

A State-Time Formulation for Dynamic Systems Simulation Using Massively Parallel Computing Resources

Kurt S. Anderson, Associate Professor (anderk5@rpi.edu)
and Mojtaba Oghbaei, Doctoral Student (oghbam@rpi.edu)
*Department of Mechanical, Aerospace, and Nuclear Engineering, Rensselaer
Polytechnic Institute, 110 8th Street, Troy, New York 12180-3590
Phone (518)276-2339 / Fax (518)276-2623*

Nonlinear Dynamics, NODY 04-72, In Press

Abstract. A novel *state-time* formulation for the simulation and analysis of the dynamic behavior of complex multibody systems is presented. The method proposes a computationally fast algorithm which is better able to fully exploit anticipated future immensely parallel computing resources (e.g. peta flop machines and beyond) than existing multibody algorithms. The intent of the algorithm is to yield significantly reduced simulation turnaround time in situations where massively parallel ($> 10^6$ processors) computing resources are available to it. It is shown that as a consequence of such a *state-time* discretization scheme, the system of governing equations yields a set of loosely coupled nonlinear algebraic equations which is at most quadratic in the *state-time* variables, with significant linear components. As such, it is well-suited in structure for nonlinear algebraic equations solvers. The linear-quadratic structure of these equations further permits the use of a special solution scheme, which is expected to yield superior performance relative to more traditional Newton-Raphson type schemes when applied to large general systems.

Keywords: Multibody dynamics, Parallel computing, Space-time formulation, Scalable algorithm

Nomenclature:

B – Typical body B with its mass center B^*
 \vec{F}_{kA} – k^{th} applied force acting on the body B
 \vec{f}_{mC} – m^{th} unknown constraint force acting on the body B
 \vec{r}_{kA} – k^{th} position vector from B^* to application point of \vec{F}_{kA}
 $\vec{r}_{B^*j_m}$ – m^{th} position vector from B^* to application point (joint) of \vec{f}_{mC}
 \vec{T}_l – l^{th} applied concentrated moment acting on the body B
 N – Newtonian reference frame
 \vec{I}^{B/B^*} – Inertia dyadic of body B w.r.t. B^*
 ${}^N\vec{\omega}^B$ – Absolute angular velocity of body B
 ${}^N\vec{\alpha}^B$ – Absolute angular acceleration of body B
 ε_{ijk} – The standard indicial cyclic permutation operator
 \vec{x}_B – Absolute displacement vector of B^*
 ${}^N\vec{\omega}_{\times}^B$ – Angular velocity cross product matrix
 $\underline{C} = {}^N\underline{C}^B$ – Direction cosine matrix from the local frame B to N
 $\psi_i(t)$ and $\varphi_i(t)$ – i^{th} member of local family of C_1 and C_0 continuous shape functions
 \underline{A} , \underline{B} , \underline{D} , \underline{E} – Coefficient matrices in the partitioned form of the discretized *Newton's 2nd Law*
 \underline{R}_1 , \underline{R}_2 – Right hand side column matrices in the partitioned form of the discretized *Newton's 2nd Law* and geometric constraint equations

© 2004 Kluwer Academic Publishers. Printed in the Netherlands.

1. Introduction

Engineering analysis involving simulation and/or virtual prototyping for the purposes of design validation, optimization, and the like are computationally intensive tasks. Given a sufficiently complex system or an application which is not implemented carefully, even the fastest computer may not provide timely results. Consequently, computational efficiency and development of algorithms leading to decreased turnaround time of solution have become an active area of research.

The design and analysis of multibody systems is of particular importance to engineers working with a wide array of machines and mechanisms ranging from terrestrial vehicles and space structures to manufacturing equipment, robots, and bio-mechanical systems. The benefits of corresponding simulations and associated analysis are realized by increased reliability through identification of design defects, optimized design parameters, increased innovation, and/or reduction in the overall cost and time of the development cycle. Furthermore, it is often desirable to obtain results from analysis requiring the multibody dynamic simulation in as timely a manner as possible. To date, parallel computing has provided relatively modest (at times disappointing) performance gains with regard to most multibody applications for a variety of reasons. First, the underlying formulations often do not offer sufficient coarse grain parallelization to allow the simulations to be distributed over more than a surprisingly small number of processors. Secondly, in the overwhelming majority of multibody applications, the spatial domain of the problems considered is very much smaller than the temporal domain (i.e. the number of generalized coordinates used to describe the configuration of the system is \ll the number of temporal integration steps necessary to perform the simulations). The traditional formulations and solution schemes are inherently serial in time and offer limited parallelization in space. The required simulation turn around time is thus dominated by serial aspects of the formulation and solution scheme.

This research improves upon the state of the art in computationally time efficient algorithm in its ability to more fully and effectively exploit current and anticipated immensely parallel computing systems. A variety of formulations and dynamic simulation algorithms have been developed by a number of individuals that are sufficiently general to handle a wide range of multibody systems. In the most general sense, a multibody system is a collection of interconnected rigid and/or flexible bodies that can move relative to one another. This motion is consistent with joints that connect the bodies and limit the relative motion of interacting bodies.

As indicated above, with most multibody simulation applications the number of generalized coordinates used to describe the system is many orders of magnitude smaller than the number of temporal steps needed for a desired simulation. Thus, these methods all show very limited *speedup* (reduced simulation turnaround) due to parallelization. This difficulty of poor scalability (simulation turn around time ideally being inversely proportional to the number of processors used) as well as undesirable growth in solution effort with problem size is manifest in many fields of engineering and science. Unfortunately, development of parallel modelling and simulation algorithms has generally lagged the development of parallel computing hardware. A premise for this work, and the NSF and NASA proposals which have spawned it, is based on the intent to produce algorithms now to take advantage of anticipated gains in parallel computing power. Through increased processor speed, increased number of processors, decreased communication costs, and improved support (parallel operating systems, programming practices, and tools) and reduced cost, tera flop computing machines and beyond will become available for more general use. As such, this work has adopted the basic goal imposed/proposed in NASA Request for Proposal *NRA 00-LaRC-1*, that algorithms be developed which are capable of “effectively exploiting concurrently operating processors whose number may be very large; hundreds of thousands, even millions”. “The intent is to stimulate innovation, including new paradigms for intrinsically concurrent solutions in analysis and optimization in major aerospace applications that are now stymied by too high cost and too long turn-around times.”

