$

TABLE 10

Derivatives of velocity vectors

```
In[224]:=
          NoxBo[1]=DvDt[N,NovBo[1]]//distributeScalars//gatherVectors
          BoxIo[1]=DvDt[N,BovIo[1]]//distributeScalars//gatherVectors
          BoxBo[1,2]=DvDt[N,BovBo[1,2]]/.qp[3]'[t] -> sp[3][t]//
          distributeScalars//gatherVectors
          BoxIo[2]=DvDt[N,BovIo[2]]//distributeScalars//gatherVectors
```

TABLE 11

Utilizing recurrence

```
In[228]=
      z[1]=Coefficient[BoxBo[1,2],b[1,1]]/ . Derivative[_][_][_]->0;
      z[2]=Coefficient[BoxBo[1,2],b[1,2]]/ . Derivative[_][_][_]->0;
      BoxBo[1,2]=BoxBo[1,2]/ . {z[1]→Z[1],z[2]→Z[2]}
Out[230]=
                                  ^
      (Z-qp [t] ṡ [t]] b  +
        1   3    1      11
                                    ^
        (Z+L ṡ [t]+ṡp [t]) b
          2  1  1     3      12
```

$$
{}^{o}_{\mathcal{N}}\vec{a}^{I_{o2}} = -(2s_1 s_3' + \dot{s}_1 q_3' + L_1 s_1'^2)\hat{b}_{11} + (\dot{s}_3' + L_1 \dot{s}_1 - q_3' s_1'^2)\hat{b}_{12}
$$

$$
- \left(2(s_1 + s_2' + s_4)\frac{\partial \tilde{u}_{22}}{\partial t} + (\dot{s}_1 + \dot{s}_2' + \dot{s}_4)\tilde{u}_{22} + x_{21}(s_1 + s_2' + s_4)^2\right)\hat{b}_{21}
$$

$$
+ \left(\frac{\partial^2 \tilde{u}_{22}}{\partial t^2} + x_{21}(\dot{s}_1 + \dot{s}_2' + \dot{s}_4) - \tilde{u}_{22}(s_1 + s_2' + s_4)^2\right)\hat{b}_{22}. \tag{8}
$$

The symbolic inputs for the acceleration terms are shown in Table 10. In the table, the output has been suppressed for brevity.

Recurrence occurs and is utilized by introducing intermediate variables such as shown in Table 11. In the table, the upper-case `Z`'s hold the place of the lower case `z`'s which are used during the simulation stage.

3.3. STRAIN ENERGY AND ASSUMED FIELD

The strain energy density functions for the beams are (assuming simple flexure)

$$
\bar{V}_1 = \tfrac{1}{2} E_1 I_1 \left(\frac{\partial^2 \tilde{u}_{12}}{\partial x_{11}^2}\right)^2, \qquad \bar{V}_2 = \tfrac{1}{2} E_2 I_2 \left(\frac{\partial^2 \tilde{u}_{22}}{\partial x_{21}^2}\right)^2. \tag{9}
$$

The corresponding computer symbolic input, including the weak form of the strain energy, takes the form shown in Table 12, where only the terms for the first beam are shown.

The deflection field is assumed to be of the form

$$
\tilde{u}(x, t) = \phi_1(t)x^2/L^4 + \phi_2(t)x^3/L^4 + \phi_3(t)x^4/L^4, \tag{10}
$$

TABLE 12

Strain energy density

```
In[231]:
         Vbar[1]=1/2 Ey[1] Ii[1] (D[u[1,2][x[1,1],t],{x[1,1],2}])^2
         dVbarUxx[1]=D[Vbar[1],D[u[1,2][x[1,1],t],{x[1,1],2}]]
```

