A Hybrid Parallelizable Low Order Algorithm for Dynamics of Multi-Rigid-Body Systems: Part I, Chain Systems

Journal *Mathematical and Computer Modelling*, vol. 30, 1999, pp. 193-215.

Kurt S. Anderson [1] and Shanzhong Duan [2]
Department of Mechanical Engineering, Aeronautical Engineering, and Mechanics
Rensselaer Polytechnic Institute
Troy, NY 12180-3590

Abstract

The paper presents a new hybrid parallelizable O(n) algorithm for the modeling the dynamic behavior of multi-rigid-body chain systems. The method is based on the cutting of certain system interbody joints so that largely independent multibody subchain systems are formed. These subchains in turn interact with one another through associated unknown constraint forces $\underline{f_c}$ at the cut joints. The underlying feature of this new hybrid low order algorithm is the increased parallelism through cutting the joints and explicit determination of associated constraint loads combined with sequential O(n) procedure. In other words, sequential O(n) procedures are performed to form and solve equations of motion within subchains and parallel strategies are used to form and solve constraint equations between subchains in parallel. It is shown that the resulting algorithm is coarse grain parallelizable up to the total number of bodies in the system. Also, the algorithm can easily accommodate the available number of processors while maintaining high efficiency. An $O[\frac{(n+m)}{N_p} + \frac{m^{(1+\gamma)}}{N_p} + m^{\gamma} log_2 N_p]$ $(0 < \gamma < 1)$ performance will be achieved with N_p processors for a chain system with n degrees of freedom and m constraints due to cutting of interbody joints.

1 Introduction

Computational efficiency of multibody systems has been receiving increasing attention since the first O(n) algorithm developed by Armstrong in 1979[1]. A myriad of formulations and algorithms have been put forward by individuals whose interests lay in a wide variety of fields (Hollerbach 1980 [2], Walker and Orin 1982 [3], Featherstone 1983 [4], Bae and Haug 1987 [5,6], Nielan 1996 [9], Anderson 1990 [7], Rosenthal 1990 [8]). All in an effort is to develop more efficient, yet general simulation algorithms for multibody systems. In many situations, computational efficiency, which manifests itself in the form of computational speed, is of paramount importance. Evidence of this may be seen with even increasing calls for a significant reduction of simulation time in the various applications of multibody algorithms. During the specialist meeting at the JPL in 1987, the simulation requirements in control systems design were described as follows [10]: "Problem solutions must be run in large numbers to arrive at design decisions, and large systems must be studied. Computational speed therefore becomes the most important single consideration in code design." This necessity was even more emphasized at the JPL summer conference on Aerospace Computational Control in 1988 [11] and was further reinforced at the NATO conference on Computer-Aided Analysis of Rigid and Flexible Mechanical Systems in Troia of Portugal [12]. As Garcia de Jalon and Bayo also state [13], "the calculation times are excessively high for even cases of average complexity, and there are important formulation and implementation problems that remain to be solved."

To this end, researchers have been trying to improve the overall computational efficiency through vastly different approaches. Some researchers pursuing improved simulation speed through the use of different dynamic analysis methods for multibody systems, such as the method based on Newton-Euler

[1] Assistant Professor
[2] Doctoral Graduate Student

equations [1,3,4], Lagrangian equations [2], Kane's equations [7,8,9] and analytical mechanics [5, 6]. Some pursuing improved simulation speed through developing new more efficient underlying algorithms, such as $O(n^3)$,$O(n^2)$ [3] and $O(n)$ [1,2,4,5,7,9] algorithms. While still other researchers direct their attention to more efficient implementation of mathematical operations within existing formulations. Whichever avenue is pursued, most dynamical formulations fall into one of the following two forms. One is the state space form (SSF)

$$\underline{\dot{q}}_I = \underline{q}_{II} \tag{1a}$$

$$\underline{\underline{M}}(t, q_I)\underline{\dot{q}}_{II} = \underline{RHS}(t, q_I, q_{II}) \tag{1b}$$

or the descriptor form (DSF)[19]

$$\underline{\dot{q}}_I - \underline{q}_{II} = \underline{0} \tag{2a}$$

$$\underline{\underline{M}}(t, q_I)\dot{q}_{II} - \underline{\underline{A}}^T(t, q_I)\underline{\lambda} - \underline{RHS}(t, q_I, q_{II}) = \underline{0} \tag{2b}$$

$$\underline{\Phi}(t, q_I) = \underline{0} \tag{2c}$$

In equations (1) and (2), $\underline{q}_I$ are the generalized coordinates used to define the configuration, $\underline{q}_{II}$ are the time derivatives of generalized coordinates, and $\dot{q}_{II}$ are the second time derivatives of generalized coordinates to be determined. The matrices $\underline{\underline{M}}$ and $\underline{RHS}$, are respectively referred to as the system mass matrix and RHS (Right Hand Side) matrix, with RHS consisting of all applied forces, stiffness terms, plus portion of inertia forces and torques associated with Coriolis and centripetal accelerations. In general, the elements of mass matrix $\underline{\underline{M}}$ and matrix $\underline{RHS}$ are nonlinear functions of the system state and time. $\underline{\underline{A}}$ is the Jacobi constraint matrix, which results from differentiating holonomic constraint equation (2c) with respect to time. The unknown Lagrange multipliers $\underline{\lambda}$ represents generalized constraint forces, and n is number of degree of freedom of chain system under considerations.

Much effort has been expended in the development of general multibody dynamics routines by various investigators, some pursing the state space form[1-9, 13-15], while the others choosing the descriptor form [15-18]. Each of these forms offer their own strengths and weaknesses. The state space form is Lagrangian equations of type two. Formulations yielding such equations is straight forward only in the case of chain or tree-configured systems when using relative state variables to describe system motion. State variables $\underline{q}_I$ and $\underline{q}_{II}$ are independent which results in a mass matrix $\underline{\underline{M}}$ which is small, but dense and the resulting equations may be treated by standard ordinary differential equation (ODE) solver. While the descriptor form is Lagrangian equations of type one. In the descriptor form, state variables $\underline{q}_I$ and $\underline{q}_{II}$ are redundant which results in system a mass matrix $\underline{\underline{M}}$ which is of larger dimension than that determined by SSF. Consequently, this mass matrix has a more sparse structure and the associated equations are represented by a set of differential algebraic equations (DAE) of index three (Brenan *et. al* [20]), which requires some special DAE solvers. Such DAE system may suffer from significant numerical difficulties and they are the topics of much on-going research[19-22]. Table 1 shows a comparison of some of the key characteristic often associated with each of these forms.

For a large system involving many dynamic degrees of freedom, purely sequential $O(n)$ algorithms have often excelled relative to the more traditional $O(n^3)$ (Featherstone [4], Anderson [14], Banerjee [23]). These recursive $O(n)$ methods tend to use the introduction of intermediate variables and recursive relationships to efficiently perform a triangular decomposition of the equations of motion (1) as they are being formed. This allows the solution of equations (1) for the system state derivatives with significantly fewer operations than by using traditional $O(n^3)$ methods. However, the recursive relationships of $O(n)$ methods severely limit the level of coarse grain parallelization, thus making these approaches much less amenable to parallel computing than the more traditional $O(n^3)$ methods (Fijany *et al.* [24,25]). But if one wisely combines parallel computing with key aspects of the $O(n)$ procedure, one may obtain a significantly higher degree of coarse grain parallelization and an associated increase in simulation speed.

Table 1: Comparison of SSF and DSF

Form	$\mathbf{q_I}, \mathbf{q_{II}}$	Lagrange	Equations	Topology Easily Accommodated
SSF	Independent	Type II	ODE	Chain, Tree
DSF	Redundant	Type I	DAE	Chain, Tree & Closed Loop

Table 2: Comparison of SSF and DSF (Continue)

Form	$\underline{M}$	State Variable	Worst Order	Pre- or Post-Processing	Constraints Accommodated
SSF	Small Dense	Relative	$\mathbf{O(n^3)}$	More	Not Easily
DSF	Large Sparse	Absolute	$\mathbf{O(n^4)}$	Less	Relatively Easily

The added parallelism for this hybrid procedure is obtained through the cutting of certain system interbody joints so that largely independent multibody subsystems are formed. These subsystems in turn interact with one another through associated unknown constraint forces f_c, as shown in figure 1. These constraint forces must be applied to the subsystem so that they behave in the resulting equations of motion as in the original multibody system. Proceeding in this way, the only coupling between the subsystems occurs through these constraint loads. This procedure increases parallelism through the explicit determination of constraint loads associated with cut joints, and forms the basis for the approaches presented and implemented in this paper [27,28].

In this algorithm, the equations of motion are derived via a new hybrid *descriptor-low order state space* formulation, which can be developed through proper manipulation and consideration of the algebraic constraint equations (2c) with equations (2a) and (2b). Specifically, the equations of motion for a chain system may be written in the form

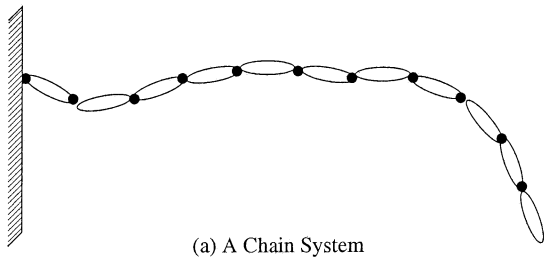
$$\underline{M}(t, q_I)\underline{\ddot{q}}_I = \underline{RHS}(t, q_I, \dot{q}_I, f_c) \tag{3a}$$

where the constraint load measure numbers f_c are determined from the companion system of linear constraint equations

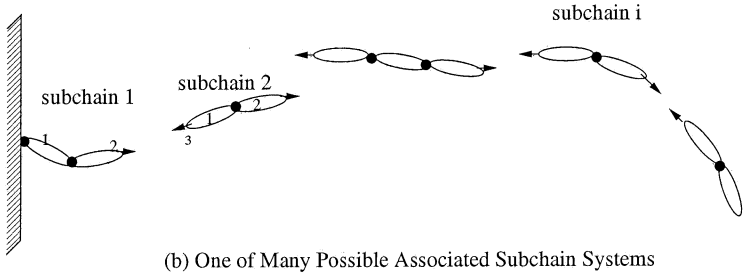
$$\underline{\Gamma}(q_I, t)\underline{f}_c = \underline{\Psi}(q_I, \dot{q}_I, t) \tag{3b}$$

The variables q_I and $\dot{q}_I$ define the system state. $\underline{\Gamma}$ is constraint force coefficient matrix and $\underline{\Psi}$ is nonlinear functions of the state variables. The detailed derivation of these quantities is presented in the formulation section.

Equations of motion (3a) are associated with each non-constrained subsystem, and each row of constraint equations (3b) is associated with one of the constraint loads associated with one of the cut joints. The unknown constraint forces f_c in (3b) are then determined using parallel iterative procedures. Once the explicit constraint forces f_c have been determined from (3b), they may be substituted into (3a), which has already been formed and now is to be solved for $\ddot{q}_I$ using an efficient hybrid $O(N)$ type procedure [27,28]. The updated values of system state, q_I and $\dot{q}_I$ may be obtained through the temporal integration of $\dot{q}_I$ and $\ddot{q}_I$, respectively, using an integration routine such as a 4-th order Runge-Kutta.



(a) A Chain System



(b) One of Many Possible Associated Subchain Systems

Figure 1: A Chain Multibody System and Associated Set of Subsystems (Subchains)

2 History Review

Until fairly recently there has been relatively little work on the development of parallel methods for simulation dynamics of multibody systems on parallel-architecture computer systems. Most algorithms and associated computer codes were developed without any regard to the potential benefits of parallel configuration of multibody systems and concurrent processing. The emphasis for the higher computational efficiency was generally placed on reducing the number of mathematical operations required to solve for the state derivative values to a minimum. In addition, initial work on parallel treatment of dynamics problems was mostly confined to the parallelization of the existing sequential codes. These procedures were mostly of the global type, with the equations of motion being developed in the form of $\underline{M}\underline{\ddot{q}} = \underline{RHS}$. For these algorithms and associated codes, the procedures for the parallel formulation and solution of equations of motion can be thought of as consisting of parallel implementation in the following four key steps:

i) Assembly of the mass matrix $\underline{M}$;

ii) Assembly of the matrix $\underline{RHS}$, which represents external, centripetal, and Coriolis forcing terms;

iii) Solution of the resulting linear system of equations for the state derivatives $\ddot{q}$;

iv) Numerical temporal integration of the state derivatives $\ddot{q}$ to update $\dot{q}$ and q.

