Equations of Motion for Multiple-Rigid-Body Systems via Symbolic Manipulation

David A. Levinson*
The University of Texas at Austin, Austin, Texas

Conventional general-purpose multibody simulation programs, although easily accessible to users, fail to provide explicit equations of motion for given spacecraft configurations. In addition, they require more computer time for performing simulations than do corresponding special-purpose programs. Nevertheless, the derivation of the equations of motion for a spacecraft consisting of two or more rigid bodies is, in general, a formidable task. In this paper, it is shown how the symbolic manipulation language FORMAC can be used in conjunction with Kane's equations to generate on a digital computer explicit equations of motion for given multiple-rigid-body systems.

Introduction

IN recent years, spacecraft designers have become increasingly involved with the dynamics of multiple-rigid-body systems. When a spacecraft is modeled as a collection of two or more rigid bodies, the derivation of the associated equations of motion becomes laborious and subject to careless errors if performed by hand. Accordingly, much attention has been directed toward the development of techniques for generating equations of motion by digital computer.

One of the earliest attempts at formulating equations of motion for multiple-rigid-body systems is attributable to Hooker and Margulies,[1] who derived dynamical equations for the attitude motion of an arbitrary number n of point-connected rigid bodies, the only restriction on the configurations included in this treatment being that chains of connected bodies must not form closed loops. The dynamical equations of Newton and Euler were written for each body, and a procedure was described for the elimination of "constraint torques" that arise at connection points having associated with them only one or two rotational degrees of freedom, thus yielding a set of $3n$ second-order scalar differential equations of motion for the system. A similar set of equations was obtained by Roberson and Wittenburg.[2] Later, Hooker[3] devised a method for eliminating the constraint torques in such a way as to reduce the total number of equations from $3n$ to the number of rotational degrees of freedom of the system. Velman[4] and Russell,[5] using a different approach, chose not to isolate each of the n point-connected rigid bodies individually, but instead constructed vector equations of motion for n different subsets of bodies, including, ultimately, a complete set of equations for the entire system. Features of both the discrete-body and nested-body approaches were combined by Frisch[6] in deriving a set of equations which forms the basis for a digital computer program. Other simulation programs have been developed by Fleischer[7] and Fleischer and Likins[8] based upon Refs. 1 and 3. More recently, Huston and Passserello[9] have made use of Lagrange's form of D'Alembert's principle as developed by Kane[10] and have, consequently, avoided the necessity of considering nonworking interbody constraint forces and torques in their derivation of the equations of motion and corresponding computer program.

All of the aforementioned formulations share certain disadvantages. First, the equations of motion for any particular multiple-rigid-body configuration exist only as indicated operations in computer programs. That is, the analyst never sees explicit terms in the equations as he would if the equations had been derived by hand and programmed term-by-term. Also, the equations must be completely constructed, using a substantial number of DO loops, at every step of a numerical integration and are not, therefore, cast in a form that can be used advantageously with certain computers, such as the CDC 6600, which have the capability of performing several simultaneous arithmetic operations on explicit expressions. Hence, there is sufficient motivation to seek a way of employing the speed and accuracy of a computer in the construction of explicit equations of motion for systems too large to be dealt with by hand. To this end, one can make use of non-numeric computation, that is, symbolic manipulation.

Any symbolic manipulation language being considered for use in formulating dynamical equations of motion must be able to carry out analytical differentiation. Among the more common such languages are SYMBAL,[11] TRIGMAN,[12] and FORMAC.[13] SYMBAL, however, being an ALGOL-based language, produces output that is not, in general, easily read or directly usable with fast FORTRAN numerical integrators. Although TRIGMAN is written in FORTRAN, its use is limited to problems having no more than 29 polynomial or 10 trigonometric variables. FORMAC, based on PL/I, is implemented easily by users familiar only with FORTRAN and contains a powerful array of manipulative functions, including provisions for releasing computer storage space that no longer is needed. Furthermore, FORMAC possesses a variety of output formats, including punched-card options available in a newer version of the language.[14] Hence, FORMAC is ideally suited for the task at hand and is, in fact, employed in the present paper.

The next choice which must be made is that of a method of analytical dynamics to be used with FORMAC. Adopted in the present work is Kane's development of Lagrange's form of D'Alembert's principle, which, as previously pointed out in connection with the work of Huston and Passerello,[9] provides for the automatic elimination of nonworking interbody constraint forces and torques. In addition, this method is applied in conjunction with Kane's u's taken as dependent variables (see Ref. 10, pp. 44-48, 188-192), these being linear combinations of time derivatives of system-generalized coordinates (for example, angular velocity measure numbers), the use of which produces equations having less complexity than those derived using independent generalized coordinates themselves as dependent variables.

This paper contains a brief account of the elements of the FORMAC language which are directly applicable to the

Presented as Paper 76-816 at the AIAA/AAS Astrodynamics Conference, San Diego, Calif., Aug. 18-20, 1976; submitted Sept. 7, 1976; revision received Feb. 22, 1977.

Index categories: Spacecraft Dynamics and Control; Spacecraft Simulation.

*Research Assistant, Department of Aerospace Engineering and Engineering Mechanics. Student Member AIAA.

formulation of equations of motion for multiple-rigid-body systems. An example configuration consisting of two point-connected rigid bodies then is analyzed in order to illustrate the use of FORMAC with Kane's equations. For this case, steps are given for generating the explicit equations of motion and for subsequently putting them into a form appropriate for numerical integration. Finally, simulation results obtained from these equations are presented as a practical application of the method.

Elements of the FORMAC Language

In order to bring to light the basic capabilities of FORMAC which are applicable to the derivation of equations of motion, a short summary of the relevant portions of the language will now be given. A complete description of the FORMAC language may be found in Refs. 13 and 14.

To begin with, FORMAC assignment statements are designed by the word LET. For example, the statement LET(X1 = C*(A + B)**3); assigns to the FORMAC variable X1 the value $C(A+B)^3$. More than one assignment statement may appear within the range of a single LET. If one wishes to express X1 in its fully expanded form using both the binomial theorem and the distributive law, one can invoke the EXPAND routine. Execution of the statement LET (X2 = EXPAND (X1); results in the assignment X2 = 3C A B^2 + 3C A^2B + C A^3 + C B^3.

Another useful routine for manipulating expressions is COEFF. Written in the general format LET (E3 = COEFF (E1,E2)); COEFF assigns to E3 the coefficient of the subexpression E2 in the expression E1. For example, with X2 as previously defined, the statement LET (X3 = COEFF (X2, C)); results in X3 = 3 A B^2 + 3 A^2 B + A^3 + B^3.

One of the most important attributes of FORMAC is its ability to deal with unspecified functions. For instance, LET (X4 = D*T*SIN(T) + Q.(T)); assigns to X4 the sum of two functions of a variable T, the first depending on T explicitly and the second, designated by a decimal point, an implicit function of T. This feature is found to be most useful in connection with DERIV, one of FORMAC's analytical differentiation routines. In this instance, the statement LET (X5 = DERIV(X4,T)); produces X5 = D T COS (T) + D SIN(T) + $Q^{(1)}$.(T), where the superscript 1 enclosed in parentheses signifies the first derivative of the function Q with respect to its argument T.

