On an Implementation of the Hilber-Hughes-Taylor Method in the Context of Index 3 Differential-Algebraic Equations of Multibody Dynamics (DETC2005-85096)

Dan Negrut*
Department of Mechanical Engineering
The University of Wisconsin
Madison, WI-53706
negrut@wisc.edu

Gisli Ottarsson
MSC.Software
Ann Arbor, Michigan 48105
gisli.ottarsson@mscsoftware.com

Rajiv Rampalli
MSC.Software
Ann Arbor, Michigan 48105
rajiv.rampalli@mscsoftware.com

Anthony Sajdak
MSC.Software
Ann Arbor, Michigan 48105
anthony.sajdak@mscsoftware.com

July 5, 2006

Abstract

The paper presents theoretical and implementation aspects related to a numerical integrator used for the simulation of large mechanical systems with flexible bodies and contact/impact. The proposed algorithm is based on the Hilber-Hughes-Taylor implicit method and is tailored to answer the challenges posed by the numerical solution of index 3 Differential-Algebraic Equations that govern the time evolution of a multibody system. One of the salient attributes of the algorithm is the good conditioning of the Jacobian matrix associated with the implicit integrator. Error estimation, integration step-size control, and nonlinear system stopping criteria are discussed in detail. Simulations using the proposed algorithm of an engine model, a model with contacts, and a model with flexible bodies indicate a 2 to 3 speedup factor when compared against benchmark MSC.ADAMS runs. The proposed HHT-based algorithm has been released in the 2005 version of the MSC.ADAMS/Solver.

INTRODUCTION

The Hilber-Hughes-Taylor method: general considerations

The Hilber-Hughes-Taylor (HHT) method (also known as the alpha-method) [22] is widely used in the structural dynamics community for the numerical integration of a linear set of second Ordinary Differential Equations (ODE). This problem is obtained at the end of a finite element discretization. Provided the finite element approach is linear, the equations of motion assume the form

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{F}(t) \tag{1}$$

The $p \times p$ mass, damping, and stiffness matrices, $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$, respectively, are constant, the force $\mathbf{F} \in \mathbb{R}^p$ depends on time t, and $\mathbf{q} \in \mathbb{R}^p$ is the set of generalized coordinates used to represent the configuration of the system.

A precursor of the HHT method is the Newmark method [30], in which a family of integration formulas that depend on two parameters β and γ is defined:

*Address all correspondence to this author.

$$\mathbf{q}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\left[(1-2\beta)\,\ddot{\mathbf{q}}_n + 2\beta\ddot{\mathbf{q}}_{n+1}\right] \tag{2a}$$

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + h\left[(1-\gamma)\,\ddot{\mathbf{q}}_n + \gamma\ddot{\mathbf{q}}_{n+1}\right] \tag{2b}$$

These formulas are used to discretize at time t_{n+1} the equations of motion (1) using an integration step size h:

$$\mathbf{M}\ddot{\mathbf{q}}_{n+1} + \mathbf{C}\dot{\mathbf{q}}_{n+1} + \mathbf{K}\mathbf{q}_{n+1} = \mathbf{F}_{n+1} \tag{2c}$$

Based on Eqs. (2a) and (2b), $\mathbf{q}_{n+1}$ and $\dot{\mathbf{q}}_{n+1}$ are functions of the acceleration $\ddot{\mathbf{q}}_{n+1}$, which in Eq. (2c) remains the sole unknown quantity that is obtained as the solution of a linear system. This method is implicit and A-stable (stable in the whole left-hand plane) [2] provided [22]

$$\gamma \geq 1/2 \qquad\qquad \beta \geq \frac{\left(\gamma + \frac{1}{2}\right)^2}{4} \tag{3}$$

The only combination of β and γ that leads to a second-order integration formula is $\gamma = \frac{1}{2}$ and $\beta = \frac{1}{4}$. This choice of parameters produces the trapezoidal method, which is both A-stable and second order. The drawback of the trapezoidal formula is that it does not induce any numerical damping in the solution, which makes it impractical for problems that have high-frequency oscillations that are of no interest or parasitic high-frequency oscillations that are a byproduct of the finite element discretization process [23, 10]. Thus, the major drawback of the Newmark family of integrators was that it could not provide a formula that was A-stable and second order and displayed a desirable level of numerical damping. The HHT method came as an improvement because it preserved the A-stability and numerical damping properties, while achieving second order accuracy when used in conjunction with the second order linear ODE problem of Eq. (1). The idea proposed in [22] actually does not pertain the expression of the Newmark integration formulas, but rather the form of the discretized equations of motion in (2c). The new equation in which the integration formulas of Eqs. (2a) and (2b) are substituted is

$$\mathbf{M}\ddot{\mathbf{q}}_{n+1} + (1+\alpha)\,\mathbf{C}\dot{\mathbf{q}}_{n+1} - \alpha\mathbf{C}\dot{\mathbf{q}}_n + (1+\alpha)\,\mathbf{K}\mathbf{q}_{n+1} - \alpha\mathbf{K}\mathbf{q}_n = \mathbf{F}\left(\tilde{t}_{n+1}\right) \tag{4}$$

where

$$\tilde{t}_{n+1} = t_n + (1+\alpha)\,h \tag{5}$$

As indicated in [23], the HHT method will possess the advertised stability and order properties provided $\alpha \in \left[-\frac{1}{3},\ 0\right]$ and

$$\gamma = \frac{1-2\alpha}{2} \qquad\qquad \beta = \frac{(1-\alpha)^2}{4} \tag{6}$$

The smaller the value of α, the more damping is induced in the numerical solution. Note that in the limit, the choice $\alpha = 0$ leads to the trapezoidal method with no numerical damping.

The multibody dynamics problem

In the case of a multibody system, without any loss of generality, the set of generalized coordinates considered henceforth is as follows: for each body i its position is described by the vector $\mathbf{r}_i = [x_i, y_i, z_i]^T$, while its orientation is given by the array of local 3-1-3 Euler angles (see, for instance, [36]), $\mathbf{e}_i = [\psi_i, \theta_i, \phi_i]^T$. Consequently, for a mechanical system containing n_b bodies, $\mathbf{q} = \left[\mathbf{r}_1^T\ \mathbf{e}_1^T\ \ldots\ \mathbf{r}_{n_b}^T\ \mathbf{e}_{n_b}^T\right]^T \in \mathbf{R}^p,\ p = 6n_b$. Note that this set of of position generalized coordinates is augmented with deformation modes when flexible bodies are present in the model.

In any constrained mechanical system, joints connecting bodies restrict their relative motion and impose constraints on the generalized coordinates. Kinematic constraints are then formulated as algebraic expressions involving generalized coordinates

$$\mathbf{\Phi}(\mathbf{q}, t) = \left[\ \Phi_1(\mathbf{q}, t)\quad \ldots\quad \Phi_m(\mathbf{q}, t)\ \right]^T = \mathbf{0} \tag{7a}$$

where m is the total number of independent constraint equations that must be satisfied by the generalized coordinates throughout the simulation. Although in its current implementation the algorithm handles nonholonomic constraints in Pfaffian form [32], for brevity, the case of holonomic constraints is assumed henceforth.

