Concurrent Multiprocessing for Calculating Nullspace and Range Space Bases for Multibody Simulation

Andrew J. Kurdila*
Texas A&M University, College Station, Texas 77843
and
Manohar P. Kamat†
Georgia Institute of Technology, Atlanta, Georgia 30332

Concurrent multiprocessing algorithms are presented for the calculation of orthonormal bases employed in the nullspace and range space methods for simulating the dynamics of multibody systems. It is shown that highly efficient parallel variants of the QR decomposition and modified Gram-Schmidt procedure can be derived by using a graph theoretic description of system connectivity. The methods developed herein regularly order the directed graph of the mechanical system, such that the transpose of the constraint matrix has block upper triangular form, and subsequently assign independent processors tasks based upon the zero fill structure of the constraint matrix. The derived algorithms are somewhat analogous to certain skyline factorization methods in finite element methods in that subdiagonal fill is limited to the original block band of the constraint matrix. Similarly, the numbering of bodies and constraints to achieve a regularly ordered graph can be likened to node numbering schemes in finite element methods that minimize bandwidth of the system matrix. A thorough comparison of the newly developed methods show that they are exceptional alternatives to conventional sequential techniques, as well as other established concurrent orthogonal basis methods.

Introduction

LITERATURE abounds in methods for simulating the dynamics of constrained multibody systems. At least two criteria are commonly employed to compare this vast collection of methods. One means of classifying the methods categorizes them into those that *symbolically* generate a system of governing equations for a problem at hand, and those that embody a general *numerical* set of equations appropriate for a wide variety of problems. Typically, the symbolic methods, such as Refs. 12 and 25, are more efficient in terms of execution time, and are often employed in controls applications. These methods may be restricted in the size of the problem they can address, due to the inordinate amount of computer memory required to explicitly factor system coefficient matrices. On the other hand, the numerical methods, such as in Ref. 22, are often more general in scope and can exploit sparsity of the governing equations, making them well-suited for problems with many degrees of freedom.

A second common means of comparing the various formulations considers the set of coordinates used in the formation of the governing equations. Many approaches use relative coordinates that determine the orientation of a particular body with respect to "previous" bodies in a system of bodies comprising a chain, or topological tree.[3] Closed-loop systems are considered by appending constraints of loop closure to those arising from a reduced system having tree topology. The relative coordinate methods have the advantage that the number of coordinates employed is minimal, but the methods can be complicated to implement.[26] A popular alternative to relative coordinates is the use of "global Cartesian" coordinates.[22,24] This choice of a redundant formulation has the disadvantage of increasing the number of coordinates beyond the minimum required, but is advantageous in its generality, ease of formulation, and ease of implementation.

One quite recent group of redundant formulation methods eliminates the constraint force contributions, and thereby avoids a potential source of numerical difficulty in the solution procedure.[2,23,26] These methods have been referred to as the nullspace methods,[5] tangent coordinate methods,[1] and methods based on Maggi's equations.[15] Another related, and in some sense complementary, redundant formulation that eliminates the constraint force contributions is the "range space" method.[1,14,19] Perhaps the single most important drawback of the nullspace formulation is that an orthogonal basis must be calculated for the nullspace of the constraint matrix at certain time steps. In some instances, the particular formulation requires the calculation of an orthogonal basis at every time step, as in Refs. 15 and 23, although the calculation is performed only at certain time steps in other formulations.[5,11,18] Similarly, the most stable implementation[7] of the range space formulation also requires the calculation of an orthogonal basis, but, in this case, for the range of the transposed constraint matrix. Different authors have spent considerable effort, and justifiably so, in deriving variants of both of these methods that are computationally efficient.

However, even though most of these methods are quite robust, they are not yet cost effective as regards applications to problems having many degrees of freedom, as is commonly encountered in the simulation of large space structures and complex flexible mechanisms. The task of accommodating more computationally intensive problems must be addressed. This point is clearly illustrated by Gluck,[8] who cites examples in which the ratio of sequential simulation time to real-time transient response time can easily exceed a factor of 800. In other words, simulation of a transient event transpiring over 1 minute may require 13 hours of cpu time, depending on the complexity of the model, equation formulation method, and the specific sequential computing machine.

The advent of concurrent multiprocessors, or parallel computers, provides a fruitful framework within which to develop new highly efficient dynamics simulation techniques. To some

Presented as Paper 88-2395 at the AIAA SDM Issues of the International Space Station, Williamsburg, VA, April 21-22, 1988; received Aug. 22, 1988; revision received April 2, 1989. Copyright © 1989 American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Assistant Professor, Department of Aerospace Engineering.

†Professor, School of Aerospace Engineering. Associate Fellow AIAA.

extent, the introduction of concurrent multiprocessors renders discussion of sequential algorithm efficiency obsolete. Some research into concurrent multiprocessing methods for multibody dynamics is already underway. In one example, Bae and Haug[3] achieve excellent results for a single problem by deriving explicit governing equations via a recursive relative coordinate formulation, and balancing the work load among processors by hand using the directed graph for the system. Still, the authors are aware of no published research into concurrent multiprocessing methods applicable to the large class of general, numerical/redundant nullspace formulations that exist today.

This paper presents concurrent multiprocessing algorithms that may be used in conjunction with either the nullspace or range space methods for simulating the transient dynamics of multibody systems. Specifically, since it is demonstrated in the paper that a dominant computational step in these methods is the calculation of an orthonormal basis, this research focuses upon means of effectively calculating, in a concurrent multiprocessing environment, the bases required by the nullspace and range space simulation methods. Concurrent algorithms designed expressly for dynamics simulations that are variants of methods such as the QR decomposition and modified Gram-Schmidt procedure are introduced. These methods use the directed graph of the system to automatically balance the work load among the independent processors. The concurrent algorithms to be described herein have been designed in the hope of being flexible and general enough so that the procedures can be exploited by many of the existing methods.

Nullspace and Range Space Equations for Dynamics

Despite the fact that the details involved in implementing a simulation method for multibody systems can vary considerably, some difficulties have been documented[23,26] in methods that explicitly retain undetermined multipliers in their integration process. Consequently, several recent methods have been developed that eliminate these "algebraic" unknowns either analytically or numerically at each time step. This class of techniques can be derived by starting with the first-order system of "differential-algebraic" equations and corresponding constraint equations governing the dynamics of a constrained mechanical system.

$$M(y,t)\dot{y} = g(y,t) + c^T\lambda \tag{1}$$

$$cy + d = 0 \tag{2}$$

In Eq. (1), M is an $N \times N$ system coefficient matrix, c is a $D \times N$ constraint matrix, y and g are N-vectors and λ, d are D-vectors. If a Lagrangian formulation, as in Ref. 22, is utilized, the vector y contains derivatives of generalized coordinates. In the case where Kane's equations,[25] Jourdain's principle,[12] or Newton-Euler methods are employed, y may contain derivatives of quasicoordinates.