This point was similarly emphasized and repeated as an intrinsic problem at the SCaLeS workshop [1]. As there is and always will be considerably more formidable systems that one wishes to model and analyze than can be treated using the most current analysis tools and computing resources, it is vital that algorithms be developed and improved so that any gains in computing resources are fully utilized, enhanced and multiplied.

As previously mentioned, a major drawback of almost all contemporary multibody algorithms is that they are inherently serial in time. Specifically, at each time step the system equations of motion are solved for the state derivatives, which are in turn integrated numerically, allowing the procedure to march forward to the next time step where the process is repeated. Due to this time marching characteristic, the focus of virtually all prior formulations has been on parallelization of the governing dynamical equations *spatially* over the current temporal integration step. Parallel implementation of these formulations in most multibody applications is limited by serial bottlenecks, and the use of

additional processors does not increase the speed of the analysis in a significant way unless these bottlenecks can be reduced. Parallelizing the simulation and all related analysis both *spatially* and *temporally* becomes possible within the proposed *state-time* formulation and results in a drastic increase in the number of coarse grain calculations that may be distributed over all the available processors.

The work described here develops the basis for a new *state-time* multibody dynamic algorithm which may better serve as an effective tool in the design and simulation of multibody systems in the context of its application using such anticipated future *immensely parallel* computing resources. This algorithm offers the potential for significantly reduced simulation turnaround time achieved through its ability to exploit effectively a very large number (if not all) of the processors made available to it in a coarse grain manner. The methodology is markedly distinct from the other traditional dynamics analysis and simulation approaches that are greatly limited in their performance due to the serial nature of major aspects of their underlying computation schemes. These gains can be accomplished by treating the *time* appearing within the equations of motion, as a variable in much the same manner as having been done on the spatial coordinates by the finite element community. This transforms the system equations of motion (most generally coupled nonlinear PDEs) to a lightly coupled system of at worst quadratic algebraic equations. These equations may in turn be solved efficiently using a special parallel solution scheme that is being developed to explicitly exploit the special structure of this resulting system of equations.

2. Traditional *State-Type* Dynamic Formulations

A substantial amount of time and effort has been put forward over the past three decades toward algorithm development for modelling, simulation and analysis of multibody dynamic systems. Initially, the majority of multibody dynamics researchers tended to utilize either a *Newton-Euler* formulation, *Lagrange's* method or a combination thereof. However, due to some computational disadvantages of these schemes, dynamic analysis techniques based on what are broadly termed *projection methods* began to appear and played a major role in contemporary multibody dynamics works. Initially, the primary concern in algorithm development was generality and robustness. However, it soon became apparent that the straight forward applications of formulations developed often exhibited very undesirable growth in computational expense (cost of function evaluation) per temporal integration step as a func-

tion of model complexity (manifesting itself as prohibitive CPU time requirements). As a consequence of this, increasing effort was expended by investigators to develop algorithms with the aim of improving simulation turnaround. Such algorithms first strove for improving simulation speed by reducing actual computational cost. However, the development of such algorithms had largely been oriented towards application on purely serial-architecture computer systems with relatively modest attention being directed to parallel computing issues.

In the mid 1980's, several researchers started developing parallel methods for dynamic simulation of multibody systems. Researchers began to rethink the use of previously developed dynamic simulation procedures, and how the benefits of concurrent processing might be realized. This effort was pioneered by Kasahara *et. al.* [2] and followed by many investigators. Some of the highlights of these parallel multibody dynamics algorithms can be found in [3]- [11].

The focus of virtually all these formulations has been to solve for the system state derivatives at the current time step in the most efficient manner. This resulted in algorithms which are inescapably limited to parallelization within the given (current) time step. As a consequence, these formulations are parallel-in-space and serial-in-time. As there will generally be many more temporal integration steps needed to perform a desirable simulation than spatial variables, proceeding in this fashion results in a method which is dominantly serial. As a consequence of *Amdahl's Law* [12] and the fact that exchange of information between processors of a parallel processing system comes with a cost, this results in often modest, if not disappointing performance gains in reducing simulation turnaround. Very roughly put, *Amdahl's Law* states that the time required to run a computer calculation in parallel is asymptotically limited by the time required to perform the sequential portions of the calculation. Because major aspects of these dynamic formulations are inherently serial, parallel implementation of these formulations are hobbled by these sequential bottlenecks.

An equally great obstacle for effective parallel computing implementation has to do with inter-processor communications cost. If the cost associated with distributing a computation (e.g. getting data from one processor to another, performing the desired calculation, and then getting the result back to the processor which needs it) requires more CPU time than the savings gained by distributing the calculation over multiple processors, then the overall calculation will actually be slowed down by distributing the computations. Indeed, the effect of communications time (cost) and *Amdahl's Law* on *Speedup* S_P (the ratio of computation time obtained by performing the calculation serially relative to in parallel) can be crudely approximated by [13]

$$S_P = \frac{T_S}{T_P} = \frac{1}{f + \frac{(1-f)}{N_P} + \frac{N_P \cdot T_{Comm}}{T_S}} \quad (1)$$

where f is the fraction of the operations which must be performed serially in the given formulation; N_P is the number of processors used in the parallel calculation; T_{Comm} is the time required for an inter-processor communication with a single processor; T_P is the time required to perform the calculation in parallel; and T_S is the time required to perform the calculation serially. From equation (1) it is clear that calculation speedup can be adversely dominated by serial bottlenecks and inter-processor communication costs. If substantial gains in simulation turnaround are to be realized, it is essential that the overall fraction of serial calculations be significantly reduced and the parallel calculations be made coarse-grain (require little inter-processor communication).