With the further development of parallel computing, more and more researchers began to consider how the potential benefits of parallel configuration of multibody systems and concurrent processing might be realized. Much of the more recent work has from its inception been dedicated to taking the advantage of parallel configuration of multibody systems and concurrent processing. Kasahara *et al* 1987 [29], Bae and Haug 1988 [30], Lee *et al* 1988 [31], Hwang *et al* 1990 [32], Anderson 1990 [7] ,1993 [33] and 1997 [27], Sharf 1993 [34], Eichberger 1994 [35,36] and Fijany *et al* 1993 and 1995 [24, 37] developed different paral-

lel algorithms to take the advantages of both parallel configuration of multibody systems and concurrent processing techniques. Some of them have $O(n)$ performance such as Anderson's and Eichberger's methods, some of them have *quasi* $O(N)$ performance such as Sharf's method and some of them have better quasi $O(log_2 N)$ performance such as the method proposed by Fijany and Sharf,and method proposed by Anderson.

Kasahara *et al* [29] in 1987 made the first contribution to parallel computing of robot dynamics simulation. They divided the computations required for the simulations into tasks and these tasks were performed in parallel as possible. The method used for the underlying formulation is method 3 of Walker and Orin [3] so that system mass matrix $\underline{M}$ was constructed by recursive means allowing each element of $\underline{M}$ to be determined at fixed cost. They set up the equations of motion by means of Newton-Euler formulation and solved for the state derivative values through the use of a parallel Gauss-Jordan procedure.

Bae *et al* in 1988 [30] applied the algorithm already developed by them to a shared memory multiprocessor system to realize parallel computing of multibody systems. An off-road vehicle with a suspension system that had eight closed loops was used to demonstrate the parallel algorithm. Though this work could be applied to general multibody systems with closed loop, the parallelization largely was limited performing calculations along independent branches of the system's spanning tree.

Based on Kane's method, Lee in 1988 [38] presented a global $O(n^3)$ algorithm for the simulation of the behavior of large flexible space structures. He designed the algorithm from its inception to develop the equations of motion in such a way as to allow the exploitation of the concurrent processing to the maximum degree on a MIMD machine. In his work, much more effort was placed on parallelizing all four of the tasks mentioned above. He demonstrated that when m processors were available with ideal communication structure, then for large systems ($n >> m$) a speedup in turnaround by a factor of $< m$ was possible. However, due to the underlying $O(n^3)$ nature of his formulation, the best he was able to obtain was a theoretical $O(n)$ performance with $O(n^2)$ processors for his systems.

Also in 1988, Lee *et al* [31] developed two parallel simulation procedures based on Walker and Orin's method 3 and method 4 with parallelization accomplished on SIMD computers. In the first procedure, the mass matrix $\underline{M}$ was evaluated in parallel by using techniques presented by Lathrop [39], and Lee and Chang [40]. The resulting equations of motion were then solved for the state derivative values $\ddot{q}$ by means of a parallel Cholesky factorization method designed for implementation on a processor array. The result indicated an $O(n)$ time complexity with $O(n^2)$ processors. In the second procedure, they chose the parallel conjugate gradient iterative method for the parallel implementation of the system equations. The theoretical time bound, which may be accomplished by their parallel conjugate gradient iterative method, is $O(n)$ for multiplication and $O(nlog_2 n)$ for addition.

Based on the $O(n)$ recursive dynamics formulation developed by Bae *et al.*, Hwang *et al.* in 1990 [32] presented a concurrent processing procedure. The algorithm could be applied to general multibody systems with closed loops. In the same way, the parallelization was largely limited performing calculations along independent branches of the spanning tree configuration.

In 1990, Anderson [7] pursued increased computational efficiency through the use of low order algorithms while also trying to exploit the inherent concurrence of the multibody systems. Based on Kane's dynamic equations, he developed his own $O(n)$ recursive procedure. Increased computational efficiency was accomplished through the production of explicit equations of motion for the specific mechanical system being analyzed via symbolic manipulation. The associated simulation code was then automatically produced from these explicit equations of motion and was devoid of extraneous computations. At the same time, the simulation code was from its inception designed for parallel implementation on an available MIMD distributed architecture computing system. The resulting algorithm could be applied to general multi-rigid body systems, possessing an arbitrary number of branches, closed loops, and described motions. At the coarsest level parallelism was approximately a function of the system configuration. Specifically, for a tree system or a tree system with closed loops which were cut at certain joint points, the flow of the

calculations performed in the simulation followed branches of systems spanning tree graph representation. In this manner, all calculations associated with independent branches of the spanning tree could be carried out in parallel.

Based on his previous work, Anderson in 1992 [14] and in 1993 [15, 33] demonstrated that significantly greater parallelism could be realized with a modified form of the $O(n)$ algorithm than was simply shown by the parallelization along independent branches of the spanning tree associated with the system. This added parallelism could be obtained through each of two approaches. The first one was the application of a various simple penalty methods for enforcing the kinematic constraints to selected joints which were cut for the purpose of producing independent subtree graphs (also used by Zeid *et. al* [26]). In this way, these subtrees were easily decoupled from one another and could be calculated in parallel. However, due to the stiffness added to the system through the simplistic introduction of the penalty methods, the numerical problems associated with the application of the penalty procedure tended to arise. The second procedure involved the generalization of the algorithm designed for the dealing with closed loops in a parallel manner. In this approach, a closed loop would be cut to produce two separate branches which could be largely analyzed in parallel. The only coupling occurred through constraint loads which had to exist at the joint if it were not cut. Similarly, the added parallelism could be obtained through the determination of the constraint forces which existed at the system joints which were cut to produce the largely independent spanning trees. This procedure forms the basis for the work presented in this paper.

Sharf *et al.* in 1993 [34] presented a new solution procedure for dynamic simulation of pure multi-rigid-body chains with rotational and/or translational interbody constraints. They used Newton-Euler method to form the equations of motions and cut at each joint to form the constraint equation. Because of the limitation to consideration of simple chain systems, the procedure produced a sparse system of linear equation for the constraints and resulted in a block tridiagonal constraint matrix. The resulting constraint equations for the unknown interbody constraint forces were solved by various iterative parallel methods. One of them was polynomial acceleration procedure, such as conjugate gradient acceleration applied to Jacobi method and Chebyshev acceleration applied to Jacobi. The constraint forces were then used to determine state derivative values. The state derivatives are in turn integrated temporally to obtain updated values for the system state. The key aspects of parallelism of this algorithm was based on determination of interbody constraint forces via an iterative procedure. $O(n^{1.12})$ computational complexity was achieved sequentially. Also, a *theoretical estimation* of the parallel efficiency of the algorithm was discussed and a predicted speed up compared with sequential $O(n)$ was shown.

Also in 1993, Chung *et al.* [41] developed a recursive $O(n^3)$ algorithm based on a variational formulation. In this algorithm, parallelism was achieved by subdividing the computational loads required for the simulation into tasks. The tasks were then performed in parallel as possible. In this way, many terms, such as the elements of the mass matrix, could be determined at a fixed cost. By means of static scheduling schemes, the computational loads were evenly distributed over the eight processors of the shared memory machine being used.

Fijany *et al.* in 1993 [24] showed that the more traditional $O(n^3)$ methods provide the highest degree of parallelism and conceivably will produce the most efficient parallel computation (that is minimum turnaround time per integration step if ideal communication and a sufficient number of processor is available). Specifically, a theoretical turnaround performance of $O(log_2 N)$ can be achieved with $O(N^3)$ processors.

Eichberger *et al.* in 1994 [36, 37, 42] introduced a new O(N) residual algorithm, which generated the equations of motion in descriptor form suitable for implicit numerical integration. The residual at a number of levels were computed iteratively so that either medium grain level parallelisms, such as parallelization of the residual algorithm, or coarse grain level parallelisms such as parallelization of the Jacobian can be accomplished on a distributed memory computing system. From the results presented it is not possible to ascertain how simulation speedup is related to the size of the model, the number of processors, or their

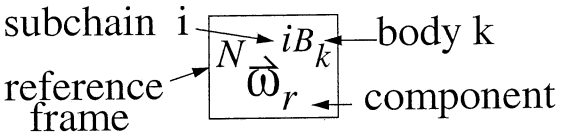


Figure 2: An Example for Notation

communication structure.

Based on Sharf's parallel strategies for constraint equations, Fijany *et al.* in 1995 [37] presented a solution procedure for forward dynamics of open multi-rigid-body chains. They uses Newton-Euler method to form the equations of motions and cuts at each joint to form the constraint equation. The algorithm is called Constraint-Force algorithm (CF) with both serial and parallel implementation. The parallel implementation is quasi $O(log_2 N)$ with $O(N)$ processors to solve both equations of motion and constraint equations. The procedure results in a theoretical lower bound on simulation turnaround time corresponding to an overall computational performance of $O(log_2 N)$ with $O(N)$ processors if calculation can be performed at a fixed number of iterations and a sufficient processors with ideal communication are available.

In 1997, Anderson developed a highly parallelizable low order algorithm for multi-rigid-body systems. This methodology generates very efficient computational algorithms in a sense of optimal amount of time and usage of available resources. The algorithm is applicable to general multi-rigid-body systems which may contain arbitrary joint type, multiple branches and/or closed loops. The method may be considered a hybrid formulation using key aspects of the sequential $O(n)$ algorithm developed by him and others, and the parallel "Full Descriptor" formulations, similar in many ways to those presented by Sharf *et al.* [34]. This method has been shown to be excellent in situations where $n >> N_p$, where n is the number of bodies and N_p is the number of processors. This paper represents considerable refinement and extension to Anderson's work.

3 Theoretical Basis

3.1 Preliminaries and Notations

Kane's dynamical equations (Kane & Levison [43]) are taken as axiomatic, and no derivation is given here. The detail of derivations of $O(n)$ method is presented in references (Anderson [7], Rosenthal [8]).

A vector dyadic representation is used in the derivation of this algorithm. If letter A is used to define different quantities, then the following notions are used in the derivations of this method.

$$scalar = A \qquad vector = \vec{A} \qquad dyadic = \vec{\vec{A}} \qquad matrix = \underline{A}$$

The left superscript refers to a reference frame, while the right superscript identifies a body or point. If no left superscript is given, then the Newtonian reference frame, N, is implied. If there are two indexes in the left superscript, then the first index represents number of subsystem (subchain) and the second refers to the frame that is fixed on the corresponding body. In the right superscript B_k represents body k, while J_k refers to joint k. Thus, $\binom{J_k}{B_k}$ refers to the joint point k located on body k. $\bar{B}_k$ represents the central mass point of body k. The right subscript refers to the components or index of vector, dyadic and matrix. The example of angular velocity is shown in figure 2.

Geometric description of a chain system is given in figure 3. The points k^- and k^+ are the *proximal hinge* point and *hinge point*, respectively, of joint k connecting body $k-1$ to body k of subchain i.