Both the nullspace and range space formulations of the governing equations can be derived from Eq. (1) by decomposing the space in which the vector $\dot{y}$ is contained. By introducing the definition of the nullspace of the constraint matrix c and the range of c^T,

$$\text{nullspace } (c) = \{x \epsilon R^N : cx = 0\} \tag{3}$$

$$\text{range } (c^T) = \{x \epsilon R^N : \exists y \epsilon R^D \text{ with } x = c^T y\} \tag{4}$$

the rank-nullity theorem guarantees that a direct sum decomposition of R^N is possible.

$$R^N = \text{nullspace } (c) \oplus \text{range } (c^T) \tag{5}$$

Consequently, the N-vector of unknown derivatives $\dot{y}$ can be expressed as the sum of two orthogonal components at a specific time t

$$\dot{y} = A\xi + c^T\eta \tag{6}$$

where ξ and η are $I = N - D$ and D-vectors, respectively, and A is an $N \times (N - D)$ matrix whose columns span the nullspace of c. By differentiating the constraints in Eq. (2) and utilizing Eq. (6), one can find that

$$\dot{y} = A\xi - c^T(cc^T)^{-1}(\dot{c}y + \dot{d}) \tag{7}$$

The nullspace equations for constrained dynamics result from substituting Eq. (7) into Eq. (1) and premultiplying by A^T, so that

$$\xi = (A^TMA)^{-1}[A^Tg + A^TMc^T(cc^T)^{-1}(\dot{c}y + \dot{d})] \tag{8}$$

and

$$\dot{y} = A(A^TMA)^{-1}[A^Tg + A^TMc^T(cc^T)^{-1}(\dot{c}y + \dot{d})]$$
$$- c^T(cc^T)^{-1}(\dot{c}y + \dot{d}) \tag{9}$$

A special case of these equations for a particular choice of nullspace basis A can be found in Ref. 23. A diverse variety of choices exist for the selection of the basis for the nullspace in Eq. (9). Several possible selections are outlined in Jalon[5] and Kurdila.[15]

It should be noted that the governing Eq. (9) has been derived by assuming that the constraints are, at worst, of the "linear, nonholonomic" type. If it is assumed that the constraints are instead of the "holonomic, scleronomic" form, an alternative form for the nullspace equations can be derived via a decomposition having the structure

$$\dot{y} = A\dot{\zeta} + \dot{A}\zeta \tag{10}$$

Although the final nullspace equations derived in this fashion are subtly different from those shown in Eq. (9), they have the same character.[1,5] That is, both representations are functions of the nullspace matrix A, and the concurrent algorithms presented in this paper are applicable.

Another way of solving for the vector of derivatives $\dot{y}$ in Eqs. (1), using the orthogonal decomposition in Eq. (6), leads to the range space equations. If one premultiplies Eq. (1) by cM^{-1} and substitutes Eq. (6) for the vector $\dot{y}$, the undetermined multipliers are then

$$\lambda = -(cM^{-1}c^T)^{-1}\{\dot{c}y + \dot{d} + cM^{-1}g\} \tag{11}$$

Hence, the governing equations can be expressed as

$$\dot{y} = M^{-1}\{g - c^T(cM^{-1}c^T)^{-1}[(\dot{c}y + \dot{d}) + cM^{-1}g]\} \tag{12}$$

Equation (12) is referred to as the "range space equation" for constrained multibody dynamics. Various forms of this equation have appeared in dynamics formulations in recent literature.[1,19,21,27]

Computational Complexity Analysis

When developing a concurrent multiprocessing algorithm for a particular task, it is imperative to identify those steps that constitute the greatest computational cost. Only in this fashion may an efficient parallel method be designed. In this section, a brief discussion of the computational cost associated with the nullspace and range space formulations is presented. Although significant portions of the overall simulation time may be attributed to both the formation of the system of nonlinear, differential-algebraic equations and to the actual solution of the system for $\dot{y}$, the focus of this paper is on the latter contributions. Surprisingly, despite the fact that the nullspace and range space formulations are in a sense complementary, it is shown in this section that the calculation of an orthogonal basis at specified time steps dominates the overall solution time in both methods.

When some forward time-marching algorithm is applied to the nullspace equation in Eq. (9), the nonlinear right-hand side of the equation must be evaluated at several specific values of t, and y must be defined by the integration rule. Assuming that the nonlinear system coefficient matrix M, constraint matrix c, and vector g have been formed at the specified integration time, then the solution of Eq. (9) for the derivatives $\dot{y}$ entails the following primary (cubic order) computational steps:

1) Calculation of an orthogonal basis for the nullspace of the constraint matrix

2) Formation of the reduced system coefficient matrix A^TMA

3) Factorization of the reduced system coefficient matrix A^TMA

The determination of a similar breakdown for the time spent solving the range space [Eq. (12)] for the derivatives $\dot{y}$ is a bit more complicated. The difficulty arises when one attempts to factor the coefficient matrix $cM^{-1}c^T$ appearing in Eq. (12). The error induced in factoring this matrix directly is bounded by[7]

$$[\kappa(c)]^2\ \kappa(M) \tag{13}$$

where $\kappa(*)$ denotes the condition number of the matrix $*$.[9] When the condition number of the constraint matrix is large, as is the case when a system of constraints becomes nearly redundant, the above bound indicates, heuristically, a relatively large induced error. A means of circumventing this problem is to employ an orthogonal factorization of the constraint matrix c^T

$$c^T = OP \tag{14}$$

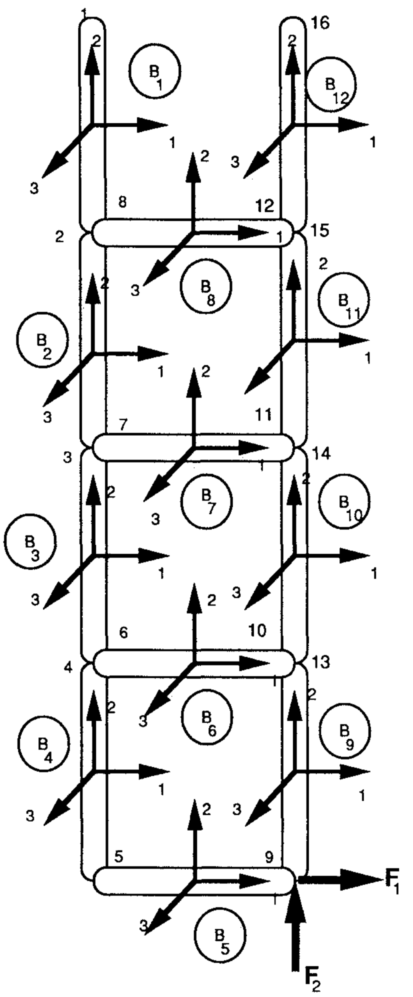

Fig. 1 A 12 rigid bar mechanism.

where O is an $N \times D$ orthogonal matrix and P is a $D \times D$ upper triangular matrix. With this factorization, the calculation of $cM^{-1}c^T$ is achieved by using the identity

$$(cM^{-1}c^T)^{-1} = P^{-T}(O^TM^{-1}O)^{-1}P^{-1} \tag{15}$$

The factorization of P and P^T above are easily achieved due to the upper triangular form of P. In addition, the factorization of $O^TM^{-1}O$ induces an error that is bounded by

$$[\kappa(O)]^2\ \kappa(M) = \kappa(M) \tag{16}$$

since O is orthogonal and $\kappa(O) = 1$. When this procedure is used to calculate the factorization of $cM^{-1}c^T$, the solution of the range space equations for the derivatives $\dot{y}$ entails the following primary computational steps:

1) Calculation of an orthogonal basis for the range space of the constraint matrix c.