What is desired is to reformulate the equations of motion and temporal integrations such that time as well as the state variables is treated in an all encompassing manner. Such a *state-time* dynamics formulation would allow the equations of motion to now be parallelized temporally as well as spatially. This would have three potential advantages: First, the system of equations may now be coarse grain parallelized to a far greater degree allowing an increased number of processors N_P , to be effectively utilized. Secondly, this would significantly reduce f , the fraction of serial operations, and thus increase *speedup* (reduced turnaround). Finally, the method allows temporal scale of each variable to be adjusted independently, and as such offer considerable potential for efficiently and accurately modelling and simulation of multiscale systems (both temporally and spatially).

3. The New *State-Time* Methodology

Formulating the problem in the proposed *state-time* method is achieved by writing the variational form of the governing equations and then applying the *Galerkin* approximation using polynomial interpolating functions. As indicated above, it is desirable to formulate the problem in such a way that it is not limited to a largely serial temporal marching scheme. One way to achieve this, is to treat the time-varying quantities appearing within the equations of motion in a manner identical to the treatment of spatially varying quantities, effectively transforming the associate equations of motion from a system of nonlinear PDE's or ODE's to an associated system of nonlinear algebraic equations.

A Newton-Raphson method is then used for the solution of this nonlinear algebraic system. As such great care must be exercised to insure

that these equations are generated in a form which provides the best convergence characteristics (i.e., largest possible radius of convergence, high convergence order, and robustness). Additionally in an effort to reduce computational cost/iteration, it is desired to generate the equations in a sparse minimally coupled form which lends itself well to rapid coarse grain parallel solution

For the purpose of generating an associated system of equations which provides the least coupling and lowest level of nonlinearity as well as the greatest degree of coarse grain parallelism, a *Newton-Euler* approach is utilized. Here, it is shown how to determine the *state-time* treatment of each of the *Euler's* equation, the generalize form of *Newton's 2nd* law, *Poisson's* kinematic equations and geometric constraint equation for a typical body within a multibody system.

3.1. STATE-TIME CONSIDERATION OF THE EULER'S EQUATION

Euler's equation in the general form can be written as

$$\begin{aligned} \sum \vec{M}^{B/B^*} &= \sum_k \vec{r}_{kA} \times \vec{F}_{kA} + \sum_m \vec{r}_{B^*j_m} \times \vec{f}_{mC} + \sum_l \vec{T}_l \\ &= \vec{I}^{B/B^*} \cdot {}^N \vec{\alpha}^B + {}^N \vec{\omega}^B \times \vec{I}^{B/B^*} \cdot {}^N \vec{\omega}^B \end{aligned} \quad (2)$$

or for simplicity, in indicial form

$$\varepsilon_{ijh} r_{kj} C_{ph} F_{kp} + \varepsilon_{ijs} r_{mj} C_{nk} f_{mn} + C_{qi} T_q = I_{ij} \alpha_j + \varepsilon_{ijt} \omega_j I_{tl} \omega_l \quad (3)$$

In this equation let us consider

$$\alpha_i = \dot{\omega}_i (= \ddot{q}_i) \quad (4)$$

where α_i and ω_i ($i=1,2,3$) represent the measure numbers in the local B basis for the angular acceleration and angular velocity in the Newtonian reference frame, respectively. Here, q exists mathematically, but not necessarily physically (q is not as a rule an orientation angle). Using q in this fashion aids in the formulation of the weak form and in the treatment of impulsive loads. Applying the weighted residual method on equation (3) and then integration by part results in

$$\begin{aligned} \int_0^1 \{ \varepsilon_{ijh} r_{kj} C_{ph} F_{kp} + \varepsilon_{ijs} r_{mj} C_{nk} f_{mn} + C_{qi} T_q \} \tilde{E} dt - \\ I_{ij} \left(\dot{q}_j \tilde{E} \Big|_0^1 - \int_0^1 \dot{q}_j \dot{\tilde{E}} dt \right) - \varepsilon_{ijt} I_{tl} \int_0^1 \dot{q}_j \dot{q}_l \tilde{E} dt = 0 \end{aligned} \quad (5)$$

Note that capital letters with *tilde* denote to the weighting functions and the barred quantities refer to the *state-time* variables. By defining

the following parameterized relations in time

$$\left. \begin{aligned} q_j &= \bar{q}_{ju} \psi_u(t) & ; & \quad C_{ph} = \bar{C}_{phx} \psi_x(t) \\ f_{mn} &= \bar{f}_{mnt} \varphi_t(t) & ; & \quad \bar{E} = \bar{E}_r \psi_r(t) \end{aligned} \right\} \quad (6)$$

the Galerkin approximation of equation (5) after simplification may be written as

$$\begin{aligned} \bar{C}_{phx} \cdot \left\{ \varepsilon_{ijh} r_{kj} F_{kp} \int_0^1 \psi_x(t) \cdot \psi_r(t) dt \right\} + \bar{C}_{qiy} \cdot \left\{ T_q \int_0^1 \psi_y(t) \cdot \psi_r(t) dt \right\} + \\ \bar{C}_{nsw} \cdot \bar{f}_{mnt} \cdot \left\{ \varepsilon_{ijs} r_{mj} \int_0^1 \psi_w(t) \cdot \varphi_t(t) \cdot \psi_r(t) dt \right\} - \\ I_{ij} \left\{ \left[\bar{q}_{ju} \dot{\psi}_u(1) \cdot \psi_r(1) - \omega_j(0) \cdot \psi_r(0) \right] - \bar{q}_{ju} \int_0^1 \dot{\psi}_u(t) \cdot \dot{\psi}_r(t) dt \right\} - \\ \bar{q}_{ju} \cdot \bar{q}_{lv} \cdot \left\{ \varepsilon_{ijl} I_{tl} \int_0^1 \dot{\psi}_u(t) \cdot \dot{\psi}_v(t) \cdot \psi_r(t) dt \right\} = 0 \quad (7) \end{aligned}$$