Differentiating Eq. (7a) with respect to time leads to the velocity kinematic constraint equation

$$\mathbf{\Phi_q}(\mathbf{q}, t)\,\dot{\mathbf{q}} + \mathbf{\Phi}_t(\mathbf{q}, t) = \mathbf{0} \tag{7b}$$

where the over-dot denotes differentiation with respect to time and the subscript denotes partial differentiation, $\mathbf{\Phi_q} = \left[\frac{\partial \Phi_i}{\partial q_j}\right]$, for $1 \leq i \leq m,\ \ 1 \leq j \leq p$. The acceleration kinematic constraint equation is obtained by differentiating Eq. (7b) with respect to time:

$$\mathbf{\Phi_q}(\mathbf{q}, t)\,\ddot{\mathbf{q}} + \left(\mathbf{\Phi_q}(\mathbf{q}, t)\dot{\mathbf{q}}\right)_{\mathbf{q}}\dot{\mathbf{q}} + 2\mathbf{\Phi}_{\mathbf{q}t}(\mathbf{q}, t)\,\dot{\mathbf{q}} + \mathbf{\Phi}_{tt}(\mathbf{q}, t) = \mathbf{0} \tag{7c}$$

The state of the mechanical system changes in time under the effect of applied forces such that Eqs. (7a)–(7c) are satisfied at all times. The time evolution of the system is governed by the Lagrange multiplier form of the constrained equations of motion (see, for instance [21, 36]),

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^T(\mathbf{q})\lambda = \mathbf{Q}\left(\dot{\mathbf{q}}, \mathbf{q}, t\right) \tag{7d}$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{p\times p}$ is the generalized mass, and $\mathbf{Q}\left(\dot{\mathbf{q}}, \mathbf{q}, t\right) \in \mathbb{R}^p$ is the action (as opposed to the reaction $\mathbf{\Phi}_{\mathbf{q}}^T(\mathbf{q})\lambda$) force acting on the generalized coordinates $\mathbf{q} \in \mathbb{R}^p$. These equations are neither linear nor ordinary differential as is the case in Eq. (1), first and foremost because the solution $\mathbf{q}(t)$ must also satisfy the kinematic constraint equations in Eq. (7a). These constraint equations lead in Eq. (7d) to the presence of the reaction force $\mathbf{\Phi}_{\mathbf{q}}^T(\mathbf{q})\lambda$, where $\lambda \in \mathbb{R}^m$ is the Lagrange multiplier associated with the kinematic constraints.

In addition to the equations of motion and kinematic constraint, several classes of equations need to be considered in a general-purpose mechanical simulation package:

1. *Ordinary differential equations* that in the most general case are provided in implicit form

$$\mathbf{d}(\dot{\mathbf{X}}, \mathbf{X}, \mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \lambda, \mathbf{V}, \mathbf{F}, t) = \mathbf{0}_{n_d} \tag{8a}$$

   This type of differential equations is encountered, for instance, when controls are active in the system, such as is the case in cars with an anti-lock braking system (ABS), active suspension control, and so forth. The state of the controller is $\mathbf{X}$, its time derivative is $\dot{\mathbf{X}}$, and the assumption is that in its implicit-form Eq. (8a) properly and uniquely defines $\dot{\mathbf{X}}$ as a function of the state of the system through the user-specified function $\mathbf{d}$. If n_d represents the number of differential states, then $\mathbf{X}, \mathbf{d} \in \mathbb{R}^{n_d}$.

2. *User-defined variables*, which can technically be regarded as aliases or definition equations. A set of n_v user-defined variables $\mathbf{V} \in \mathbb{R}^{n_v}$ is typically specified through an equation of the form

$$\mathbf{V} - \mathbf{v}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{X}, \lambda, \mathbf{V}, \mathbf{F}, t) = \mathbf{0}_{n_v} \tag{8b}$$

   which, during the solution sequence, are solved (or rather evaluated) simultaneously with the equations of motion and the kinematic constraint equations. Here $\mathbf{v} \in \mathbb{R}^{n_v}$ is a user-defined function that depends on other system states as indicated in Eq. (8b).

3. *External force definition*, $\mathbf{F}$, which allows the user to define the set of n_f applied forces $\mathbf{F} \in \mathbb{R}^{n_f}$ that act on the system. This is the mechanism through which a complex tire model can be interfaced to a vehicle model that supports user-defined bushing elements, custom nonlinear damper and friction models, and the like.

$$\mathbf{F} - \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{X}, \dot{\mathbf{X}}, \lambda, \mathbf{V}, \mathbf{F}, t) = \mathbf{0}_{n_f} \tag{8c}$$

Equations (7a)–(7d) comprise a system of index 3 DAE [7], where the notion of DAE index, as far as its definition and meaning are concerned, continues to be a research topic and the interested reader is referred to [37] and the recent work [26] for a more in depth analysis of the concept. It is known that differential-algebraic equations are not ordinary differential equations [33]. Analytical solutions of Eqs. (7a) and (7d) automatically satisfy Eqs. (7b) and (7c), but this is no longer true for numerical solutions. In general, the task of obtaining a numerical solution of the DAE of Eqs. (7a)–(7d) is substantially more difficult and prone to intense numerical computation than that of solving ordinary differential equations. For an account of relevant work in the area of numerical integration methods for the DAE of multibody dynamics the reader is referred to [2, 7, 4, 16, 1, 20, 28, 34] and references therein.

The theory and attractive features associated with the HHT method have been derived in conjunction with a linear second-order ODE. The only similarity between Eqs. (1) and (7d) is that they are both second order and qualitatively obtained from Newton's second law. In [38], for the purpose of stability and convergence analysis, the constrained equations of motion are tackled in a stabilized index 2 DAE framework. The HHT method is also discussed in [14] and more recently in [13], where the proposed implementation is based on a technique that accounts for violations in the position and velocity constraints in a stabilization framework similar to the one proposed in [5]. There are also several Runge-Kutta-based approaches for highly oscillatory mechanical system simulation that, like the HHT method, display the attractive attribute of selectively damping frequency at the high end of the spectrum. In [31], a Singly Diagonal Runge-Kutta (SDIRK)-based method allows the user to choose, within certain bounds, the diagonal value in the formula and thus control the amount of numerical damping associated with the algorithm. The role of the diagonal element in the formula becomes similar to that of the α parameter in the HHT method. An approach based on additive Runge-Kutta methods that has the potential to accurately handle highly oscillatory multibody dynamics simulation was introduced in [24], and further discussed in [35]. These novel Runge-Kutta-based algorithms are mathematically sound, but they require more time to achieve, vis-a-vis industrial-strength applications, the level of acceptance currently associated with the well-established HHT method.

The method used in this paper was also considered in [9], and then revisited in [10]. Thus, this work does not propose a new integration method (the HHT method), but rather an algorithm based on this method, which (a) proposes a step-size control strategy at the position level that works in conjunction with both the second order differential equations of motion and the first order equations that typically come from hydraulic or control systems; (b) proposes a choice of unknowns (accelerations and Lagrange multipliers, respectively) in the Newton-type algorithm associated with the implicit discretization scheme that results in optimal condition number with extremely positive impact on the robustness of the algorithm; (c) defines a stopping criteria for the nonlinear system that is tied to the level of user specified accuracy in an analytically sound way; and (d) withstood the test of industrial strength applications as it was implemented and released in a commercial simulation software that used it successfully in conjunction with models with more than 18,000 states (engine model).