If one desires to assign a name to the first derivative of an unspecified function, one can apply the FORMAC routine pair DIFF, CHAIN. With X4 retaining its previous value, the sequence of statements LET(DIFF(Q) = CHAIN(QDOT); X6 = DERIV(X4, T)); leads to X6 = QDOT + D T COS (T) + D SIN (T), where X6 differs from X5 by the fact that, in X6, Q(1)·(T) has been replaced with QDOT.

One can avail oneself of FORMAC's substitution capability by employing the routine REPLACE, which, in its general format, is written as LET(E = REPLACE (E1,A1,B1,A2,B2,...,AN,BN)); where E1,A1,B1,...,AN,BN are arbitrary FORMAC expressions. Execution is accomplished as follows: Each occurrence of A1 in E1 is replaced with B1 to yield an expression E2. Then each occurrence of A2 in E2 is replaced with B2 to produce an expression E3. The process is continued until each occurrence of AN in EN is replaced with BN, at which time the resulting expression is returned as the value of E. As an illustration, LET(X7 = REPLACE(X6, SIN(T),ST,COS(T),CT)); results in X7 = QDOT + ST D + CT D T, where X6 is defined as in the preceding example.

Printed output in FORMAC is handled by the PRINT_OUT statement. For example, PRINT_OUT (X7); produces the printed line X7 = QDOT + ST D + CT D T. Note that the expression has been automatically edited to afford facility in reading. In the latest version of the language[14] FORMAC expressions also may be punched on cards. The routine PUNCH_F, used in the same manner as PRINT_OUT, results in the punched output of syntactically correct FORTRAN statements. In addition, there are similar punching routines for PL/I and double-precision FORTRAN.

The processing of long FORMAC expressions requires a significant amount of computer core. FORMAC provides two routines for circumventing major storage difficulties, SAVE and ATOMIZE. SAVE (E1; E2;...;EN); copies the expressions named by E1,...,EN into secondary storage while releasing the space in core that they previously occupied. A subsequent reference to any EI (I = 1,...,N) results in the automatic return to core of the expression that EI names. ATOMIZE(E1; E2;...;EN); causes the space occupied by the expressions named by E1,...,EN to be released. The value of any FORMAC variable that has been "atomized" is represented by a null string. For example, ATOMIZE(X7); yields, upon output, X7 = .

One now can begin to comprehend the potential utility of the FORMAC capability, of which the preceding discussion covers but a small part. The idea behind a FORMAC-based scheme for deriving equations of motion is this: rather than create large general-purpose multibody programs that numerically construct and integrate equations of motion for any spacecraft configuration, one can proceed from the concept of treating each multiple-rigid-body system as a separate case and using FORMAC to derive its explicit equations of motion once and for all. The example that follows is intended to demonstrate the feasibility of such an approach.

Application

The system to be considered is a spacecraft (see Fig. 1) consisting of two rigid bodies: a main body B, and a rotor R that is connected to B at point P by a hinge whose axis ζ maintains a fixed orientation relative to both B and R. Lines ξ_i, η_i $(i=1,2,3)$ are, respectively, principal axes of inertia of B and R for their mass centers B^* and R^*, and ζ is presumed to be parallel to ξ_1 (and hence to η_1). Thus, if one introduces a dextral set of unit vectors $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, $\boldsymbol{b}_3$, respectively parallel to ξ_1, ξ_2, ξ_3, and unit vectors $\boldsymbol{r}_i$ parallel to η_i $(i=2,3)$ such that $\boldsymbol{b}_1$, $\boldsymbol{r}_2$, $\boldsymbol{r}_3$ also form a dextral set, then R can be brought into a general orientation with respect to B by initially aligning $\boldsymbol{r}_i$ with $\boldsymbol{b}_i$ $(i=2,3)$ and then subjecting R to a rotation about ζ relative to B characterized by $q\boldsymbol{b}_1$. The mass of B is m_B, and the moments of inertia of B about ξ_i are β_i $(i=1,2,3)$. Similarly, R has mass m_R, and its moments of inertia about η_i are denoted by ρ_i $(i=1,2,3)$. The spacecraft is assumed to be moving in a Newtonian reference frame N.

The first step in the process is to record the vector and dyadic quantities from which Kane's equations of motion for this system can be constructed. To begin with, the unit vectors $\boldsymbol{r}_2$, $\boldsymbol{r}_3$ can be written as

$$\boldsymbol{r}_2 = \cos q\, \boldsymbol{b}_2 + \sin q\, \boldsymbol{b}_3 \tag{1}$$

$$\boldsymbol{r}_3 = -\sin q\, \boldsymbol{b}_2 + \cos q\, \boldsymbol{b}_3 \tag{2}$$

so that all vectors and dyadics herein ultimately can be exhibited in the basis $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, $\boldsymbol{b}_3$ for the sake of uniformity. Next, the angular velocity ${}^N\boldsymbol{\omega}^B$ of B in N always can be expressed in terms of functions u_1, u_2, u_3 of time t as

$${}^N\boldsymbol{\omega}^B = u_1\boldsymbol{b}_1 + u_2\boldsymbol{b}_2 + u_3\boldsymbol{b}_3 \tag{3}$$

so that the angular velocity ${}^N\boldsymbol{\omega}^R$ of R in N then is given by

$${}^N\boldsymbol{\omega}^R = (u_1 + u_4)\boldsymbol{b}_1 + u_2\boldsymbol{b}_2 + u_3\boldsymbol{b}_3 \tag{4}$$

where

$$u_4 \triangleq \dot{q} \tag{5}$$

The angular acceleration ${}^N\boldsymbol{\alpha}^B$ of B in N can be written as

$${}^N\boldsymbol{\alpha}^B = \dot{u}_1\boldsymbol{b}_1 + \dot{u}_2\boldsymbol{b}_2 + \dot{u}_3\boldsymbol{b}_3 \tag{6}$$

whereas ${}^N\boldsymbol{\alpha}^R$, the angular acceleration of R in N, is furnished by the equation

$$ {}^N\boldsymbol{\alpha}^R = {}^B\mathrm{d}/\mathrm{d}t({}^N\boldsymbol{\omega}^R) + {}^N\boldsymbol{\omega}^B \times {}^N\boldsymbol{\omega}^R \tag{7} $$

where ${}^B\mathrm{d}/\mathrm{d}t$ denotes differentiation with respect to t in B.