TABLE 13

Forces and torques

```
In[236]:=
     Ib[1]=Integrate[rho[1] NoxBo[1], {x[1,1],0,L[1]}]
     +Integrate[rho[1] BoxIo[1],{x[1,1],0,L[1]}]//.
     {Dt[phi[i__][t],t] -> Dtphi[i][t],
     Dt[phi[i__][t],{t,2}] -> Dt[Dtphi[i][t], t]}//
     distributeScalars//gatherVectors
In[240]:=
     Fb[1]=Integrate[-rho[1] g n[2],{x[1,1],0,L[1]}]
In[243]:=
     Jb[1]=Integrate[rho[1] BorIo[1] >< NoxBo[1],{x[1,1],0,L[1]}]+
     Integrate[rho[1] BorIo[1] >< BoxIo[1],{x[1,1],0,L[1]}]//.
     {Dt[phi[i__][t],t] -> Dtphi[i][t],
     Dt[phi[i__][t],{t,2}] -> Dt[Dtphi[i][t],t]}//
     distributeScalars//gatherVectors
In[246]:=
     Tb[1]=Integrate[BorIo[1] >< (-rho[1] g n[2]),{x[1,1],0,L[1]}]+
     (Ta[1]-Ta[2]) b[1,3]//gatherVectors
```

similar for each beam for use in the weak formulation. It is noted that this assumed field may not be judicious for rigorous models; however, it is demonstrative of the tools described herein.

3.4. ORDINARY DIFFERENTIAL EQUATIONS

The ordinary differential equations governing the angular speeds are obtained from

$$0 = \frac{\partial_{\mathcal{M}}^{o}\vec{v}^{b_{o1}}}{\partial s_1}[\vec{F}_{B_1} - \vec{I}_{B_1}] + \frac{\partial^{\mathcal{M}}\vec{\omega}^{B_1}}{\partial s_1}[\vec{T}_{B_1} - \vec{J}_{B_1}] + \frac{\partial_{\mathcal{M}}^{o}\vec{v}^{b_{o2}}}{\partial s_1}[\vec{F}_{B_2} - \vec{I}_{B_2}] + \frac{\partial^{\mathcal{M}}\vec{\omega}^{B_2}}{\partial s_1}[\vec{T}_{B_2} - \vec{J}_{B_2}] \tag{11}$$

for s_1, and

$$0 = \frac{\partial_{\mathcal{M}}^{o}\vec{v}^{b_{o1}}}{\partial s_4}[\vec{F}_{B_1} - \vec{I}_{B_1}] + \frac{\partial^{\mathcal{M}}\vec{\omega}^{B_1}}{\partial s_4}[\vec{T}_{B_1} - \vec{J}_{B_1}] + \frac{\partial_{\mathcal{M}}^{o}\vec{v}^{b_{o2}}}{\partial s_4}[\vec{F}_{B_2} - \vec{I}_{B_2}] + \frac{\partial^{\mathcal{M}}\vec{\omega}^{B_2}}{\partial s_4}[\vec{T}_{B_2} - \vec{J}_{B_2}] \tag{12}$$

for s_4. The forces and torques (applied and inertia) are defined in symbolic form as shown in Table 13. The derivative transformations are executed to maintain a first order form for the equations of motion. Kinematic differential equations will be added.

TABLE 14

Recurrence again

```
In[260]:=
   z[14]=Coefficient[Jb[2],b[2,1]><b[1,1]]/.Derivative[1][_][_]->0;
   Jb[2]=Jb[2]/.{z[14] -> Z[14]}
```

TABLE 15

Ordinary equation of motion

```
In[270]:=
          eom[1]=Pvel[NovBo[1],s[1][t]] . (Fb[1]-Ib[1])+
          Pvel[NwB[1],s[1][t]] . (Tb[1]-Jb[1])+
          Pvel[(NovBo[1]+BovBo[1,2]),s[1][t]] . (Fb[2]-Ib[2])+
          Pvel[NwB[2],s[1][t]] . (Tb[2]-Jb[2])// . {Dt[sp[2][t],t] ->
          Dt[Dt[u(1,2][x[1,1],t],x[1,1]],{t,2}],Dt[sp[3][t],t] ->
          Dt[u[1,2][x[1,1],t],{t,2}],x[1,1] -> L[1],
          Dt[phi[i__][t],{t,2}] -> Dt[Dtphi[i][t],t]}//gatherDots
```

As before, recurrence can be typically utilized by writing intermediate variables; for another example, see Table 14.

