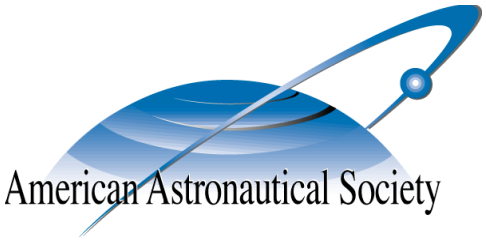

An Instantaneous Eigenstructure Quasivelocity Formulation for Nonlinear Multibody Dynamics

John L. Junkins and Hanspeter Schaub

Simulated Reprint from

Journal of the Astronautical Sciences
Vol. 45, No. 3, July–Sept. 1997, Pages 279–295

A publication of the
American Astronautical Society
AAS Publications Office
P.O. Box 28130
San Diego, CA 92198

AAS 97-119

AN INSTANTANEOUS EIGENSTRUCTURE QUASI-COORDINATE FORMULATION FOR NONLINEAR MULTIBODY DYNAMICS

John L. Junkins* **and Hanspeter Schaub†**

A novel method is presented to solve the equations of motion for a large class of constrained and unconstrained dynamical systems. Given an analytic expression for the system mass matrix, quasi-coordinate equations of motion are derived in a manner that generates equations analogous to the dynamics/kinematics partitioning in Eulerian rigid body dynamics. This separation is accomplished by introducing a new quasi velocity coordinate η which yields a dynamical system with an identity mass matrix. The problem of inverting a complex mass matrix is replaced by the problem of solving two first order differential equations for the mass matrix eigenfactors. Dynamic constraint equations are incorporated directly into the new η differential equation, forgoing any need to solve the algebraic constraint equations simultaneously with the differential equations of motion.

INTRODUCTION

The equations of motion of complex dynamical systems are usually second order nonlinear differential equations which require taking the inverse of a time-varying, configuration variable dependent mass matrix in some manner. Such dynamical systems could be a large nonlinear deformation model for an arbitrary body, a multi-body system or a multi-link robot arm. One reason why the resulting dynamics are complicated is that they are usually written in a way that combines coordinates natural to the momentum or energy description with those natural to the displacement description. The result is a split between momentum differential equations and kinematic differential equations. This natural splitting is typically destroyed when the generalized methods of mechanics are employed and result in a more complicated mass matrix. This occurs when the classical Lagrange equations of motion are written in terms of a generalized coordinate and their time derivatives. By using Newton-Eulerian mechanics or the Boltzmann-Hamel version of Lagrange's equations, it is possible to introduce quasi-coordinates which separate the decision of choosing displacement coordinates and velocity (momentum) coordinates. As is well-known, (e.g. Eulerian rigid body dynamics), this process often leads to much more attractive equations than those that result from "brute force" application of Lagrange's equations. It is possible to bring

*George Eppright Chair Professor of Aerospace Engineering, Aerospace Engineering Department, Texas A&M University, College Station TX 77843, Fellow AAS.

†Graduate Research Assistant, Aerospace Engineering Department, Texas A&M University, College Station TX 77843.

the equations of motion to their most convenient form with a constant mass matrix.[1,2] For general configuration-variable mass matrices there has not been a generally applicable method to accomplish an analogous transformation.

Several methods have been proposed to carry out the mass matrix inverse[2,3] ranging from taking an algebraic inverse, to using traditional numerical inverse methods (such as a Cholesky decomposition) to the elegant method of using the innovations factorization.[2] Naturally each method has its advantages and disadvantages. The algebraic inverse in only feasible for relatively small systems, even with symbol manipulation programs such as Mathematica and Maple. Taking a numerical inverse at each integration step is computationally costly and difficult. The method proposed by Ref. 2 uses the innovations factorizations technique to parameterize the mass matrix and recursively approximate its inverse. The mass matrix factors involved are obtained from a recursive filter. However, this recursive filter is conveniently applicable only to a linked body chain and other kinematically recursive topologies.

This paper presents a method to solve a very general class of constrained and unconstrained dynamical systems and avoids the necessity of inverting a configuration variable mass matrix to obtain instantaneous accelerations. The equations of motion will be separated into dynamical and kinematic differential equations somewhat analogous to classical developments in rigid body dynamics. A method outlined in Ref. 4 will be used to replace the mass matrix inverse problem with one of solving the corresponding eigenfactor differential equations. The new formulation will allow any Pfaffian constraints to be easily incorporated into the equations of motion, thus avoiding having coupled algebraic constraint equations to be solved simultaneously with the original equations of motion and reducing the overall order of the system.

PROBLEM FORMULATION

The equations of motion for a dynamical system can be derived by first formulating the kinetic energy T and the potential energy V. Let the system Lagrangian $\mathcal{L}$ be defined as

$$\mathcal{L} = T - V \tag{1}$$

Let x be the generalized configuration coordinate vector for the system, then the potential energy is given by

$$V = V(x) \tag{2}$$

The kinetic energy can be written in terms of the state vector derivative $\dot{x}$ or in terms of a quasi-velocity vector y defined as

$$y = P(x)\dot{x} \tag{3}$$

A field where quasi-velocities are often preferred over configuration coordinate derivatives is in rigid body attitude dynamics. For example, it is much simpler to write the system kinetic energy in terms of the body angular velocity ω then in terms of the Euler attitude angle derivatives $\dot{\theta}$. Let $\bar{M}(x,t)$ be the mass matrix for a system described with y, then the total kinetic energy for the system is given by

$$T = T_2 + T_1 + T_0 = \frac{1}{2} y^T \bar{M}(x,t) y + \bar{G}^T(x,t) y + T_0(x,t) \tag{4}$$

where the T_1 and T_0 terms only appear in unnatural systems. However, to find the traditional version of Lagrange's equations of motion the kinetic energy needs to be written in terms of generalized coordinate derivative $\dot{x}$, not quasi-velocities y. Using Eq. (3), the kinetic energy can thus be rewritten in terms of $\dot{x}$.