One can express the position vectors $\boldsymbol{p}^{P/B^*}$ and $\boldsymbol{p}^{P/R^*}$ of P relative to B^* and R^* as

$$ \boldsymbol{p}^{P/B^*} = b_1\boldsymbol{b}_1 + b_2\boldsymbol{b}_2 + b_3\boldsymbol{b}_3 \tag{8} $$

$$ \boldsymbol{p}^{P/R^*} = r_1\boldsymbol{b}_1 + r_2\boldsymbol{r}_2 + r_3\boldsymbol{r}_3 \tag{9} $$

where b_i, r_i $(i=1,2,3)$ are constants, and $\boldsymbol{p}^{R^*/B^*}$, the position vector of R^* relative to B^*, as

$$ \boldsymbol{p}^{R^*/B^*} = \boldsymbol{p}^{P/B^*} - \boldsymbol{p}^{P/R^*} \tag{10} $$

The velocity ${}^N\boldsymbol{v}^{B^*}$ of B^* in N can be written in terms of functions u_5, u_6, u_7 of t as

$$ {}^N\boldsymbol{v}^{B^*} = u_5\boldsymbol{b}_1 + u_6\boldsymbol{b}_2 + u_7\boldsymbol{b}_3 \tag{11} $$

so that ${}^N\boldsymbol{v}^{R^*}$, the velocity of R^* in N, is obtained from

$$ {}^N\boldsymbol{v}^{R^*} = {}^N\boldsymbol{v}^{B^*} + \frac{{}^B\mathrm{d}}{\mathrm{d}t}(\boldsymbol{p}^{R^*/B^*}) + {}^N\boldsymbol{\omega}^B \times \boldsymbol{p}^{R^*/B^*} \tag{12} $$

The accelerations ${}^N\boldsymbol{a}^{B^*}$, ${}^N\boldsymbol{a}^{R^*}$ of B^* and R^* in N then are given by

$$ {}^N\boldsymbol{a}^{B^*} = \frac{{}^B\mathrm{d}}{\mathrm{d}t}({}^N\boldsymbol{v}^{B^*}) + {}^N\boldsymbol{\omega}^B \times {}^N\boldsymbol{v}^{B^*} \tag{13} $$

$$ {}^N\boldsymbol{a}^{R^*} = \frac{{}^B\mathrm{d}}{\mathrm{d}t}({}^N\boldsymbol{v}^{R^*}) + {}^N\boldsymbol{\omega}^B \times {}^N\boldsymbol{v}^{R^*} \tag{14} $$

and inertia dyadics $\boldsymbol{I}^{B/B^*}$ and $\boldsymbol{I}^{R/R^*}$ of B for B^* and R for R^* can be written as

$$ \boldsymbol{I}^{B/B^*} = \beta_1\boldsymbol{b}_1\boldsymbol{b}_1 + \beta_2\boldsymbol{b}_2\boldsymbol{b}_2 + \beta_3\boldsymbol{b}_3\boldsymbol{b}_3 \tag{15} $$

$$ \boldsymbol{I}^{R/R^*} = \rho_1\boldsymbol{b}_1\boldsymbol{b}_1 + \rho_2\boldsymbol{r}_2\boldsymbol{r}_2 + \rho_3\boldsymbol{r}_3\boldsymbol{r}_3 \tag{16} $$

thus providing the quantities necessary for the construction of inertia forces $(\boldsymbol{F}^*)_B$, $(\boldsymbol{F}^*)_R$ and inertia torques $(\boldsymbol{T}^*)_B$, $(\boldsymbol{T}^*)_R$ for B and R in N, given by (see Ref. 10, pp. 91, 115-116)

$$ (\boldsymbol{F}^*)_B = -m_B\,{}^N\boldsymbol{a}^{B^*} \tag{17} $$

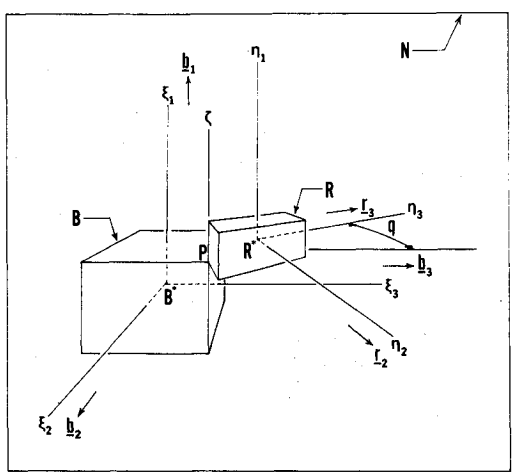

Fig. 1 Schematic representation of spacecraft.

$$ (\boldsymbol{F}^*)_R = -m_R\,{}^N\boldsymbol{a}^{R^*} \tag{18} $$

$$ (\boldsymbol{T}^*)_B = (\boldsymbol{I}^{B/B^*}\cdot{}^N\boldsymbol{\omega}^B)\times{}^N\boldsymbol{\omega}^B - \boldsymbol{I}^{B^*/B^*}\cdot{}^N\boldsymbol{\alpha}^B \tag{19} $$

$$ (\boldsymbol{T}^*)_R = (\boldsymbol{I}^{R/R^*}\cdot{}^N\boldsymbol{\omega}^R)\times{}^N\boldsymbol{\omega}^R - \boldsymbol{I}^{R/R^*}\cdot{}^N\boldsymbol{\alpha}^R \tag{20} $$

The system of contact and body forces acting on B is assumed to be equivalent to a force $(\boldsymbol{F})_B$, whose line of action passes through B^*, given by

$$ (\boldsymbol{F})_B = F_1\boldsymbol{b}_1 + F_2\boldsymbol{b}_2 + F_3\boldsymbol{b}_3 \tag{21} $$

and a couple of torque $(\boldsymbol{T})_B$ given by

$$ (\boldsymbol{T})_B = T_1\boldsymbol{b}_1 + T_2\boldsymbol{b}_2 + T_3\boldsymbol{b}_3 \tag{22} $$

whereas R is assumed to be free of such forces.

Generalized active forces F_r and generalized inertia forces F_r^* associated with u_r $(r=1,\ldots,7)$then can be obtained from the expressions

$$ F_r = \frac{\partial {}^N\boldsymbol{v}^{B^*}}{\partial u_r}\cdot(\boldsymbol{F})_B + \frac{\partial {}^N\boldsymbol{\omega}^B}{\partial u_r}\cdot(\boldsymbol{T})_B \qquad (r=1,\ldots,7) \tag{23} $$

$$ F_r^* = \frac{\partial {}^N\boldsymbol{v}^{B^*}}{\partial u_r}\cdot(\boldsymbol{F}^*)_B + \frac{\partial {}^N\boldsymbol{v}^{R^*}}{\partial u_r}\cdot(\boldsymbol{F}^*)_R + \frac{\partial {}^N\boldsymbol{\omega}^B}{\partial u_r}\cdot(\boldsymbol{T}^*)_B + \frac{\partial {}^N\boldsymbol{\omega}^R}{\partial u_r}\cdot(\boldsymbol{T}^*)_R \qquad (r=1,\ldots,7) \tag{24} $$

so that, finally, one arrives at Kane's equations, given by

$$ F_r + F_r^* = 0 \quad (r=1,\ldots,7) \tag{25} $$

(Details of Kane's theory may be found throughout Ref. 10.)