2) Factorization of the system coefficient matrix M.

3) Formation of the reduced system coefficient matrix $cM^{-1}c^T$.

4) Factorization of the reduced system coefficient matrix $cM^{-1}c^T$.

Of the operations tabulated above for the nullspace and range space formulations, concurrent strategies for matrix factorization have been explored extensively. Furthermore, the products $cM^{-1}c^T$ and A^TMA are rather straightforward to calculate in a concurrent computing environment. Hence, the focus of this paper is the investigation of concurrent orthogonal basis algorithms for multibody simulation.

Sequential Computing Example

Although the last section enables one to identify the primary computational steps, a concrete example helps one to fully appreciate the extent to which the calculation of an orthogonal basis dominates the solution time in the nullspace and range space methods. The example mechanism considered in this paper is shown in Fig. 1. It is comprised of 12 uniform, rigid bars having 6 deg of freedom each. The system is connected by 16 spherical joints. A body fixed coordinate system B_i is attached to the center of mass of each bar i. A dextral set of basis vectors for each frame B_i is denoted $(\hat{b}_m)_i$, $m = 1\cdots3$. A system of basis vectors for the inertial frame I are given by $\hat{i}_n$, $n = 1\cdots3$, and l_{mn} are the direction cosines forming the orthogonal transformation of basis

$$\hat{b}_m = l_{mn}\hat{i}_n \tag{17}$$

The geometric and material properties of each bar are

$$\begin{array}{lll} M & = 44.32 \text{ kg} & = \text{Mass} \\ A & = 2.827E-3 \text{ m}^2 & = \text{Cross-sectional area} \\ L & = 2.0 \text{ m} & = \text{Length} \\ I_t & = 14.77 \text{ kg-m}^2 & = \text{Transverse inertia} \\ I_a & = 0.020 \text{ kg-m}^2 & = \text{Axial inertia} \\ \rho & = 7838 \text{ kg/m}^3 & = \text{Density} \end{array} \tag{18}$$

When the derivatives of the quasicoordinates for body i are selected to be

$$(\dot{X}_{c_l})_i = (v_c \cdot \hat{i}_l)_i \tag{19}$$

$$(\omega_l)_i = ({}^I\omega^B \cdot \hat{b}_l)_i \tag{20}$$

where $(\dot{X}_{c_l})_i$ is the lth coordinate of the center of mass of body i, and $(\omega_l)_i$ is the lth component of the angular velocity of body i in the I frame, a quasicoordinate method can be employed to generate a system of governing equations having the form shown in Eq. (1).[14]

$$M(y,t)\dot{y} = g(y,t) + c^T\lambda \tag{21}$$

$$cy + d = 0 \tag{22}$$

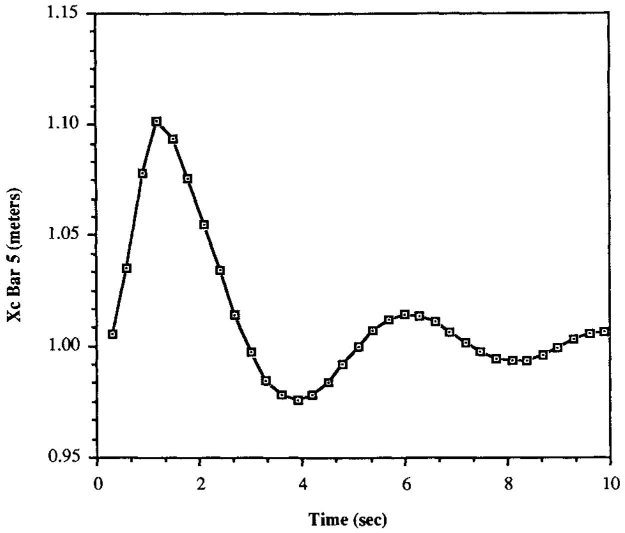

Fig. 2 *X* **location, center of mass bar 5.**

With this choice of method for formulating the equations of motion, the parameters N, I, D from the second section are

$$\begin{aligned} N &= 12 \times 6 = 72 \\ D &= 16 \times 3 = 48 \\ I &= N - D = 24 \end{aligned} \tag{23}$$

Thus, M is a 72 × 72 matrix, c is a 48 × 72 matrix, g is a vector of length 72, and λ is a vector of length 24. Since the system shown has closed loop topology, the number of independent constraints may be less than D for some (or all) configurations of the system. However, during the simulation carried out in this example, the constraint matrix maintained a full row rank equal to D.

In the simulation, the system is subject to two forces. The horizontal force F_1 and the vertical force F_2 shown in Fig. 1 are given by

$$F_1 = -20 \sin\omega t \, \vec{\imath}_1 \text{ Newtons}, \qquad 0 < t < 1.5 \text{ s} \tag{24}$$

$$F_2 = -50 \, \vec{\imath}_2 \text{ Newtons} \qquad t > 0 \tag{25}$$

The transient response is calculated for roughly 13 s, using an Adams-Bashforth-Moulton explicit predictor-implicit corrector integration scheme with a constant time step of 0.075 s. Both the nullspace and range space formulations are employed to evaluate derivatives $\dot{y}$ and yield the same results. The X location of the center of mass of bar 5 during the simulation is shown in Fig. 2. As expected, the response peaks while the force F_1 is nonzero and rapidly decays after the force is removed. The maximum error norm $\epsilon_{\max}$ of constraint violation at the spherical joints

$$\epsilon_{\max} = \frac{\text{maximum}}{k = 1{,}16} \left\| R_{s1}^k - R_{s2}^k \right\| \tag{26}$$

remained on the order of 10^{-6} throughout the time period of interest. In the above equation, R_{s1}^k and R_{s2}^k are the locations in the inertial frame of the kth spherical joint connecting bodies $s1$ and $s2$, respectively.

The lists below summarize the time spent during several phases of the solution process using the nullspace and range space methods. A Givens-based QR decomposition and the modified Gram-Schmidt procedure are used to calculate the bases for the nullspace and range space, respectively, of the constraint matrix c. Both tables indicate the times in terms of 1/50 of a second for a single evaluation of the right-hand side of Eqs. (9) and (12) to calculate the derivative terms $\dot{y}$.