$$T_2 = \frac{1}{2}\dot{x}^T P(x)^T \bar{M}(x,t) P(x)\dot{x} = \frac{1}{2}\dot{x}^T M(x,t)\dot{x} \tag{5}$$

$$T_1 = \bar{G}^T(x,t)P(x)\dot{x} = G^T(x,t)\dot{x} \tag{6}$$

where $M(x,t) = P(x)^T \bar{M}(x,t)P(x)$ is the system mass matrix for the state vector $(x,\dot{x})$ and $G(x,t) = P^T(x)\bar{G}(x,t)$. For mechanical systems $M(x,t)$ will always by symmetric positive definite. Let Q be a non-conservative forcing term and let $A^T\lambda$ be the constraint force, then the Lagrange equations of motion are defined as

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}}\right) - \frac{\partial \mathcal{L}}{\partial x} = Q - A^T\lambda \tag{7}$$

with the Pfaffian non-holonomic constraint being

$$A(x)\dot{x} + b(t) = 0 \tag{8}$$

The partial derivatives of the system Lagrangian $\mathcal{L}$ are

$$\frac{\partial \mathcal{L}}{\partial \dot{x}} = M(x,t)\dot{x} + G(x,t) \tag{9}$$

and

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{1}{2}\dot{x}^T \frac{\partial M(x,t)}{\partial x}\dot{x} + \frac{\partial G^T(x,t)}{\partial x}\dot{x} + \frac{\partial T_0(x,t)}{\partial x} - \frac{\partial V}{\partial x} \tag{10}$$

where the term $\dot{x}^T \frac{\partial M(x,t)}{\partial x}\dot{x}$ is a column vector. The resulting standard Lagrange equations of motion are

$$M(x,t)\ddot{x} + \left(\dot{M} - \frac{1}{2}\dot{x}^T \frac{\partial M(x,t)}{\partial x} - \frac{\partial G^T(x,t)}{\partial x}\right)\dot{x} + \dot{G}(x,t) - \frac{\partial T_0(x,t)}{\partial x} + \frac{\partial V}{\partial x} = Q - A^T\lambda \tag{11}$$

or more compactly

$$M(x,t)\ddot{x} + H(x,\dot{x},t) + \frac{\partial V}{\partial x} = Q - A^T\lambda \tag{12}$$

The above equations of motion are a second order nonlinear differential equation, obviously generally not a simple task to solve. In particular, the time and state dependence of the mass matrix poses a particular difficulty. These standard equations of motion, when coupled to the constraint equations in Eq. (8), pose a significant challenge for high dimensioned systems. The necessity of solving systems of order $n+m$ to obtain $(\ddot{x},\lambda)$ for each $(x,\dot{x},t)$ lies at the heart of the difficulty.

THE BOLTZMANN-HAMEL EQUATIONS OF MOTION

We motivate this development using rigid body dynamics wherein it is common practice to separate the momentum dynamics and kinematics. Euler's equation of motion are usually

written in terms of the body angular velocity ω, not in terms of the time derivative of the attitude coordinate vector θ.

$$\Im\dot{\omega} = -[\tilde{\omega}]\Im\omega + u \tag{13a}$$

$$\dot{\theta} = f(\theta)\omega \tag{13b}$$

Eq. (13a) describes the system momentum time rate of change and Eq. (13b) describes the kinematic relationship between the body angular velocity and the attitude coordinate derivative. Using only θ and its inertial derivatives would yield a much more complex second order differential equation.

This separation of dynamics and kinematics in the equations of motion cannot be accomplished in more general dynamical systems. However, we show a way to accomplish an analogous structure in the system equations, at the expense of increasing the number of differential equations to be solved. This involves projecting the configuration coordinate derivative into a moving reference frame[1,2] by introducing a quasi-velocity vector which diagonalizes the mass matrix. Since the mass matrix M is always symmetric and positive definite, it can be spectrally decomposed using the orthogonal real eigenvector matrix E and the diagonal positive real eigenvalue matrix D. Instead of using E directly, using $C = E^T$ instead will simplify the following development.

$$M = C^T D C \qquad CC^T = I \qquad D = diag(\lambda_i) \tag{14}$$

Let the diagonal S matrix be defined as the positive square root of the eigenvalue matrix D.

$$S = \sqrt{D} = diag\left(+\sqrt{\lambda_i}\right) \qquad D = S^T S \tag{15}$$

Substituting Eqs. (14) and (15) into Eq. (5) yields the following kinetic energy expression.

$$T_2 = \frac{1}{2}\dot{x}^T C^T S^T S C \dot{x} \tag{16}$$

By introducing the quasi-velocity coordinate vector η

$$\eta = SC\dot{x} \qquad \eta = \eta(\lambda_i(x), c_i(x), \dot{x}) \tag{17}$$

we obtain a new simplified expression for the kinetic energy. The mass matrix associated with η is the identity matrix.

$$T^* = T_2^* + T_1^* + T_0^* = \frac{1}{2}\eta^T\eta + G^T(x,t)C^T S^{-1}\eta + T_0(x,t) \tag{18}$$

Note that T_2^* depends explicitly on η. However, if we choose $(x, \dot{x})$ as the independent set for taking partial derivatives, we must recall that η depends on $(x, \dot{x})$. The x dependence is implicit in Eq. (14), (15), (17) because *S(x), C(x)* parameterize $M(x) = C^T S^T SC$. Also note that T^* is equal to T (both represent the same physical kinetic energy quantity), they differ only in their algebraic formulations.