Equation (25) represents a set of first-order ordinary differential equations of the form

$$ X\dot{u} = Y \tag{26} $$

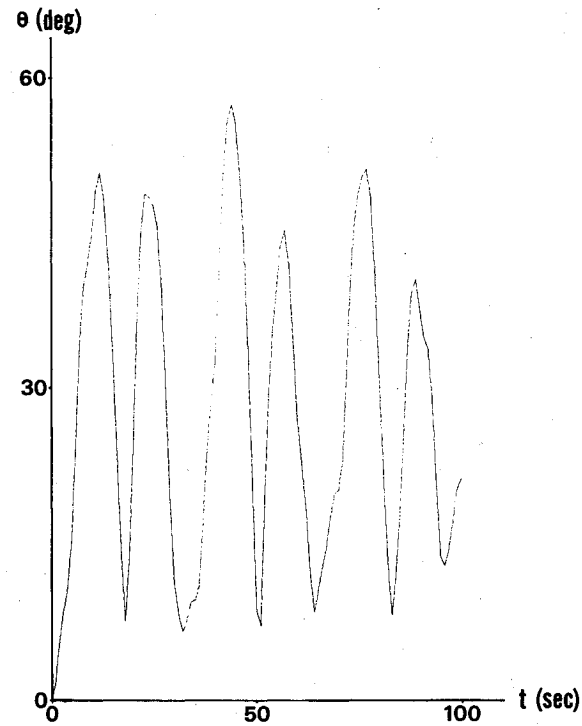

Fig. 2 Simulation results.

where X is a 7×7 matrix of functions of q, Y is a 7×1 matrix of functions of q and u_r $(r=1,\ldots,7)$, and $\dot{u}$ is the 7×1 matrix containing $\dot{u}_1,\ldots,\dot{u}_7$. Note that one cannot obtain a complete description of the motion of the spacecraft from Eq. (26) alone. One must supply, in addition, a set of kinematical equations that relate $u_1,\ldots,u_7$ to variables characterizing the orientation of R relative to B, the orientation of B in N, and the position of B^* in N. The first of these equations is furnished by Eq. (5), and the remainder may be chosen at one's discretion. For example, if the relationship between $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, $\boldsymbol{b}_3$ and a dextral set of mutually perpendicular unit vectors $\boldsymbol{n}_1$, $\boldsymbol{n}_2$, $\boldsymbol{n}_3$ fixed in N is specified by the Euler parameters, ϵ_1, ϵ_2, ϵ_3, ϵ_4, and if the position vector $\boldsymbol{p}^{B^*/O}$ of B^* relative to a point O fixed in N is written in the form $\boldsymbol{p}^{B^*/O}=x_1\boldsymbol{n}_1+x_2\boldsymbol{n}_2+x_3\boldsymbol{n}_3$, where x_i $(i=1,2,3)$ are functions of t, then the associated kinematical equations are given by[15]

$$\dot{\epsilon}_1 = \tfrac{1}{2}(u_1\epsilon_4-u_2\epsilon_3+u_3\epsilon_2) \tag{27}$$

$$\dot{\epsilon}_2 = \tfrac{1}{2}(u_1\epsilon_3+u_2\epsilon_4-u_3\epsilon_1) \tag{28}$$

$$\dot{\epsilon}_3 = \tfrac{1}{2}(-u_1\epsilon_2+u_2\epsilon_1+u_3\epsilon_4) \tag{29}$$

$$\dot{\epsilon}_4 = \tfrac{1}{2}(-u_1\epsilon_1-u_2\epsilon_2-u_3\epsilon_3) \tag{30}$$

$$\dot{x}_1 = u_5C_{11}+u_6C_{12}+u_7C_{13} \tag{31}$$

$$\dot{x}_2 = u_5C_{21}+u_6C_{22}+u_7C_{23} \tag{32}$$

$$\dot{x}_3 = u_5C_{31}+u_6C_{32}+u_7C_{33} \tag{33}$$

where the direction cosines $C_{ij}=\boldsymbol{n}_i\cdot\boldsymbol{b}_j$ $(i,j=1,2,3)$ are given by

$$C_{11} = 1-2\epsilon_2^2-2\epsilon_3^2 \tag{34}$$

$$C_{12} = 2(\epsilon_1\epsilon_2-\epsilon_3\epsilon_4) \tag{35}$$

$$C_{13} = 2(\epsilon_3\epsilon_1+\epsilon_2\epsilon_4) \tag{36}$$

$$C_{21} = 2(\epsilon_1\epsilon_2+\epsilon_3\epsilon_4) \tag{37}$$

$$C_{22} = 1-2\epsilon_3^2-2\epsilon_1^2 \tag{38}$$

$$C_{23} = 2(\epsilon_2\epsilon_3-\epsilon_1\epsilon_4) \tag{39}$$

$$C_{31} = 2(\epsilon_3\epsilon_1-\epsilon_2\epsilon_4) \tag{40}$$

$$C_{32} = 2(\epsilon_2\epsilon_3+\epsilon_1\epsilon_4) \tag{41}$$

$$C_{33} = 1-2\epsilon_1^2-2\epsilon_2^2 \tag{42}$$

Although Eqs. (1-4) and (6-24) are comparatively simple expressions when written in vector-dyadic form, complications arise when one attempts to use them to construct Eq. (26) by hand. In the first place, the process of generating Eq. (25) requires the evaluation of numerous derivatives, cross products, and dot products. To make matters worse, obtaining Eq. (26) from Eq. (25) involves the identification and isolation of 56 rather complicated subexpressions. Appendix A contains the listing of a FORMAC program that performs all of the aforementioned manipulations and produces as output the elements of the matrices X and Y punched on cards as FORTRAN statements. The program works with the b_i $(i=1,2,3)$ measure numbers of the vectors appearing in Eqs. (1-4) and (6-24). A listing of the punched output, produced by the Amdahl 470 computer at Texas A&M University via a remote terminal located at The University of Texas at Austin, may be found in Appendix B. Similar punched decks can be obtained for other spacecraft configurations by writing and implementing the corresponding FORMAC programs. In this way, one can create a "library" consisting of decks containing equations of motion for particular spacecraft of interest. Such decks can be employed directly with a linear algebraic equation solver routine and a numerical integrator to perform simulations on a digital computer.

Simulation

As a check on the correctness of the equations listed in Appendix B, a test simulation was performed using values of the system parameters chosen as follows:

m_B	= 500 kg	m_R	= 10 kg
β_1	= 110 kg-m^2	ρ_1	= 50 kg-m^2
β_2	= 100 kg-m^2	ρ_2	= 45 kg-m^2
β_3	= 90 kg-m^2	ρ_3	= 10 kg-m^2
b_1	= 3 m	r_1	= −0.1 m
b_2	= 4 m	r_2	= −0.2 m
b_3	= 5 m	r_3	= −3.5 m

The quantities F_i, T_i $(i=1,2,3)$ were set equal to zero, and initial conditions for Eqs. (5) and (26-33) were taken to be

u_i	= 0	$(i=1,2,3,5,6,7)$
u_4	= 1 rad/sec	
q	= 0	
x_i	= 0	$(i=1,2,3)$
ϵ_i	= 0	$(i=1,2,3)$
ϵ_4	= 1	

A sample of the results obtained from the simulation is presented in Fig. 2 in the form of a 100-sec time-plot of the angle θ between $\boldsymbol{n}_1$ and $\boldsymbol{b}_1$. Values of θ were computed at each time step of the numerical integration from the expression [see Eq. (34)].

$$\theta=\cos^{-1}(C_{11}) \tag{43}$$

One can see from Fig. 2 that the spacecraft is performing motions in N which are oscillatory in nature.