The separation of points k^- and k^+ is given by the vector ${}^{iB_{(k-1)}}\vec{d}^{i\left({}^{J_{k^+}}_{B_{k-1}}\right)\left({}^{J_{k^-}}_{B_k}\right)}$, where ${}^{iB_{(k-1)}}\vec{d}^{i\left({}^{J_{k^+}}_{B_{k-1}}\right)\left({}^{J_{k^-}}_{B_k}\right)}$ is parallel to the translation of k^+ relative to k^- and has a magnitude equal to the amount of this translation. The position vector ${}^{iB_{(k-1)}}\vec{s}^{i\left({}^{J_{(k-1)^+}}_{B_k}\right)\left({}^{J_{k^-}}_{B_k}\right)}$ points from $(k-1)^+$ to k^-, while the vector ${}^{iB_k}\vec{r}^{i\left({}^{J_{k^+}}_{B_k}\right)\bar{B}_k}$ is the position vector from k^+ to $\bar{k}$. The location of k^+ relative to $(k-1)^+$ is defined as

$$
\begin{aligned}
{}^{iB_{k-1}}\vec{\gamma}^{i\left({}^{J_{k-1}}_{B_{k-1}}\right)\left({}^{J_k}_{B_k}\right)} \triangle & {}^{iB_{(k-1)}}\vec{s}^{i\left({}^{J_{(k-1)^+}}_{B_k}\right)\left({}^{J_{k^-}}_{B_k}\right)} + {}^{iB_{(k-1)}}\vec{d}^{i\left({}^{J_{k^+}}_{B_{k-1}}\right)\left({}^{J_{k^-}}_{B_k}\right)} \\
= &
\end{aligned}
$$

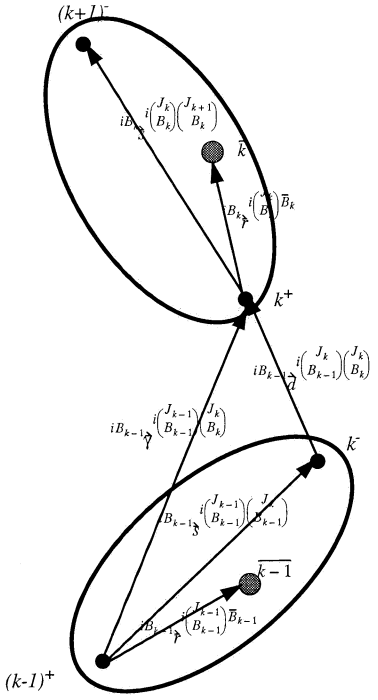


Figure 3: Adjacent Bodies of Subchain i

4 Derivation of New Hybrid Algorithm

Consider a system as shown in figure 1. For the purpose of this derivation, a system of cut subchains as shown is used. After cutting, constraint forces $\vec{f}_c^{\,i}$ and constraint moments $\vec{l}_c^{\,i}$ appear between the terminal body of the subchain i and the base body of the subchain $i+1$ to insure the system of subchains continuing to behave as the original interconnected systems as shown in figure 4. Within each subchain i the numbering of each body increases sequentially from the base body 1 to the terminal body n^i.

4.1 Forming recursive kinematic relations

The generalized acceleration matrix of the k^{th} body of the i^{th} subchain with respect to the inertia reference frame is represented by

$$
\underline{\mathcal{A}}^{iB_k} \triangleq \begin{bmatrix} \vec{\alpha}^{iB_k} \\ \vec{a}^{i\left({}^{J_k}_{B_k}\right)} \end{bmatrix} \qquad \in \mathcal{R}^{6\times 1} \tag{4}
$$

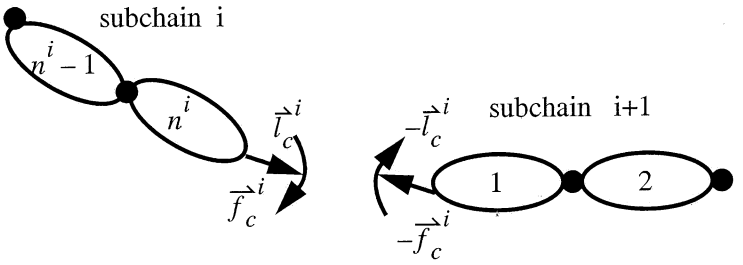


Figure 4: Constraint Forces between Subchains

where $\vec{\alpha}^{iB_k} \in \mathcal{R}^{3\times 1}$ and $\vec{a}^{i\binom{J_k}{B_k}} \in \mathcal{R}^{3\times 1}$ are the angular acceleration of body k and the acceleration of the inboard joint point k^+ of body k of subchain i, respectively.

To obtain the explicit expression form of $\underline{\ddot{q}}_I$ as in equation (3a), we divide $\underline{\mathcal{A}}^{iB_k}$ into two parts in the following equation

$$\underline{\mathcal{A}}^{iB_k} = \underline{\tilde{\mathcal{A}}}^{iB_k} + \underline{\mathcal{A}}_t^{iB_k} \tag{5}$$

where $\underline{\tilde{\mathcal{A}}}^{iB_k}$ is that portion of $\underline{\mathcal{A}}^{iB_k}$ expressed explicitly in the unknown second derivatives of generalized coordinates $\ddot{q}_I$, while $\underline{\mathcal{A}}_t^{iB_k}$ represents all of the remaining acceleration terms.

From recursive kinematic relations of $O(n)$ methods (Anderson [7], Rosenthal [8]), we can develop the following kinematic equations recursively body by body outward from base body 1 to terminal body n^i, specifically,

$$\underline{\mathcal{A}}^{iB_k} = \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \underline{\tilde{\mathcal{A}}}^{iB_{k-1}} + \underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \cdot \underline{\ddot{q}}^{ik} + \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \underline{\mathcal{A}}_t^{iB_{k-1}} + {}^{iB_{k-1}}\underline{\mathcal{A}}_t^{iB_k} \tag{6}$$

where, by definition,

$$\underline{\mathcal{P}}^{i\binom{J_k}{B_k}} = \begin{bmatrix} \vec{\omega}_r^{iB_k} \\ \vec{v}_r^{\,i\binom{J_k}{B_k}} \end{bmatrix} \tag{7}$$

$${}^{iB_{k-1}}\underline{S}^{iB_k} = \begin{bmatrix} \underline{U} & {}^{iB_{k-1}}\underline{\gamma}^{i\binom{J_{k-1}}{B_{k-1}}\binom{J_k}{B_k}\times} \\ \underline{0} & \underline{U} \end{bmatrix} \tag{8}$$

In equation (6), ${}^{iB_{k-1}}\underline{\mathcal{A}}_t^{iB_k}$ represents all of the terms of relative acceleration matrix of body k with respect to body $k-1$ of subchain i which are not explicit in $\underline{\ddot{q}}^{ik}$. $\underline{\ddot{q}}^{ik}$ is matrix of unknown second time derivatives of the generalized coordinates associated with the joint connecting body $(k-1)$ to body k of i^{th} subchain. In equations (7) and (8), $\underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \in \mathcal{R}^{6\times\nu^{ik}}$ represents the *free mode of motion* (Schwertassek *et al.* [19]) of joint k of subchain i. Here ν^{ik} is the number of degrees of freedom associated with the joint which connects body $(k-1)$ to body k in subchain i. ${}^{iB_{k-1}}\underline{S}^{iB_k} \in \mathcal{R}^{6\times 6}$ is a linear transformation "*shift matrix*" (transforming vector or dyadic quantities associated with one point to equivalent system of quantities associated with another point of same body), and $\underline{U} \in \mathcal{R}^{3\times 3}$ is an identity matrix. Finally, ${}^{iB_{k-1}}\underline{\gamma}^{i\binom{J_{k-1}}{B_{k-1}}\binom{J_k}{B_k}\times} \in \mathcal{R}^{3\times 3}$ represents a skew matrix, which is the matrix representation of the vector cross product operator between the joint position vector ${}^{iB_{k-1}}\vec{\gamma}^{i\binom{J_{k-1}}{B_{k-1}}\binom{J_k}{B_k}}$ and the matrix representation of any other vector.

4.2 Shifting applied and inertia forces

With the shift matrix so specified, the system of forces $\vec{f}^{i\binom{J_k}{B_k}}$ and moments $\vec{l}^{i\binom{J_k}{B_k}}$ acting at joint point k^+ of body k can be transformed to an equivalent systems of forces $\vec{f}^{i\binom{J_{k-1}}{B_{k-1}}}$ and moments $\vec{l}^{i\binom{J_{k-1}}{B_{k-1}}}$ acting at joint point $(k-1)^+$ of body $k-1$ by the relation

$$\underline{R}^{i\binom{J_{k-1}}{B_{k-1}}} = \begin{bmatrix} \underline{U} & {}^{iB_{k-1}}\underline{\gamma}^{i\binom{J_{k-1}}{B_{k-1}}\binom{J_k}{B_k}\times} \\ \underline{0} & \underline{U} \end{bmatrix} \cdot \underline{R}^{i\binom{J_k}{B_k}} \tag{9}$$

with $\underline{R}^{i\binom{J_k}{B_k}}$ defined as

$$\underline{R}^{i\binom{J_k}{B_k}} = \begin{bmatrix} \sum \vec{l}^{i\binom{J_k}{B_k}} \\ \sum \vec{f}^{i\binom{J_k}{B_k}} \end{bmatrix} \qquad \in \mathcal{R}^{6\times 1} \tag{10}$$

Similarly, we can transform the *inertia matrix* ${}^{iB_k}\underline{I0}^{i\bar{B}_k} \in \mathcal{R}^{6\times 6}$ of body k with respect to its center of mass, to the inboard joint point k^+ via the parallel axis theorem.

$${}^{iB_k}\underline{I1}^{i\binom{J_k}{B_k}} \underset{=}{\triangle} {}^{iB_k}\underline{S1}^{i\binom{J_k}{B_k}\bar{B}_k} \cdot {}^{iB_k}\underline{I0}^{i\bar{B}_k} \cdot ({}^{iB_k}\underline{S1}^{i\binom{J_k}{B_k}\bar{B}_k})^T \qquad \in \mathcal{R}^{6\times 6} \tag{11}$$

where, by similar definition,

$${}^{iB_k}\underline{S1}^{i\binom{J_k}{B_k}\bar{B}_k} \underset{=}{\triangle} \begin{bmatrix} \underline{U} & {}^{iB_k}\underline{r}^{i\binom{J_k}{B_k}\bar{B}_k\times} \\ \underline{0} & \underline{U} \end{bmatrix} \tag{12}$$

In equation (11), ${}^{iB_k}\underline{I1}^{i\binom{J_k}{B_k}} \in \mathcal{R}^{6\times 6}$ is the inertia moment matrix of body k associated with joint point k^+ on body k. ${}^{iB_k}\underline{r}^{i\binom{J_k}{B_k}\bar{B}_k\times} \in \mathcal{R}^{3\times 3}$, in equation (12), is another skew matrix, which is the matrix representation of the vector cross product operator between the position vector ${}^{iB_k}\vec{r}^{i\binom{J_k}{B_k}\bar{B}_k}$ from inboard joint point k^+ of body k to the center of mass of body k, and another vector.

At this time, according to equations (5), (10) and (11), we can define the force matrix for body k of the subchain i in the form

$$\underline{F}^{iB_k} = \underline{R}^{i\binom{J_k}{B_k}} - \left({}^{iB_k}\underline{I1}^{i\binom{J_k}{B_k}} \cdot \underline{\mathcal{A}}_t^{iB_k} + \underline{F}_t^{iB_k *} \right) \in \mathcal{R}^{6\times 1} \tag{13}$$

where $\underline{F}_t^{iB_k *}$ is inertia force remainder term matrix resulted from centripetal and Coriolis accelerations. Similarly, the constraint force matrix with respect to joint point k^+ of body k of subchain i can be defined by

$$\underline{F}_c^{iB_k} = \begin{bmatrix} \sum \vec{l}_c^{i\binom{J_k}{B_k}} \\ \sum \vec{f}_c^{i\binom{J_k}{B_k}} \end{bmatrix} \qquad \in \mathcal{R}^{6\times 1} \tag{14}$$

4.3 Triangularization of equations

Through kinematic relations and shifting procedures, we get the basic mathematical relations for the acceleration matrix, the inertia matrices and force matrices. Now we can use them for the recursive triangularization of "Kane's dynamic equations" [43], $\mathcal{F}^{ik} + \mathcal{F}^{ik*} = 0$. From Kane's dynamic equation and $O(n)$ method developed by Anderson [7], we will have

$$\underline{\mathcal{F}}^{ik} + \underline{\mathcal{F}}^{ik*} \equiv (\underline{\mathcal{P}}^{i\binom{J_k}{B_k}})^T \cdot (-{}^{iB_k}\hat{\underline{I}}^{i\binom{J_k}{B_k}} \cdot \underline{\tilde{\mathcal{A}}}^{iB_k} + \underline{\hat{F}}^{iB_k} + \underline{\hat{F}}_c^{iB_k}) = \underline{0} \qquad \in \mathcal{R}^{\nu^{ik}\times 1} \tag{15}$$

The matrices $\underline{\mathcal{F}}^{ik}$ and $\underline{\mathcal{F}}^{ik*}$ represent the generalized active forces and generalized inertia forces associated with the time derivative of generalized coordinates which are used to describe the degrees of freedom of inboard joint k of body k of subchain i.