Table 1 Nullspace timing results

Orthogonal basis calculation	914	83%
Formation/factorization	143	13%
Quadratic order	41	4%

Table 2 Range space timing results

Orthogonal basis calculation	440	40%
Formation	497	45%
Factorization	136	12%
Quadratic order	12	1%

Clearly, the calculation of an orthogonal basis constitutes a significant portion of the overall solution time in both methods. From the first table, the calculation of an orthogonal basis for the nullspace of the constraints using a Givens QR decomposition is seen to require over 80% of the total solution time. The formation and factorization of the system coefficient matrix A^TMA requires a mere 13% by comparison. When the range space equations are employed, the calculation of an orthogonal basis via the modified Gram-Schmidt procedure for the range of the constraints takes 40% of the solution time.

Concurrent Orthogonal Basis Calculation

Graph theoretic principles have been employed in the simulation of multibody systems for the last twenty years. The now classical multibody formalism of Wittenburg[27] is one example of a robust method that makes use of graph theory in the formation and solution of the governing system of differential-algebraic equations. Perhaps the primary advantage associated with a graph theoretic approach to multibody dynamics is that the formulation permits elegant analytical solutions to be written for quite complicated systems. In addition, the method naturally lends itself to the considerable "accounting" required in a general multibody simulation program. In this section, it is demonstrated that there are significant computational reasons for employing a graph theoretic approach in the range space and nullspace methods for simulating transient response. Specifically, the process of generating a "regularly ordered" graph associated with a multibody system enables extremely efficient concurrent algorithms for the calculation of orthogonal bases for the range space and nullspace of the constraints.

Graph Theory and Constraint Matrix Sparsity

A complete description of the application of graph theory to the transient simulation of multibody systems is beyond the scope of this paper. A treatment more detailed than that which follows can be found in Wittenburg.[27] Fortunately, only the most fundamental concepts from a graph theoretic approach to multibody dynamics is required in this paper. In particular, there are diverse ways of mathematically representing the interconnection structure of a mechanical system via the "directed graph" of the mechanism. For the purposes of the concurrent methods to follow, however, it suffices to describe the "incidence matrix" of (the directed graph associated with) a mechanical system. In an intuitive sense, that incidence matrix specifies which bodies are connected by a certain constraint. The elements of the incidence matrix are defined by

$$\Lambda_{kl} = \begin{cases} +1 & \text{if constraint } l \text{ leaves body } k \\ -1 & \text{if constraint } l \text{ enters body } k \\ 0 & \text{otherwise} \end{cases} \tag{27}$$

In other words, each column of the incidence matrix corresponds to a certain hinge. There are exactly two nonzero entries in any column after the first column, each entry representing one of the two bodies that are connected by the constraint. For example, the incidence matrix for (the directed graph of) the system in Fig. 1 is

$$\Lambda = \begin{bmatrix} 1 & 1 & & & & & & 1 & & & & \\ & -1 & 1 & & & & 1 & & & & & \\ & & -1 & 1 & & 1 & & & & & & \\ & & & -1 & 1 & & & & & & & \\ & & & & -1 & & & & 1 & & & \\ & & & & & -1 & & & & 1 & & \\ & & & & & & -1 & & & & 1 & \\ & & & & & & & -1 & & & & 1 \\ & & & & & & & & -1 & & & \\ & & & & & & & & & -1 & & \\ & & & & & & & & & & -1 & \\ & & & & & & & & & & & -1 \end{bmatrix} \tag{28}$$

For different numberings of the constraints and bodies, not every incidence matrix has the upper triangular form shown here. However, every (directed graph corresponding to a) mechanical system with tree topology can be numbered such that the incidence matrix has the upper triangular form of the example above.[27] For specific classes of constraints and choices of generalized coordinates, the structure of the incidence matrix can reflect the block sparsity structure of the actual constraint matrix.[13,14]

Perhaps the most straightforward way to illustrate exactly how the block sparsity structure of the constraint matrix is affected by system connectivity is by considering an example. Referring to the example presented in section 2, constraint that body i is connected to body j by a spherical joint is written as

$$\{v_c + {}^I\omega^B \times r_s\}_i - \{v_c + {}^I\omega^B \times r_s\}_j = 0 \tag{29}$$

$$\{\dot{X}_{c_l} + l_{kl}e_{kmn}\omega_m r_{s_n}\}_i - \{\dot{X}_{c_l} + l_{kl}e_{kmn}\omega_m r_{s_n}\}_j = 0 \tag{30}$$

$$[C_{\text{spherical}}]\begin{Bmatrix} \dot{X}_c \\ {}^I\omega^B \end{Bmatrix}_i - [C_{\text{spherical}}]\begin{Bmatrix} \dot{X}_c \\ {}^I\omega^B \end{Bmatrix}_j = 0 \tag{31}$$

where r_s is the location in the body-fixed basis of the spherical joint and e_{kmn} is the Levi-Civita permutation symbol. The set of constraints corresponding to all of the hinges in the mechanism in Fig. 1 can be formed simply by rewriting the constraint equation above for every pair of bodies that are connected by a ball joint. When all of these constraints are assembled together, they can be collected in matrix form, as discussed in the second section

$$Cy + d = 0 \tag{32}$$

where the system constraint matrix is comprised of specific constraint matrices, such as $[C]_s$ above. The block fill pattern of the particular constraint matrix for the mechanism in Fig. 1 is depicted below.

●	●						●				
	●	●				●					
		●	●		●						
			●	●							
				●				●			
					●				●		
						●				●	
							●				●
								●			
									●		
										●	
											●

It is clear from this illustration that the block sparsity pattern of the constraint matrix in this case depends upon the numbering scheme for the bodies and hinges making up the mechanism. Specifically, the transpose of the constraint matrix has a block structure that is identical to the zero/nonzero structure of the incidence matrix of the associated directed graph.

In Kurdila,[13,14] it is shown that this observation can be generalized to hold for a wide class of formulations under mild restrictions regarding the choice of coordinates and type of constraints. More importantly, the procedure above can be extended to form the foundation of efficient concurrent multiprocessing techniques for the nullspace and range space simulation techniques. In the next section, methods are described that utilize the zero/nonzero structure of the incidence matrix to assign independent processors "noncolliding" tasks when calculating orthogonal bases for the range or nullspace of the constraints.

Concurrent Columnwise QR Decomposition

The QR decomposition of an $N \times D$ matrix c^T is defined by the relationship

$$c^T = QR = [Q_1 \; Q_2]\begin{bmatrix} R_u \\ 0 \end{bmatrix} \tag{33}$$

where Q is an $N \times N$ matrix, R is an $N \times D$ upper triangular matrix, Q_1 is an $N \times D$ matrix, Q_2 is an $N \times (N - D)$ matrix, and R_u is a $D \times D$ upper triangular matrix. The useful characteristics of this decomposition to multibody simulation procedures are that the columns of Q_1 form an orthonormal basis for the range of the matrix c^T, whereas the columns of Q_2 form a orthonormal basis of the nullspace of c. Perhaps the most common means of calculating the QR decomposition of a matrix c^T is by the Householder orthogonalization procedure. In this method, a sequence of orthogonal Householder transformations are applied to the matrix c^T. Each transformation has the property that it zeroes all subdiagonal elements of one of the columns of the matrix c^T. Many of the most general commercially available versions of software to perform a QR decomposition use this approach. It should be noted that this technique often makes no use of sparsity, or any special structure, of the matrix c^T.