During the simulation, computations also were made of the kinetic energy K of the spacecraft and the magnitude H of the system's central angular momentum vector. It was found that K and H retained their respective initial values of 86.4500 N-m and 365.7642 kg-m^2/sec throughout the numerical integration. Since conservation of both of these quantities must follow from the conditions $F_i=0$, $T_i=0$ $(i=1,2,3)$, the claim that the results given in Appendix B (and in Fig. 2) are correct is well founded.

Conclusion

A successful attempt has been made to derive the explicit equations of motion for a multiple-rigid-body spacecraft using symbolic manipulation. The particular configuration considered herein was chosen because its equations of motion are of small enough size to be accommodated in an appendix to the paper, and yet possess sufficient complexity to warrant the use of a computer in their derivation. Although, in this investigation, FORMAC was not applied to the analysis of systems composed of more than two rigid bodies, the prospects for doing so appear quite good, particularly since the program listed in Appendix A consumed only 27 sec of CPU time while requiring 196K bytes of computer memory in producing the equations tabulated in Appendix B.

Appendix A: FORMAC Program

```
     ROTOR1: PROCEDURE OPTIONS(MAIN);
     FORMAC_OPTIONS;
     F_FUNCTION(SQUISH);
/*                                                                    */
/*                                                                    */
/******************** FUNCTION PROCEDURE "SQUISH" ********************/
/*                                                                    */
     SQUISH: F_PROCEDURE(S);
             LET(S=REPLACE(S,SIN(Q.(T)),SQ,
                           COS(Q.(T)),CQ));
             DO JSQ=1 TO 7; LET(JSQ="JSQ");
             LET(S=REPLACE(S,U(JSQ).(T),U(JSQ)));
             END;
             F_RETURN(S);
     F_END SQUISH;
/*                                                                    */
/*                                                                    */
/******************** SUBSCRIPTS FOR CROSS PRODUCTS ********************/
/*                                                                    */
     LET(P1(1)=2; P1(2)=3; P1(3)=1;
         P2(1)=3; P2(2)=1; P2(3)=2);
/*                                                                    */
/*                                                                    */
/************************ ASSIGNMENT OF NAMES TO ************************/
/************************ TIME DERIVATIVES OF U'S ************************/
/************************ AND TIME DERIVATIVE OF Q ************************/
/*                                                                    */
     DO J=1 TO 7; LET(J="J");
     LET(DIFF(U(J))=CHAIN(UD(J)));
     END;
     LET(DIFF(Q)=CHAIN(U(4).(T)));
/*                                                                    */
/*                                                                    */
/******************** ANGULAR VELOCITY MEASURE NUMBERS ********************/
/*                                                                    */
     DO J=1 TO 3; LET(J="J");
     LET(OB(J)=U(J).(T); OR(J)=U(J).(T));
     END;
     LET(OR(1)=OR(1)+U(4).(T));
/*                                                                    */
/*                                                                    */
/************** ANGULAR ACCELERATION MEASURE NUMBERS FOR R **************/
/*                                                                    */
     DO J=1 TO 3; LET(J="J");
     LET(ALR(J)=DERIV(OR(J),T)+OB(P1(J))*OR(P2(J))
                              -OB(P2(J))*OR(P1(J)));
     END;
/*                                                                    */
/*                                                                    */
/************************** DIRECTION COSINES **************************/
/************************** OF R RELATIVE TO B **************************/
/*                                                                    */
     LET(CR(1,1)=1; CR(1,2)=0;          CR(1,3)= 0;
         CR(2,1)=0; CR(2,2)=COS(Q.(T)); CR(2,3)=-SIN(Q.(T));
         CR(3,1)=0; CR(3,2)=SIN(Q.(T)); CR(3,3)= COS(Q.(T)));
/*                                                                    */
/*                                                                    */
/***************** MEASURE NUMBERS OF POSITION VECTOR *****************/
/************************ OF R* RELATIVE TO B* ************************/
/*                                                                    */
     DO J=1 TO 3; LET(J="J");
     LET(P(J)=B(J)-(R(1)*CR(J,1)+R(2)*CR(J,2)+R(3)*CR(J,3)));
     END;
/*                                                                    */
/*                                                                    */
/***************** MEASURE NUMBERS OF VELOCITY VECTORS *****************/
/********************** OF MASS CENTERS OF BODIES **********************/
/*                                                                    */
     DO J=1 TO 3; LET(J="J");
     LET(VB(J)=U(J+4).(T);
         VR(J)=VB(J)+DERIV(P(J),T)+OB(P1(J))*P(P2(J))
                                  -OB(P2(J))*P(P1(J)));
     END;
/*                                                                    */
/*                                                                    */
```

```
/*************** MEASURE NUMBERS OF ACCELERATION VECTORS **************/
/********************** OF MASS CENTERS OF BODIES **********************/
/*                                                                     */
     DO J=1 TO 3; LET(J="J");
     LET(AB(J)=DERIV(VB(J),T)+OB(P1(J))*VB(P2(J))
                             -OB(P2(J))*VB(P1(J));
         AR(J)=DERIV(VR(J),T)+OB(P1(J))*VR(P2(J))
                             -OB(P2(J))*VR(P1(J)));
     END;
/*                                                                     */
     DO J=1 TO 3; LET(J="J");
     LET(VB(J)=SQUISH(VB(J)); VR(J)=SQUISH(VR(J));
         AB(J)=SQUISH(AB(J)); AR(J)=SQUISH(AR(J)));
     END;
/*                                                                     */
/*                                                                     */
/************************** MOMENTS OF INERTIA *************************/
/*                                                                     */
     DO J=1 TO 3; LET(J="J");
        DO K=1 TO 3; LET(K="K");
        LET(IR(J,K)=0);
           DO L=1 TO 3; LET(L="L");
           LET(IR(J,K)=IR(J,K)+RHO(L)*CR(J,L)*CR(K,L));
           END;
        END;
     END;
/*                                                                     */
/*                                                                     */
/******************* INERTIA TORQUE MEASURE NUMBERS ********************/
/*                                                                     */
     DO J=1 TO 3; LET(J="J");
     LET(TB(J)=U(P1(J))*U(P2(J))*(BETA(P1(J))-BETA(P2(J)))
                                                    -UD(J)*BETA(J));
     LET(TR(J)=0);
        DO K=1 TO 3; LET(K="K");
        LET(TR(J)=TR(J)+(IR(P1(J),K)*OR(K))*OR(P2(J))
                       -(IR(P2(J),K)*OR(K))*OR(P1(J))
                        -IR(J,K)*ALR(K));
        END;
     END;
/*                                                                     */
     DO J=1 TO 3; LET(J="J");
     LET(OB(J)=SQUISH(OB(J)); OR(J)=SQUISH(OR(J));
         TB(J)=SQUISH(TB(J)); TR(J)=SQUISH(TR(J)));
     ATOMIZE(ALR(J));
        DO K=1 TO 3; LET(K="K");
        ATOMIZE(IR(J,K); CR(J,K));
        END;
     END;

/*                                                                     */
/*                                                                     */
/************************** EQUATIONS OF MOTION ************************/
/*                                                                     */
     DO J=1 TO 7; LET(J="J");
     LET(E(J)=0);
        DO K=1 TO 3; LET(K="K");
        LET(E(J)=E(J)+(F(K)-MB*AB(K))*DERIV(VB(K),U(J))
                      -MR*AR(K) *DERIV(VR(K),U(J))
                  +(T(K)+TB(K))*DERIV(OB(K),U(J))
                      +TR(K) *DERIV(OR(K),U(J)));
        END;
     PRINT_OUT(E(J)); PUT SKIP(4);
     SAVE(E(J));
     END; PUT SKIP(4);
/*                                                                     */
     DO J=1 TO 3; LET(J="J");
     ATOMIZE(AB(J); AR(J); TB(J); TR(J));
     END;
/*                                                                     */
     DO J=1 TO 7; LET(J="J");
     LET(XE(J)=EXPAND(E(J)));
        DO K=1 TO 7; LET(K="K");
        LET(X(J,K)=COEFF(XE(J),UD(K)));
        LET(X(J,K)=EXPAND(REPLACE(X(J,K),CQ*CQ,1-SQ*SQ));
            X(J,K)=REPLACE(X(J,K),MR,0)+MR*COEFF(X(J,K),MR));
        PRINT_OUT(X(J,K)); PUT SKIP(4);
        PUNCH_F(X(J,K));
        ATOMIZE(X(J,K));
        END; PUT SKIP(4);
```

```
      ATOMIZE(XE(J));
      LET(Y(J)=-REPLACE(E(J),UD(1),0,UD(2),0,UD(3),0,
                                UD(4),0,UD(5),0,UD(6),0,UD(7),0));
      SAVE(Y(J));
      ATOMIZE(E(J));
      END;
/*                                                                    */
      DO J=1 TO 7; LET(J="J");
      PRINT_OUT(Y(J)); PUT SKIP(4);
      PUNCH_F(Y(J));
      ATOMIZE(Y(J));
      END;
/*                                                                    */
      END ROTOR1;
```