This is the *state-time* representation of the *Euler's* equation which in general constitutes $3 \cdot k \cdot n_e$ quadratic algebraic equations and $9 \cdot k \cdot n_e + 3 \cdot m \cdot (p \cdot n_e + 1) + 3 \cdot k \cdot n_e$ unknown *state-time* variables. k and p denote to the order of the shape functions associated with spatial and force variables respectively, and in general are different so that the *Babuška-Brezzi condition* [14] is satisfied. n_e is the number of temporal elements and m is the number of geometric constraints.

3.2. STATE-TIME CONSIDERATION OF THE GENERALIZATION OF NEWTON'S 2nd LAW

In *Euler's* generalization of the *Newton's 2nd* law of motion as stated below, it is assumed that all active forces included in \vec{F}^B are known. However, in many cases some of these constituent forces might be functions of the state of the system, e.g. the spring force or viscous force. We apply the same discretization scheme on these state-dependent forces.

$$\vec{F}^B + \sum_m \vec{f}_{mC} = m_B \ddot{\vec{x}}_B \quad (8)$$

where $\vec{F}^B = \sum_k \vec{F}_{kA}^B$. For simplicity, we neglect the superscript B in the subsequent equations. Following a similar scheme as described above, i.e. applying weighted residual method, integrating by parts, and then writing the Galerkin approximation of equation (8), results in

$$\left\{ \bar{x}_{ij} \dot{\psi}_j(1) \cdot \dot{\psi}_r(1) - \dot{x}_i(0) \cdot \dot{\psi}_r(0) \right\} - \bar{x}_{ij} \int_0^1 \dot{\psi}_j(t) \cdot \dot{\psi}_r(t) dt -$$

$$\frac{1}{m} \left\{ F_i \int_0^1 \psi_r(t) dt + \sum_m \left[\bar{f}_{mk} \cdot \int_0^1 \varphi_k(t) \cdot \psi_k(t) dt \right] \right\} = 0 \quad i = 1, 2, 3 \quad (9)$$

Equations (9) are the *state-time* representation of *Newton's 2nd Law* which constitutes $3 \cdot k \cdot n_e$ linear algebraic equations and introduces $3 \cdot k \cdot n_e$ additional unknown *state-time* variables pertaining to translational DOF of the body B .

3.3. STATE-TIME CONSIDERATION OF THE POISSON'S EQUATIONS

In the spatial dynamics of MBS, the components may undergo large translational and rotational motions. To define the configuration of each body with respect to neighboring reference frame, it is convenient to assign a coordinate reference frame to each body.

One of the most widely used parameterizations in describing reference orientations are the use of three independent orientation angles (of which *Euler's angles* are a subset). A three parameter set is attractive since it has the same number of parameters as there are rotational DOF for a body. A disadvantage of this representation is that no three-parameter set can be both global and nonsingular [15]. Additionally, this type of parameterization is undesirable in this context for describing the required basis transformation matrix, because the direction cosine terms which arise within the equations of motion are in general third order nonlinear transcendental functions, which further couple in a nonlinear fashion with system variables within the proposed state-time framework.

To be both nonsingular and global, more than three parameters are required to set up the transformation matrix. *Euler's parameters* are a set of four parameters that are well suited for computation and have been used in several general-purpose multibody programs since they do not suffer from singularity. Unfortunately, the *kinematical differential equations* associated with Euler's parameters are individually quadratic in the system variables. This results in final *state-time* equations that would be cubic in *state-time* variables. Hence, these parameters also yield a level of coupling and nonlinearity in the subsequent calculations that is not desirable from the viewpoint of this research.

An alternate means for describing general three dimensional rotations is through the direct use of direction cosines as rotation parameters. Due to their increased number (9) and associated higher system dimensionality, this parametrization method is not generally used in conventional/traditional MBS formulations. However, in this *state-time* formulation, the use of direction cosines directly as state variables produces a set of algebraic equations that are at worst quadratic in

state-time variables with the lowest level of coupling. With this being the case, the kinematical differential equations, which relate the time derivative of the transformation matrix to the angular velocity vector of the body are *Poisson's* equations, specifically

$${}^N \dot{\underline{C}}^B = {}^N \underline{C}^B \cdot \left[{}^N \underline{\omega}_\times^B \right]_B \quad (10)$$

or in indicial form

$$\dot{C}_{il} = -\varepsilon_{mlk} \omega_k C_{im} \quad (11)$$

Applying the weighted residual and substituting the approximation relations yields

$$\bar{C}_{iln} \cdot \int_0^1 \dot{\psi}_n(t) \psi_r(t) dt + \bar{C}_{imj} \cdot \left\{ \varepsilon_{mlk} \bar{q}_{kjp} \int_0^1 \dot{\psi}_p(t) \cdot \psi_j(t) \cdot \psi_r(t) dt \right\} = 0 \quad (12)$$

Equations (12) are the *state-time* representation of *Poisson's* equations which constitutes $9 \cdot k \cdot n_e$ quadratic algebraic equations and there are no additional unknown *state-time* variables introduced by them.

3.4. STATE-TIME CONSIDERATION OF THE GEOMETRIC CONSTRAINT EQUATIONS

Consider the body B with m neighbors as illustrated in Figure 1.