The new quasi-velocity coordinate η_i can also be expressed as

$$\eta_i = s_i c_i^T \dot{x} \tag{19}$$

Or in other words, η_i is the projection of the velocity vector $\dot{x}$ onto the i-th eigenvector c_i and scaled by the i-t eigenvalue square root s_i. The new velocity coordinate not only

contains information about the standard velocity coordinate, but it also linked to the system mass matrix itself. The inverse mapping of Eq. (17) describes the kinematic relationship between $\dot{x}$ and η similarly to the relationship of $\dot{\theta}$ and ω in Eq. (13b). Since C is orthogonal and the diagonal entries of S are positive the inverse mapping is trivial and singularity free.

$$\dot{x} = C^T S^{-1} \eta \tag{20}$$

The partial derivatives of the system Lagrangian $\mathcal{L}$ are now rewritten in terms of the new generalized velocity vector η using the chain rule as[1]

$$\frac{\partial \mathcal{L}}{\partial \dot{x}} = \frac{\partial T^*}{\partial \dot{x}} + \frac{\partial \eta}{\partial \dot{x}}^T \frac{\partial T^*}{\partial \eta} = C^T S \frac{\partial T^*}{\partial \eta} \tag{21}$$

and

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial T^*}{\partial x} + J^T \frac{\partial T^*}{\partial \eta} - \frac{\partial V}{\partial x} \tag{22}$$

where J is the sensitivity matrix of η with respect to the state vector x. This matrix is non-zero since the C and S both indirectly depend on x.

$$J \equiv \frac{\partial \eta}{\partial x} = \left[\frac{\partial \eta}{\partial x_1}, \cdots, \frac{\partial \eta}{\partial x_n} \right] \tag{23}$$

Using the chain rule $\partial \eta / \partial x_k$ is expressed as

$$\frac{\partial \eta}{\partial x_k} = \left(\frac{\partial S}{\partial x_k} + S \frac{\partial C}{\partial x_k} C^T \right) S^{-1} \eta \tag{24}$$

However, finding $\partial S / \partial x_k$ and $\partial C / \partial x_k$ is difficult to do without resolving the eigenvector, eigenvalue problem. One method to find these partial derivatives is to use the same square root algorithm used in this paper to find $\dot{S}$ and $\dot{C}$. This method is covered in the companion technical report in Ref. 5. However, this method only allowed the partial eigenfactor derivatives to be evaluated at the discrete integration time steps, not in between them. Because of this, methods such as the Runge-Kutta methods which require intermediate steps could not be used to forward integrate the equations of motions. Thus Ref. 5 uses a predictor-corrector type integration method instead.

To avoid solving for the computationally expensive $\partial S / \partial x_k$ and $\partial C / \partial x_k$ terms, $\partial \mathcal{L} / \partial x$ can be solved using $T(x, \dot{x})$ instead of $T^*(x, \eta)$ as was done in Eq. (10). The $\dot{x}$ velocities in Eq. (10) could be replaced by the new velocity vector η through Eq. (20), but it is computationally more efficient to keep the $\dot{x}$ terms.

The partial derivative of T^* with respect to η is

$$\frac{\partial T^*}{\partial \eta} = \eta + S^{-1} C G(x, t) \tag{25}$$

Using Eqs. (10), (21) and (25), the Lagrange equations of motion in Eq. (7) become

$$\frac{d}{dt} \left(C^T S \eta + G \right) - \frac{1}{2} \dot{x}^T \frac{\partial M}{\partial x} \dot{x} - \frac{\partial G^T}{\partial x} \dot{x} - \frac{\partial T_0}{\partial x} + \frac{\partial V}{\partial x} = Q - A^T \lambda \tag{26}$$

After carrying out the time derivative and using the orthogonality of the C matrix, the following first order differential equation is obtained.

$$\dot{\eta} + S^{-1}\left(C\dot{C}^T S + \dot{S}\right)\eta - S^{-1}C\left(\frac{1}{2}\dot{x}^T\frac{\partial M}{\partial x}\dot{x} - \frac{\partial G^T}{\partial x}\dot{x}\right) = S^{-1}CF - B^T\lambda \tag{27}$$

where

$$B = AC^T S^{-1} \tag{28}$$

and

$$F = Q - \frac{\partial V}{\partial x} - \dot{G} + \frac{\partial T_0}{\partial x} \tag{29}$$

The two first order equations (20) and (27) replace the classical second order equations of motion in Eq. (12). Eq. (27) is an interesting new form of the well-known the Boltzmann-Hamel equation[1,6] for our choice of quasi-coordinates η. This diagonalized equation of motion is very similar to the one introduced in Ref. 2, except that our new velocity vector η is not equal to their υ since our parameterization of the mass matrix is different. Note that Eq. (27) requires no matrix inverse to be taken thanks to the orthogonality of the C matrix. Inverting the S matrix is trivial since it is a positive diagonal matrix. At this stage the complex problem of finding the instantaneous matrix inverse has been traded for another problem of solving the eigenfactor differential equations.

MASS MATRIX EIGENFACTOR DERIVATIVES

To solve the above Boltzmann-Hamel equation auxiliary differential equations are required to yield the eigenfactor derivatives with respect to time. A square root algorithm developed by Oshman and Bar-Itzhack to solve the matrix Riccati differential equation[7] was enhanced and extended in Ref. 4 to robustly handle the mass matrix eigenfactor derivatives. The method allows the calculation of the instantaneous C and S matrices and can handle both repeated eigenvalues and eigenvalue derivatives. Since C is an orthogonal matrix it satisfies a differential equation of the form[4,8–10]

$$\dot{C} = -\Omega C \tag{30}$$

where Ω is a skew-symmetric matrix. Analogously to the attitude dynamics problem where the Ω matrix represents body angular velocities, for the eigenvector dynamics each Ω_{ij} terms represents an eigenvector axis angular velocity. All eigenfactor derivatives of M are expressed by a projection onto C in terms of μ_{ij} as[4,7,11,12]

$$\mu_{ij} = c_j^T \dot{M}(x,t) c_i \tag{31}$$

The method in Ref. 4 defines Ω_{ij} matrix elements as[7,11]

$$[\Omega_{ij}] = \begin{cases} \frac{\mu_{ij}}{s_j^2 - s_i^2} & \text{for } |s_j^2 - s_i^2| \geq \epsilon \\ \Omega_{ij}(t_0) + \dot{\Omega}_{ij}(t_0)(t_1 - t_0) & \text{for } |s_j^2 - s_i^2| < \epsilon \end{cases} \tag{32}$$

where ϵ is the smallest allowable numerical difference in eigenvalues before numerical problems occur in calculating Ω_{ij}. When λ_i differs from λ_j by less than ϵ then the unknown Ω_{ij} term at t_1 is linearly interpolated from known terms at t_0. This approximation is shown

to have minimal impact on the numerical accuracy of the solution. To enhance longterm stability of the eigenfactor integration and handle cases where the eigenvectors are discontinuous Ref. 4 performs a Jacobi sweep[4, 13, 14] whenever the Ω_{ij} term is being approximated. Discontinuous eigenvectors of a symmetric positive definite matrix are mathematically only possible for the rare case where both the corresponding eigenvalues and their derivatives are repeated. As a side note, no mechanical systems studied so far have exhibited crossing eigenvalues with a corresponding non-zero μ_{ij} term. Thus evaluating Ω_{ij} has never posed any numerical problems. Should a system be found where numerical problems occur, the method in Ref.[4] was designed to handle even the worst of cases.