Appendix B: Listing of Punched Output

```
 X(1,1) = -BETA(1)-RHO(1) + MR*(CQ*B(3)*R(3)**2 + CQ*R(2)*B(2)**2 + S
CQ*B(3)*R(2)**2-SQ*R(3)*B(2)**2-B(3)**2-R(2)**2-R(3)**2-B(2)**2)
 X(1,2) = MR*(-CQ*B(1)*R(2) + CQ*R(1)*R(2) + SQ*B(1)*R(3)-SQ*R(1)*R
C(3) + B(1)*B(2)-R(1)*B(2))
 X(1,3) = MR*(-CQ*B(1)*R(3) + CQ*R(1)*R(3)-SQ*B(1)*R(2) + SQ*R(1)*R
C(2) + B(3)*B(1)-B(3)*R(1))
 X(1,4) = -RHO(1) + MR*(CQ*B(3)*R(3) + CQ*R(2)*B(2) + SQ*B(3)*R(2)-
CSQ*R(3)*B(2)-R(2)**2-R(3)**2)
 X(1,5) = 0
 X(1,6) = MR*(B(3)-CQ*R(3)-SQ*R(2))
 X(1,7) = MR*(-B(2) + CQ*R(2)-SQ*R(3))
 X(2,1) = MR*(-CQ*B(1)*R(2) + CQ*R(1)*R(2) + SQ*B(1)*R(3)-SQ*R(1)*R
C(3) + B(1)*B(2)-R(1)*B(2))
 X(2,2) = -BETA(2)-RHO(2) + MR*(-CQ*SQ*R(2)*R(3)**2 + CQ*B(3)*R(3)**2
C + SQ*B(3)*R(2)**2 + B(1)*R(1)**2-SQ**2*R(2)**2 + SQ**2*R(3)**2-B(3)
C**2-B(1)**2-R(1)**2-R(3)**2)-RHO(3)*SQ**2 + RHO(2)*SQ**2
 X(2,3) = MR*(CQ*SQ*R(2)**2-CQ*SQ*R(3)**2-CQ*B(3)*R(2)-CQ*R(3)*B(2)
C + SQ*B(3)*R(3)-SQ*R(2)*B(2) + B(3)*B(2) + R(2)*R(3)-SQ**2*R(2)*R(
C3)**2) + RHO(3)*CQ*SQ-CQ*RHO(2)*SQ
 X(2,4) = MR*(-CQ*B(1)*R(2) + CQ*R(1)*R(2) + SQ*B(1)*R(3)-SQ*R(1)*R
C(3))
 X(2,5) = MR*(-B(3) + CQ*R(3) + SQ*R(2))
 X(2,6) = 0
 X(2,7) = MR*(B(1)-R(1))
 X(3,1) = MR*(-CQ*B(1)*R(3) + CQ*R(1)*R(3)-SQ*B(1)*R(2) + SQ*R(1)*R
C(2) + B(3)*B(1)-B(3)*R(1))
 X(3,2) = MR*(CQ*SQ*R(2)**2-CQ*SQ*R(3)**2-CQ*B(3)*R(2)-CQ*R(3)*B(2)
C + SQ*B(3)*R(3)-SQ*R(2)*B(2) + B(3)*B(2) + R(2)*R(3)-SQ**2*R(2)*R(
C3)**2) + RHO(3)*CQ*SQ-CQ*RHO(2)*SQ
 X(3,3) = -BETA(3)-RHO(3) + MR*(CQ*SQ*R(2)*R(3)*2 + CQ*R(2)*B(2)**2-
CSQ*R(3)*B(2)**2 + B(1)*R(1)**2 + SQ**2*R(2)**2-SQ**2*R(3)**2-B(1)**2
C-R(1)**2-R(2)**2-B(2)**2) + RHO(3)*SQ**2-RHO(2)*SQ**2
 X(3,4) = MR*(-CQ*B(1)*R(3) + CQ*R(1)*R(3)-SQ*B(1)*R(2) + SQ*R(1)*R
C(2))
 X(3,5) = MR*(B(2)-CQ*R(2) + SQ*R(3))
 X(3,6) = MR*(-B(1) + R(1))
 X(3,7) = 0
 X(4,1) = -RHO(1) + MR*(CQ*B(3)*R(3) + CQ*R(2)*B(2) + SQ*B(3)*R(2)-
CSQ*R(3)*B(2)-R(2)**2-R(3)**2)
 X(4,2) = MR*(-CQ*B(1)*R(2) + CQ*R(1)*R(2) + SQ*B(1)*R(3)-SQ*R(1)*R
C(3))
 X(4,3) = MR*(-CQ*B(1)*R(3) + CQ*R(1)*R(3)-SQ*B(1)*R(2) + SQ*R(1)*R
C(2))
 X(4,4) = -RHO(1) + MR*(-R(2)**2-R(3)**2)
 X(4,5) = 0
 X(4,6) = MR*(-CQ*R(3)-SQ*R(2))
 X(4,7) = MR*(CQ*R(2)-SQ*R(3))
 X(5,1) = 0
 X(5,2) = MR*(-B(3) + CQ*R(3) + SQ*R(2))
 X(5,3) = MR*(B(2)-CQ*R(2) + SQ*R(3))
 X(5,4) = 0
 X(5,5) = -MB-MR
 X(5,6) = 0
 X(5,7) = 0
 X(6,1) = MR*(B(3)-CQ*R(3)-SQ*R(2))
 X(6,2) = 0
 X(6,3) = MR*(-B(1) + R(1))
 X(6,4) = MR*(-CQ*R(3)-SQ*R(2))
 X(6,5) = 0
 X(6,6) = -MB-MR
 X(6,7) = 0
 X(7,1) = MR*(-B(2) + CQ*R(2)-SQ*R(3))
```

```
      X(7,2) = MR*(B(1)-R(1))
      X(7,3) = 0
      X(7,4) = MR*(CQ*R(2)-SQ*R(3))
      X(7,5) = 0
      X(7,6) = 0
      X(7,7) = -MB-MR

     Y(1) = -T(1) + MR*(B(2)-CQ*R(2) + SQ*R(3))*(CQ*R(3)*U(4)**2 + SQ*R
    C(2)*U(4)**2 + U(1)*(U(6) + CQ*R(3)*U(4) + SQ*R(2)*U(4)-U(1)*(B(3)-
    CCQ*R(3)-SQ*R(2)) + U(3)*(B(1)-R(1))) + U(1)*(CQ*R(3)*U(4) + SQ*R(2
    C)*U(4))-U(2)*(U(5) + U(2)*(B(3)-CQ*R(3)-SQ*R(2))-U(3)*(B(2)-CQ*R(2
    C) + SQ*R(3)))) + MR*(-B(3) + CQ*R(3) + SQ*R(2))*(CQ*R(2)*U(4)**2-S
    CQ*R(3)*U(4)**2-U(1)*(U(7)-CQ*R(2)*U(4) + SQ*R(3)*U(4) + U(1)*(B(2)
    C-CQ*R(2) + SQ*R(3))-U(2)*(B(1)-R(1)))-U(1)*(-CQ*R(2)*U(4) + SQ*R(3
    C)*U(4)) + U(3)*(U(5) + U(2)*(B(3)-CQ*R(3)-SQ*R(2))-U(3)*(B(2)-CQ*R
    C(2) + SQ*R(3))))-U(2)*U(3)*(BETA(2)-BETA(3)) + U(2)*U(3)*(RHO(3)*C
    CQ**2 + RHO(2)*SQ**2)-U(2)*U(3)*(RHO(3)*SQ**2 + CQ**2*RHO(2)) + U(2
    C)**2*(-RHO(3)*CQ*SQ + CQ*RHO(2)*SQ)-U(3)**2*(-RHO(3)*CQ*SQ + CQ*RH
    CO(2)*SQ)

     Y(2) = -T(2) + MR*(B(3)-CQ*R(3)-SQ*R(2))*(U(2)*(U(7)-CQ*R(2)*U(4)