A second, normally less efficient, means of calculating the QR decomposition of a matrix c^T can be obtained by using orthogonal Givens transformations. The Givens orthogonalization approach is similar to the Householder method in that the QR decomposition is achieved by applying a sequence of orthogonal transformations $Q_{(k)}^T$ to the matrix c^T.

$$Q_{(k)}^T \cdots Q_1^T c^T = \begin{bmatrix} R_u \\ 0 \end{bmatrix} \tag{34}$$

Each Givens transformation has the form of the identity matrix "modulo, a 2 × 2 rotation matrix." Details of the definition of a Givens rotation matrix can be found in Ref. 9.

As opposed to the Householder method, the parameter β in each Givens transformation is selected such that only one subdiagonal entry in c^T is zeroed upon application. From the form of the Givens rotation matrix, it is evident that premultiplication of a matrix by $Q_{(k)}$ alters only two rows of that matrix. Consequently, by applying several rotations in a sequence,[9] all subdiagonal entries of the matrix c^T can be zeroed, and the QR decomposition of c^T is generated.

$$c^T = Q_{(1)} \cdots Q_{(k)}\begin{bmatrix} R_u \\ 0 \end{bmatrix} \tag{35}$$

Although a Givens based QR decomposition method is usually not as efficient as the Householder algorithm, the fact that the Givens method can selectively zero elements of a matrix has made it attractive in certain circumstances. For applications involving sparse matrices[10] and updates of QR decompositions,[6] the method is quite convenient.

Still another area in which the Givens-based QR decomposition has enjoyed much success is in the field of parallel computation. Concurrent multiprocessing algorithms for Givens-based QR decompositions have appeared in Cosnard,[4] Lord,[17] and Sameh.[20] The philosophy behind each of these techniques is, in fact, quite similar. Since the application of a Givens rotation affects only two rows of the matrix to which it is premultiplied, Cosnard,[4] Lord,[17] and Sameh[20] have suggested different schemes in which "noncolliding" Givens transformations are applied concurrently to the same matrix by independent processors. In effect, each method creates a more complicated orthogonal transformation that zeroes several subdiagonal elements of the matrix c^T by creating the "direct sum" of several Givens orthogonal rotation matrices. Several different ways of collecting the Givens rotations into groups that can be applied concurrently are possible, and some are summarized in Ref. 29.

However, for application to problems in simulating constrained dynamical systems, the methods in Cosnard, Lord, and Sameh have the disadvantage that they are effective only for densely filled matrices. Consider the "greedy" algorithm discussed in Cosnard.[4] In this algorithm, subdiagonal elements of the matrix c^T are zeroed out, starting with the lower left corner element and proceeding upwards and to the right. No provision is made for the fact that the matrix c^T may be sparse or block structured, and considerable fill-in of nonzero elements can occur as the algorithm proceeds. This is a serious deficiency for sparse or highly structured matrices.

This fill-in problem, which necessitates zeroing many more entries in the matrix c^T than are originally nonzero, can be alleviated by developing a concurrent algorithm that utilizes the potential band structure of the constraint matrix. This goal can be achieved by first numbering the bodies and hinges of the mechanical system to generate a regularly ordered graph. Using the system of coordinates selected in this work, the transposed constraint matrix is then guaranteed to block upper triangular form. It is then possible to derive a Givens annihilation pattern that minimizes the number of rotation matrices that must be applied, and reduces the subdiagonal fill-in to within the original block banded region.[13,14] The algorithm is based upon the fact that Givens rotations can be used to zero one-half of the elements in a particular column that lie below all nonzero entries in all columns to its left. This algorithm is denoted the concurrent columnwise QR decomposition and is summarized as follows:

$$c^T = [c_1^T \cdots c_D^T] \tag{36}$$

$$c^T = QR = [Q_1 Q_2]R \tag{37}$$

1) Order the directed graph of the constrained system to generate a regularly ordered graph, and drive c^T to block upper triangular form. Store the length of the nonzero portion of each column.

$$l_{c_j} = \text{length of column } j,\ j = 1 \cdots D \tag{38}$$

2) Generate the columnwise sequence of Givens transformations based upon the upper banded structure of c^T. The details involved in deriving the transformation sequence can be found in Refs. 13 and 14. Store the sequence in which the Givens transformations are to be applied:

$$Q^T(i,j;\ I_p,\ N_{\text{cycle}}) \tag{39}$$

where i and j are the row numbers affected by this transformation, I_p is the processor to apply the transformation, and N_{cycle} is the concurrent cycle in which the transformation is applied.

3) Loop for the number of concurrent cycles: $k = 1 \cdots N_{\text{cycle}}$.

4) Concurrently apply all Givens transformations for this concurrent cycle k.

$$\left.\begin{bmatrix}\text{create and apply}\\ Q^T(i_1,\ j_1;\ 1,\ k)\end{bmatrix}\right\}\text{processor 1}$$
$$\left.\begin{bmatrix}\text{create and apply}\\ Q^T(i_2,\ j_2;\ 2,\ k)\end{bmatrix}\right\}\text{processor 2}$$
$$\vdots$$
$$\left.\begin{bmatrix}\text{create and apply}\\ Q^T(i_{N_p},\ j_{N_p};\ N_p,\ k)\end{bmatrix}\right\}\text{processor } N_p \tag{40}$$

Concurrent Modified Gram Schmidt Procedure

In the last section, it was demonstrated that an efficient concurrent multiprocessing algorithm can be derived for a Givens-based QR decomposition by using the fact that the transposed constraint matrix has block upper triangular form. The resulting concurrent QR decomposition can be employed in either the nullspace or range space methods for multibody simulation. In this section, a similar strategy enables efficient concurrent multiprocessing versions of the modified Gram-Schmidt procedure to be developed. Various forms of the sequential version of the Gram-Schmidt procedure have appeared in the literature on the nullspace equations for multibody simulation.[5,16]

In terms of a decomposition, the modified Gram-Schmidt procedure can be viewed as another means of achieving an orthogonal factorization of the $N \times D$ matrix c^T

$$c^T = OP \tag{41}$$

In the above equation, O is an $N \times D$ matrix, and P is a $D \times D$ upper triangular matrix. It is a property of the decomposition that the columns of O form an orthonormal basis for the range of c^T. The procedure by which the matrices O and P are generated has a simple geometric interpretation. Starting with the first (left-most) column of c^T, each column is normalized to unit length, and subsequently, its orthogonal projection is subtracted from all columns to its right. In this fashion, an orthonormal basis for the range of c^T is generated. Because the algorithm is structured to subtract the orthonormal component of a certain column from several columns to its right, it is not difficult to devise a scheme whereby several independent processors perform the orthonormalization concurrently. In effect, each processor "cascades" over all columns to the right of the current column, subtracting the orthogonal component of the current column from every N_pth column, where N_p is the number of processors. This algorithm is summarized as follows:

Concurrent Modified Gram-Schmidt

$$c^T = [c_1^T \cdots c_D^T] \tag{42}$$

1) Loop for columns: $(j = 1 \cdots D)$.