THE PROPOSED ALGORITHM

In addition to the Newmark formulas of Eq. (2) an integration formula is required for the solution of the first-order ODE in Eq. (8a). The general formula considered is

$$\mathbf{X}_{n+1} = \mathbf{X}_n + h\dot{\mathbf{X}}_n + h\rho\left(\dot{\mathbf{X}}_{n+1} - \dot{\mathbf{X}}_n\right) \tag{9a}$$

in which case

$$\frac{\partial \mathbf{X}_{n+1}}{\partial \dot{\mathbf{X}}_{n+1}} = \rho h\, \mathbf{I}_{n_d} \tag{9b}$$

where throughout this paper $\mathbf{I}_s$ stands for the identity matrix of dimension s. The discussion about the choice of the parameter ρ is deferred to a later section.

For the multibody dynamics problem stated, the unknowns of interest are the generalized positions, velocities, and accelerations $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$, respectively, the Lagrange multipliers λ, the applied-force states $\mathbf{F}$, the user-defined variables (aliases) $\mathbf{V}$, and the states associated with the user-defined ordinary differential equations, that is, $\mathbf{X}$ and $\dot{\mathbf{X}}$. The index 3 DAE multibody dynamics problem that must be solved to compute these quantities is neither linear nor ordinary differential, and the HHT method is thus applied for a different class of problems from that originally designed for. Rather than approaching the solution within an index 2 framework [38] or using a stabilization approach [14, 13], the proposed algorithm uses the implicit Newmark formulas to discretize the equations of motion and requires that the position-level kinematic constraint equations be satisfied at the end of each time step. This is a direct index 3 approach, and it requires at each integration time step the solution of a nonlinear system of equations. The theoretical foundation of this algorithm is provided by the stability and convergence results and observations in [27, 6, 9, 10]. However, we are not aware of a global convergence proof for the HHT method when used in conjunction with the fully nonlinear index 3 problem of multibody dynamics. Current work is undergoing for a rigorous global convergence proof for a stabilized index 2 formulation [25]. The global convergence for the direct HHT-based index 3 formulation remains very much an open question, but the results obtained with the proposed algorithm provide an early validation of the good properties associated with the HHT method.

At the cornerstone of the proposed algorithm lies the simple idea on which the HHT method is built: recycle the Newmark integration formulas, but slightly change the equations of motion to account for the set of forces acting on the system at two consecutive integration points. The algorithm is modified to a small extent to accommodate the set of differential and algebraic equations (8a) through (8c) that a general-purpose simulation package would have to handle. Thus, in the spirit of the original HHT formulation, the discretization of the multibody dynamics equations of motion yields

$$\left(\mathbf{M}\ddot{\mathbf{q}}\right)_{n+1} + (1+\alpha)\left(\Phi_{\mathbf{q}}^T\lambda - \mathbf{Q}\right)_{n+1} - \alpha\left(\Phi_{\mathbf{q}}^T\lambda - \mathbf{Q}\right)_{n} = \mathbf{0} \tag{10}$$

For notational simplicity, when obvious, the dependency of some quantities on $\mathbf{q}$ and/or $\dot{\mathbf{q}}$ and/or time t will be omitted, as was done in Eq. (10). From an implementation standpoint it is more advantageous to scale the previous equation by $\frac{1}{(1+\alpha)}$ to obtain the equivalent form

$$\frac{1}{1+\alpha}\left(\mathbf{M}\ddot{\mathbf{q}}\right)_{n+1} + \left(\Phi_{\mathbf{q}}^T\lambda - \mathbf{Q}\right)_{n+1} - \frac{\alpha}{1+\alpha}\left(\Phi_{\mathbf{q}}^T\lambda - \mathbf{Q}\right)_{n} = \mathbf{0} \tag{11a}$$

Discretization of the rest of the problem equations leads to

$$\mathbf{\Phi}\left(\mathbf{q}_{n+1}, t_{n+1}\right) = \mathbf{0} \tag{11b}$$

$$\mathbf{d}\left(\dot{\mathbf{X}}_{n+1}, \mathbf{X}_{n+1}, \mathbf{q}_{n+1}, \dot{\mathbf{q}}_{n+1}, \ddot{\mathbf{q}}_{n+1}, \lambda_{n+1}, \mathbf{V}_{n+1}, \mathbf{F}_{n+1}, t_{n+1}\right) = 0 \tag{11c}$$

$$\mathbf{V}_{n+1} - \mathbf{v}\left(\mathbf{q}_{n+1}, \dot{\mathbf{q}}_{n+1}, \ddot{\mathbf{q}}_{n+1}, \mathbf{X}_{n+1}, \lambda_{n+1}, \mathbf{V}_{n+1}, \mathbf{F}_{n+1}, t_{n+1}\right) = 0 \tag{11d}$$

$$\mathbf{F}_{n+1} - \mathbf{f}\left(\mathbf{q}_{n+1}, \dot{\mathbf{q}}_{n+1}, \ddot{\mathbf{q}}_{n+1}, \mathbf{X}_{n+1}, \dot{\mathbf{X}}_{n+1}, \lambda_{n+1}, \mathbf{V}_{n+1}, \mathbf{F}_{n+1}, t_{n+1}\right) = 0 \tag{11e}$$

Everywhere in Eq. (11), in an index 3 DAE direct approach, the Newmark integration formulas of Eq. (2) are used to express $\mathbf{q}$ and $\dot{\mathbf{q}}$ as a function of $\ddot{\mathbf{q}}$, while Eq. (9a) is used to discretize the set of ordinary differential equations (express $\mathbf{X}$ as a function of $\dot{\mathbf{X}}$). A Newton-like algorithm [15] is used to solve the resulting system of nonlinear equations for the set of unknowns (in this order) $\ddot{\mathbf{q}}$, λ, $\dot{\mathbf{X}}$, $\mathbf{V}$, and $\mathbf{F}$. Note that the generalized force $\mathbf{Q}$ of Eq. (7d) is obtained by projecting the force states $\mathbf{F}$ along the generalized coordinates $\mathbf{q}$; that is, $\mathbf{Q}(\mathbf{q}, \mathbf{F}) = \mathbf{\Pi}\ \mathbf{F}$, where the projection operator $\mathbf{\Pi} = \mathbf{\Pi}(\mathbf{q})$ depends on the choice of generalized coordinates. The iterative algorithm requires at each iteration (k) the solutions of the linear system

$$\begin{bmatrix} \hat{\mathbf{M}} & \mathbf{\Phi}_{\mathbf{q}}^T & 0 & 0 & -\mathbf{\Pi} \\ \mathbf{\Phi}_{\mathbf{q}} & 0 & 0 & 0 & 0 \\ \left(\mathbf{d}_{\ddot{\mathbf{q}}} + \gamma h \mathbf{d}_{\dot{\mathbf{q}}} + \beta h^2 \mathbf{d}_{\mathbf{q}}\right) & \mathbf{d}_{\lambda} & \mathbf{d}_{\dot{\mathbf{X}}} + \rho h\, \mathbf{d}_{\mathbf{X}} & \mathbf{d}_{\mathbf{V}} & \mathbf{d}_{\mathbf{F}} \\ -\left(\mathbf{v}_{\ddot{\mathbf{q}}} + \gamma h \mathbf{v}_{\dot{\mathbf{q}}} + \beta h^2 \mathbf{v}_{\mathbf{q}}\right) & -\mathbf{v}_{\lambda} & -\rho h\, \mathbf{v}_{\mathbf{X}} & \mathbf{I} - \mathbf{v}_{\mathbf{V}} & -\mathbf{v}_{\mathbf{F}} \\ -\left(\mathbf{f}_{\ddot{\mathbf{q}}} + \gamma h \mathbf{f}_{\dot{\mathbf{q}}} + \beta h^2 \mathbf{f}_{\mathbf{q}}\right) & -\mathbf{f}_{\lambda} & -\mathbf{f}_{\dot{\mathbf{X}}} - \rho h\, \mathbf{f}_{\mathbf{X}} & -\mathbf{f}_{\mathbf{V}} & \mathbf{I} - \mathbf{f}_{\mathbf{F}} \end{bmatrix} \begin{bmatrix} \Delta\ddot{\mathbf{q}} \\ \Delta\lambda \\ \Delta\dot{\mathbf{X}} \\ \Delta\mathbf{V} \\ \Delta\mathbf{F} \end{bmatrix}^{(k)} = \begin{bmatrix} -\mathbf{e}_1 \\ -\mathbf{e}_2 \\ -\mathbf{e}_3 \\ -\mathbf{e}_4 \\ -\mathbf{e}_5 \end{bmatrix}^{(k)} \tag{12}$$