    C+ SQ*R(3)*U(4) + U(1)*(B(2)-CQ*R(2) + SQ*R(3))-U(2)*(B(1)-R(1))) +
    C U(2)*(-CQ*R(2)*U(4) + SQ*R(3)*U(4))-U(3)*(U(6) + CQ*R(3)*U(4) + S
    CQ*R(2)*U(4)-U(1)*(B(3)-CQ*R(3)-SQ*R(2)) + U(3)*(B(1)-R(1)))-U(3)*(
    CCQ*R(3)*U(4) + SQ*R(2)*U(4))) + MR*(-B(1) + R(1))*(CQ*R(3)*U(4)**2
    C + SQ*R(2)*U(4)**2 + U(1)*(U(6) + CQ*R(3)*U(4) + SQ*R(2)*U(4)-U(1)
    C*(B(3)-CQ*R(3)-SQ*R(2)) + U(3)*(B(1)-R(1))) + U(1)*(CQ*R(3)*U(4) +
    C SQ*R(2)*U(4))-U(2)*(U(5) + U(2)*(B(3)-CQ*R(3)-SQ*R(2))-U(3)*(B(2)
    C-CQ*R(2) + SQ*R(3)))) + RHO(1)*U(3)*(U(1) + U(4))-U(1)*U(3)*(BETA(
    C3)-BETA(1))-U(2)*(U(1) + U(4))*(-RHO(3)*CQ*SQ + CQ*RHO(2)*SQ)-U(3)
    C*(U(1) + U(4))*(RHO(3)*CQ**2 + RHO(2)*SQ**2) + (-RHO(3)*CQ*SQ + CQ
    C*RHO(2)*SQ)*(U(1)*U(2)-U(2)*(U(1) + U(4))) + (-U(1)*U(3) + U(3)*(U
    C(1) + U(4)))*(RHO(3)*SQ**2 + CQ**2*RHO(2))

     Y(3) = -T(3) + MR*(B(1)-R(1))*(CQ*R(2)*U(4)**2-SQ*R(3)*U(4)**2-U(1
    C)*(U(7)-CQ*R(2)*U(4) + SQ*R(3)*U(4) + U(1)*(B(2)-CQ*R(2) + SQ*R(3)
    C)-U(2)*(B(1)-R(1)))-U(1)*(-CQ*R(2)*U(4) + SQ*R(3)*U(4)) + U(3)*(U(
    C5) + U(2)*(B(3)-CQ*R(3)-SQ*R(2))-U(3)*(B(2)-CQ*R(2) + SQ*R(3)))) +
    C MR*(-B(2) + CQ*R(2)-SQ*R(3))*(U(2)*(U(7)-CQ*R(2)*U(4) + SQ*R(3)*U
    C(4) + U(1)*(B(2)-CQ*R(2) + SQ*R(3))-U(2)*(B(1)-R(1))) + U(2)*(-CQ*
    CR(2)*U(4) + SQ*R(3)*U(4))-U(3)*(U(6) + CQ*R(3)*U(4) + SQ*R(2)*U(4)
    C-U(1)*(B(3)-CQ*R(3)-SQ*R(2)) + U(3)*(B(1)-R(1)))-U(3)*(CQ*R(3)*U(4
    C) + SQ*R(2)*U(4)))-RHO(1)*U(2)*(U(1) + U(4))-U(1)*U(2)*(-BETA(2) +
    C BETA(1)) + U(2)*(U(1) + U(4))*(RHO(3)*SQ**2 + CQ**2*RHO(2)) + U(3
    C)*(U(1) + U(4))*(-RHO(3)*CQ*SQ + CQ*RHO(2)*SQ) + (-RHO(3)*CQ*SQ +
    CCQ*RHO(2)*SQ)*(-U(1)*U(3) + U(3)*(U(1) + U(4))) + (RHO(3)*CQ**2 +
    CRHO(2)*SQ**2)*(U(1)*U(2)-U(2)*(U(1) + U(4)))

     Y(4) = MR*(-CQ*R(2) + SQ*R(3))*(CQ*R(3)*U(4)**2 + SQ*R(2)*U(4)**2
    C+ U(1)*(U(6) + CQ*R(3)*U(4) + SQ*R(2)*U(4)-U(1)*(B(3)-CQ*R(3)-SQ*R
    C(2)) + U(3)*(B(1)-R(1))) + U(1)*(CQ*R(3)*U(4) + SQ*R(2)*U(4))-U(2)
    C*(U(5) + U(2)*(B(3)-CQ*R(3)-SQ*R(2))-U(3)*(B(2)-CQ*R(2) + SQ*R(3))
    C)) + MR*(CQ*R(2)*U(4)**2-SQ*R(3)*U(4)**2-U(1)*(U(7)-CQ*R(2)*U(4) +
    C SQ*R(3)*U(4) + U(1)*(B(2)-CQ*R(2) + SQ*R(3))-U(2)*(B(1)-R(1)))-U(
    C1)*(-CQ*R(2)*U(4) + SQ*R(3)*U(4)) + U(3)*(U(5) + U(2)*(B(3)-CQ*R(3
    C)-SQ*R(2))-U(3)*(B(2)-CQ*R(2) + SQ*R(3))))*(CQ*R(3) + SQ*R(2)) + U
    C(2)*U(3)*(RHO(3)*CQ**2 + RHO(2)*SQ**2)-U(2)*U(3)*(RHO(3)*SQ**2 + C
    CQ**2*RHO(2)) + U(2)**2*(-RHO(3)*CQ*SQ + CQ*RHO(2)*SQ)-U(3)**2*(-RH
    CO(3)*CQ*SQ + CQ*RHO(2)*SQ)

     Y(5) = -F(1) + MB*(-U(6)*U(3) + U(7)*U(2)) + MR*(U(2)*(U(7)-CQ*R(2
    C)*U(4) + SQ*R(3)*U(4) + U(1)*(B(2)-CQ*R(2) + SQ*R(3))-U(2)*(B(1)-R
    C(1))) + U(2)*(-CQ*R(2)*U(4) + SQ*R(3)*U(4))-U(3)*(U(6) + CQ*R(3)*U
    C(4) + SQ*R(2)*U(4)-U(1)*(B(3)-CQ*R(3)-SQ*R(2)) + U(3)*(B(1)-R(1)))
    C-U(3)*(CQ*R(3)*U(4) + SQ*R(2)*U(4)))


     Y(6) = -F(2) + MB*(U(5)*U(3)-U(7)*U(1)) + MR*(CQ*R(2)*U(4)**2-SQ*R
    C(3)*U(4)**2-U(1)*(U(7)-CQ*R(2)*U(4) + SQ*R(3)*U(4) + U(1)*(B(2)-CQ
    C*R(2) + SQ*R(3))-U(2)*(B(1)-R(1)))-U(1)*(-CQ*R(2)*U(4) + SQ*R(3)*U
    C(4)) + U(3)*(U(5) + U(2)*(B(3)-CQ*R(3)-SQ*R(2))-U(3)*(B(2)-CQ*R(2)
    C + SQ*R(3))))


     Y(7) = -F(3) + MB*(-U(5)*U(2) + U(6)*U(1)) + MR*(CQ*R(3)*U(4)**2 +
    C SQ*R(2)*U(4)**2 + U(1)*(U(6) + CQ*R(3)*U(4) + SQ*R(2)*U(4)-U(1)*(
    CB(3)-CQ*R(3)-SQ*R(2)) + U(3)*(B(1)-R(1))) + U(1)*(CQ*R(3)*U(4) + S
    CQ*R(2)*U(4))-U(2)*(U(5) + U(2)*(B(3)-CQ*R(3)-SQ*R(2))-U(3)*(B(2)-C
    CQ*R(2) + SQ*R(3))))
```