The equations of motion for the regular generalized speeds s_1 and s_4 are written in computer form (mimicking equations (11) and (12)) as shown in Table 15. The operator `Pvel[_,_]` is used for partial differentiation. A similar formula for `eom[2]` except with `s[1][t]` replaced by `s[4][t]` is also utilized.

The transformation of vector products is performed via expressions as shown in Table 16, where `co[1][t]` and `co[2, 4]` imply $\cos q_1$ and $\cos(q_2 + q_4)$, respectively. The other transformation terms are similarly defined.

3.5. FIELD EQUATIONS

The formula for the field equation governing deflection of B_1 is

$$0 = \frac{\partial}{\partial x_{11}}\left(\frac{\partial \bar{V}_1}{\partial \tilde{u}_{12,1}}\right) - \frac{\partial^2}{\partial x_{11}^2}\left(\frac{\partial \bar{V}_1}{\partial \tilde{u}_{12,11}}\right) - \rho_A^o \tilde{a}^{I_{o1}} \cdot \hat{b}_{12}. \tag{13}$$

The structural damping term is not shown explicitly, but will be added later. The boundary conditions are given as

$$\frac{\partial \bar{V}_1}{\partial \tilde{u}_{12,1}} - \frac{\partial}{\partial x_{11}}\left(\frac{\partial \bar{V}_1}{\partial \tilde{u}_{12,11}}\right) = g'_{12}, \qquad \frac{\partial \bar{V}_1}{\partial \tilde{u}_{12,11}} = k'_{12} \tag{14, 15}$$

at $x_{11} = L_1$, and

$$\tilde{u}_{12} = \tilde{u}_{12,1} = 0 \tag{16}$$

TABLE 16

Basic transformation

```
In[271]:=
          eom[1]=eom[1] // . {b[1,1] . n[1] -> co[1][t],
          b[2, 2] . n[2] -> co[1, 2, 4][t]}
```

TABLE 17

Weak field equation

```
In[292]:=
     intterm[1][psi_]:=-rho[1]gb[1,2] . n[2]psi-
     rho[1] b[1,2] . (NoxBo[1]+BoxIo[1])psi-
     dVbarUxx[1] D[psi,{x[1,1],2}]+
     2gam[1]D[D[u[1,2][x[1,1,t],{x[1,1],2}],t]psi
In[293]:=
     bndry[1][spd_,psi_]:=(Pvel[NovBo[1],spd] . (Fb[1]-Ib[1])+
     Pvel[NwB[1],spd] . (Tb[1]-Jb[1])+
     Pvel[(NovBo[1]+BovBo[1,2]),spd] . (Fb[2]-Ib[2])+
     Pvel[NwB[2],spd] . (Tb[2]-Jb[2]))psi
In[294]:=
     eom[3]=Integrate[intterm[1][x[1,1]^2/L[1]^4],{x[1,1],0,L[1]}]+
     bndry[1][sp[2][t],Dt[x[1,1]^2/L[1]^4,x[1,1]]]+
     bndry[1][sp[3][t],x[1,1]^2/L[1]^4]//.
     {Dt[sp[2][t],t] -> Dt[Dt[u[1,2][x[1,1],t],x[1,1]],{t,2}],
     Dt[sp[3][t],t] -> Dt[u[1,2][x[1,1],t],{t,2}],x[1,1] -> L[1],
     Dt[phi[i__][t],t] -> Dtphi[i][t],
     Dt[phi[i__][t],{t,2}] -> Dt[Dtphi[i][t],t]}//gatherDots
```

at $x_{11} = 0$. The terms on the right sides of the boundary conditions are defined as

$$g'_{12} = \frac{\partial^o_{\mathcal{N}} \vec{v}^{b_{o1}}}{\partial s'_3}[\vec{F}_{B_1} - \vec{I}_{B_1}] + \frac{\partial^{\mathcal{N}} \vec{\omega}^{B_1}}{\partial s'_3}[\vec{T}_{B_1} - \vec{J}_{B_1}] + \frac{\partial^o_{\mathcal{N}} \vec{v}^{b_{o2}}}{\partial s'_3}[\vec{F}_{B_2} - \vec{I}_{B_2}] + \frac{\partial^{\mathcal{N}} \vec{\omega}^{B_2}}{\partial s'_3}[\vec{T}_{B_2} - \vec{J}_{B_2}]$$