where $\mathbf{e}_i$ are the residuals in satisfying the set of discretized equations of motion, constraint equations, discretized first order differential equations, variable definition equations, and applied force definition equations, respectively, and unless otherwise specified, all the quantities in $\mathbf{e}_1$ through $\mathbf{e}_5$ are evaluated at time t_{n+1}:

$$\begin{aligned}
\mathbf{e}_1 &= \frac{1}{1+\alpha}\left(\mathbf{M}\ddot{\mathbf{q}}\right)_{n+1} + \left(\Phi_{\mathbf{q}}^T\lambda - \mathbf{Q}\right)_{n+1} - \frac{\alpha}{1+\alpha}\left(\Phi_{\mathbf{q}}^T\lambda - \mathbf{Q}\right)_n \\
\mathbf{e}_2 &= \frac{1}{\beta h^2}\Phi\left(\mathbf{q}, t\right) \\
\mathbf{e}_3 &= \mathbf{d}\left(\dot{\mathbf{X}}, \mathbf{X}, \mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \lambda, \mathbf{V}, \mathbf{F}, t\right) \\
\mathbf{e}_4 &= \mathbf{V} - \mathbf{v}\left(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{X}, \lambda, \mathbf{V}, \mathbf{F}, t\right) \\
\mathbf{e}_5 &= \mathbf{F} - \mathbf{f}\left(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{X}, \dot{\mathbf{X}}, \lambda, \mathbf{V}, \mathbf{F}, t\right)
\end{aligned} \tag{13}$$

The matrix $\hat{\mathbf{M}}$ in Eq. (12) is defined as

$$\hat{\mathbf{M}} = \frac{\partial \mathbf{e_1}}{\partial \ddot{\mathbf{q}}} = \mathbf{M} - h\gamma\frac{\partial \mathbf{Q}}{\partial \dot{\mathbf{q}}} + \beta h^2\left[\left(\mathbf{M}\ddot{\mathbf{q}}\right)_{\mathbf{q}} + \left(\Phi_{\mathbf{q}}^T\lambda\right)_{\mathbf{q}} - \frac{\partial \mathbf{Q}}{\partial \mathbf{q}}\right] \tag{14}$$

Note that the nonlinear equations associated with the position kinematic constraints are scaled by $\frac{1}{\beta h^2}$ in order to improve the conditioning of the coefficient matrix in Eq. (12). This is a compromise reached after considering the following options: (a) have the level-zero positions, $\mathbf{q}$, and differential states, $\mathbf{X}$, be the unknowns (replacing $\ddot{\mathbf{q}}$ and $\dot{\mathbf{X}}$), in which case some entries in the Jacobian matrix in Eq. (12) end up divided by βh^2; (b) have $\ddot{\mathbf{q}}$ and $\dot{\mathbf{X}}$ be the unknowns, in which case the second row in the Jacobian matrix comes multiplied by βh^2; (c) same as in (b), except that the set of positions kinematic constraint equations are scaled by $\frac{1}{\beta h^2}$. Option (a) is implemented by the default integrator used in the MSC.ADAMS simulation package [29] (here entries get divided by a factor $\beta_0 h$ rather than βh^2, as the second-order equations of motion are reduced to an equivalent first-order system of differential equations that is then solved with a BDF-type integrator [18]). On numerous occasions this has been observed to be the cause of numerical problems once the step-size becomes very small and consequently some entries in the Jacobian become extremely large. A bad Jacobian condition number ensues, and the quality of the Newton corrections becomes poor. The option (b) was not embraced because the problem at (a) plagues this approach as well, though in a more subtle way. If h becomes very small, the second row of the Jacobian matrix is scaled by βh^2, which practically makes all the entries in this row very small and thus leads to ill conditioning. Option (c) proved a good solution because typically the type of error that one sees in satisfying the position kinematic constraint equations is very small. It is never that these constraint equations are problematic in a simulation but rather some discontinuity in the model that causes the step-size h to assume small values. But if h is small, when advancing the simulation the position constraint violation stays very small, and the norm of $\mathbf{e}_2$ always remains bounded. A formal proof of this result is provided in [17], which also discusses the nonsingular character of the coefficient matrix in Eq. (12) when $h \to 0$, and the convergence of the iterative Newton scheme. Thus, a salient feature of the approach is that it eliminates the ill conditioning typically associated with the index 3 integration of the DAE of multibody dynamics. Two factors are responsible for this: (i) the position kinematic constraint equations are appropriately scaled, and (ii) the set of unknowns $\ddot{\mathbf{q}}$ and λ are consistent in the sense that they are qualitatively of the same kinematic level, that is, two (as opposed to mixing $\mathbf{q}$, which is level zero, with λ, which is level two).

With the corrections computed as the solution of the linear system of Eq. (12), the numerical approximation of the solution is improved at each iteration as $\ddot{\mathbf{q}}^{(k+1)} = \ddot{\mathbf{q}}^{(k)} + \Delta\ddot{\mathbf{q}}^{(k)}$, $\lambda^{(k+1)} = \lambda^{(k)} + \Delta\lambda^{(k)}$, $\dot{\mathbf{X}}^{(k+1)} = \dot{\mathbf{X}}^{(k)} + \Delta\dot{\mathbf{X}}^{(k)}$, $\mathbf{V}^{(k+1)} = \mathbf{V}^{(k)} + \Delta\mathbf{V}^{(k)}$, $\mathbf{F}^{(k+1)} = \mathbf{F}^{(k)} + \Delta\mathbf{F}^{(k)}$. The following sections present in detail the answer to three key questions: (a) When is the computed solution accurate enough? (b) How to select the integration step-size h? and (c) When should one stop the Newton-like iterative process that computes at each integration step the unknowns $\ddot{\mathbf{q}}$, λ, $\dot{\mathbf{X}}$, $\mathbf{V}$, and $\mathbf{F}$? Recall that once $\ddot{\mathbf{q}}$ and $\dot{\mathbf{X}}$ are available, Eqs. (2a), (2b), and (9a) are used to evaluate $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\mathbf{X}$, respectively.

IMPLEMENTATION DETAILS FOR PROPOSED ALGORITHM

Estimating the local integration error

Since an approximation of the *global* error at time t_{n+1} cannot be obtained in general, the goal is to produce an approximation of the *local* integration error in advancing the simulation from step n to $n+1$. Once the *local* integration error is available, an algorithm is implemented to ensure that this error stays smaller than a user-prescribed tolerance. Based on a linearization of the equations of motion in Eq. (7d) along with an asymptotic expansion of the solution $\mathbf{q}$, a strategy for estimating the local integration error in positions is presented. Following the same approach, namely, linearization and asymptotic expansion, an estimate of the local integration error is also provided for the state $\mathbf{X}$ associated with the ordinary differential equations defined by Eq. (8a).