In equation (15), composite inertia matrix ${}^{iB_k}\hat{\underline{I}}^{i\binom{J_k}{B_k}}$, composite force matrix $\hat{\underline{F}}^{iB_k}$ and composite constraint force matrix $\hat{\underline{F}}_c^{iB_k}$ are obtained by using inward recursive triangularization of $O(n)$ methods (Anderson [7]). They are expressed in the following form

$$ {}^{iB_k}\hat{\underline{I}}^{i\binom{J_k}{B_k}} = {}^{iB_k}\underline{I}^{i\binom{J_k}{B_k}} + {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot {}^{iB_{k+1}}\hat{\underline{I}}^{i\binom{J_{k+1}}{B_{k+1}}} \cdot \left({}^{iB_k}\underline{S}^{iB_{k+1}}\right)^T \qquad \in \mathcal{R}^{6\times 6} \tag{16} $$

$$ \hat{\underline{F}}^{iB_k} = \underline{F}^{iB_k} + {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot \hat{\underline{F}}^{iB_{k+1}} \qquad \in \mathcal{R}^{6\times 1} \tag{17} $$

$$ \hat{\underline{F}}_c^{iB_k} = \underline{F}_c^{iB_k} + {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot \hat{\underline{F}}_c^{iB_{k+1}} \qquad \in \mathcal{R}^{6\times 1} \tag{18} $$

where ${}^{iB_{k-1}}\underline{\mathcal{T}}^{iB_k}$ is the triangularization matrix which performs the task of recursively triangularizing the system mass matrix $\underline{M}$ in equation (3a) as it is being formed. According to $O(n)$ method (Anderson [7], Rosenthal [8]), this matrix can be defined as

$$ {}^{iB_{k-1}}\underline{\mathcal{T}}^{iB_k} = {}^{iB_k}\underline{S}^{iB_{k+1}} \cdot \left\{ \underline{U} - \frac{1}{M_{kk}^i} \cdot {}^{iB_k}\hat{\underline{I}}^{i\binom{J_k}{B_k}} \cdot \underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \cdot \left(\underline{\mathcal{P}}^{i\binom{J_k}{B_k}}\right)^T \right\} \in \mathcal{R}^{6\times 6} \tag{19} $$

In equation (19),

$$ M_{kk}^i = \left(\underline{\mathcal{P}}^{i\binom{J_k}{B_k}}\right)^T \cdot {}^{iB_k}\hat{\underline{I}}^{i\binom{J_k}{B_k}} \cdot \underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \tag{20} $$

where M_{kk}^i is the diagonal element of the system mass matrix $\underline{M}$ in equation (3a) associated with the degree of freedom currently being considered.

For subchain i, there are no constraint forces acting on any body except for the terminal and base bodies. In other words, if $k \neq$ base body 1 or terminal body n^i, then from equation (18), we have

$$ \hat{\underline{F}}_c^{iB_k} = {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot \hat{\underline{F}}_c^{iB_{k+1}} \qquad \in \mathcal{R}^{6\times 1} \tag{21} $$

To use sequential $O(n)$ procedure, massless auxiliary bodies or frame may be added to describe floating joint on base body 1 of subchain i due to cutting (Anderson [7], Rosenthal [8]), then body number k is mapping from $k = 1, ..., n^i$ to $k = 1, ..., \nu^i$ (ν^i is total degrees of freedom of subchain i). Now recursively using equation (21), we have

$$ \hat{\underline{F}}_c^{iB_k} = {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot \ldots \cdot {}^{iB_5}\underline{\mathcal{T}}^{iB_6} \cdot \underline{F}_c^{iB_6} + {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot \ldots \cdot {}^{iB_{\nu^i-1}}\underline{\mathcal{T}}^{iB_{\nu^i}} \cdot \underline{F}_c^{iB_{\nu^i}} \in \mathcal{R}^{6\times 1} \qquad (k < 6) \tag{22a} $$

and

$$ \hat{\underline{F}}_c^{iB_k} = {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot \ldots \cdot {}^{iB_{\nu^i-1}}\underline{\mathcal{T}}^{iB_{\nu^i}} \cdot \underline{F}_c^{iB_{\nu^i}} \in \mathcal{R}^{6\times 1} \qquad (k > 6) \tag{22b} $$

Similarly, with the definition of shift matrix ${}^{iB_{k-1}}\underline{S}^{iB_k}$, direction cosine matrix ${}^{iB_{k-1}}\underline{C1}^{iB_k}$ and Boolean matrix $\underline{E}^{i(\nu^i+1)}$, we can shift , transform and expand matrix of constraint force measure number at joint basis to matrix representing constraint load vectors at body basis as follow

$$ \hat{\underline{F}}_c^{iB_k} = \underline{\tau}_b^{ik} \cdot \underline{f}_c^{i6} + \underline{\tau}_t^{ik} \cdot \underline{f}_c^{i(\nu^i+1)} \in \mathcal{R}^{6\times 1} \qquad (k < 6) \tag{23a} $$

$$ \hat{\underline{F}}_c^{iB_k} = \underline{\tau}_t^{ik} \cdot \underline{f}_c^{i(\nu^i+1)} \in \mathcal{R}^{6\times 1} \qquad (k > 6) \tag{23b} $$

where

$$ \underline{\tau}_b^{ik} = {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot \underline{\tau}_b^{i(k+1)} \in \mathcal{R}^{6\times 5} \qquad (k < 6) \tag{24a} $$

$$\underline{\tau}_t^{ik} = {}^{iB_k}\underline{\mathcal{T}}^{iB_{k+1}} \cdot \underline{\tau}_t^{i(k+1)} \in \mathcal{R}^{6\times 5} \tag{24b}$$

with the boundary condition,

$$\underline{\tau}_b^{i6} = {}^{iB_5}\underline{S}^{iB_6} \cdot {}^{iB_5}\underline{C1}^{iB_6} \cdot \underline{E}^{i(\nu^i+1)} \in \mathcal{R}^{6\times 5} \tag{25a}$$

$$\underline{\tau}_t^{i\nu^i} = {}^{iB_{\nu^i}}\underline{S}^{iB_{\nu^i+1}} \cdot {}^{iB_{\nu^i}}\underline{C1}^{iB_{\nu^i+1}} \cdot \underline{E}^{i(\nu^i+1)} \in \mathcal{R}^{6\times 5} \tag{25b}$$

4.4 Back substitution

During the procedure of the triangularization, within each subchain the inward sweeping is recursively performed from terminal body to base body. Once the base body is reached, we can obtain its portion of generalized acceleration which has resulted from non-constraint forces. Then starting with the base body, we proceed outward to get the portion of generalized acceleration, η^{ik} associated with the unconstrained system (i.e. if constraint forces f_c were not applied), of each body. For the k^{th} degree of freedom ($k = 1, 2, ..., \nu^i$), the total generalized acceleration can be expressed in the form

$$\ddot{q}^{ik} = \eta^{ik} + \zeta^{ik} \qquad (k = 1, 2, ..., \nu^i) \tag{26}$$

where scalar quantity ζ^{ik} is that part of generalized acceleration which results from the unknown constraint forces. The scalar quantity η^{ik} can be determined using the recursive relations (Anderson [7])

$$\eta^{ik} = \frac{\left(\underline{\mathcal{P}}^{i\left(J_k\right)}_{\left(B_k\right)}\right)^T}{M^i_{kk}} \cdot \left[-{}^{iB_k}\underline{\hat{I}}^{i\left(J_k\right)}_{\left(B_k\right)} \cdot \left({}^{iB_k}\underline{S}^{iB_{k+1}}\right)^T \cdot \underline{\tilde{\eta}}^{i(k-1)} + \underline{\hat{F}}^{iB_k}\right] \qquad (k = 1, 2, ..., \nu^i) \tag{27}$$

and

$$\underline{\tilde{\eta}}^{ik} = \left({}^{iB_k}\underline{S}^{iB_{k+1}}\right)^T \cdot \underline{\tilde{\eta}}^{i(k-1)} + \underline{\mathcal{P}}^{i\left(J_k\right)}_{\left(B_k\right)} \cdot \eta^{ik} \qquad (k = 1, 2, ..., \nu^i) \tag{28}$$

with boundary condition $\underline{\tilde{\eta}}^{i0} = \underline{0} \in \mathcal{R}^{6\times 1}$.

Similarly, the scalar quantity ζ^{ik} can be determined using the recursive relations (Anderson [7])

$$\zeta^{ik} = \frac{\left(\underline{\mathcal{P}}^{i\left(J_k\right)}_{\left(B_k\right)}\right)^T}{M^i_{kk}} \cdot \left[-{}^{iB_k}\underline{\hat{I}}^{i\left(J_k\right)}_{\left(B_k\right)} \cdot \left({}^{iB_k}\underline{S}^{iB_{k+1}}\right)^T \cdot \underline{\tilde{\zeta}}^{i(k-1)} + \underline{\hat{F}}_c^{iB_k}\right] \qquad (k = 1, 2, ..., \nu^i) \tag{29}$$

and

$$\underline{\tilde{\zeta}}^{ik} = \left({}^{iB_k}\underline{S}^{iB_{k+1}}\right)^T \cdot \underline{\tilde{\zeta}}^{i(k-1)} + \underline{\mathcal{P}}^{i\left(J_k\right)}_{\left(B_k\right)} \cdot \zeta^{ik} \qquad (k = 1, 2, ..., \nu^i) \tag{30}$$

with boundary condition $\underline{\tilde{\zeta}}^{i0} = \underline{0} \in \mathcal{R}^{6\times 1}$.