2) Normalize column j.

$$P_{jj} = (c_j^T,\ c_j^T)^{1/2} \tag{43}$$

$$c_j^T = c_j^T / P_{jj} \tag{44}$$

3) Calculate the number of concurrent cycles: N_{cycle}.

$$N_{\text{cycle}} = (N_p + D - 1)/N_p \tag{45}$$

4) Loop for concurrent cycles: $l = 1 \cdots N_{\text{cycle}}$.

5) Concurrently subtract orthogonal components.

$$\left.\begin{bmatrix} k_1 = j + (l-1)*N_p + 1 \\ P_{jk_1} = (c_j^T, c_{k_1}^T) \\ c_{k_1}^T = c_{k_1}^T - P_{jk_1} c_j^T \end{bmatrix}\right\} \text{processor 1}$$

$$\left.\begin{bmatrix} k_2 = j + (l-1)*N_p + 2 \\ P_{jk_2} = (c_j^T, c_{k_2}^T) \\ c_{k_2}^T = c_{k_2}^T - P_{jk_2} c_j^T \end{bmatrix}\right\} \text{processor 2}$$

$$\vdots$$

$$\left.\begin{bmatrix} k_{N_p} = j + (l-1)*N_p + N_p \\ P_{jk_{N_p}} = (c_j^T, c_{k_{N_p}}^T) \\ c_{k_{N_p}}^T = c_{k_{N_p}}^T - P_{jk_{N_p}} c_j^T \end{bmatrix}\right\} \text{processor } N_p \qquad (46)$$

Because the intent of this paper is to derive concurrent algorithms expressly for problems arising in the simulation of system dynamics, still further improvements to the above technique are possible. Just as in the columnwise QR method discussed in the last section, one can make use of the block sparsity structure of c^T to increase the efficiency of the modified Gram-Schmidt procedure. By simply using a regular ordering for the specific constrained mechanical system, one is assured that much of the above work in the modified Gram-Schmidt procedure need not be carried out. Once the length of each column is known, the inner products and vector updates in the above algorithm need only be carried out for the nonzero portion of the column. Normally, these savings in computational work would not be worth the effort involved in determining the column lengths. Yet, since the constraint matrix is certain to have block upper triangular form, and because the column lengths must be calculated only once, the savings in execution time are significant.

Comparison of the Concurrent Orthogonal Basis Methods

A preliminary comparison of the concurrent multiprocessing algorithms described in the last three sections of this paper has been conducted using the FLEX 32 concurrent multiprocessing computer at the Georgia Institute of Technology. The system is comprised of eight independent processors, each having two megabytes of local memory accessible by only one processor, and four megabytes of shared, or global, memory accessible by all processors. The primary goals of the investigation have been to determine the following:

1) What is the asymptotic performance increase relative to sequential computing techniques of each algorithm? What parameters control this performance increase?

2) At what point do "diminishing returns" obviate the need for additional processors?

3) What effect does regular ordering of the directed graph have upon the performance of the various algorithms?

The study has been conducted in two phases. First, the asymptotic performance of each algorithm has been measured experimentally as a function of the constraint matrix order and number of processors. In this part of the study, the matrices are created automatically, and are either full or block upper triangular, depending upon the algorithm to be tested. Obviously, for some of the algorithms the fill pattern has no effect upon its efficiency, whereas on others it has a profound effect. The second phase of the study looks at the performance of two of the algorithms during the simulation of a constrained mechanical system.

Asymptotic Performance

Ideally, the introduction of N_p independent processors for calculating the solution to a given problem should result in an execution time that is N_p times faster than that achieved with only one processor. Such an increase in performance is realizable, unfortunately, only in the case of problems that are comprised of N_p uncoupled subproblems. In actuality, synchronization and communication can result in decreases in execution time that are considerably less than this optimum. This effect lessens in severity as the fraction of computational work that can be done in parallel becomes much greater than that which must be done sequentially. In other words, problems of greater size typically have execution times that more closely approach the theoretical limit for a given algorithm. Clearly, any viable concurrent multiprocessing algorithm should approach this ideal level of efficiency in the limit of large problems.

Although it is certainly a sound policy to require all viable concurrent algorithms to approach this ideal performance in the limit of large problems, one must consider the performance of the algorithms relative to other potential methods. For example, even though a specific algorithm may be "highly parallelizable" and closely approach its ideal execution time, it may still be slower than other methods. For these reasons, it has become necessary to select relative performance measures for the concurrent algorithms. Although several comparative measures of efficiency and performance of parallel algorithms have been defined, no single index seems universally appropriate.[29] In this paper, the relative speed-up of an algorithm has been selected because of its simplicity in form. The relative speed-up of a parallel algorithm is defined to be

$$\phi_r = \frac{\text{execution time of the fastest algorithm}}{\text{execution time of the algorithm with } N_p \text{ processors}} \qquad (47)$$

where N_p is the number of processors employed by the method. One source of difficulty in defining the relative speed-up, as defined above, is that it requires an agreement on what constitutes the "best" or "fastest" sequential algorithm available. For the class of problems at hand, this choice is not clear. Although the modified Gram-Schmidt procedure is approximately one-half as costly as a Householder QR decomposition, and several times as efficient as a singular value decomposition, it can generate an orthonormal basis that is of a slightly poorer quality.[9] This characteristic of the method can be a source of concern in application to problems in constrained dynamics.[16] To give as objective a comparison as

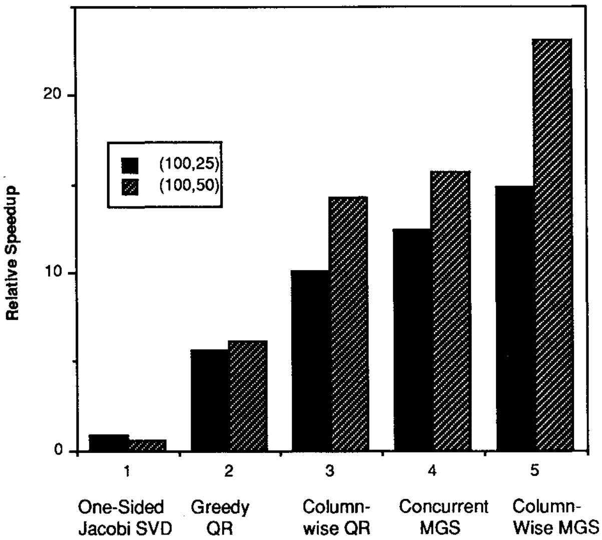

Fig. 3 Relative speed-up for range space basis (QR).