The time derivative of the eigenvalues λ_i are also defined in terms of μ matrix entries.[4, 7, 11, 12]

$$\dot{\lambda}_i = \mu_{ii} \tag{33}$$

However, the time derivative of the eigenvalues is not used directly, but the derivative of the eigenvalue square root. Let s_i be the i-th entry of the S matrix. Using the chain rule, the derivative of s_i is

$$\dot{s}_i = \frac{1}{2s_i}\dot{\lambda}_i \tag{34}$$

This is written in a more compact form using the diagonal matrix $\Gamma = diag(\mu_{ii})$ as[7]

$$\dot{S} = \frac{1}{2}\Gamma S^{-1} \tag{35}$$

Substituting Eq. (30) into Eq. (27), the Boltzmann-Hamel equations are now reduced to

$$\dot{\eta} + S^{-1}\left(\Omega S + \dot{S}\right)\eta - S^{-1}C\left(\frac{1}{2}\dot{x}^T\frac{\partial M}{\partial x}\dot{x} - \frac{\partial G^T}{\partial x}\dot{x}\right) = S^{-1}CF - B^T\lambda \tag{36}$$

At first glance, Eq. (36) may seem more complicated than than the original equations of motion. Keep in mind, however, that S and Γ are diagonal matrices which greatly reduces the computational burden. The most costly terms to compute when evaluating the eigenfactor derivatives are the μ_{ij} terms in Eq. (31). Finding the instantaneous eigenfactors would be a very attractive task for massively parallel computer system since the μ_{ij} calculation is trivially parallelizable.

PFAFFIAN NON-HOLONOMIC CONSTRAINTS

If the dynamical system is unconstrained, then the Pfaffian constraint matrix B will be zero and Eq. (36) is fully defined. However, if the dynamics are constrained through the Pfaffian constraint surface given in Eq. (8), then Eq. (36) will need to be solved simultaneously with the constraint equation. Using Eq. (20) we rewrite the Pfaffian constraint in terms of the new velocity vector η.

$$AC^T S^{-1}\eta + b = 0 \tag{37}$$

which can be simplified using Eq. (28) to

$$B\eta + b = 0 \tag{38}$$

The dynamic constraint equations is obtained by taking the first time derivative of Eq. (38).

$$B\dot{\eta} + \dot{B}\eta + \dot{b} = 0 \tag{39}$$

Using Eqs. (35), (30) and (28) $\dot{B}$ can be expressed as

$$\dot{B} = (\dot{A}C^T + A\dot{C}^T - B\dot{S})S^{-1} \tag{40}$$

To determine the vector $(\dot{\eta}, \lambda)$ Eq. (36) will need to be solved simultaneously with Eq. (39). This leads to the differential-algebraic equations (DAE)

$$\begin{bmatrix} I & B^T \\ B & 0 \end{bmatrix} \begin{pmatrix} \dot{\eta} \\ \lambda \end{pmatrix} = \begin{pmatrix} -S^{-1}\left(\Omega S + \dot{S}\right)\eta + S^{-1}C\left(\frac{1}{2}\dot{x}^T\frac{\partial M}{\partial x}\dot{x} - \frac{\partial G^T}{\partial x}\dot{x} + F\right) \\ -\dot{B}\eta - \dot{b} \end{pmatrix} \tag{41}$$

which can be written in more compact form as

$$M_2 \begin{pmatrix} \dot{\eta} \\ \lambda \end{pmatrix} = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \tag{42}$$

Since B is a mxn matrix, M_2 is a symmetric $(n+m)$x$(n+m)$ matrix. A partitioned matrix inversion formula[11] is used to find the inverse of M_2. Because of the use of the quasi-coordinates η, the upper left partition of M_2 is a nxn identity matrix which simplifies the partitioned inverse immensely. For this case the mxm Schur complement Δ reduces to[11]

$$\Delta = BB^T \tag{43}$$

making the partitioned inverse of M_2

$$M_2^{-1} = \begin{bmatrix} I - B^T\Delta^{-1}B & B^T\Delta^{-1} \\ \Delta^{-1}B & -\Delta^{-1} \end{bmatrix} \tag{44}$$

Using M_2^{-1} in Eq. (44) the constrained differential equation of motion for η is

$$\dot{\eta} = (I - B^T\Delta^{-1}B)a_1 + B^T\Delta^{-1}a_2 \tag{45}$$

The Lagrange constraint vector λ is

$$\lambda = \Delta^{-1}Ba_1 - \Delta^{-1}a_2 \tag{46}$$

Note that if no constraints are imposed on the dynamical system then Eq. (45) collapses back to Eq. (36). If the number of system constraints m is small, then the mxm matrix Δ^{-1} could be inverted for each time step. However, as m grows larger taking a numerical inverse quickly grows in computational complexity.