If subchain i's base body interacts with another subchain, then there are constraint forces and/or moments acting directly on this base body. In such a case the quantity $\underline{\tilde{\zeta}}^{i1}$ would contain constraint loads $\vec{f}_c^{\,i\left(J_{\nu^i+1}\right)}_{\left(B_{\nu^i+1}\right)}$ and $\vec{l}_c^{\,i\left(J_{\nu^i+1}\right)}_{\left(B_{\nu^i+1}\right)}$, which act on the subchain terminal body n^i and now act on base body 1 as equivalent system through the *shifting* and *triangularization* procedure. The constraint loads $\vec{f}_c^{\,i\left(J_6\right)}_{\left(B_6\right)}$ and $\vec{l}_c^{\,i\left(J_6\right)}_{\left(B_6\right)}$, by comparison act directly on the base body of the subchain i and result from interaction with the proximal subchain of the i^{th} subchain. So, substituting equations (23) into (29) and applying recursive relations, we obtain

$$\zeta^{ik} = \underline{D_b}^{iB_k} \cdot \underline{f}_c^{i6} + \underline{D_t}^{iB_k} \cdot \underline{f}_c^{in^i+1} \qquad (k = 1, 2, ..., \nu^i) \tag{31}$$

with

$$\underline{D_t}^{iB_k} = \frac{\left(\underline{\mathcal{P}}^{i\binom{J_k}{B_k}}\right)^T}{M^i_{kk}} \cdot \left[\underline{\tau}_t^{ik} - {}^{iB_k}\underline{\hat{I}}^{i\binom{J_k}{B_k}} \cdot \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \underline{\bar{D}_t}^{iB_{k-1}}\right] \qquad (k = 1, 2, ..., \nu^i) \tag{32}$$

and

$$\underline{D_b}^{iB_k} = \frac{\left(\underline{\mathcal{P}}^{i\binom{J_k}{B_k}}\right)^T}{M^i_{kk}} \cdot \left[\underline{\tau}_b^{ik} - {}^{iB_k}\underline{\hat{I}}^{i\binom{J_k}{B_k}} \cdot \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \underline{\bar{D}_b}^{iB_{k-1}}\right] \qquad (k = 1, 2, ..., \nu^i) \tag{33}$$

where

$$\underline{\bar{D}_t}^{iB_k} = \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \underline{\bar{D}_t}^{iB_{k-1}} + \underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \cdot \underline{D_t}^{iB_k} \qquad (\underline{\bar{D}_t}^{iB_0} = 0) \tag{34}$$

$$\underline{\bar{D}_b}^{iB_k} = \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \underline{\bar{D}_b}^{iB_{k-1}} + \underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \cdot \underline{D_b}^{iB_k} \qquad (\underline{\bar{D}_b}^{iB_0} = 0) \tag{35}$$

$$\underline{\tau}_b^{ik} = 0 \qquad (k > 6) \tag{36}$$

4.5 Assembling constraint force coefficient matrix

To insure that the subchain system's behavior (shown in figure 1(b)) remains as the original system (show in figure 1(a)), the displacement constraints must be enforced between the terminal body n^i of subchain i and the base body 1 of adjacent subchain $i+1$. To do so, we differentiate the displacement constraint equations (2c) twice with respect to time to yield the acceleration constraint equations

$$\underline{\mathcal{A}}^{(i+1)B_1} = \underline{\mathcal{A}}^{iB_{n^i+1}}. \tag{37}$$

Due to introduction of auxiliary bodies of sequential $O(n)$, discussed previously, we have the following constraint equations

$$\underline{\mathcal{A}}^{(i+1)B_6} = \underline{\mathcal{A}}^{iB_{\nu^i+1}}. \tag{38}$$

From (6) and (38), we have

$$\left({}^{iB_5}\underline{S}^{iB_6}\right)^T \cdot \underline{\tilde{\mathcal{A}}}^{(i+1)B_5} + \underline{\mathcal{P}}^{(i+1)\binom{J_6}{B_6}} \cdot \ddot{q}^{(i+1)6} + \underline{\mathcal{A}}_t^{(i+1)B_6} =$$

$$\left({}^{iB_{\nu^i}}\underline{S}^{iB_{\nu^i+1}}\right)^T \cdot \underline{\tilde{\mathcal{A}}}^{iB_{\nu^i}} + \underline{\mathcal{P}}^{i\binom{J_{\nu^i+1}}{B_{\nu^i+1}}} \cdot \ddot{q}^{i(\nu^i+1)} + \underline{\mathcal{A}}_t^{iB_{\nu^i+1}}. \tag{39}$$

Matrix $\underline{q}^{i(\nu^i+1)}$ represents the generalized coordinates which describe the rigid body degrees of freedom of the base body of subchain $i+1$ relative to the terminal body ν^i of subchain i. They are degrees of freedom allowed by the joint ν^i+1, which would connect these two bodies in the original system, and which was cut so to produce subchain $i+1$. These generalized coordinates $\underline{q}^{i(\nu^i+1)}$ form a redundant set of coordinates in the cut subchain system. Thus equations (26) and (39) represent a system of $n+m$ equations and $n+2m$ unknowns, where n is the total number of degree of freedom possessed by the unconstrained system of subchains and m is the total number of constraint loads which must act on the resulting subchain system to insure its behaves as the original un-cut system. To make this system of equations solvable it is necessary to eliminate redundant m unknowns from the system of equations given by (26) and (39). This can be achieved by eliminating unknown $\underline{q}^{i(\nu^i+1)}$ $(i = 1, 2, ..., Nsub-1)$ from all equations (39). Here $Nsub$ is the total number of subchains in the system of subchains. Specifically, we can accomplish this by multiplying both sides of equations (39) by the orthogonal complement to the free mode of motion matrix $\underline{\mathcal{P}}^{i\binom{J_{\nu^i+1}}{B_{\nu^i+1}}}$. Since

$$\underline{\mathcal{O}}^{i(\nu^i+1)} \cdot \underline{\mathcal{P}}^{i\binom{J_{\nu^i+1}}{B_{\nu^i+1}}} = \underline{\mathcal{O}}^{i(\nu^i+1)} \cdot \underline{\mathcal{P}}^{i\binom{J_6}{B_6}} = \underline{0} \tag{40}$$

multiplying equation (39) through by $\underline{\mathcal{O}}^{i(\nu^i+1)}$ yields

$$\underline{\mathcal{O}}^{i(\nu^i+1)} \cdot \left({}^{iB_5}\underline{S}^{(i+1)B_6}\right)^T \cdot \underline{\tilde{\mathcal{A}}}^{(i+1)B_5} + \underline{\mathcal{A}}_t^{(i+1)B_6}) = \underline{\mathcal{O}}^{i(\nu^i+1)} \cdot (({}^{iB_{\nu^i}}\underline{S}^{iB_{\nu^i+1}})^T \cdot \underline{\tilde{\mathcal{A}}}^{iB_{\nu^i}} + \underline{\mathcal{A}}_t^{iB_{\nu^i+1}}). \tag{41}$$

From (5) and (6), we have

$$\underline{\tilde{\mathcal{A}}}^{iB_k} = \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \underline{\tilde{\mathcal{A}}}^{iB_{k-1}} + \underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \cdot \ddot{q}^{ik}. \tag{42}$$

Recursively applying (42) to $\underline{\tilde{\mathcal{A}}}^{iB_{\nu^i}}$ and $\underline{\tilde{\mathcal{A}}}^{(i+1)B_5}$ with $\underline{\tilde{\mathcal{A}}}^{iB_0} = 0$ and $\underline{\tilde{\mathcal{A}}}^{(i+1)B_0} = 0$, we will have

$$\underline{\tilde{\mathcal{A}}}^{iB_k} = \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \left({}^{iB_{k-2}}\underline{S}^{iB_{k-1}}\right)^T \cdot \ldots \cdot \left({}^{iB_1}\underline{S}^{iB_2}\right)^T \cdot \underline{\mathcal{P}}^{i\binom{J_1}{B_1}} \cdot \ddot{q}^{i1} + \left({}^{iB_{k-1}}\underline{S}^{iB_k}\right)^T \cdot \left({}^{iB_{k-2}}\underline{S}^{iB_{k-1}}\right)^T$$

$$\cdot\ldots \cdot \left({}^{iB_2}\underline{S}^{iB_3}\right)^T \cdot \underline{\mathcal{P}}^{i\binom{J_2}{B_2}} \cdot \ddot{q}^{i2} + \ldots + \underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \cdot \ddot{q}^{ik} \qquad (k = 1, 2, \ldots, \nu^i) \tag{43}$$

$$\underline{\tilde{\mathcal{A}}}^{(i+1)B_5} = \left({}^{(i+1)B_4}\underline{S}^{(i+1)B_5}\right)^T \cdot \left({}^{(i+1)B_3}\underline{S}^{(i+1)B_4}\right)^T \cdot \ldots \cdot \left({}^{(i+1)B_1}\underline{S}^{(i+1)B_2}\right)^T \cdot \underline{\mathcal{P}}^{(i+1)\binom{J_1}{B_1}} \cdot \ddot{q}^{(i+1)1}$$

$$+\ldots + \underline{\mathcal{P}}^{(i+1)\binom{J_5}{B_5}} \cdot \ddot{q}^{(i+1)k}. \tag{44}$$

Substituting (43) and (44) into (41) and replacing $u^(ik)$ with equation (5) and (31), we have

$$\left[\underline{G}_1^{i1}\ \underline{G}_1^{i2} \ldots \underline{G}_1^{i\nu^i}\right] \cdot \left(\begin{bmatrix} \eta^{i1} \\ \eta^{i2} \\ \vdots \\ \eta^{i\nu^i} \end{bmatrix} + \begin{bmatrix} \underline{D_b}^{iB_1} \cdot \underline{f}_c^{i6} \\ \underline{D_b}^{iB_2} \cdot \underline{f}_c^{i6} \\ \vdots \\ \underline{D_b}^{iB_\nu^i} \cdot \underline{f}_c^{i6} \end{bmatrix} + \begin{bmatrix} \underline{D_t}^{iB_1} \cdot \underline{f}_c^{i\nu^i} \\ \underline{D_t}^{iB_2} \cdot \underline{f}_c^{i\nu^i} \\ \vdots \\ \underline{D_t}^{iB_\nu^i} \cdot \underline{f}_c^{i\nu^i} \end{bmatrix} \right) + \underline{\mathcal{O}}^{i(\nu^i+1)} \cdot \underline{\mathcal{A}}_t^{iB_{\nu^i+1}} =$$

$$\left[\underline{G}_2^{(i+1)1}\ \underline{G}_2^{(i+1)2} \ldots \underline{G}_2^{(i+1)6}\right] \cdot \left(\begin{bmatrix} \eta^{(i+1)1} \\ \eta^{(i+1)2} \\ \vdots \\ \eta^{(i+1)6} \end{bmatrix} + \begin{bmatrix} \underline{D_b}^{(i+1)B_1} \cdot \underline{f}_c^{(i+1)6} \\ \underline{D_b}^{(i+1)B_2} \cdot \underline{f}_c^{(i+1)6} \\ \vdots \\ \underline{D_b}^{(i+1)B_6} \cdot \underline{f}_c^{(i+1)6} \end{bmatrix} + \begin{bmatrix} \underline{D_t}^{(i+1)B_1} \cdot \underline{f}_c^{(i+1)\nu^i} \\ \underline{D_t}^{(i+1)B_2} \cdot \underline{f}_c^{(i+1)\nu^i} \\ \vdots \\ \underline{D_t}^{(i+1)B_6} \cdot \underline{f}_c^{(i+1)\nu^i} \end{bmatrix} \right) +$$

$$\underline{\mathcal{O}}^{i(\nu^i+1)} \cdot \underline{\mathcal{A}}_t^{(i+1)B_6} \tag{45}$$

where

$$\underline{G}_1^{ik} = \underline{L}_1^{ik} \cdot \underline{\mathcal{P}}^{i\binom{J_k}{B_k}} \tag{46}$$

$$\underline{L}_1^{ik} = \underline{L}_1^{i(k+1)} \cdot \left({}^{iB_k}\underline{S}^{iB_{(k+1)}}\right)^T \tag{47}$$

$$\underline{L}_1^{i(\nu^i+1)} = \underline{\mathcal{O}}^{i(\nu^i+1)} \tag{48}$$

and

$$\underline{G}_2^{(i+1)k} = \underline{L}_2^{(i+1)k} \cdot \underline{\mathcal{P}}^{(i+1)\binom{J_k}{B_k}} \qquad (k < 7) \tag{49}$$

$$\underline{L}_2^{(i+1)k} = \underline{L}_2^{(i+1)(k+1)} \cdot \left({}^{(i+1)B_k}\underline{S}^{(i+1)B_{(k+1)}}\right)^T \qquad (k < 6) \tag{50}$$

$$\underline{L}_2^{(i+1)6} = \underline{\mathcal{O}}^{i(\nu^i+1)}. \tag{51}$$

According Newtonian third law, we know

$$\underline{f}_c^{i6} = -\underline{f}_c^{(i-1)\nu^{i-1}} \tag{52}$$

$$\underline{f}_c^{i\nu^i} = -\underline{f}_c^{(i+1)6}. \tag{53}$$

Substituting (52) and (53) into (45), we obtain constraint equation of cut joint i $(i = 1, ..., Nsub - 1)$ as follow