Local integration error in positions coordinates.

The approximation of the *local* integration error is similar to the approach proposed in [39] for the Newmark method. The discussion is going to focus on Eq. (4), because locally a linearization of Eq. (7d) leads to the previous form. Thus, Eq. (4) is rewritten as

$$\mathbf{M}\ddot{\mathbf{q}}_{n+1} + (1+\alpha)\left(\mathbf{C}\dot{\mathbf{q}}_{n+1} + \mathbf{K}\mathbf{q}_{n+1}\right) - \alpha\left(\mathbf{C}\dot{\mathbf{q}}_{n} + \mathbf{K}\mathbf{q}_{n}\right) = \mathbf{F}\left(\tilde{t}_{n+1}\right) \tag{15}$$

with $\tilde{t}_{n+1}$ defined as in Eq. (5).

For the purpose of computing the local integration error, the usual assumption is that the configuration at time t_n, $(\mathbf{q}_n, \dot{\mathbf{q}}_n, \ddot{\mathbf{q}}_n)$ is perfectly consistent. That is, it satisfies the equations of motion, along with the time derivatives of the equations of motion:

$$\mathbf{M}\ddot{\mathbf{q}}_n + \mathbf{C}\dot{\mathbf{q}}_n + \mathbf{K}\mathbf{q}_n = \mathbf{F}_n \tag{16a}$$

$$\mathbf{M}\dddot{\mathbf{q}}_n + \mathbf{C}\ddot{\mathbf{q}}_n + \mathbf{K}\dot{\mathbf{q}}_n = \dot{\mathbf{F}}_n \tag{16b}$$

The Newmark integration formula of Eq. (2) is rewritten in the equivalent form

$$\mathbf{q}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\ddot{\mathbf{q}}_n + \beta h^2 \mathbf{x} \tag{17a}$$

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + h\ddot{\mathbf{q}}_n + h\gamma\mathbf{x} \tag{17b}$$

$$\ddot{\mathbf{q}}_{n+1} = \ddot{\mathbf{q}}_n + \mathbf{x} \tag{17c}$$

where the unknown $\mathbf{x}$ represents the change in the value of acceleration from time t_n to t_{n+1}. The goal is to compute an estimate of the error at the end of one integration step (the *local* integration error)

$$\delta^q_{n+1} = \mathbf{q}_{n+1} - \tilde{\mathbf{q}}_{n+1} \tag{18}$$

where $\tilde{\mathbf{q}}_{n+1}$ is the *exact* solution of the initial value problem

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{F} \tag{19}$$

that starts in the configuration $(\mathbf{q}_n, \dot{\mathbf{q}}_n, \ddot{\mathbf{q}}_n)$ at $t = t_n$.
Using Taylor's theorem, one obtains $\tilde{\mathbf{q}}_{n+1}$ as

$$\tilde{\mathbf{q}}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\ddot{\mathbf{q}}_n + \frac{h^3}{6}\dddot{\mathbf{q}}_n + O\left(h^4\right) \tag{20}$$

The local integration error δ_{n+1}^q becomes available as soon as the acceleration correction $\mathbf{x}$ is available. In order to obtain an estimate for $\mathbf{x}$, based on Eqs. (15) and (17)

$$\mathbf{M}\left(\ddot{\mathbf{q}}_n+\mathbf{x}\right)+(1+\alpha)\left[\mathbf{C}\left(\dot{\mathbf{q}}_n+h\ddot{\mathbf{q}}_n+h\gamma\mathbf{x}\right)+\mathbf{K}\left(\mathbf{q}_n+h\dot{\mathbf{q}}_n+\frac{h^2}{2}\ddot{\mathbf{q}}_n+\beta h^2\mathbf{x}\right)\right]$$
$$-\alpha\left(\mathbf{C}\dot{\mathbf{q}}_n+\mathbf{K}\mathbf{q}_n\right)=\mathbf{F}_n+(1+\alpha)\,\dot{\mathbf{F}}_n h+O\left(h^2\right) \tag{21}$$

where Taylor's theorem was used to expand $\mathbf{F}\left(t_n+(1+\alpha)\,h\right)$. Using Eqs. (16a) and (16b),

$$\left[\mathbf{M}+(1+\alpha)\,h\gamma\mathbf{C}+(1+\alpha)\,\beta h^2\mathbf{K}\right]\mathbf{x}=(1+\alpha)\,\mathbf{M}\dddot{\mathbf{q}}_n h+O\left(h^2\right) \tag{22}$$

Denoting $\mathbf{D}=\mathbf{M}+(1+\alpha)\,h\gamma\mathbf{C}+(1+\alpha)\,\beta h^2\mathbf{K}$, since $\mathbf{D}^{-1}=\mathbf{M}^{-1}+O\left(h\right)\cdot\mathbf{I}_p$, the equation

$$\mathbf{D}\,\mathbf{x}=(1+\alpha)\,\mathbf{M}\dddot{\mathbf{q}}_n h+O\left(h^2\right) \tag{23}$$

leads to

$$\mathbf{x}=(1+\alpha)\,\dddot{\mathbf{q}}_n h+O\left(h^2\right) \tag{24}$$

and therefore

$$\delta_{n+1}^q=\mathbf{q}_{n+1}-\tilde{\mathbf{q}}_{n+1}=h^3\left[\beta\,(1+\alpha)-\frac{1}{6}\right]\cdot\dddot{\mathbf{q}}_n+O\left(h^4\right) \tag{25}$$

Substituting for $\dddot{\mathbf{q}}_n$ from Eq. (24) and ignoring the higher-order terms leads to

$$\delta_{n+1}^q\approx\left[\beta-\frac{1}{6\,(1+\alpha)}\right]h^2\mathbf{x} \tag{26}$$

which provides an effective way of computing the local integration error, since the required quantities are available at the end of the corrector stage.

Local integration error in X states.

A necessary condition for the differential equations of Eq. (8a) to be locally well defined is that [7] $\det\left(\frac{\partial\mathbf{d}}{\partial\dot{\mathbf{X}}}\right)\neq 0$ holds in a neighborhood of the current system configuration. Assuming that the user-defined form for $\mathbf{d}$ satisfies this requirement, by using the implicit function theorem and Taylor's theorem, $\dot{\mathbf{X}}$ can be *locally* expressed explicitly as a function of $\mathbf{X}$ and time t:

$$\dot{\mathbf{X}}=\mathbf{A}\mathbf{X}+\mathbf{b}\,(t) \tag{27}$$

where $\mathbf{A}$ is a constant matrix that depends on the configuration of the system at the time when the linearization is carried out, and $\mathbf{b}$ is a function of time. One additional time derivative leads to

$$\ddot{\mathbf{X}}=\mathbf{A}\dot{\mathbf{X}}+\dot{\mathbf{b}}\,(t) \tag{28}$$

The integration formula used to integrate the differential equations in Eq. (8a) is equivalently expressed as

$$\mathbf{X}_{n+1}=\mathbf{X}_n+h\dot{\mathbf{X}}_n+\rho h\mathbf{x}_d \tag{29}$$

where $\mathbf{x}_d=\dot{\mathbf{X}}_{n+1}-\dot{\mathbf{X}}_n$. The goal is to produce an approximation of the local integration error when advancing the simulation from t_n to t_{n+1}. To this end, suppose that $\tilde{\mathbf{X}}_{n+1}$ is the exact solution at t_{n+1} starting from (t_n,X_n), while $\mathbf{X}_{n+1}$ is the approximate solution as computed by the proposed algorithm. By using Taylor's theorem,

$$\tilde{\mathbf{X}}_{n+1}=\mathbf{X}_n+h\dot{\mathbf{X}}_n+\frac{1}{2}h^2\ddot{\mathbf{X}}_n+O\left(h^3\right) \tag{30}$$