Since Δ, for linearly independent constraints, is a positive definite symmetric matrix by Eq. (43), it can be decomposed using the eigenfactor parameterization analogous to the mass matrix parameterization. Let C_Δ be the transpose of the eigenvector matrix of Δ, and let S_Δ be a diagonal matrix whose entries are the positive roots of Δ eigenvalues. Then through a spectral decomposition Δ can be written as

$$\Delta = C_\Delta^T S_\Delta^T S_\Delta C_\Delta \tag{47}$$

Since C_Δ is an orthogonal matrix and the diagonal entries of S_Δ are all positive, the inverse of Δ is

$$\Delta^{-1} = C_\Delta^T S_\Delta^{-2} C_\Delta \tag{48}$$

This direct inverse formulation reduces Eq. (45) to the following matrix inverse free formulation.

$$\dot{\eta} = (I - B^T C_\Delta^T S_\Delta^{-2} C_\Delta B) a_1 + B^T C_\Delta^T S_\Delta^{-2} C_\Delta a_2 \tag{49}$$

Keep in mind that S_Δ is a diagonal matrix with positive entries. Therefore finding its inverse involves only scalar inversions.

To update the C_Δ and S_Δ matrices without resolving the eigenvalue, eigenvector problem, their time derivatives are found using the square root eigenfactor algorithm[4] analogously to finding the time derivatives of C and S of the mass matrix M. Let c_{Δ_i} be the i-th eigenvector of C_Δ, then β_{ij} is defined as

$$\beta_{ij} = c_{\Delta_j} \dot{\Delta} c_{\Delta_i}^T \tag{50}$$

where the time derivative of Δ is

$$\dot{\Delta} = \dot{B} B^T + B \dot{B}^T \tag{51}$$

and $\dot{B}$ was defined in Eq. (40). The diagonal matrix Γ_Δ and the skew-symmetric matrix Ω_Δ are then defined as

$$\Gamma_\Delta = diag(\beta_{ii}) \tag{52}$$

$$[\Omega_{\Delta_{ij}}] = \begin{cases} \frac{\beta_{ij}}{s_{\Delta_j}^2 - s_{\Delta_i}^2} & \text{for } |s_{\Delta_j}^2 - s_{\Delta_i}^2| \geq \epsilon \\ \Omega_{\Delta_{ij}}(t_0) \ + \dot{\Omega}_{\Delta_{ij}}(t_0)(t_1 - t_0) & \text{for } |s_{\Delta_j}^2 - s_{\Delta_i}^2| < \epsilon \end{cases} \tag{53}$$

The time derivatives of C_Δ and S_Δ are written as[7]

$$\dot{S}_\Delta = \frac{1}{2} \Gamma_\Delta S_\Delta^{-1} \tag{54}$$

$$\dot{C}_\Delta = -\Omega_\Delta C_\Delta \tag{55}$$

MULTI-LINK SIMULATION

To demonstrate the eigenfactor square root algorithm, a constrained multi-link motion is simulated. The system layout is shown in Fig. 1. The mass m_1 is attached to the link l_1 through the spring $K_1 = 1$ which simulates stretching, but no bending of the link l_1. The mass m_2 is attached to the point $(0, 4)$ through the spring K_2 with stiffness 1. The rod l_3 is attached to l_1 through the torsional spring K_3 with stiffness 0.5 and a neutral state of $\pi/2$ radians. The links are assumed to be mass-less and have lengths of $l_1 = 0.5$, $l_2 = 1/\sqrt{2}$ and $l_3 = 0.5$. All rod tips have unit masses attached to them. A constraint is imposed on the system which restricts the hand of l_2 at the coordinates (x, y) to only move horizontally. Since only conservative forces or torques are acting on the system, the total system energy will remain constant.

The hand coordinates (x,y) are given as

$$x = (l_1 + \theta_3) \cos\theta_1 + l_2 \cos\theta_2 \tag{56}$$

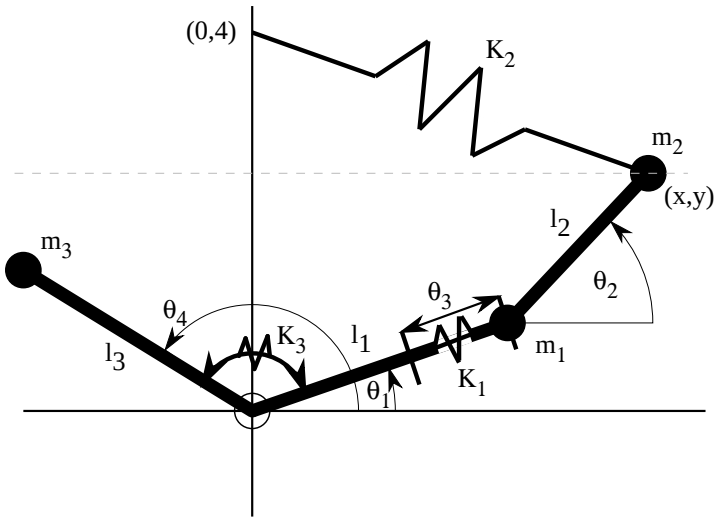

Figure 1 Constrained Dual-Link Manipulator Layout

$$y = (l_1 + \theta_3)\sin\theta_1 + l_2 \sin\theta_2 \tag{57}$$

The system potential energy is the total spring energy given by

$$V(\theta) = \frac{1}{2}K_1\theta_3^2 + \frac{1}{2}K_2\left(x^2 + (4-y)^2\right) + \frac{1}{2}K_3(\theta_4 - \theta_1 - \frac{\pi}{2})^2 \tag{58}$$

The system kinetic energy is given as

$$\begin{gathered} T = \tfrac{1}{2}m_1\left((l_1+\theta_3)^2\dot{\theta}_1^2 + \dot{\theta}_3^2\right) + \tfrac{1}{2}m_3 l_3^2\dot{\theta}_4^2 \\ +\tfrac{1}{2}m_2\left((l_1+\theta_3)^2\dot{\theta}_1^2 + l_2^2\dot{\theta}_2^2 + \dot{\theta}_3^2 + 2(l_1+\theta_3)l_2\cos(\theta_1-\theta_2)\dot{\theta}_1\dot{\theta}_2 + 2l_2\sin(\theta_1-\theta_2)\dot{\theta}_2\dot{\theta}_3\right) \end{gathered} \tag{59}$$

From the kinetic energy T the system mass matrix can be extracted.