TABLE 18

I-Matrix element

```
In[272]:=
          iM[1,1]=-D[eom[1],Dt[s[1][t],t]]//Simplify//
          gatherScalars//miniDivisions
Out[272]=
                                2
          rho (co [t] (L L  +
            2    24    1 2
          (L (20 phi [t]+15 phi [t]+12 phi [t])
            2       221       222        223
          «terms now shown»
          (264600 (rho (L+L phi [t] phi [t])+
                     1  1  1   121    122
          «terms now shown»
          8820 L rho (20 phi [t]+15 phi [t]+
                2   2      221        222
             12 phi [t]) si [t])) / 793800
                   223      24
```

TABLE 19

Generating the right side

```
In[280]:=
          rhs[1]=eom[1]// . Derivative[1][_][_]->0//
          Simplify//gatherScalars//minDivisions
```

and

$$k'_{12} = \frac{\partial^o_{\mathcal{N}} \vec{v}^{b_{o1}}}{\partial s'_2}[\vec{F}_{B_1} - \vec{I}_{B_1}] + \frac{\partial^{\mathcal{N}} \vec{\omega}^{B_1}}{\partial s'_2}[\vec{T}_{B_1} - \vec{J}_{B_1}] + \frac{\partial^o_{\mathcal{N}} \vec{v}^{b_{o2}}}{\partial s'_2}[\vec{F}_{B_2} - \vec{I}_{B_2}] + \frac{\partial^{\mathcal{N}} \vec{\omega}^{B_2}}{\partial s'_2}[\vec{T}_{B_2} - \vec{J}_{B_2}].$$

The formula for the field equation for B_2 is

$$0 = \frac{\partial}{\partial x_{21}}\left(\frac{\partial \bar{V}_2}{\partial \tilde{u}_{22,1}}\right) - \frac{\partial^2}{\partial x_{21}^2}\left(\frac{\partial \bar{V}_2}{\partial \tilde{u}_{22,11}}\right) - \rho^o_{\mathcal{N}} \vec{a}^{I_{o2}} \hat{b}_{22}\,; \tag{17}$$

damping will be added later. The boundary conditions are

$$\frac{\partial \bar{V}_2}{\partial \tilde{u}_{22,1}} - \frac{\partial}{\partial x_{11}}\left(\frac{\partial \bar{V}_2}{\partial \tilde{u}_{22,11}}\right) = 0, \qquad \frac{\partial \bar{V}_2}{\partial \tilde{u}_{22,11}} = 0 \tag{18, 19}$$

at $x_{21} = L_2$, and

$$\tilde{u}_{22} = \tilde{u}_{22,1} = 0 \tag{20}$$

at $x_{21} = 0$.

The integral weak form of the field equations can be generated for each assumed shape function via *Mathematica* as shown in Table 17, where `psi` is the shape function. The equation for the first beam, first shape function (x^2), is given as `eom[3]`. These formulas can be exercised for each mode shape. For the second beam we have similar formulas but the boundary term is zero. The term containing `gam[1]` is the structural damping term, where `gam[1]` is the damping factor.