Considering the definition of the local truncation error $\delta_{n+1}^d\equiv\mathbf{X}_{n+1}-\tilde{\mathbf{X}}_{n+1}$, based on Eq. (29) and Eq. (30),

$$\delta_{n+1}^d=\rho h\mathbf{x}_d-\frac{1}{2}h^2\ddot{\mathbf{X}}_n+O\left(h^3\right)$$

Thus δ_{n+1}^d is available as soon as $\mathbf{x}_d$ becomes available. Note that $\mathbf{x}_d$ should satisfy

$$\dot{\mathbf{X}}_n + \mathbf{x}_d = \mathbf{A}\mathbf{X}_n + \mathbf{b}_n + \mathbf{A}\left(h\dot{\mathbf{X}}_n + \rho h\mathbf{x}_d\right) + h\dot{\mathbf{b}}_n + O\left(h^2\right) \tag{31}$$

Substituting for $\dot{\mathbf{X}}_n$ from Eq. (27) into Eq. (31) and performing simple manipulations yields

$$\mathbf{x}_d = h\ddot{\mathbf{X}}_n + O\left(h^2\right) \tag{32}$$

Therefore,

$$\delta_{n+1}^d = \left(\rho - \frac{1}{2}\right) h^2\ddot{\mathbf{X}}_n + O\left(h^3\right) \tag{33}$$

which leads to

$$\delta_{n+1}^d \approx \left(\rho - \frac{1}{2}\right) h\mathbf{x}_d \tag{34}$$

This is an effective way of computing the local integration error because all the quantities in the right side of the previous equation are available at the end of the corrector phase.

Note that Eq. (33) is relevant for $\rho \neq 0.5$. The choice $\rho = 0.5$ corresponds to the trapezoidal formula, for which one additional term in the Taylor expansion would need to be considered throughout the derivation. This is qualitatively similar to the presentation herein and is not detailed further. Other choices of $\rho \in (\frac{1}{2}, 1]$ are viable, and it is insightful to compare Eq. (9) with the Newmark formula of Eq. (2b). This idea can be taken one step further and combined with the introduction of a fictitious variable $\mathbf{Z}$, defined as $\dot{\mathbf{Z}} = \mathbf{X}$. In this case Eq. (8a) leads to a second-order equation in $\mathbf{Z}$, in which case straight Newmark can be applied to find the solution $\mathbf{X}$. This approached is followed in [8].

The accuracy test

With the local truncation error in positions $\mathbf{q}$ and differential states $\mathbf{X}$ obtained as indicated in Eqs. (26) and (34), the numerical integrator has to certify at time t_{n+1} the accuracy of the newly computed solution. Two tests performed to this end are used to accept or reject the integration step. The tests are based on the value of the position and differential states composite errors, e^q and e^d, respectively:

$$e^q = \sqrt{\frac{1}{p}\sum_{i=1}^{p}\left(\frac{\delta_{i,n+1}^q}{Y_i^q}\right)^2} \qquad e^d = \sqrt{\frac{1}{n_d}\sum_{i=1}^{n_d}\left(\frac{\delta_{i,n+1}^d}{Y_i^X}\right)^2} \tag{35}$$

where $Y_i^q = \max(1, \max_{j=1,\ldots,n}|q_{i,j}|)$, and $\delta_{i,n+1}^q$, $1 \leq i \leq p$, is the i^{th} component of δ_{n+1}^q.

The composite error is compared with the user-prescribed error ϵ. Introducing the notation

$$\psi_q \equiv \frac{p\epsilon^2}{\left[\beta - \frac{1}{6(1+\alpha)}\right]^2} \tag{36a}$$

the error test $e^q \leq \epsilon$ is equivalently expressed as

$$\|\mathbf{x}\|_q^2 \leq \frac{\psi_q}{h^4} \tag{36b}$$

where $\|\cdot\|_q$ represents a weighted norm [3] defined as $\|\mathbf{x}\|_q \equiv \left[\sum_{i=1}^{p}\left(\frac{\mathbf{x}_i}{Y_i^q}\right)^2\right]^{\frac{1}{2}}$.

For the local integration error in the differential states, introducing the notation

$$\psi_d = \frac{n_d \cdot \epsilon^2}{\left(\rho - \frac{1}{2}\right)^2} \tag{37a}$$

the accuracy test $e^d \leq \epsilon$ leads to the requirement

$$\|\mathbf{x}_d\|_X^2 \leq \frac{\psi_d}{h^2} \tag{37b}$$

where the corresponding weighted norm is defined as $\|\mathbf{x}\|_X \equiv \left[\sum_{i=1}^{n_d} \left(\frac{\mathbf{x}_i}{Y_i^X}\right)^2\right]^{\frac{1}{2}}$. Note that all the quantities that enter the accuracy tests in Eqs. (36b) and (37b) are available after the nonlinear discretization system of Eq. (11) is solved, and a decision is made at that point whether the newly computed solution is accepted or rejected.

The step-size selection

Step-size selection plays a central role in the numerical integration algorithm. If $e^q \ll \epsilon$ and $e^d \ll \epsilon$, CPU time is wasted in computing a solution that unnecessarily exceeds the user-demanded accuracy. At the other end of the spectrum, a step-size selection mechanism that is too aggressive leads to a large number of integration steps at the end of which the user accuracy requirements are not met. The effort to perform such an integration step is wasted, as the integration step is discarded for a new attempt with a more conservative step-size h. To strike the right note, the integration step-size is always chosen such that the error at the end of the next integration step is precisely equal to the one deemed acceptable by the user and quantitatively defined by ϵ. By ignoring the terms of order h^4 and higher, and denoting $c_i = \left[\beta\,(1+\alpha) - \frac{1}{6}\right] \cdot \ddot{\mathbf{q}}_{i,n}$, Eq. (25) suggests that the position composite error is proportional to the cube of the step-size h. Ideally, the new step-size h_{new} is selected such that

$$\epsilon = h_{new}^3 \left[\frac{1}{p}\sum_{i=1}^{p}\left(\frac{c_i}{Y_i}\right)^2\right]^{\frac{1}{2}}$$

Therefore, $\frac{e}{\epsilon} = \frac{h^3}{h_{new}^3}$, from where $h_{new} = h\,\epsilon^{\frac{1}{3}}\left[\frac{1}{\sqrt{p}}\left(\beta - \frac{1}{6(1+\alpha)}\right)h^2 \cdot \|\mathbf{x}\|_q\right]^{-\frac{1}{3}}$. By defining

$$\Theta_q = \frac{\|\mathbf{x}\|_q^2 \cdot h^4}{\psi_q} \tag{38a}$$

the position-based criterion for selecting the step-size becomes

$$h_{new}^q = \frac{s\ h}{\Theta_q^{\frac{1}{6}}} \tag{38b}$$

As recommended in [20], a safety factor $s = 0.9$ was used to scale the value of the new step-size. The superscript q was added to indicate that this value of the new step-size is based on position considerations.