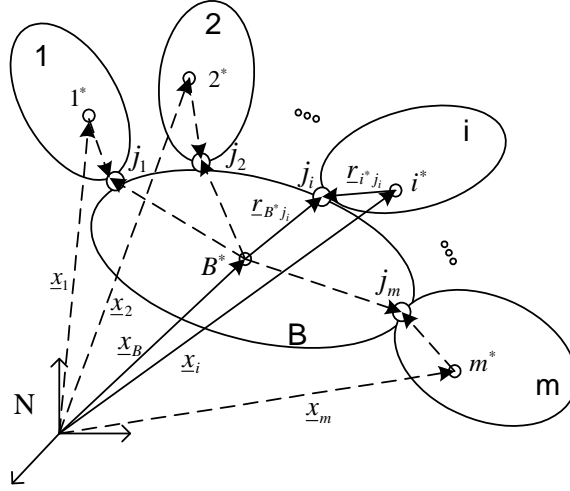


Figure 1. A multibody system in tree structure

For each joint J_i we can write

$$\underline{x}_B + {}^N \underline{C}^B \underline{r}_{B^*j_i} = \underline{x}_i + {}^N \underline{C}^i \underline{r}_{i^*j_i} \xrightarrow{\text{vel. level}} \dot{\underline{x}}_B + {}^N \dot{\underline{C}}^B \underline{r}_{B^*j_i} = \dot{\underline{x}}_i + {}^N \dot{\underline{C}}^i \underline{r}_{i^*j_i} \quad (13)$$

Applying the weighted residual method on each of these equations gives

$$\int_0^1 \left\{ \dot{\underline{x}}_B + {}^N \underline{\dot{C}}^B \underline{r}_{B^*j_i} \right\} \tilde{G}_i dt = \int_0^1 \left\{ \dot{\underline{x}}_i + {}^N \underline{\dot{C}}^i \underline{r}_{i^*j_i} \right\} \tilde{G}_i dt \quad i = 1, 2, \dots, m \quad (14)$$

The Galerkin approximation of equations (14) after parameterization in time may be written as

$$\begin{aligned} \bar{x}_{Btj} \int_0^1 \dot{\psi}_j(t) \cdot \varphi_z(t) dt + {}^N \bar{C}_{tkm}^B \int_0^1 r_{B^*j_k} \dot{\psi}_m(t) \cdot \varphi_z(t) dt - \\ \bar{x}_{its} \int_0^1 \dot{\psi}_s(t) \cdot \varphi_z(t) dt + {}^N \bar{C}_{tl_n}^i \int_0^1 r_{i^*j_l} \dot{\psi}_n(t) \cdot \varphi_z(t) dt = 0 \\ i = 1, 2, \dots, m \end{aligned} \quad (15)$$

Equations (15) are the *state-time* representation of geometric constraint equations that are linear in the *state-time* variables. Before counting the number of new equations and independent unknowns due to these equations, let us first look at the correspondent *Euler's* equation, generalization of the *Newton's 2nd* law, and *Poisson's* equation associated with each of the attached bodies. In a similar manner as elaborated for body *B*, we can find the number of new equations and unknowns introduced by each of these equations for these bodies. At the end, after counting the number of independent equations and unknowns pertaining to the *state-time* representation of the common geometric constraint equations, the number of algebraic *state-time* equations and unknowns for the entire system are determined to be

$$15 k n_e (m + 1) + 3 m (p n_e + 1) \quad (16)$$

The resulting nonlinear algebraic system, though large in dimension, exhibits only quadratic nonlinearity and light nearest neighbor coupling in temporal domain. Consequently, if one solves these equations through the use of *Newton-Raphson* type methods, the associated tangent matrix is very sparse and banded in structure. This enables parallel sparse matrix solvers to be used effectively.

4. Linear-Quadratic Structure of the *state-time* Equations

Earlier work by the authors (submitted for *Multibody System Dynamics Journal*) utilized a *Newton-Raphson* approach for the solution of the entire set of *state-time* equations. This approach was shown to be simple and effective, but was felt to be more general and expensive than might actually be needed for the solution of the system equations

which result from the parameterization associated with the *state-time* formulation. Inspection of the *state-time* equations derived in section 3 indicates a special *linear - loosely coupled - quadratic* structure. Due to this system of equation's light coupling and limited quadratic terms, a special solution approach was developed which exploits this structure, with the aim of producing solution methods, while specific to this structure, are both more efficient and robust than the more general Newton-Raphson type methods. As shown in the section 3, *state-time* equations for a multibody system consists of a substantial linear portion, associated with the *state-time* representation of *Newton's 2nd Law*, the kinematic constraint equations, and a significant fraction of the *Euler's* equations. The remaining portion involves loosely coupled quadratic terms arising in the *state-time* representation of *Euler's* equations and *Poisson's* equations. Within these equations the *state-time* discretization produces quadratic terms of the form $\bar{C}_{ij}\bar{f}_k$; $\bar{C}_{ij}\bar{q}_k$; and $\bar{q}_i\bar{q}_j$ ($i, j, k = 1, 2, 3$). The *state-time* representation of *Newton's 2nd Law* and geometric constraint relations being purely linear in the constituent variables do not have to appear within the Newton-Raphson iteration at all, and may instead be used to reduce out a like number of variables from the system of equations. Eliminating the linear part from the Newton-Raphson iterate results in the system of quadratic equations of reduced dimension and considerably smaller eigenvalue spectrum. The resulting method requires a substantially smaller set of initial values to get started, on a more local set of equations. The results from the linear subset of equations are used to produce a reduced dimension consistent tangent matrix associated with the *state-time* representation of *Euler's* equations of motion and *Poisson's* kinematical equations.