3.6. SIMULATION

The simulation was expedited by arranging the equations of motion in the form

$$\begin{bmatrix} I_{11} & \cdots & I_{18} \\ \vdots & \ddots & \vdots \\ I_{81} & \cdots & I_{88} \end{bmatrix} \begin{Bmatrix} \dot{s}_1 \\ \vdots \\ Dt\dot{\phi}_{223} \end{Bmatrix} = \begin{Bmatrix} f_1 \\ \vdots \\ f_8 \end{Bmatrix}, \tag{21}$$

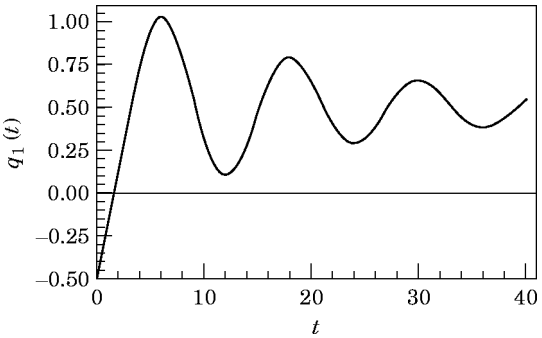

Figure 2. Co-ordinate q_1 versus time.

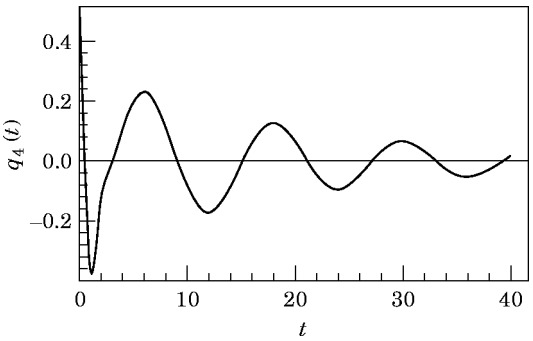

Figure 3. Co-ordinate q_4 versus time.

with kinematic differential equations of the form

$$\begin{aligned} \dot{q}_1 &= s_1, \\ &\vdots \\ \dot{\phi}_{223} &= Dt\phi_{223}. \end{aligned} \tag{22}$$

The I matrix is formed via commands such as are shown in Table 18, where only the I_{11} term is shown. The large integers are the result of simplifying the factors and divisors.

The right sides can be found via commands demonstrated in Table 19 and will include the intermediate Z variables.

Operations similar to those shown in Tables 18 and 19 can be exercised for each discrete and field variable.

The equations of motion were output to a file using the intrinsic FortranForm in *Mathematica*. The explicit function of the time notation and the indices of the symbols were changed appropriately for FORTRAN coding; another function was devised for this purpose.

A Gear multi-step integrator was utilized for the simulation. The applied torques were assumed to be of the form

$$T_{a1} = -D_1 s_1 - K_1 (q_1 - Q_1), \qquad T_{a2} = -D_2 s_4 - K_1 (q_4 - Q_4),$$

where the D's are simple damping terms and the K's are proportional gains; the commanded angles are the Q's. This control law is simplistic and used only to obtain a maneuver profile to complete the example.

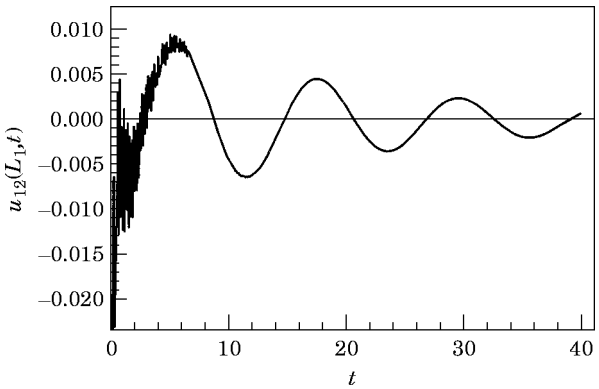

Figure 4. The beam B_1 tip deflection.