The approach for computing h_{new}^d follows step by step the position-based selection of h_{new}^q. Defining $s_d = \frac{\rho^{-\frac{1}{2}}}{\sqrt{n_d}}\left[\sum_{i=1}^{n_d}\left(\frac{\ddot{X}_{i,n}}{Y_i}\right)^2\right]^{\frac{1}{2}}$, it can be concluded that the error depends quadratically on the step-size h, like $e^d = s_d\ h^2$. Therefore, h_{new} should be selected such that $\epsilon = s_d\ h_{new}^2$, which leads to $h_{new} = h\,\epsilon^{\frac{1}{2}}\,e^{-\frac{1}{2}}$. Hence, $h_{new} = h\left[\frac{\psi_d}{h^2 \cdot \|\mathbf{x}_d\|_X^2}\right]^{\frac{1}{4}}$. By defining

$$\Theta_d = \frac{\|\mathbf{x}_d\|_X^2 \cdot h^2}{\psi_d} \tag{39a}$$

the differential-based criterion for selecting the step-size becomes

$$h_{new}^d = \frac{s\ h}{\Theta_d^{\frac{1}{4}}} \tag{39b}$$

where $s = 0.9$ is a safety factor [20].

The correction stage

The last issue that needs to be addressed is how accurate the quantities $\mathbf{x}$ and $\mathbf{x}_d$ of Eqs. (17) and (29), respectively, should be computed. These quantities are obtained as the solution of an iterative Newton-like algorithm that recycles the coefficient matrix in Eq. (12) and its LU factorization, and which requires at least one evaluation of the residuals in Eq. (13), followed by a forward/backward substitution to retrieve the corrections in the unknowns. However, one corrector iteration might be as expensive as doing all of the above but preceded by an evaluation and LU factorization of the coefficient matrix of the linear system of Eq. (12), two operations that are expensive and should be kept to a minimum.

Suppose that $\mathbf{x}$ is approximated by $\mathbf{x}^{(k)}$, the value obtained after k corrector iterations. Therefore, according to Eqs. (25) and (35), the composite error e^q is actually computed based not on the value $\mathbf{x}$, but rather on $\mathbf{x}^{(k)}$, which will lead to a value $e^{q,(k)}$. It is therefore important to have a good approximation $\mathbf{x}^{(k)}$ for $\mathbf{x}$ if the algorithm is to produce a reliable measure of the local integration error (a similar argument holds for the differential error e^d). Another reason for having an accurate approximation is that the stability and convergence results associated with a numerical integrator are derived under the assumption that the numerical solution is computed to the specifications of the integration formula; in other words, no room is left for errors in finding the numerical solution at the end of one integration step. Finding an approximate solution translates into solving a different initial value problem, which can be close to or far from the original problem based on how accurate the nonlinear system of Eq. (11) is solved and the nature of the original initial value problem itself. In summary, based on these two remarks, the corrector stopping criterion adopted here is that the relative difference between e and $e^{(k)}$ should stay smaller than a threshold value denoted by c. A typical value recommended in the literature is $c = 0.001$ [20]. The local integration error at the end of one time step is $e^q = \left[\beta - \frac{1}{6(1+\alpha)}\right] \frac{h^2}{\sqrt{p}} \|\mathbf{x}\|_q$. After iteration k, the approximation obtained is $e^{q,(k)} = \left[\beta - \frac{1}{6(1+\alpha)}\right] \frac{h^2}{\sqrt{p}} \|\mathbf{x}^{(k)}\|_q$. The question is what k should be such that $e^{q,(k)}$ is close to e^q within 0.1% (c = 0.001); that is, $|e^q - e^{q,(k)}| \leq c\,|e^q|$. Since e^q is not available, the test is replaced by

$$\left|\frac{e^q - e^{q,(k)}}{\epsilon}\right| \leq c \tag{40}$$

where ϵ is the user-prescribed error. Note that the goal of the step-size control is to keep e^q as close as possible to ϵ; therefore, substituting the original condition with Eq. (40) is acceptable. Then,

$$\left|e^q - e^{q,(k)}\right| \leq \left|\beta - \frac{1}{6\,(1+\alpha)}\right| \frac{h^2}{\sqrt{p}} \|\mathbf{x} - \mathbf{x}^{(k)}\|_q \tag{41}$$

and an approximation for $\|\mathbf{x} - \mathbf{x}^{(k)}\|_q$ is needed. Since for the Newton-like algorithm employed the convergence is linear, there is a constant ξ that for convergence must satisfy $0 \leq \xi < 1$ such that [15]

$$\|\triangle\mathbf{x}^{(k+1)}\|_q \leq \xi \cdot \|\triangle\mathbf{x}^{(k)}\|_q \tag{42}$$

where $\triangle\mathbf{x}^{(k)}$ represents the correction at iteration k, $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \triangle\mathbf{x}^{(k)}$. Immediately,

$$\|\mathbf{x} - \mathbf{x}^{(k+1)}\|_q \leq \|\triangle\mathbf{x}^{(k)}\|_q \cdot \frac{\xi}{1-\xi} \tag{43}$$

The value ξ is going to be approximated by

$$\xi \approx \xi_k = \frac{\|\triangle\mathbf{x}^{(k)}\|_q}{\|\triangle\mathbf{x}^{(k-1)}\|_q} \tag{44}$$

Based on Eq. (41),

$$\left|e^q - e^{q,(k)}\right| \leq \left[\beta - \frac{1}{6\,(1+\alpha)}\right] \frac{h^2}{\sqrt{p}} \|\triangle\mathbf{x}^{(k)}\|_q \cdot \frac{\xi}{1-\xi} \tag{45}$$

The condition of Eq. (40) is then satisfied as soon as

$$\left(\frac{\xi}{1-\xi}\right)^2 \|\triangle \mathbf{x}^{(k)}\|_q^2 \leq c^2 \cdot \frac{\psi_q}{h^4} \tag{46}$$

Note that at the right of the inequality sign are quantities that remain constant during the corrector iterative process, while at the left are quantities that change at each iteration. Likewise, note that the stopping criterion of Eq. (46) can be used only at the end of the second iteration because only then can an approximation of the convergence rate ξ be produced. In other words, the proposed approach will not be able to stop the iterative process after the first iteration. This is not a matter of great concern, however, because models as simple as a one-body pendulum are already nonlinear and require more than one iteration.

Qualitatively, the same approach used for the positions-based stopping criterion is used for the differential states. Without getting into details, this will lead to the following stopping criterion:

$$\left(\frac{\xi_d}{1-\xi_d}\right)^2 \cdot \|\triangle \mathbf{x}_d^{(k)}\|_X^2 \leq c^2 \cdot \frac{\psi_d}{h^2} \tag{47}$$

The prediction stage

For the Newton-like algorithm used to find the solution of Eq. (13), a good starting point is essential both for convergence and for reducing the effort in finding the approximation of the solution at time t_{n+1}. In [23], the generalized accelerations prediction is obtained by taking $\ddot{\mathbf{q}}_{n+1} = \ddot{\mathbf{q}}_n$ and $\dot{\mathbf{X}}_{n+1} = \dot{\mathbf{X}}_n$, which is equivalent to setting $\mathbf{x} = \mathbf{0}$ and $\mathbf{x}_d = \mathbf{0}$. A new strategy is proposed based on polynomial extrapolation, in which a polynomial of order up to three is used to produce an initial guess for the unknowns. The approach used in [23] is thus obtained by setting the degree of the interpolation polynomial to zero. The polynomial extrapolation is based on Newton divided differences and uses Horner's scheme for evaluation of the interpolant at time t_{n+1} [3]. The heuristics for choosing the degree of the interpolant would be based on the size of the numerical derivatives of the solution (evaluated by divided differences), and on the "smoothness" of the numerical solution as reflected in the number of rejected integration steps.