More specifically, since the discretized form of *Newton's 2nd Law* and the *geometric constraint equations* are linear in \bar{x} 's and \bar{f} 's and \bar{x} 's and \bar{C} 's, respectively, these subsets of the system equations can be always partitioned in the following form

$$[\underline{A} \ \underline{B}] \begin{bmatrix} \bar{x} \\ \bar{f} \end{bmatrix} = [\underline{R}_1] \quad (17)$$

and

$$[\underline{D} \ \underline{E}] \begin{bmatrix} \bar{x} \\ \bar{C} \end{bmatrix} = [\underline{R}_2] \quad (18)$$

which consists of a consistent set of equations and unknowns. If we assume that we have an initial guess for all the *state-time* values asso-

ciated with the elements of direction cosine matrices $\underline{\bar{C}}$, then one can solve for $\underline{\bar{f}}$'s by the following relations

$$\underline{\bar{f}} = \left[\underline{D} \underline{A}^{-1} \underline{B} \right]^{-1} \left[\underline{D} \underline{A}^{-1} \underline{R}_1 + \underline{E} \underline{C} - \underline{R}_2 \right] \quad (19)$$

and $\underline{\bar{x}}$'s can be simply found from (17) as

$$\underline{\bar{x}} = \underline{A}^{-1} \left[\underline{R}_1 - \underline{B} \underline{\bar{f}} \right]. \quad (20)$$

Having the values of $\underline{\bar{f}}$'s from equation (19) and using the initial guess on $\underline{\bar{q}}$'s, equations (7) and (12) become quadratic in the variables $\underline{\bar{q}}$'s and $\underline{\bar{C}}$'s. Newton-Raphson iterates on these equations results in updated values for these variables and the whole process may be repeated until the convergence criterium is met. The chart shown in Figure 2 outlines the above procedure.

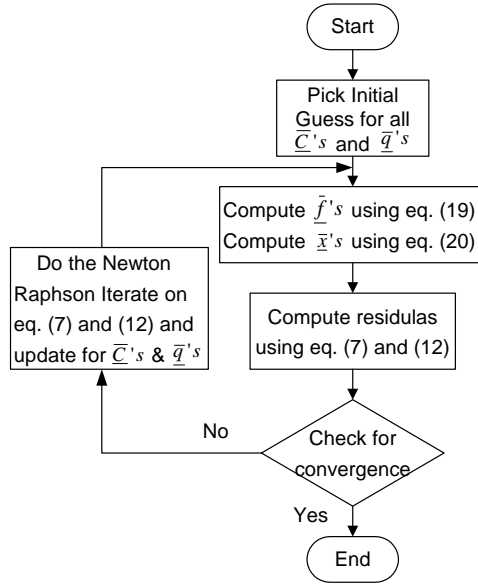


Figure 2. Flow chart of the operation using new approach

The advantages in solving the *state-time* system of equations in the described manner, relative to the direct use of *state-time* equations with a more traditional Newton-Raphson iteration type approach are classified in the Table I

Table I. Characteristic Comparison of the two approaches

Newton-Raphson iteration on all <i>state-time</i> equations and variables simultaneously	Exploitation of the linear-quadratic structure of <i>state-time</i> equations
Need for initial guess on all <i>state-time</i> variables, \bar{q} , \bar{x} , \bar{C} , and \bar{f}	Need to have initial guess for just \bar{q} , and \bar{C}
Perform Newton-Raphson iteration on all variables and equations	Perform Newton-Raphson iteration only on considerably <i>smaller</i> reduced system
More effort in realizing an intelligent initial guess on full set of variables	Easier to choose a set of intelligent initial guess due to the constraints on \bar{C} 's
Populated block matrix per each equation in certain temporal element	Smaller blocks involve fewer nonzero terms
Highest dimension system of equations	Considerably smaller system of equations
Equations quadratic	Equations still quadratic

5. Interpolating Functions and Example

In this section, we will briefly discuss some implementation issues as they pertain to the performance of interpolation functions, and demonstrate typical results of dynamic simulation of a simple nonlinear system which is obtained using the *state-time* methodology.

5.1. SELECTION OF INTERPOLATING FUNCTIONS

In general, the order of the polynomial interpolating functions associated with spatial variables and constraint forces can and should be different from each other. Briefly to put, it is noticed that the set of governing equations herein is very much of the same structure as the *mixed problem* in FEM literature. Thus, care should be taken to avoid spurious modes in the resulting system of algebraic constraint equations through satisfaction of the *Babuška-Brezzi condition* [14]. This in many cases requires the use of lower order interpolation functions for the force-type variables than for the spatial variables.

5.2. OTHER CLASSES OF INTERPOLATING FUNCTIONS

There are a variety of interpolation techniques which may be employed in the parameterization of equations. Each introduces various properties and characteristics inherent to the interpolation functions and

techniques used. In the example given here, Lagrange family of polynomial shape functions are used to interpolate the *state-time* variables. However, there are other classes of approximating techniques such as those involving *Hermite* interpolating functions, *wavelet* basis and so forth that have been shown to be useful in capturing some important part of solutions in other applications. Their well-known capabilities can also be potentially used within this formulation which shows great promise when dealing with dynamic simulation of complex multibody systems, particularly as it relates to modelling impact and/or impulsive behavior.

5.3. EXAMPLE: PLANAR DOUBLE PENDULUM

Below are presented some typical results of the *state-time* simulation of a dynamic system, specifically a planar double pendulum as depicted in Figure 3. The plots given in this section provide representative results associated with the 5 second simulation of this pendulum based on the *state-time* (ST) formulation, as well as obtained from the system equations of motion determined by traditional means and direct integration using the MATLAB ODE45 solver. The results for each of these cases are superimposed onto the same set of axes for better comparison.