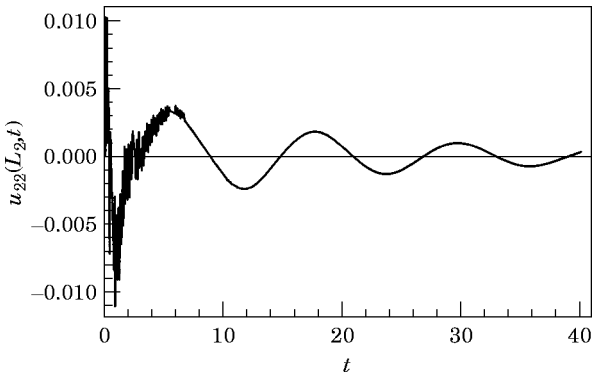

Figure 5. The beam B_2 tip deflection.

For the simulation, the beams were 3 ft ($\approx 0{\cdot}9$ m) long each with a cross-section of 2 in (≈ 5 cm) by $\frac{3}{4}$ in (≈ 2 cm). The material properties of aluminum were used with the mass density per unit length. The initial conditions for the angles were $q_1(0) = -0{\cdot}5$ rad and $q_4(0) = 0{\cdot}5$ rad; all other initial conditions were zero. The commanded angles were $Q_1 = 0{\cdot}5$ rad and $Q_4 = 0$ rad. The torque constants were $D_1 = 15$, $D_2 = 10$ ft-lb-s ($\approx 13{\cdot}6$ N m s) and $K_1 = 40$, $K_2 = 30$ ft-lb ($\approx 40{\cdot}7$ N m). The structural damping was taken as $\gamma = 0{\cdot}1$ lb s ($\approx 0{\cdot}5$ N s) for both beams. The simulation was allowed to run for 40 s.

The response of the base angle is given in Figure 2. The response of the second angle is given in Figure 3. The tip deflection of the first beam as seen from the rotating co-ordinate frame riding with B_1 is shown in Figure 4. The deflections stayed within the linear deflection assumption over the length of the beam. The tip deflection of the second beam as seen from frame B_2 is shown in Figure 5. As stated above, the control law is rudimentary as can be seen from the response and time elapsed; it was provided to demonstrate the tools, not its own merits.

4. SUMMARY AND DISCUSSION

In this paper, a set of vector algebra tools based on a commercial symbolic manipulation package were presented. These tools were explicitly formulated to assist in the closed form modelling of HPMBS via modelling algorithms previously developed by the author. A demonstration of the modelling technique and computer tools was presented. The example included the modelling and simulation of a simplified HPMBS, which was complicated to a degree that allowed the deftness of the modelling tools to be elucidated.

It can be seen by the results herein that the method described in references [20–22] can rigorously and systematically model complicated systems. The utility of utilizing pseudo co-ordinates and speeds is clear by the simplicity of the way in which the technique handles boundary conditions (see equations (14) and (15)). Creating a computer tool to directly imitate the notation of the modelling method allows one to sit down and write the equations of motion at the computer, mirroring the method of pencil and paper. The vector-based computer tool also allows for the generation of lean equations of motion. Redundant operations are relegated to memory reads. These tools will be used as a basis for fully automated closed form non-holonomic HPMBS modelling software, as well as in research and education. In education, not having a fully automated package is desirable because it allows the student to do by computer what is normally done by hand, but for systems of fair complexity; the computer input and output mimicks hand calculations. In

research, the symbolic tool as provided is useful for an analyst that requires complete control of the formulation.

ACKNOWLEDGMENTS

The author thanks the ASEE/NASA Summer Faculty Fellows program for allowing the author to work on HPMBS modelling research. The author also thanks the Center for Applied Automation and Research at Texas Tech University, and the Amarillo National Resource Center for Plutonium for their support.