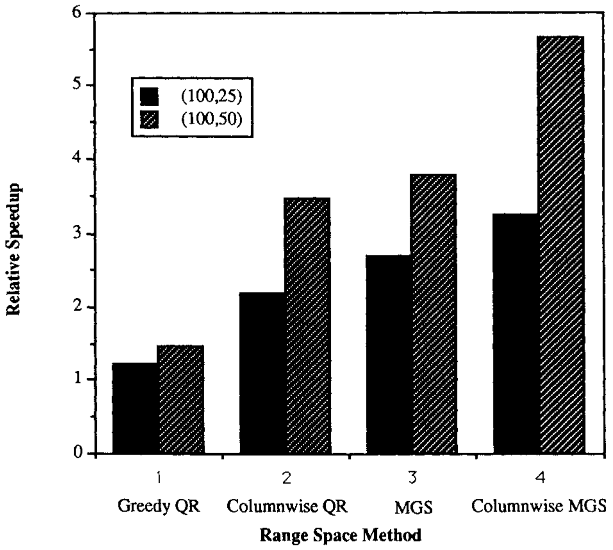

Fig. 4 Relative speed-up for range space basis (MGS).

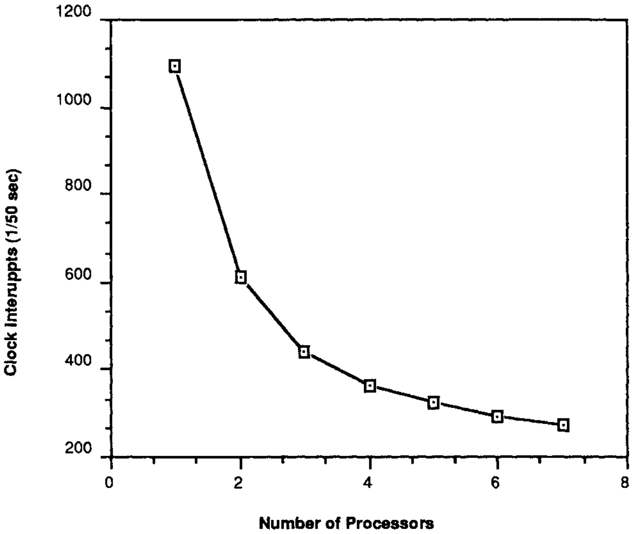

Fig. 5 Execution time for nullspace solution of derivatives.

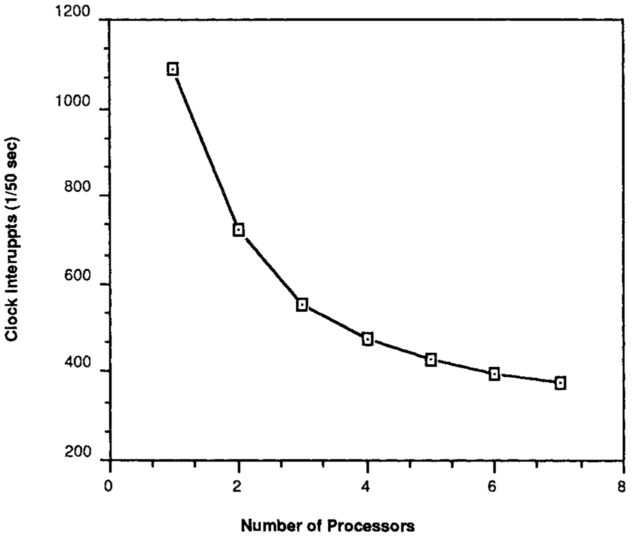

Fig. 6 Execution time for range space solution of derivatives.

possible, the relative speed-up of the various techniques have been normalized with respect to two different methods. One comparison employs the sequential modified Gram-Schmidt procedure as the standard for comparison. A second relative speed-up is calculated using the sequential Greedy QR decomposition. This latter algorithm has been selected because it is in some sense the "optimal" Givens annihilation pattern, and has a concurrent version.[4]

Figures 3 and 4 succinctly represent the outcome of the comparison of the various concurrent techniques in terms of the relative speed-ups defined above. With respect to the execution time for the Greedy QR decomposition, the concurrent modified Gram-Schmidt procedures are by far the most efficient algorithms for calculating an orthogonal basis for the range space of the constraints. The concurrent and columnwise modified Gram-Schmidt procedures have yielded speed-ups for the 100 × 50 matrix of approximately 15 and 23, respectively. In other words, it is estimated that on 7 processors, these concurrent algorithms require around 1/15 and 1/23 of the execution time required by the sequential Householder QR decomposition to calculate an orthonormal basis for the range of the constraint matrix. After the modified Gram-Schmidt methods, the Givens-based concurrent columnwise and concurrent Greedy QR decompositions are seen to have relative speed-ups near 15 and 5, respectively, for the 100 × 50 matrices using 7 processors. It must be kept in mind when interpreting these results, however, that these encouraging relative speed-ups are due to several different factors:

1) The parallelism induced by the algorithms

2) The columnwise algorithms that order the constraint matrix first and then operate only on the nonzero block band, thereby resulting in a faster sequential method

3) The sequential modified Gram-Schmidt procedure that is (roughly four times) faster than the sequential Greedy QR decomposition

Consequently, the relative speed-ups are not due solely to parallelism, but due to the fact that the sequential methods appearing in the literature make no use of the special block upper triangular structure of the constraint matrix. These comparisons are important to those currently employing "off the shelf" matrix packages that do not consider the sparsity of the matrix.

A more pessimistic comparison of the merits of the various algorithms is achieved by normalizing the relative speed-ups with respect to the sequential modified Gram-Schmidt procedure. Figure 4 illustrates that the columnwise QR, concurrent modified Gram-Schmidt, and columnwise modified Gram-Schmidt require about 1/4, 1/4, and 1/6, respectively, of the execution time of the sequential modified Gram-Schmidt procedure.

Performance for the Example Mechanism

As a final comparison, two of the concurrent algorithms have been employed as part of the nullspace and range space formulations for the problem considered in the third section. The same transient response is obtained with both methods, and is identical to that achieved in the sequential simulations shown in Fig. 2. Figures 5 and 6 depict the time required to calculate once the derivatives $\dot{y}$ in Eqs. (9) and (12). As illustrated in Fig. 5, the solution time for the nullspace equations is reduced by a factor of 4 when the columnwise QR decomposition is used to calculate the nullspace basis. Similarly, the solution time for the range space equations is reduced by a factor of 3 when the columnwise modified Gram-Schmidt is used to solve for the derivatives.

Conclusions

Several recent papers have presented methods that simulate the transient response of constrained dynamical systems by eliminating the constraint contributions to the governing equations. One class of these techniques generates a basis for the tangent, or nullspace, of the constraint matrix; another generates a basis for the range space to eliminate the constraint forces and torques. Much of the recent work in this area focuses upon means of estimating and improving the computational efficiency of the methods, due to the cost incurred in the governing nonlinear system of equations.

However, the methods presented have been designed solely for utilization on sequential processing computers. To some degree, comparisons of the efficiency of these methods based upon sequential processing methods have been rendered inadequate, due to the introduction of parallel processors. In this paper, three concurrent multiprocessing algorithms for calculating an orthonormal basis for the range space and/or nullspace of the constraints have been derived. These methods are appropriate for use with a wide class of equation formulation techniques. A study and comparison of the performance and efficiency of the methods has shown that they provide exceptional alternatives to conventional sequential algorithms, as well as other established concurrent methods.