$$M(\theta) = \begin{bmatrix} (m_1+m_2)(l_1+\theta_3)^2 & m_2(l_1+\theta_3)l_2\cos(\theta_1-\theta_2) & 0 & 0 \\ m_2(l_1+\theta_3)l_2\cos(\theta_1-\theta_2) & m_2 l_2^2 & m_2 l_2\sin(\theta_1-\theta_2) & 0 \\ 0 & m_2 l_2\sin(\theta_1-\theta_2) & m_1+m_2 & 0 \\ 0 & 0 & 0 & m_3 l_3^2 \end{bmatrix} \tag{60}$$

The eigenfactor square root algorithm requires an algebraic expression for $\dot{M}$ and $\partial M/\partial\theta_k$. They are found directly from the system mass matrix M in Eq. (60). The system constraint requires that $\dot{y} = 0$. Using Eq. (57) this can be expressed as $A(\theta)\dot{\theta} = 0$ where

$$A(\theta) = \begin{bmatrix} (l_1+\theta_3)\cos\theta_1 & l_2\cos\theta_2 & \sin\theta_1 & 0 \end{bmatrix} \tag{61}$$

Note that the Lagrange equations of motion did not have to get solved specifically for this conservative multi-link system. After finding the kinetic energy of the system and the corresponding mass matrix formulation, the necessary time and spatial derivatives of the mass matrix are found. The simulation is started with all links at rest and let run for 10 seconds. The initial configuration is $\theta_1 = 30°$, $\theta_2 = 60°$, $\theta_3 = 0$, $\theta_4 = 135°$. The resulting

motion up to $t = 3.5$ seconds is shown in Figure 2. All integrations were performed with a 3rd/4th order variable step size Runge Kutta method. Note that the hand of link l_2 moves only in the horizontal direction as required by the system constraint. The link l_1 is first compressed until θ_2 is greater than 90°, than l_1 is stretched upwards. Since $|\theta_4 - \theta_1| > 90°$ initially, the mass m_3 is first pulled toward m_1, but is then repelled as m_1 begins to move upward.

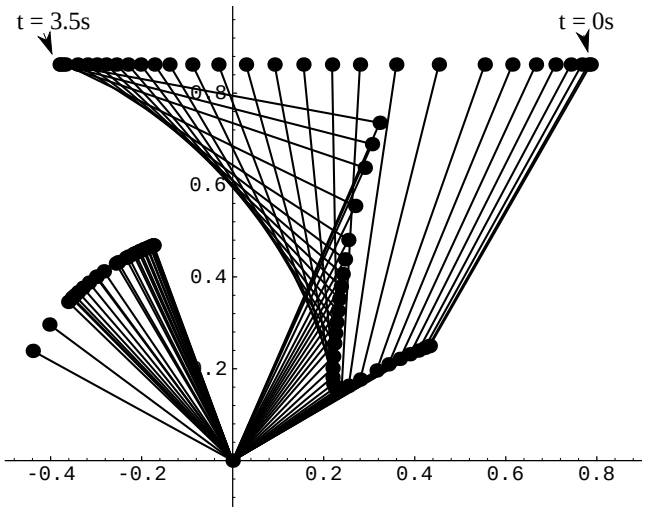

Figure 2 Unconstrained Dual-Link Manipulator Motion

The system state time histories found by using the new quasi-velocities η and the auxiliary differential equations $\dot{C}$ and $\dot{S}$ are shown in Fig. 3. Note that the stretching θ_3 of link l_1 has been scaled in this figure by a factor of 100 to make it visible on the same scale as the system angles. These trajectories were verified by integrating the classical Lagrange equations of motion with the same variable step size integration technique. An exact, algebraic expression was used to perform the time varying mass matrix inverse.

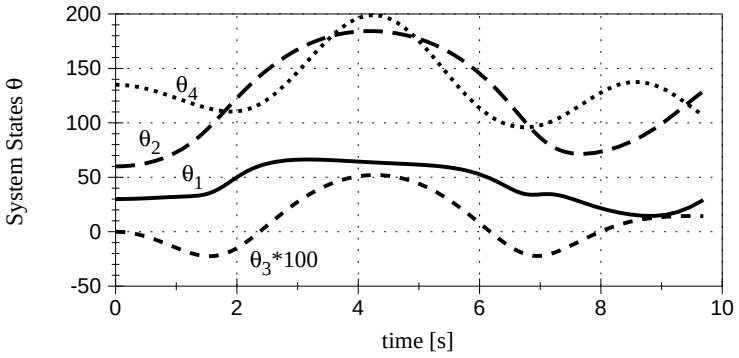

Figure 3 System State Time Histories.

The time histories of the new quasi-velocities η are shown in Fig. 4. They are well behaved and smooth throughout the maneuver. Keep in mind that these new velocity coordinates are influenced by the system mass matrix through Eq. (19). However, since η only depends

on the square root of the mass matrix, namely the C and S matrices, any rapid changes in the mass matrix would exhibit itself in the η time history in a smoother manner.

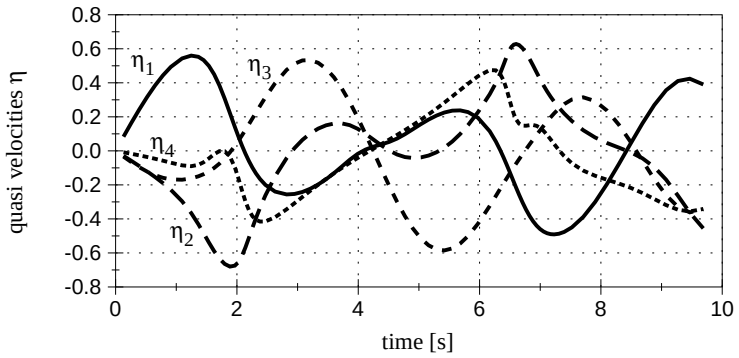

Figure 4 Quasi-Velocity Time Histories.