REFERENCES

1. A. L. Hale and L. Meirovitch 1977 *Proceedings of a Symposium on Dynamics and Control of Large Flexible Spacecraft*, 367–385. A special purpose symbolic manipulation program for the derivation of the equations of motion for large flexible structures.
2. D. B. Schaechter and D. A. Levinson 1988 *The Journal of Astronautical Sciences* **36**, 365–388. Interactive computerized symbolic dynamics for the dynamicist.
3. M. Sherman 1988 *Proceedings of the Workshop on Multibody Simulation JPL D*-5190, 692–726. SD/EXACT and SD/FAST symbolic multibody codes.
4. R. S. Sharp 1994 *Journal of Automobile Engineering* **208**(1), 55–61. Application of multi-body computer codes to road vehicle dynamics modelling problems.
5. H. Yoshimura, H. Nakano and T. Kawase 1993 *Proceedings of the* 1993 *ASME Winter Annual Meeting DSC*: *Automated Modeling for Design*, 63–94. Modelling of multibody dynamics and a recursive symbolic generation scheme.
6. P. Davison, D. K. Longmore and C. R. Burrows 1993 *Proceedings of the* 1993 *ASME Winter Annual Meeting DSC*: *Automated Modeling for Design*, 33–45. Analysis of a flexible multibody system using an efficient automated modelling method.
7. A. A. Shabana, Y. L. Hwang and R. A. Wehage 1992 *International Journal for Numerical Methods in Engineering* **35**, 1927–1939. Projection methods in flexible multibody dynamics, part I: kinematics.
8. R. A. Wehage, A. A. Shabana and Y. L. Hwang 1992 *International Journal for Numerical Methods in Engineering* **35**, 1941–1966. Projection methods in flexible multibody dynamics, part II: dynamics and recursive projection methods.
9. P. Tomei and A. Tornambe 1988 *IEEE Transactions on Systems*, *Man and Cybernetics* **18**(5), 831–840. Approximate modeling of robots having elastic links.
10. J. Lieh and I.-U. Haque 1991 *Journal of Mechanisms*, *Transmissions*, *and Automation in Design* **113**, 124–132. Symbolic closed-form modeling and linearization of multibody systems subject to control.
11. N. Rentia and N. Vira 1991 *Computers in Industry* **17**, 49–62. Why symbolic computation in robotics?
12. V. J. Modi, A. Suleman, A. C. Ng and Y. Morita 1991 *International Journal for Numerical Methods in Engineering* **32**, 1727–1748. Approach to dynamics and control of orbiting flexible structures.
13. P.-R. Chang 1991 *International Journal of Control* **53**, 1205–1232. Computer-aided-design tool for simplification of robot dynamic model based on manipulator system performance.
14. S. Cetinkunt and W. J. Book 1989 *Robotics and Computer-Integrated Manufacturing* **5**(4), 301–310. Symbolic modeling and dynamic simulation of robotic manipulators with compliant links and joints.
15. S. Ravichandran, G. Gaonkar, J. Nagabhushanam and T. S. R. Reddy 1990 *Journal of Sound and Vibration* **137**, 495–507. Study of symbolic processing and computational aspects in helicopter dynamics.
16. H. Ashrafiuon and N. K. Mani 1990 *Journal of Mechanisms*, *Transmissions*, *and Automation in Design* **112**, 200–207. Analysis and optimal design of spatial mechanical systems.
17. M. W. Sayers 1991 *Journal of Guidance*, *Control*, *and Dynamics* **14**, 1153–1163. Symbolic computer language for multibody systems.
18. M. W. Sayers 1991 *Journal of Guidance*, *Control*, *and Dynamics* **14**, 1240–1250. Symbolic vector/dyadic multibody formalism for tree-topology.

19. S. THOMAS 1991 *Simulation* **57**, 56–72. Dynamics of spacecraft and manipulators.
20. A. A. BARHORST and L. J. EVERETT 1995 *The International Journal of Nonlinear Mechanics* **30**(1), 1–21. Modeling hybrid parameter multiple body systems: a different approach.
21. A. A. BARHORST and L. J. EVERETT 1995 *Journal of Dynamic Systems, Measurement, and Control* **117**(4), 559–569. Contact/impact in hybrid parameter multiple body systems.
22. A. A. BARHORST *Journal of Sound and Vibration*. On modeling variable structure dynamics of hybrid parameter multiple body systems.
23. S. WOLFRAM 1991 *Mathematica—A System for Doing Mathematics by Computer*. Redwood City, California: Addison-Wesley; second edition.