$$-\underline{\Gamma}_{i(i-1)} \cdot \underline{f}_c^{(i-1)(\nu^{(i-1)}+1)} + \underline{\Gamma}_{ii} \cdot \underline{f}_c^{i(\nu^i+1)} - \underline{\Gamma}_{i(i+1)} \cdot \underline{f}_c^{(i+1)(\nu^{(i+1)}+1)} = \underline{\Psi}_i \tag{54}$$

where

$$\underline{\Gamma}_{i(i-1)} = \underline{G1}^i \cdot \underline{D_b}^i \tag{55}$$

$$(\underline{\Gamma})_{ii} = \underline{G1}^i \cdot \underline{D_t}^i + \underline{G2}^{(i+1)} \cdot \underline{D_{bb}}^{(i+1)} \tag{56}$$

$$\underline{\Gamma}_{i(i+1)} = \underline{G2}^{(i+1)} \cdot \underline{D_{tb}}^{(i+1)} \tag{57}$$

$$(\underline{\Psi})_i = \underline{G2}^{(i+1)} \cdot \underline{\eta}_b^{(i+1)} - \underline{G1}^i \cdot \underline{\eta}^i + \underline{\mathcal{O}}^{i(n^i+1)} \cdot \left(\underline{\mathcal{A}}_t^{(i+1)B_6} - \underline{\mathcal{A}}_t^{iB_{n^i}} \right) \tag{58}$$

with

$$\underline{G1}^i = \left[\underline{G}_1^{i1} \; \underline{G}_1^{i2} \; ... \; \underline{G}_1^{i\nu^i} \right] \tag{59}$$

$$\underline{G2}^{(i+1)} = \left[\underline{G}_2^{(i+1)1} \; \underline{G}_2^{(i+1)2} \; ... \; \underline{G}_2^{(i+1)6} \right] \tag{60}$$

$$\underline{D_t}^i = \begin{bmatrix} \underline{D}_t^{iB_1} \\ \underline{D}_t^{iB_2} \\ \vdots \\ \underline{D}_t^{iB_\nu^i} \end{bmatrix} \tag{61}$$

$$\underline{D_b}^i = \begin{bmatrix} \underline{D}_b^{iB_1} \\ \underline{D}_b^{iB_2} \\ \vdots \\ \underline{D}_b^{iB_\nu^i} \end{bmatrix} \tag{62}$$

$$\underline{D_{tb}}^{(i+1)} = \begin{bmatrix} \underline{D}_t^{(i+1)B_1} \\ \underline{D}_t^{(i+1)B_2} \\ \vdots \\ \underline{D}_t^{(i+1)B_6} \end{bmatrix} \tag{63}$$

$$\underline{D_{bb}}^{(i+1)} = \begin{bmatrix} \underline{D}_b^{(i+1)B_1} \\ \underline{D}_b^{(i+1)B_2} \\ \vdots \\ \underline{D}_b^{(i+1)B_6} \end{bmatrix}. \tag{64}$$

In this way, cut joint by cut joint we can assemble the system of linear equations for the constraint load magnitudes of the entire system of subchains. All m of the constraint load magnitudes $\underline{f_c} \in \Re^{m \times 1}$ acting on each of the subchains for the whole of the cut system can be expressed as follows

$$\underline{\Gamma} \cdot \underline{f_c} = \underline{\Psi} \tag{65}$$

where matrices $\underline{\Gamma}$ and $\underline{\Psi}$ have the exact meaning as in equation (3).

5 New Hybrid Algorithm

The parallel implementation of this new hybrid algorithm is briefly presented in the following pseudo code

Step I. Preprocessing

For i=1 to Nsub (do parallel by all processors)

♠ For k=1 to ν^i (do recursively for each body)

1) Compute all inertia momentum matrix, such as ${}^{iB_k}\underline{I1}^{i\binom{J_k}{B_k}}$ from (11) and (12).

2) Compute all external force matrix and shift them to inboard joint from (10).

3) Compute free mode of motion $\underline{\mathcal{P}}^{i\binom{J_k}{B_k}}$ from (7) and its orthogonal complement $\underline{\mathcal{O}}^{i(\nu^i+1)}$ algorithm in [44].

4) Map constant variables between global chain and local subchains.

♠ End do

End do parallel

Step II. Mapping from global to local

if $t \neq 0$

For i=1 to Nsub-1 (do parallel by 1:Nsub-1 processors)

1) Transform $\vec{\omega}^{iB_{\nu^i}}$ and $\vec{\omega}^{(i+1)B_6}$ from body basis to Newtonian frame.

2) Send transformed $\vec{\omega}^{iB_{\nu^i}}$ of processor i to processor $i+1$.

3) Compute $u^{(i+1)(\nu^i+1)}$ and $q^{(i+1)(\nu^i+1)}$

End do parallel

Endif

Step III. Map variables changing with time between global chain and local subchains

For i=1 to Nsub (do parallel by all processors)

♠ For k=1 to ν^i (do recursively for each body)

1) Map state variables between global chain and local subchains.

2) Compute cosine transformation matrices ${}^{iB_{k-1}}\underline{c}^{iB_k}$, ${}^{iB_{k-1}}\underline{C1}^{iB_k}$, ${}^{iB_0}\underline{c}^{iB_k}$ and ${}^{iB_0}\underline{C1}^{iB_k}$.

3) Form skew matrix ${}^{iB_{k-1}}\underline{\gamma}^{i\binom{J_{k-1}}{B_{k-1}}}$ and shift matrix ${}^{iB_{k-1}}\underline{S}^{iB_k}$ for joint position vector ${}^{iB_{k-1}}\vec{\gamma}^{i\binom{J_{k-1}}{B_{k-1}}\binom{J_k}{B_k}}$ from (8).

4) Form multiplication of shift matrix and cosine transformation matrix ${}^{iB_{k-1}}\underline{S}^{iB_k} \cdot {}^{iB_{k-1}}\underline{C1}^{iB_k}$.

5) Transform $\underline{R}^{i\binom{J_k}{B_k}}$ from Newtonian frame to body k basis.

♠ End do

End do parallel

Step IV. The first outward sweeping to form recursive kinematic parameters

For i=1 to Nsub (do parallel by all processors)

♠ For k=1 to ν^i (do recursively for each body)

1) Compute $\vec{\omega}^{iB_k}$ and $\vec{v}^{i\binom{J_k}{B_k}}$ with $\vec{\omega}^{iB_0} = 0$ and $\vec{v}^{i\binom{J_0}{B_0}} = 0$ to form velocity matrix $\underline{V}^{iB_k}$ from $O(n)$ algorithm in [14].

2) Compute $\underline{\mathcal{A}}_t^{iB_k}$ and ${}^{iB_{k-1}}\underline{\mathcal{A}}_t^{iB_k}$ with $\underline{\mathcal{A}}_t^{iB_0} = 0$ from $O(n)$ algorithm in [14].

3) Compute $\underline{F}_t^{iB_k*}$ from algorithms in [27].

4) Compute $\underline{F}^{iB_k}$ from (13).

5) Compute multiplications of $\underline{\mathcal{O}}^{(i-1)(\nu^{(i-1)}+1)} \cdot \underline{\mathcal{A}}_t^{iB_6}$ and $\underline{\mathcal{O}}^{i(\nu^i+1)} \cdot \underline{\mathcal{A}}_t^{iB_{\nu^i+1}}$.

♠ End do

End do parallel

Step V. The inward sweeping for recursive triangularization to form kinetic parameters

For i=1 to Nsub (do parallel by all processors)

♠ For k= ν^i to 1 (do recursively for each body)

1) Compute ${}^{iB_k}\underline{\hat{I}}^{i\binom{J_k}{B_k}}$ with ${}^{iB_{\nu i}}\underline{\hat{I}}^{i\binom{J_{\nu^i}}{B_{\nu^i}}} = {}^{iB_{\nu i}}\underline{I}^{i\binom{J_{\nu^i}}{B_{\nu^i}}}$ from (16).

2) Compute $\underline{\hat{F}}^{iB_k}$ with $\underline{\hat{F}}^{iB_{\nu^i}} = \underline{F}^{iB_{\nu^i}}$ from (17).

3) Compute M^i_{kk} from (20).

4) Compute ${}^{iB_{k-1}}\underline{\mathcal{T}}^{iB_k}$ from (19).

5) Compute $\underline{\tau}_b^{ik}$ and $\underline{\tau}_t^{ik}$ from (24) and (25).

6) Compute $\underline{L}_1^{ik}$ and $\underline{L}_2^{(i+1)k}$ from (47), (48), (50) and (51).

7) Compute $\underline{G}_1^{ik}$ and $\underline{G}_2^{(i+1)k}$ from (46) and (49).

♠ End do

End do parallel

Step VI. The second outward sweeping for back substitution to form equations (3)

For i=1 to Nsub (do parallel by all processors)

♠ For k= 1 to ν^i (do recursively for each body)

1) Compute $\underline{\tilde{\eta}}^{ik}$ with $\underline{\tilde{\eta}}^{i0} = \underline{0}$ from (28).

2) Compute $\underline{\eta}^{ik}$ from (27).

3) Compute $\underline{D_t}^{iB_k}$ from (32).

4) Compute $\underline{D_b}^{iB_k}$ from (33).

5) Compute $\underline{\bar{D}_t}^{iB_k}$ and $\underline{\bar{D}_b}^{iB_k}$ from (34) and (35).

6) Compute $\underline{\Gamma}_{i(i-1)}$ from (55)

7) Compute multiplication of $\underline{G1}^i \cdot \underline{D_t}^i$.

8) Compute multiplication of $\underline{G2}^i \cdot \underline{D_{bb}}^i$.

9) Compute $\underline{\Gamma}_{(i-1)i}$ from (57).

10) Compute multiplication of $\underline{G2}^i \cdot \underline{\eta}_b^i$.

11) Compute multiplication of $\underline{G1}^i \cdot \underline{\eta}^i$.

♠ End do

End do parallel

Step VII. Assembling the constraint equations

For i=1 to Nsub-1 (do parallel by processors 1 to Nsub-1)

1) Send $\underline{\mathcal{O}}^{i(\nu^i+1)} \cdot \underline{\mathcal{A}}_t^{(i+1)B_6}$, $\underline{\Gamma}_{i(i+1)}$, $\underline{G2}^{(i+1)} \cdot \underline{\eta}_b^{(i+1)}$, and $\underline{G2}^{(i+1)} \cdot \underline{D_{bb}}^{(i+1)}$ to processor i.

2) Compute $(\underline{\Gamma})_{ii}$ from (56).

3) Compute $(\underline{\Psi})_i$ from (58).

♠ End do

End do parallel

Step VIII. Solving the constraint equations

At first direct solver such as LU decomposition is used to sovle (65) to obtain initial values of $\underline{f}_c$, then preconditional conjugate gradient algorithm in [45] is used to solve the constraint equation in parallel. The procedure is expressed as following

if t==0

1) solve $\underline{f}_c$ from (65) by LU decompostion.

2) compute residual $\underline{r}_0 = \underline{\Psi} - \underline{\Gamma} \cdot \underline{f}_c$

3) solve $\underline{\Gamma}_d \cdot \underline{z}_0 = \underline{r}_0$ by LU decompostion.

4) assign $\underline{p}_0 = \underline{z}_0$

5) MPI-SCATTER $(\underline{r_0})_i$, $(\underline{z_0})_i$ and $(\underline{p_0})_i$ from root processor to processor i.

else

For j=0, 1, ... till convergence do (iteration)

For i=1 to Nsub (do parallel by all processors)

6) computer dot product of $[(\underline{r_j})_i, (\underline{z_j})_i]$ and $[(\underline{\Gamma})_i \cdot (\underline{p_j})_{i3}, (\underline{p_j})_{i1}]$

7) MPI-ALLREDUCE both products in 6).

8) compute $\alpha_j = \frac{[\underline{r}_j, \underline{z}_j]}{[(\underline{\Gamma})_j \cdot \underline{p}_j, \underline{p}_j]}$.