The mass matrix eigenvalues for this maneuver are shown in Fig. 5. While some eigenvalues do periodically become close, no eigenvalues actually cross during this maneuver. The proximity of two eigenvalues is never closer than 0.03, thus not enough to cause numerical problems evaluating Ω_{ij} in Eq. (32). To only mechanical systems found to have crossing eigenvalues also always had the corresponding μ_{ij} equal to zero. Having $\mu_{ij} = 0$ throughout the maneuver means geometrically that the c_i and c_j eigenvectors always have zero relative angular velocity. In this case numerical problems typically only occur when λ_i was exactly equal to λ_j. How to handle this case was shown in Ref.[4]

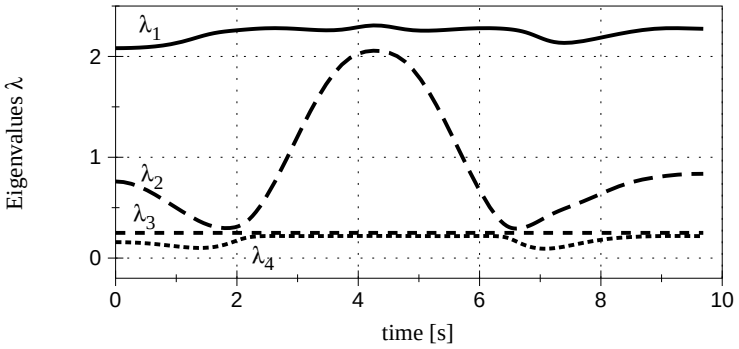

Figure 5 Eigenvalue Time Histories.

In simulations we found that whenever two eigenvalues approach eachother, the corresponding eigenaxis angular velocity Ω_{ij} becomes more active. This is easily seen in Fig. 6 as it occurs around $t = 2$ sec, 4.25 sec and 6.5 sec. At time t = 2 sec the second and fourth eigenvalue become close and Fig. 6 clearly shows a local rise in Ω_{24}. What is happening here geometrically is that the corresponding eigenvectors are being rotated in the plane spanned by them such that they roughly trade positions. The closer the eigenvalues approach eachother, the sharper this rotation would be. These isolated sharp changes in the square root of the eigenvalues and in the eigenvectors is the reason why it is very beneficial to use a variable step size integration technique with this instantaneous eigenstructure quasi-coordinate formulation. Note however that eigenvalues approaching λ_3 have no effect.

This is because the μ_{3i} terms, and therefore the Ω_{3i} terms, are always zero for this system.

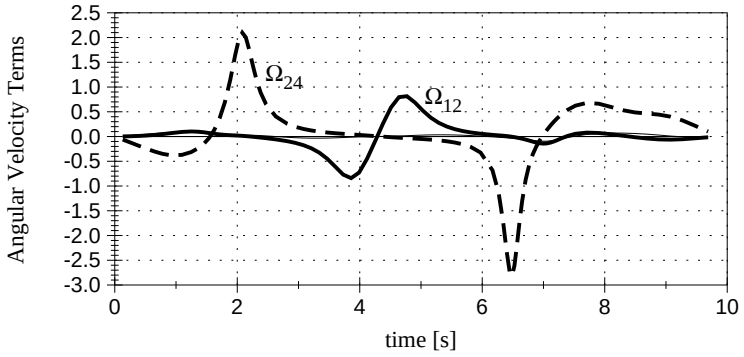

Figure 6 Eigenaxis Angular Velocity Time Histories.

The system constraint violations $\Delta y(t) = |y(t) - y(t_0)|$ for the instantaneous eigenstructure quasi-coordinate formulation and for a "brute force" method are shown in Fig. 7. The brute force method integrates the classical Lagrange equations of motion and uses an exact, algebraic inverse of the system mass matrix. The average integration step size for these simulations was $\bar{h} = 0.024$ seconds. Instead of increasing the order of the system to $n+m$ by adding m constraints to an n-th order system, the instantaneous eigenfactor quasi-coordinate formulation is able to maintain the system at the order n. This is due to the mass matrix corresponding to the η velocity variables being the identity matrix. This allows for a simpler partitioned matrix inverse which incorporates the constraint equations directly into the equations of motion. This simpler calculation keeps the constraint violation for the new method slightly lower than for the brute force method.

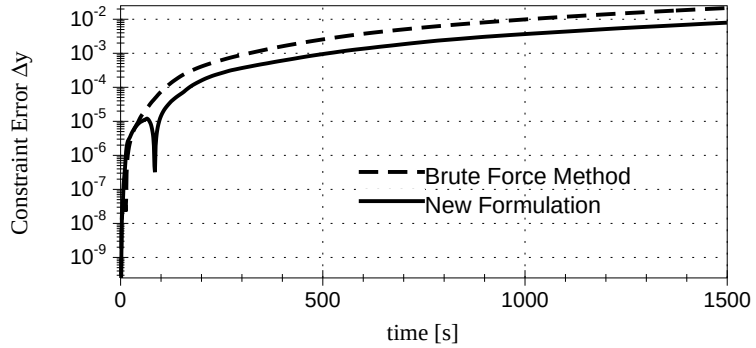

Figure 7 System Constraint Violation.

To measure the integration error over the entire maneuver, the following norm was defined

$$|\Delta P| = \int_0^{t_f} |P(t) - P(t_0)| dt \tag{62}$$

where $P = T + V$ is the total energy of the system. Since no non-conservative forces are present the total energy will remain constant. Summing up any changes in P provides for a convenient way to measure the error accumulation during the simulation. Fig. 8 compares the numerical accuracy of the instantaneous eigenstructure quasi-coordinate formulation to

the "brute force inverse" technique. Obviously performing an algebraic inverse of a mass matrix can only be performed for relatively low order systems such as this one. It would be extremely difficult to beat the accuracy of this brute force formulation. It does provide us however with a good base line of what type of accuracy would be possible under the best of situations. As Fig. 8 shows us, the instantaneous eigenstructure quasi-coordinate formulation is only slightly less accurate than the brute force integration method. This is a promising result and will lead to further investigation of the accuracy and efficiency of this new formulation.