It has been shown that the improvement in the performance of the algorithms presented in this paper can be attributed to both 1) the parallelism induced in each technique, and 2) the block upper triangular form induced in the constraint matrices by employing a regularly ordered directed graph for the mechanical system. This last factor emphasizes the computational advantages associated with the use of graph theory for nullspace methods. Finally, it is shown that the newly developed columnwise QR decomposition and columnwise modified Gram-Schmidt have characteristics reminiscent of "skyline" equation solvers employed in finite element methods. The approaches limit subdiagonal fill-in to below the main diagonal only within the block lower band of the matrix.

References

[1] Agrawal, O. P., "Dynamic Analysis of Multi-body Systems Using Tangent Coordinates," *The Theory of Machines and Mechanisms,* edited by E. Bautista, J. Garcia-Lomas, and A. Navarro, Pergamon Press, Oxford, England, 1987, pp. 533-536.

[2] Amirouche, F. M., and Jia, T., "Automatic Elimination of the Undetermined Multipliers in Kane's Equations Using a Pseudo-uptriangular Decomposition (PUTD) Method," *Computers and Structures,* Vol. 27, No. 2, 1987, pp. 203-210.

[3] Bae, D., and Haug, E. J., "A Recursive Formulation for Constrained Mechanical System Dynamics: Parts i,ii,iii," *Mechanics of Structures and Machines* (to be published).

[4] Cosnard, M., and Robert, Y., "Complexity of the Parallel QR Decomposition of a Rectangular Matrix," *Parallel Computing 85,* edited by M. Feilmeier, North-Holland, NY, 1986, pp. 123-128.

[5] Garcia de Jalon, J., and Jimenez, J. M., "Basis for the Nullspace for the Jacobean of Constrained Multibody Systems," *The Theory of Machines and Mechanisms,* edited by E. Bautista, J. Garcia-Lomas, and A. Navarro, Pergamon, Oxford, England, 1987, pp. 501-504.

[6] Dongarra, J. J., Biench, J. R., Moler, C. B., and Stewart, G. W., "LINPACK Users' Guide," Society of Industrial and Applied Mathematics, Philadelphia, PA, 1979.

[7] Gill, P. E., Murray, W., and Wright, M. H., "Practical Optimization," Academic, London, England, 1981.

[8] Gluck, R., "Hope for Simulating Flexible Spacecraft," *Aerospace America,* Nov. 1986, pp. 40-44.

[9] Golub, G. H., and Van Loan, C. F., *Matrix Computations,* The Johns Hopkins University, Baltimore, MD, 1983.

[10] Heath, M. T., Plemmons, R. J., and Ward, R. C., "Sparse Orthogonal Schemes for Structural Optimization Using the Force Method," *Society of Industrial and Applied Mathematics Journal of Scientific and Statistical Computing,* Vol. 5, No. 3, Sept. 1984, pp. 514-528.

[11] Kim, S. S., and Vanderploeg, M. J., "QR Decomposition for State Space Representation of Constrained Mechanical Dynamic Systems," *Journal of Mechanisms, Transmissions, and Automation in Design,* 108, June 1986, pp. 183-188.

[12] Kreuzer, Edwin J., and Schiehlen, Werner O., "Symbolic Computerized Derivation of Equations of Motion," *Dynamics of Multibody Systems,* edited by K. Magnus, Springer-Verlag, Berlin, Federal Republic of Germany, 1978, pp. 290-305.

[13] Kurdila, A. J., "Concurrent Multiprocessing Methods for Calculating Nullspace and Range Space Bases for Multibody Simulation," *Proceedings of the AIAA/ASME/ASCE/AHS/ASC 29th Structures, Structural Dynamics and Materials Conference,* Washington, DC, 1988, pp. 1447-1459.

[14] Kurdila, A. J., *Concurrent Nullspace and Range Space Methods for Constrained Dynamic Analysis,* Ph.D. Thesis, Georgia Institute of Technology, Atlanta, GA, 1988.

[15] Kurdila, Andrew J., Papastavridis, J. G., and Kamat, M. P., "On the Role of Maggi's Equations in Computational Methods for Constrained Multibody Systems," *Journal of Guidance, Control, and Dynamics,* Vol. 13, No. 1, 1990, pp. 113-120.

[16] Liang, C. G., and Lance, G. M., "A Differentiable Null Space Method for Constrained Dynamic Analysis," *Journal of Mechanisms, Transmissions, and Automation in Design,* Vol. 109, Sept. 1987, pp. 405-411.

[17] Lord, R. E., Kowalik, J. S., and Kumar, S. P., "Solving Linear Algebraic Equations on a Mind Computer," *Journal of the Association for Computing Machinery,* Vol. 30, No. 1, 1983, pp. 103-117.

[18] Mani, N. K., *Use of Singular Value Decomposition for Analysis and Optimization of Mechanical System Dynamics,* Ph.D. Thesis, University of Iowa, Iowa City, IA, 1984.

[19] Placek, B., "Contribution to the Solution of the Equations of Motion of the Discrete Dynamical System with Holonomic Constraints," *The Theory of Machines and Mechanisms,* edited by E. Bautista, J. Garcia-Lomas, and A. Navarro, Pergamon, Oxford, England, 1987, pp. 379-382.

[20] Sameh, A. H., and Kuck, D. J., "A Parallel QR Algorithm for Symmetric Tridiagonal Matrices," *IEEE Transactions on Computers,* Vol. 26, No. 2, 1977, pp. 147-152.

[21] Schielen, W. O., "Dynamics of Complex Multibody Systems," *Solid Mechanics Archives,* Vol. 9, 1984, pp. 159-195.

[22] Shabana, A. A., and Wehage, R. A., "A Coordinate Reduction Technique for Dynamic Analysis of Spatial Substructures with Large Angular Rotations," *Journal of Structural Mechanics,* Vol. 11, No. 3, 1983, pp. 401-413.

[23] Singh, R. P., and Likins, P. W., "Singular Value Decomposition for Constrained Dynamical Systems," *Journal of Applied Mechanics,* Vol. 52, Dec. 1985, pp. 943-948.

[24] Song, J. O., and Haug, E. J., "Dynamic Analysis of Planar Flexible Mechanisms," *Computer Methods in Applied Mechanics and Engineering,* Vol. 24, 1980, pp. 359-381.

[25] Wampler, C., Buffinton, K., and Shu-hui, J., "Formulation of Equations of Motion for Systems Subject to Constraints," *Journal of Applied Mechanics,* Vol. 52, June 1985, pp. 465-470.

[26] Wehage, R. A., and Haug, E. J., "Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems," *Journal of Mechanical Design,* Vol. 104, Jan. 1982, pp. 247-255.

[27] Wittenburg, J., *Dynamics of Systems of Rigid Bodies,* B. G. Teubner, Stuttgart, Federal Republic of Germany, 1977.