9) compute $(\underline{r_{(j+1)}})_i = (\underline{r_j})_i - \alpha_j \cdot (\underline{\Gamma})_i \cdot (\underline{p_j})_{i3}$

10) solve $\underline{\Gamma}_d \cdot (\underline{z_{(j+1)}})_i = (\underline{r_{(j+1)}})_i$ by LU decompostion.

11) MPI-ALLREDUCE dot product of $[\underline{r}_{(j+1)}, \underline{z}_{(j+1)}]$.

12) compute $\beta_j = \frac{[\underline{r}_{(j+1)}, \underline{z}_{(j+1)}]}{[\underline{r}_j, \underline{z}_j]}$.

13) compute $(\underline{p_{(j+1)}})_{i1} = (\underline{z}_{(j+1)})_i + \beta_j \cdot (\underline{p_j})_{i1}$.

11) MPI-GATHER processor i get $\underline{p_{(j+1)}}$ from processor $i-1$ and $i+1$.

End do parallel

End iteration

Endif

Step IX. The generalized accelerations

For i=1 to Nsub (do parallel by all processors)

♠ For k= 1 to ν^i (do recursively for each body)

1) Compute multiplication of $\underline{D_t}^{iB_k} \cdot \underline{f}_c^{i\nu^i}$.

2) Compute multiplication of $\underline{D_b}^{iB_2} \cdot \underline{f}_c^{(i-1)\nu^{(i-1)}}$

3) Compute $\ddot{q}^{ik}$ from (26).

♠ End do

End do parallel

Step X. The temporal integration

For i=1 to Nsub (do parallel by all processors)

♠ For k= 1 to ν^i (do recursively for each body)

1) Integrate $\ddot{q}^{ik}$ twice to update $\dot{q}^{ik}$ and q^{ik}.

♠ End do

End do parallel

The temporal integration includes the procedures from step II to step X.

The operational numbers can be minimized by optimizing operational counting of the new hybrid parallelizable $O(n)$ algorithm at different levels, such as avoiding duplicate calculations, taking advantages of sparseness and symmetry of some dynamic quantities, using body fixed coordinate systems and so on. These procedures of saving computational cost were used by Featherstone [4] and Fijany [37] respectively for the operational counting of their algorithms. Further, with the consideration of computational costs of shift matrix and cosine transformation matrix, we have operational counting of this algorithm for parallel implementation as shown in table 3.

$m-$ multiplication cost $\alpha-$ communication set-up cost

$a-$ addition cost $\beta-$ single floating point number communication cost

Table 3: Computational and communication costs of the new hybrid parallelizable $O(n)$ algorithm

Step	Computational cost	Communication cost
II	$(9m+12a)\frac{N}{Nsub}$	$\alpha+6\beta$
III	$(116m+102a)\frac{N}{Nsub}$	0
IV (1st out)	$(83m+68a)\frac{N}{Nsub}+(32m+22a)\frac{Nsub-1}{Nsub}$	0
V (inward)	$(309m+225a)\frac{N}{Nsub}+(248m+116a)\frac{Nsub-1}{Nsub}$ $-(94m+60a)\frac{N}{Nsub^2}$	0
VI (2nd out)	$(88m+72a)\frac{N}{Nsub}+(35m+25a)\frac{N}{Nsub^2}$ $+(180m+119a)\frac{Nsub-1}{Nsub}-(20m+16a)\frac{1}{Nsub}$	0
VII (assembling)	$40a\frac{Nsub-1}{Nsub}$	$\alpha+70\beta$
VIII (solving)	$162m+(150+3log_2Nsub)a$	$3\alpha log_2Nsub$ $+(6log_2Nsub+10)\beta$
IX (acceleration)	$(10m+10a)\frac{N}{Nsub}+(24m+26a)\frac{Nsub-1}{Nsub}$	0
total	$(615m+489a)\frac{N}{Nsub}+(484m+323a)\frac{Nsub-1}{Nsub}$ $-(59m+35a)\frac{N}{Nsub^2}-(20m+16a)\frac{1}{Nsub^2}$ $+162m+(150+3log_2Nsub)a$	$(2+3log_2Nsub)\alpha$ $+(6log_2Nsub+86)\beta$

6 Discussion and Conclusion

We have presented new hybrid parallelizable $O(n)$ algorithm. This novel procedure may be viewed as a hybrid between more traditional methods in a number of ways. The first aspect of hybridization is due to the fact that the procedure takes advantage of elements of improved computing efficiency from both parallel computations and the sequential *O(N)* algorithm to achieve the higher overall computing efficiency. Secondly the algorithm uses a hybrid direct and iterative solution scheme which allows a substantially higher degree of parallelization than is generally obtainable using the more conventional recursive *O(N)* procedures. Finally the dynamical formulation is a hybrid form of state space and descriptor representations, involving aspects of the recursive order-n and more tradition Newton-Euler approaches.

Due to these facts, the algorithm has several advantages for improvement of computational efficiency. The method permits the hybridization of parallel computation allowed by *full descriptor procedure* [11] and efficient sequential *O(N)* procedures in an optimal way. So it should more easily accommodate the available (often sub-optimal) number of processors ($N_p << N$) while maintaining higher efficiency than other parallel procedures.

If we limit our attention to chain systems, a chain system topology will produce a block tridiagonal coefficient matrix as shown in figure 5. This matrix will have some desirable properties and structure (sparseness, narrow bandwidth, positive definiteness), which allow us to develop an efficient hybrid direct-iterative method to form and solve equations of motion and constraint equations. Golub shows that narrow band width not only saves the storage but also reduces flop counts (a float point operation count [46]). The sparseness and narrow band width will be suitable for different Krylov iterative methods such as Preconditional Conjugate Gradient (PCG) for symmetric and positive definite systems (SPD), Generalized Minimal Residual (GMRES) for non-symmetric systems (NS) and Quasi-Minimal Residual (QMR) for non-positive definite systems (NPS). The sparseness and narrow band width associated with this formulation lend themselves well to the use of parallel data structure such as a block mapping and a cyclic mapping. These data structures combined with different Krylov iterative methods may form fewer parallel synchronization points so that communication cost will be less if a message passing parallel strategy

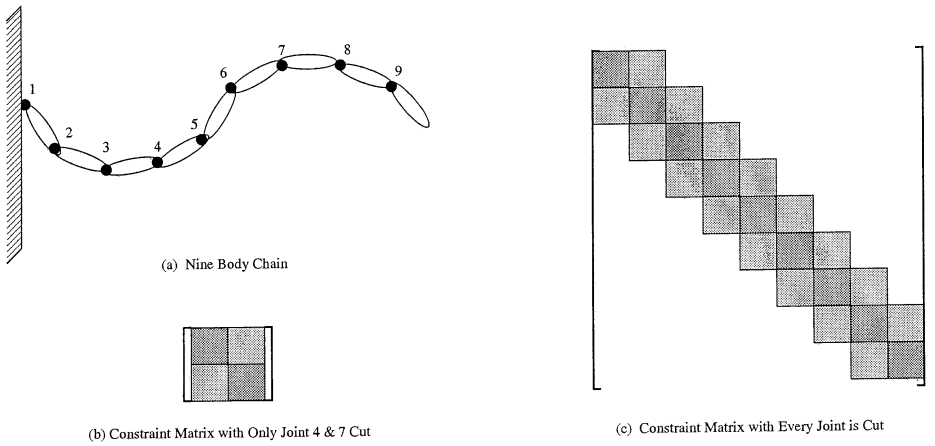


Figure 5: A Chain System

such as Message Passing Interface (MPI) is used. In the other words, the sparseness and narrow band width favors the combination of different parallel strategies with different iterative methods (Sadd[45], Quinn[47], Pacheco[48], Foster[49]). One advantage of this hybrid method is that just a few joints are cut as necessary, as opposed to all joints being cut in the *full descriptor representation.* The hybrid method is thus much less prone to a variety of numerical difficulties as the system of equations formed by the full descriptor method. In *full descriptor representation*, cutting each joint results in a large dimension of equations of (3). The large size of constraint equations (3b) can have a significant affect on the number of iterations required, due to the broader eigenvalue spectra of the system and thus adversely affect the performance of the simulation, if an iterative method is used. In addition, all algebraic constraints at all the joints of the system are enforced at the acceleration level rather than the displacement level, thus zero eigenvalues are introduced for each of these constraints. This will result in an eventual unstable growth in constraint violation error at each cut joint if the simulation is allowed to proceed and no constraint error stabilization scheme is used (Anderson [7]). Figure 6 shows the theoretically required simulation time versus the number of system dynamic degrees of freedom which can be expected using the traditional $O(n^3)$ approach, the sequential $O(n)$ approach, full descriptor $O(n^{1.12})$ or quasi $O(n)$ approach presented by Sharf *et al.*[33], and parallel $O(log_2 N_B)$ approach with approximately $O(N_B)$ processors available. The curve for the new hybrid parallel $O(n)$ algorithm will locate between range bounded by the curves of sequential $O(n)$ and parallel $O(log_2 N)$.

The implementation of this procedure in a general purpose simulation code, plus the development of improved special iterative parallel methods for efficiently solving for the constraint load magnitudes $\underline{f}_c$ are areas of ongoing work. Specifically, improved algorithms need to be developed for the parallel solution of (65) when the associated coefficient matrix is not symmetric positive definite and contains coupling terms well off the diagonal, such as can be expected in systems involving closed loops.

Acknowledgment

Authors wish to acknowledge and thank the National Science Foundation for its support of this work through award No. ECS-9596237.

References

[1] Armstrong, W. W., "Recursive Solution of the Equations of Motion of an N-link Manipulator", *Proceedings of the Fifth World Congress on the Theory of Machines and Mechanisms*, Montreal, Canada,

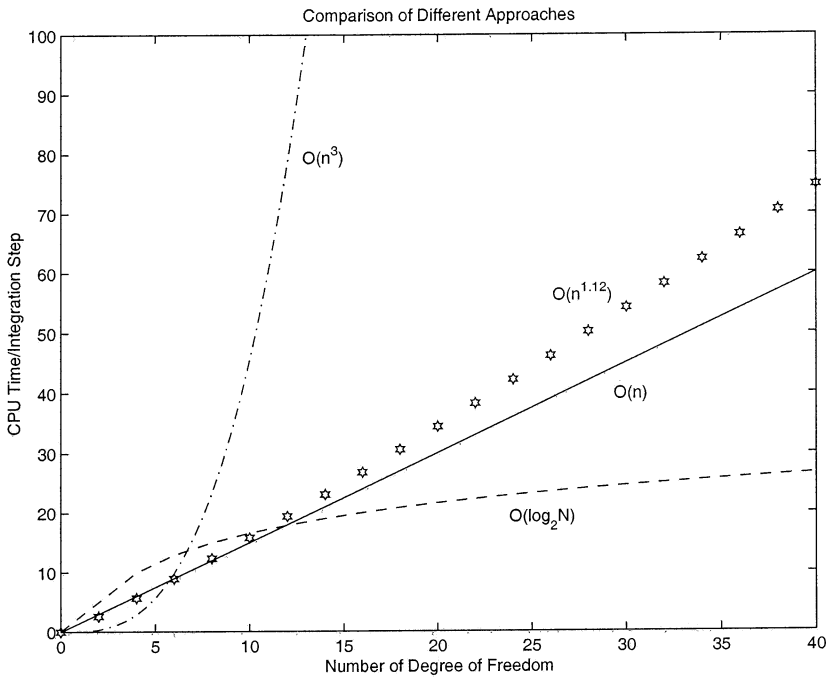


Figure 6: Comparison of Different Approaches

1979, Vol. 2, pp. 1342- 1346.

[2] Hollerbach, J. M.,"A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity," *IEEE Trans. Systems, Man, and Cybernetics*, Vol. SMC-10, No. 11, November 1980, pp. 730-736.

[3] Walker, M.W., and Orin, D. E., "Efficient Dynamic Computer Simulation of Robotic Mechanisms", *Journal of Dynamic Systems, Measurements, and Control*, Vol. 104, Sept. 1982, pp. 205-211.