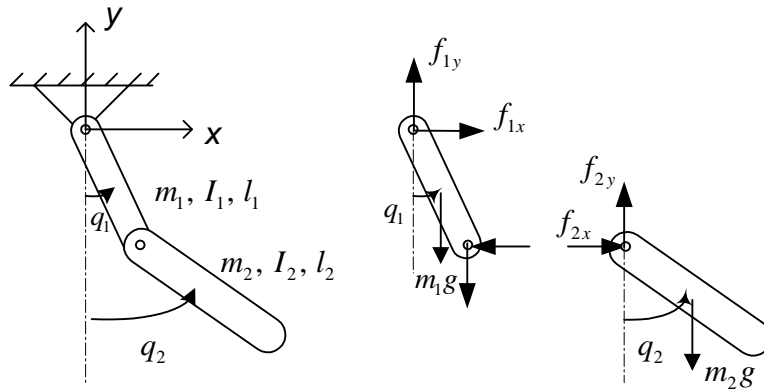


Figure 3. Planar Double Pendulum

The full descriptor form of *Newton-Euler's* equations, along with the Poisson's kinematical equations of each body and two geometric constraint equations are used. This yields a system of 14 equations and 14 unknowns in which the elements of the direction cosine matrices are treated as independent variables. We have used two different sets of polynomial interpolating functions: linear-quadratic (LQ) and quadratic-cubic (QC) shape functions. In each case, the lower order

interpolating function is used to approximate constraint forces and the higher order function is associated with the spatial variables.

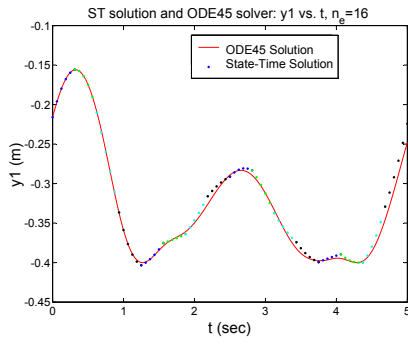
The parameters used for this simulation are $m_1 = 1$ kg, $l_1 = 0.8$ m, $\theta_{10} = 1$ rad, $\omega_{10} = 1$ rad/sec, $m_2 = 0.5$ kg, $l_2 = 1$ m, $\theta_{20} = 1.2$ rad, $\omega_{20} = -1$ rad/sec, and $g = 2$ m/sec². The plots in Figure 4 are associated with a 5 second simulation of this system using $n_e = 16$ temporal elements.

Comparison of the plots demonstrated in Figure 4 shows graphically how the quality of the solution is improved, particularly for the velocity level quantities and the constraint forces as a result of increasing the order of shape functions from quadratic to cubic. Table II quantifies this improvement for different numbers of temporal elements for both the linear-quadratic (LQ) and quadratic-cubic (QC) cases. In this table, "error" ε is defined as the L_2 norm of the area encapsulated between the solution curves from the *state-time* solution and ODE45 solver of a unit time simulation of the double pendulum without considering the error due to the angular velocity quantities.

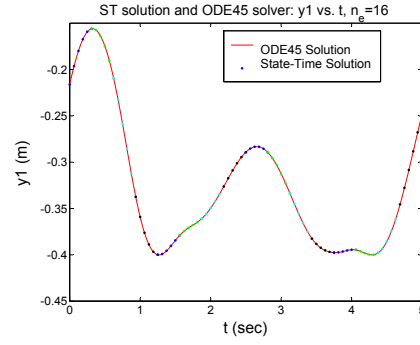
Inspection of the results presented in Table II indicates the method exhibits a super-linear performance of 1.76 (that is the error decreases as $(1/n_e)^{1.76}$) when using the Linear-Quadratic interpolation set, and super-quadratic performance of 2.09 for the Quadratic-Cubic interpolation set. The accuracy of the approach thus appears to be dominated by the order of force associated interpolation functions. Additionally, it is seen that the increase in the interpolation set order from LQ to QC, which represents an 1.4 factor increase in computational cost, results in significantly improved accuracy particularly for the velocity level quantities and the constraint forces. It appears that for the systems thus far considered, one can expect greater enhancement (superior accuracy to cost ratio) of the method through p-refinement (refinement through higher order interpolation functions), than through h-refinement (refinement through increased number of elements) [16].

Table II. Accuracy as function of the number of temporal elements and interpolation order

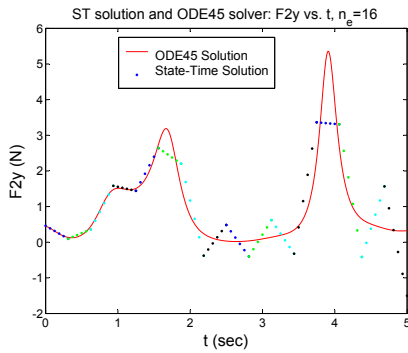
Number of temporal elements (n)	1	2	4	8	16	32	64
L2 norm of the error (LQ), ε_{LQ}	0.9038	0.3502	0.1273	0.0272	0.0062	0.0024	7.99e-4
L2 norm of the error (QC), ε_{QC}	0.2834	0.1094	0.0169	0.0041	9.69e-4	2.37e-4	5.95e-5
Error Ratio, $\varepsilon_{LQ}/\varepsilon_{QC}$	3.189	3.201	7.532	6.634	6.399	10.126	13.429



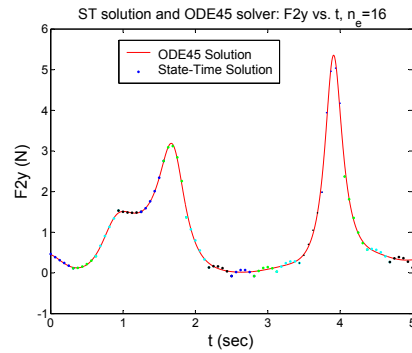
(a) Position of the mass center, y_1 vs. t (LQ)