Acknowledgments

I take this opportunity to thank Wallace T. Fowler of The University of Texas at Austin for introducing me to FORMAC. I am also grateful to Chuck Adams of Texas A&M University, whose efforts made possible "The Aggie Connection."

References

[1]Hooker, W.W. and Margulies, G., "The Dynamical Attitude Equations for an *n*-Body Satellite," *The Journal of the Astronautical Sciences,* Vol. XII, Winter 1965, pp. 123-128.

[2]Roberson, R.E. and Wittenburg, J., "A Dynamical Formalism for an Arbitrary Number of Interconnected Rigid Bodies, with Reference to the Problem of Satellite Attitude Control," *Proceedings of the Third Congress of the International Federation of Automatic Control,* Vol. 1, Book, 3, London, June 20-25, 1966, pp. 46D.1-46D.9.

[3]Hooker, W.W., "A Set of *r* Dynamical Attitude Equations for an Arbitary *n*-Body Satellite Having *r* Rotational Degrees of Freedom," *AIAA Journal,* Vol. 8, July 1970, pp. 1205-1207.

[4]Velman, J.R., "Simulation Results for a Dual-Spin Spacecraft," *Proceedings of the Symposium on Attitude Stabilization and Control of Dual-Spin Spacecraft,* El Segundo, Calif., Aug. 1-2, 1967, SAMSO Rept. TR-68-191, Nov. 1967.

[5]Russell, W.J., "On the Formulation of Equations of Rotational Motion for an *N*-Body Spacecraft," Aerospace Corp., El Segundo, Calif., TR-0200 (4133)-2, Feb. 1969.

[6]Frisch, H.P., "A Vector-Dyadic Development of the Equations of Motion for *N*-Coupled Rigid Bodies and Point Masses," NASA TN D-7767, Oct. 1974.

[7]Fleischer, G.E., "Multi-Rigid-Body Attitude Dynamics Simulation," TR 32-1516, Jet Propulsion Lab., Pasedena, Calif., Feb. 15, 1971.

[8]Fleischer, G.E. and Likins, P.W., "Attitude Dynamics Simulation Subroutines for Systems of Hinge-Connected Rigid Bodies," Jet Propulsion Lab., Pasadena, Calif., TR 32-1592, May 1, 1974.

[9]Huston, R.L. and Passerello, C.E., "On the Dynamics of Chain Systems," American Society of Mechanical Engineers, Paper 74-WA/Aut-11, 1974.

[10]Kane, T.R., *Dynamics,* Holt, Rinehart and Winston, Inc., New York, 1968, pp. 177-238.

[11]Engeli, M.E., "User's Manual for the Formula Manipulation Language SYMBAL," Univ. of Texas at Austin Computation Center, July 1969.

[12]Jefferys, W.H., "TRIGMAN—A System for Algebraic Manipulation of Poisson Series," Univ. of Texas at Austin Computation Center, May 1973.

[13]Tobey, R., et al., "PL/I—FORMAC Symbolic Mathematics Interpreter," IBM Corp., Program 360D-03.3.004, Jan. 1971.

[14]Bahr, K., "SHARE FORMAC/FORMAC73," SHARE Program Library Agency, Triangle Univ. Computation Center, Research Triangle Park, N.C., Program 360D-03.3.013, July 1975.

[15]Kane, T.R. and Likins, P.W., "Kinematics of Rigid Bodies in Spaceflight," Dept. of Applied Mechanics, Stanford Univ., TR 204, May 1971, pp. 27, 119-120.