[4] Featherstone, R., "The Calculation of Robotic Dynamics Using Articulated Body Inertias", *International Journal of Robotics Research*, Vol. 2, No. 1, Spring 1983, pp. 13-30.

[5] Bae, D. S. and Haug, E. J., "A recursive Formation for Constrained Mechanical Systems Dynamics: Part I, Open Loop Systems," *Mechanisms, Structures, and Machines*, Vol. 15, No. 3, 1987, pp. 359-382.

[6] Bae, D. S. and Haug, E. J., "A recursive Formation for Constrained Mechanical Systems Dynamics: Part II, Closed Loop Systems," *Mechanisms, Structures, and Machines*, Vol. 15, No. 4, 1987, pp. 481-506.

[7] Anderson, K. S.,"Recursive Derivation of Explicit Equations of motion for Efficient Dynamic/Control Simulation of Large Multibody Systems". *Ph.D. Dissertation*, Stanford University,1990.

[8] Rosenthal, D. E., "An Order n Formulation for Robotic Systems", *Journal of Astronautical Sciences*, Vol. 38, No. 4, Dec. 1990, pp. 511-529.

[9] Nielan, P.E., "Efficient Computer Simulation of Motions of Multibody Systems", *Ph.D. Dissertation*, Stanford University, 1986.

[10] Jones, R. E., "Multi-flex body dynamics for control design", *Proceedings of the Workshop on Multibody Simulation*, JPL D-5190, Pasadena, California, 1988, pp. 543-549.

[11] Schwertassek, R., "Reduction of Multibody Simulation Time by Appropriate Formulation of Dynamical System Equations", *Proc. of the NATO Advanced Study Institute on Computer-Aided Analysis of Rigid and Flexible Mechanical Systems*, Troia, Portugal, June 27-July 9, 1993, pp. 447-482.

[12] Garcia de Jalon, J. and Bayo, E., *Kinematic and Dynamic Simulation of Multibody Systems: The*

Real-Time Challenge, Springer-Verlag, New York, 1994.

[13] Rosenthal, D. E., and Sherman, M. A.,"High Performance Multibody Simulations via Symbolic Equation Manipulation and Kane's Method", *Journal of the Astronautical Sciences*, Vol. 34, No. 3, July-September, 1986, pp. 223-239.

[14] Anderson, K. S., "An Order-N Formulation for Motion Simulation of General Constrained Multi-Rigid-Body Systems", *Computers and Structures*, Vol. 43, No. 3, 1992, pp. 565-572.

[15] Anderson, K. S., "An Efficient Formulation for the Modeling of General Multi-Flexible-Body Constrained Systems", *International Journal of Solids and Structures*, Vol. 30, No. 7, 1993, pp. 921-945.

[16] Orlandea, N., Chace, M. A., and Calahan, D.A., " A Sparsity-Oriented Approach to the Dynamics Analysis and Design of Mechanical Systems- Parts 1 & 2", *Journal of Engineering for Industry*, Vol. 43, August 1977, pp. 773-784.

[17] Chace, M. A.,"Method and Experience in Computer Aided Design of Large-Displacement Mechanical Systems," *Computer Aided Analysis and Optimizations of Mechanical System Dynamics*, E.J. Haug (Ed.), Springer-Verlag, Berlin, 1984, pp. 233-259.

[18] Lubich, C., Nowak, U., Pohle, U., and Engstler, C."MEXX-Numerical Software for the Integration of Constrained Mechanical Systems", Berlin, SC92-12, 1992.

[19] Roberson, R. E. and Schwertassek, R., *Dynamics of multibody systems*, Springer-Verlag, New York, 1988.

[20] Brenan, K. E., Campbell, S. L. and Petzold, L. R., *Numerical solution of initial-value problems in differential-algebraic equations*, North-Holland, New York, 1989.

[21] Petzold, L. R., "Differential/Algebraic Equations are Not ODE's", *SIAM Journal of Science, Statistics and Computation*, Vol. 3, No. 3, September 1982, pp. 367-384.

[22] Petzold, L. R. and Lotstedt, P., "Numerical Solution of Nonlinear Differential Equations with Algebraic Constraints II: Practical Implications", *SIAM Journal of Science, Statistics and Computation*, Vol. 7, No. 3, July 1986, pp. 720-733.

[23] Banerjee, A. K., "Block-diagonal Equations for Multibody Elastrodynamics with Geometric Stiffness and Constraints", *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 6, Nov.-Dec. 1993, pp. 1092-1100.

[24] Fijany, A. and Bejczy, A. K., "Parallel Computation of Forward Dynamics of Manipulators", *JPL New Technology Report NPO-18706, NASA Technical Brief*, Vol. 17, No. 12, Item #82, December 1993.

[25] Fijany, A. and Bejczy, A. K., "Techniques for Parallel Computation of Mechanical Manipulator Dynamics. Part II Forward Dynamics", *Advances in Control and Dynamic Systems*, Vol. 40: Advances in Robotic Systems and Control, C.T. Leondes (Editor), Academic Press, March 1991, pp. 357-410.

[26] Zeid, A. A., Overholt, J. L. and Beck, R. R., "Modeling of Multibody Systems For Controls Using General Purpose Simulation Languages", *Simulation*, January 1994, pp. 6-19.

[27] Anderson, K. A., "Parallel $O(log_2 N)$ Algorithm for the Motion Simulation of General Multi-Rigid-Body Mechanical Systems", *Proceeding of DETC'97, 1997 ASME Design Engineering Technical Conference*, September 14-17, 1997, Sacramento, California, CDROM Paper # DETC97/VIB-4226.

[28] Anderson, K. A. and Duan, S., "Highly Parallelizable Low Order Algorithm for the Dynamics of Complex Multi Rigid Body Systems", Submitted to *Journal of Guidance, Control, and Dynamics*,Feb. 1997.

[29] Kasahara, H., Fujii, H. and Iwata, M., "Parallel Processing of Robot Motion Simulation", *Proc. IFAC 10th World Conference*, Munich, FRG, July, 1987.

[30] Bae, D.S., Kuhl, J.G., and Haug,E.J., "A recursive Formation for Constrained Mechanical System Dynamics: Part III, Parallel Processing Implementation", *Mechanisms, Structures, and Machines*, Vol. 16, 1988, pp. 249-269.

[31] Lee, C. S. G., and Chang, P. R.,"Efficient Parallel Algorithms for Robot Forward Dynamics", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 18, No. 2, 1988, pp. 238-251.

[32] Hwang, R. S., Bae, D. S., Kuhl, J. G., and Haug, E. J., "Parallel Processing for Real-Time Dynamic System Simulation", *ASME Journal of Mechanical Design*, Vol. 112, No. 4, 1990, pp. 520-528.

[33] Anderson, K. S., "An Efficient Modeling of Constrained Multibody Systems for Application with Parallel Computing", *Zeitschrift fur Angewante Mathematic un Mechanik*, Vol. 73, No. 6, 1993, pp. 520-528.

[34] Sharf, I. and D'Eleuterio, G.M.T., "An Iterative Approach to Multibody Simulation Dynamics Suitable for Parallel Implementation." *Journal of Dynamic Systems, Measurement and Control*, Vol. 115, Dec. 1993, pp. 730-735.

[35] Eichberger, A. "Transputer-Based Multibody System Dynamic Simulation, Part I: The Residual Algorithm-A Modified Inverse Dynamic Formulation", *Mechanics Of Structures And Machines*, Vol. 22, No. 2, 1994, pp. 211-237.

[36] Eichberger, A. "Transputer-Based Multibody System Dynamic Simulation, Part II: The Residual Algorithm-A Modified Inverse Dynamic Formulation", *Mechanics Of Structures And Machines*, Vol. 22, No. 2, 1994, pp. 239-261.

[37] Fijany, Amir, Sharf,Inna and D'Eleuterio, G.M.T.,"Parallel O(log N) Algorithms for Computation of Manipulator Forward Dynamics." *IEEE Transactions on Robotics and Automation*, Vol. 11, No. 3, June 1995, pp. 389-400.

[38] Lee, S. S., "Symbolic Generation of Equation of Motion for Dynamics/Control Simulation of Large Flexible Multibody Space Systems," *Ph.D Dissertation*, University of California, Los Angeles,

[39] Lathrop, L. H., "Parallelism in Manipulator Dynamics", *International Journal of Robotics Research*, Vol. 4, No. 2, 1985, pp. 80-102.

[40] Lee, C. S. G., and Chang, P. R., "Efficient Parallel Algorithms for Robot Inverse Dynamics", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 16, No. 4, 1988, pp. 532-542.

[41] Chung, S. and Haug, E. J., "Real-Time Simulation of Multibody Dynamics on Shared Memory Multiprocessors", *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 115, December 1993, pp. 730-735.

[42] Eichberger, A. and Fuhrer, C. and Schwertassek, R., "The Benefits of Parallel Multibody Simulation." *International Journal for Numerical Methods in Engineering*, Vol. 37, 1994, pp. 1557-1572.

[43] Kane, T.R., and Levinson, D.A., *Dynamics: Theory and Application*, McGraw Hill, NY, 1985.

[44] Ider, S. K. and Amirouche, F. M. L., "Coordinate reduction in the dynamics of constrained multibody systems - A new approach", *Journal of Applied Mechanics*, Vol. 55, December 1988, pp. 899-904.

[45] Saad, Y. *Iterative Methods for Sparse Linear Systems*, PWS, Boston, 1996.

[46] Golub, G. H., *Matrix Computations*, Second Edition 1993, Johns Hopkins, Baltimore.

[47] Quinn, M., *Parallel Computing: Theory and Practice*, Second Edition
, McGraw-Hill, Inc., NY, 1994.

[48] Pacheco, P. S., *Parallel Programming with MPI*, Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1997.

[49] Foster, I., *Designing and Building Parallel Programs*, Addison-Wesley, Reading, MA, 1995.

Sequential computational cost of hybrid O(n) with considerations of shifting and transformation

Step	No cutting	Constraints due to cutting	Hybrid O(n)
1st out	$(83m+68a)N$	$(32m+22a)(Nsub-1)$	(83m+68a)N+(32m+22a)(Nsub-1)
inward	$(215m+165a)N$ $-(215m+165a)$	$(94m+60a)N+(463m+281a)(Nsub-1)$ $-(94m+60a)\frac{N}{Nsub}$	$(309m+225a)N+(248m+116a)(Nsub-1)$ $-(94m+60a)\frac{N}{Nsub}$
2nd out	$(53m+47a)N$	$(35m+25a)N-(35m+25a)\frac{N}{Nsub}$ $+(180m+119a)(Nsub-1)-(20m+16a)$	$(88m+72a)N-(35m+25a)\frac{N}{Nsub}$ $+(180m+119a)(Nsub-1)-(20m+16a)$
assembling	0	$40a(Nsub-1)$	$40a(Nsub-1)$
acceleration	0	$(10m+10a)N+(24m+26a)(Nsub-1)$	$(10m+10a)N+(24m+26a)(Nsub-1)$
total (1-3)	$(351m+280a)N$ $-(215m+165a)$	$(139m+95a)N+(667m+466a)(Nsub-1)$	$(490m+375a)N+((484m+323a)(Nsub-1)$
(2-3)	$(268m+212a)N$ $-(215m+165a)$	$-(129m+85a)\frac{N}{Nsub}-(20m+16a)$	$-(129m+85a)\frac{N}{Nsub}-(20m+16a)$

m - multiplication cost a - addition cost Nsub - total numbers of subchain N - total numbers of bodies

Comparison of sequential O(n)

	ABI O(N)	K O(N) with shifting	K O(N) without shifting
Computational cost	$(199m+174a)N-(198m-173a)$	$(268m+212a)N-(215m+165a)$	$(181m+124a)N-(128m+77a)$

Comparison of o(logN)

	NE $O(logN)$	K $O(logN)$ with cosine for I	K $O(logN)$ without cosine for I
Computational cost	$(791m+521a)+(34m+76a)log_2N$ $+(732m+653a)log_2N$		