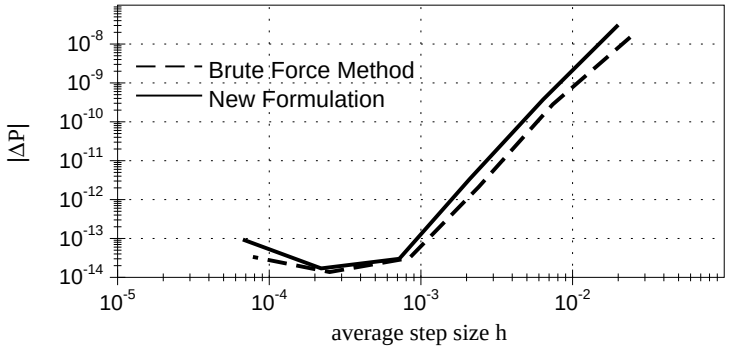

Figure 8 Integration Accuracy Comparison.

CONCLUSION

The method presented brings a general class of constrained multi-body dynamics to a form which completely avoids the necessity of inverting configuration-variable matrices to obtain instantaneous accelerations. The form of the equations is very analogous to classical "dynamics/kinematics" quasi-coordinate development of rigid body dynamics. The eigenvalue, eigenvector problem is only solved numerically once to find the initial $S(t_0)$, $C(t_0)$, $S_\Delta(t_0)$ and $C_\Delta(t_0)$ matrices. From there on the eigenfactors are solved through their differential equations. Instead of using the generalized coordinate derivative $\dot{x}$ as the velocity measure, a new quasi-velocity η is introduced to which corresponds an identity mass matrix.

To evaluate the eigenfactor derivatives it is assumed that $\dot{M}(x,t)$ and $\partial M/\partial x_k(x,t)$ are available algebraically. This is a feasible assumption, especially in view of the several modern software packages like Maple and Mathematica which can derive the mass matrix in an explicit algebraic form and automate the generation of, for example, the C-code to compute $\dot{M}(x,t)$ and $\partial M(x,t)/\partial x_k$.

For a constrained dynamical system, traditional processes lead to the classical Lagrange equations of motion coupled to second order differential constraint equations where a time and configuration variable mass matrix needs to be inverted. In the present development, there are no matrix inverse operations. These equations are mapped into a set of simpler nonlinear first order differential equations. The second order differential equation for $\ddot{x}$ is replaced with two first order differential equations $\dot{\eta}$ and $\dot{x}$. The mass matrix inverse problem is side-stepped by introducing the mass matrix eigenfactor matrices and solving their usually well-behaved differential equations for $\dot{S}$ and $\dot{C}$ instead. This method has no second coupled constraint equation, since the constraint force was already solved for and back-substituted into the equation of motion for η. However, this involved taking the inverse

of a symmetric Schur matrix Δ. This inverse can also be avoided very simply by using the eigenfactor matrices of the Schur matrix instead of the Schur matrix itself. Therefore, again a matrix inverse is replaced by solving two first order differential equations for $\dot{S}_\Delta$ and $\dot{C}_\Delta$. Contrary to Ref. 5 any integration method can be used to solve the $\dot{x}$ and $\dot{\eta}$ equations, since $\partial M/\partial x_k$ is given algebraically. This makes this method much more flexible and versatile.

REFERENCES

1. M. H. Rheinfurth and H. B. Wilson. "Methods of Applied Dynamics," NASA Reference Publication 1262, NASA, May 1991.
2. A. Jain and G. Rodriguez. "Diagonalized Lagrangian Robot Dynamics," *IEEE Trans. on Robotics and Automation*, Vol. 11, No. 4, Aug. 1995, pp. 571–584.
3. E. Bayo and R. Ledesma. "Augmented Lagranian and Mass-Orthogonal Projection Methods for Constrained Multibody Dynamics," *Journal of Nonlinear Dynamics*, Vol. 9, 1996, pp. 113–130.
4. J. L. Junkins and H. Schaub. "Orthogonal Square Root Eigenfactor Parameterization of Mass Matrices," *35th Aerospace Sciences and Meeting*, Reno, NV, Jan. 1997. paper No. 97-0248.
5. J. L. Junkins and H. Schaub. "An Eigenfactor Square Root Algorithm Formulation for Nonlinear Dynamics," technical report, Aerospace Engineering Department, Texas A&M University, May 1996.
6. J. G. Papastavridis. "On the Boltzmann-Hamel Equations of Motion: A Vectorial Treatment," *Journal of Applied Mechanics*, Vol. 61, June 1994, pp. 453–459.
7. Y. Oshman and I. Y. Bar-Itzhack. "Eigenfactor Solution of the Matrix Riccati Equation — A Continuous Square Root Algorithm," *IEEE Trans. on Automatic Control*, Vol. AC-30, No. 10, Oct. 1985, pp. 971–978.
8. J. L. Junkins and J. D. Turner. *Optimal Spacecraft Rotational Maneuvers.* Elsevier Science Publishers, Netherlands, 1986.
9. H. Schaub, P. Tsiotras, and J. L. Junkins. "Principal Rotation Representations of Proper NxN Orthogonal Matrices," *International Journal of Engineering Science*, Vol. 33, No. 15, Elsevier Science Ltd., Great Britain, 1995, pp. 2277–2295.
10. M. D. Shuster. "A Survey of Attitude Representations," *Journal of the Astronautical Sciences*, Vol. 41, No. 4, 1993. pp. 439–517.
11. J. L. Junkins and Y. Kim. *Introduction to Dynamics and Control of Flexible Structures.* AIAA Education Series, Washington D.C., 1993.
12. Y. Kim, S. Lee, and J. L. Junkins. "Eigenvector Derivatives for Mechanical Second - Order Systems," *Journal of Guidance, Control and Dynamics*, Vol. 18, No. 4, July–Aug. 1995, pp. 899–906.
13. C. G. J. Jacobi. "Über ein leichtes Verfahren die in der Theorie der Säculärstörungen vorkommenden Gleichungen numerisch aufzulösen," *Crelle's Journal*, Vol. 30, 1846, pp. 51–94.
14. K. Bathe. *Finite Element Procedures in Engineering Analysis.* Prentice-Hall, Inc., New Jersey, 1982.