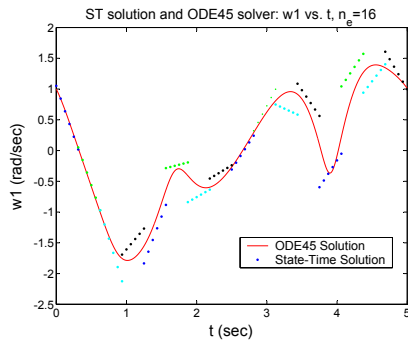
(b) Position of the mass center, y_1 vs. t (QC)



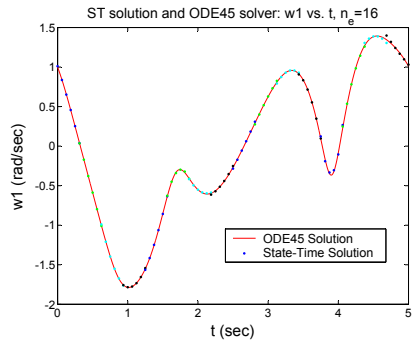
(c) Constraint force, f_{2y} vs. t (LQ)



(d) Constraint force, f_{2y} vs. t (QC)



(e) Angular velocity, w_1 vs. t (LQ)



(f) Angular velocity, w_1 vs. t (QC)

Figure 4. Simulation results of the planar double pendulum; left column: using linear-quadratic element & right column: using quadratic-cubic element

6. Conclusion

In traditional dynamic simulation, the governing system of differential algebraic equations (DAEs) must be solved at the current time step for the system state derivatives, which are then temporally integrated. The results from current integration step are then used to update the system state and the process repeats as the simulation marches sequentially forward in time. This inherently serial in time numerical solution greatly restricts the extent to which parallel computing can be effectively employed for most multibody applications, and as such greatly limits any gains in reduced turn around time which might be realized through availability of massively parallel computing resources. Additionally, these serial time steps often reflect the finest governing time scales within the system at the instant under consideration (except where more sophisticated multirate integrations schemes are used), and as such may reduce simulation speed because integration is dictated by these scales. Such drawbacks cause unavoidable computational cost for the entire simulation, and result in simulations which lack scalability in time. The proposed methodology demonstrates great promise (in the context of immensely parallel computing) for improved speedup which can be achieved by circumventing many of the shortcomings of conventional dynamic simulation algorithms. This work is distinct from the other space-time formulations that have been applied to date in dynamic contexts, in that all those implementations tend to result in a dense, highly nonlinear and coupled structure made their solution difficult, if not intractable for anything but simple systems [17]. The presented *state-time* formulation has a highly desirable structure, namely a hybrid linear-quadratic system of loosely coupled equations in spatial and force variables with only nearest neighbor coupling in time variables. It shows the capability of scaling both spatially and temporally, resulting in a drastic increase in the number of coarse grain calculations that can be effectively and efficiently distributed over all the available processors of the parallel computing system.

Acknowledgements

Support for this work has been provided by National Science Foundation (NSF) under the Award No. CMS-0219734 and is gratefully appreciated.

References

1. 'Science Case for Large-scale Simulation', Washington D.C., U.S.A, June 24 & 25, 2003, <http://www.pnl.gov/scales/>
2. Kasahara, H., Fujii, H., and Iwata, M., 'Parallel processing of robot motion simulation', In Proceedings IFAC 10th World Conference, 1987.
3. 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, (16), 1988, pp.249-269.
4. Fijany, A., and Bejczy, A. K., 'Techniques for parallel computation of mechanical manipulator dynamics, part II Forward Dynamics', Advances in Robotic Systems and Control, Vol. 40, Academic Press, March 1991, pp. 357-410.
5. 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, (115), Dec. 1993, pp.730-735.
6. Fijany, A., and Bejczy, A. K., 'Parallel computation of forward dynamics of manipulators', NASA Technical Brief Report NPO-18706 12, NASA Jet Propulsion Laboratory, Item # 82 1993.
7. Anderson, K. S., 'Efficient modeling of constrained multibody systems for application with parallel computation', Zeitschrift für Angewandte Mathematik und Mechanik, Vol. 73, No. 6, 1993, pp. 935-939.
8. Fijany, A., Sharf, I., and D'Eleuterio, G. M. T., 'Parallel $O(\log n)$ algorithms for computation of manipulator forward dynamics', IEEE Transactions on Robotics and Automation, 11(3), 1995, pp.389-400
9. Anderson, K. S., and Duan, S., 'A highly parallelizable low-order algorithm for dynamics of multi-rigid-body systems: Part I, chain systems', Mathematical and Computer Modeling, (30), 1999, pp. 193-215.
10. Featherstone, R., 'A divide-and-conquer articulated body algorithm for parallel $O(\log_2 n)$ calculation of rigid body dynamics, Part 1: Basic algorithm', International Journal of Robotics Research, 18(9), Sep. 1999, pp.867-875.
11. Critchley J. H., and Anderson, K. S., 'On Parallel Methods of Multibody Dynamics', ASME DETC'03 Computers and Information in Engineering Conference, Chicago, Illinois, USA, September 2003.
12. Almassi, G. S. and Gotlieb, A., Highly Parallel Computing, Benjamin-Cummings, Menlo Park, CA, 2nd edition, 1994.
13. Quinn, M. J., Parallel Computing Theory and Practice, McGraw-Hill, 1994.
14. Hughes, T. J. R., The finite element method, Linear static and dynamic finite element analysis' Dover Publications Inc., New York, 1987.
15. Hughes, P. C., Spacecraft attitude dynamics, Wiley, 1986.
16. Biswas, R., and Devine, K. D., and Flaherty, J. E., 'Parallel, adaptive finite element methods for conservation laws', Applied Numerical Mathematics, v. 14, n. 1-3, April 1994, pp. 255-283.
17. Kunz, D. L., 'Multibody System Analysis Based on Hamilton's Weak Principle', 42nd Structures, Structural Dynamics and Materials Conference and Exhibit, AIAA 2001-1296, Seattle, WA, 16-19 April 2001.