Summary of key formulas

Summarized below are the answers to the questions (a) What is the stopping criteria for the nonlinear discretization algebraic system? (b) How is the integration error computed? and (c) How is the step-size controlled?

Summary of key formulas for handling of the generalized coordinates.

Notation:

$$\psi_q \equiv \frac{p\epsilon^2}{\left[\beta - \frac{1}{6(1+\alpha)}\right]^2} \qquad \Theta_q = \frac{\|\mathbf{x}\|_q^2 \cdot h^4}{\psi_q} \tag{48a}$$

Prediction: Performed based on divided differences (Newton interpolation and Horner's scheme for extrapolation at t_{n+1}).

Correction: Linear convergence rate allows for computation of ξ (Eq. (44)). Stopping criterion:

$$\left(\frac{\xi}{1-\xi}\right)^2 \|\triangle \mathbf{x}^{(k)}\|_q^2 \leq c^2 \, \frac{\psi_q}{h^4}, \qquad (c = 0.001) \tag{48b}$$

Accuracy Check: Performed after corrector converged,

$$\Theta_q \leq 1 \tag{48c}$$

Step-Size Selection: With a safety factor $s = 0.9$,

$$h_{new}^{q} = \frac{s\ h}{\Theta_q^{\frac{1}{6}}} \tag{48d}$$

Summary of key formulas for handling of the differential states.

Notation:

$$\psi_d = \frac{n_d \cdot \epsilon^2}{\left(\rho - \frac{1}{2}\right)^2} \qquad\qquad \Theta_d = \frac{\|\mathbf{x}_d\|_X^2 \cdot h^2}{\psi_d} \tag{49a}$$

Correction: Linear convergence rate allows for computation of ξ_d. Stopping criterion:

$$\left(\frac{\xi_d}{1-\xi_d}\right)^2 \|\triangle \mathbf{x}_d^{(k)}\|_X^2 \leq c^2\ \frac{\psi_d}{h^2}, \qquad (c = 0.001) \tag{49b}$$

Accuracy check: Performed after corrector converged,

$$\Theta_d \leq 1 \tag{49c}$$

Step-size selection: With a safety factor $s = 0.9$,

$$h_{new}^{d} = \frac{s\ h}{\Theta_d^{\frac{1}{4}}} \tag{49d}$$

In multibody dynamics simulations, the number of differential states is orders of magnitude smaller than the number of states associated with the generalized coordinates; that is, $n_d \ll p$. Nevertheless, the stopping criteria as well as the selection of the new step-size h_{new} take into account both the position and differential states. For stopping the Newton-like algorithm, iterations are carried out until the conditions of Eqs. (48b) and (49b) are simultaneously satisfied. The new step size is chosen as $h_{new} = \min\left(h_{new}^{q}, h_{new}^{d}\right)$. An integration step is not accepted unless both accuracy conditions of Eqs. (48c) and (49c) are satisfied.

NUMERICAL EXPERIMENTS

The proposed algorithm has been implemented in the commercial simulation package MSC.ADAMS and released in its 2005 version. The algorithm has been extensively tested with more than 1,600 mechanical systems of various complexity. Three representative numerical experiments aimed at comparing the HHT-based algorithm and GSTIFF, the default integrator in the MSC.ADAMS [29] simulation package, are presented herein. The comparison primarily focuses on efficiency issues, although the accuracy of the results is touched upon.

A Poly-V belt model

The model in Fig .1 is an accessory drive for a car engine with a poly-V belt (V-ribbed belt) wrapped around three pulleys (crank, water pump, and alternator), and one tensioner (deviation pulley). The poly-V belt provides drive by adhesion, and compared with conventional belts of the same width, it augments the contact area, increasing power transfer. The larger of the pulleys (the lowest one in the picture) is the engine crankshaft pulley. Right above it is the water-pump pulley, and at the left is the alternator pulley. There is a tensioner in between the crankshaft and alternator pulleys; its pivot point is shown in the figure as the little ring just outside the belt. The tensioner uses a rotational spring element that includes damping and stiffness effects. The units used for this model are Newton, kilograms, milliseconds, and millimeters.

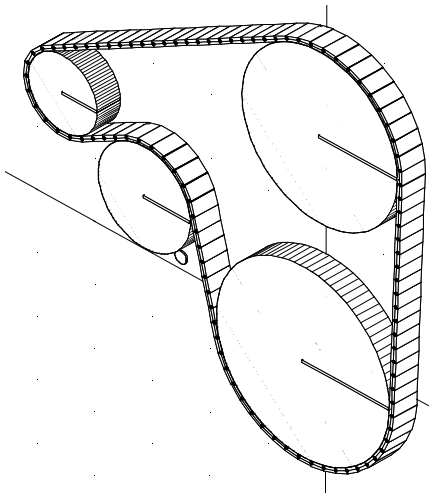

Figure 1: Poly-V accessory belt.

One driving torque is on the crankshaft, and two resisting torques act on the alternator and water pulleys. The belt is modeled by using 100 segments connected by a set of 400 VFORCE elements [29]. The length of the simulation was 200 milliseconds, which results in more than a full revolution of the belt. The GSTIFF integrator was run with ERROR=1.E-4, while HHT was run with ERROR=1.E-5 (this is the ϵ variable of Eqs. (48a) and (49a)). Because of the different error control strategies employed by the two integrators, it has been noticed that GSTIFF typically can be run with a more lax ERROR for results that are qualitatively similar to HHT. The simulation time for GSTIFF was 1286 seconds, while HHT completed the simulation in 485 seconds. The simulation was run on a Windows 2000 machine, with Pentium III CPU, and 512 MB RAM.

Figure 2 shows the X-component of the reaction force in the revolute joint that connects the alternator with the rest of the system. The agreement of the results is very good: the peak difference between the two sets of results is less than 3%.

Figure 4 shows the time variation of the angular velocity of the alternator. The plot displays good correlation between the results obtained with the GSTIFF and the HHT integrators. Figure 5 confirms that the difference between the angular velocity computed with GSTIFF and HHT integrator is less than 1%. This value is smaller than the 3% noticed for the difference in force in the alternator joint. This is an expected trend all across the simulation results, where the quality of the velocity level variables is better than the quality of the force/acceleration variables. Although not shown here, the position-level variables for the two integrators are practically identical, and in general they are qualitatively better than the velocity-level results obtained with the two integrators.

A track model

The track presented in Fig. 6 is a detailed model of a subsystem of a low-mobility hydraulic mining excavator. Weight and extreme operating conditions cause high mechanical stresses on crawler tracks especially in the case of big hydraulic excavators of 1,000 tons and higher. Long haulage distances, frequent place changes, and demanded 90% machine availabilities are standard requirements in the industry. The type of simulation reported here is typical in the virtual prototyping cycle when trying to meet these requirements and extend life cycles of the track and drive systems.

The model in Fig. 6 contains a set of 61 moving parts. It has one planar joint, 54 revolute joints, 1 translational joint, 4 fixed joints (a joint that removes all six degrees of freedom), one inline joint primitive, and one motion. This results in a model with 61 degrees of freedom. The relatively large number of degrees of freedom indicates that the motion of this system is not controlled as much through constraints (joints) as through the geometry of the components and 690 three-dimensional contact elements (sprocket-track, roller-track, track-track, and track-ground contacts). The HHT integrator resulted in a set of 4,675 equations for the dynamic analysis of this model.

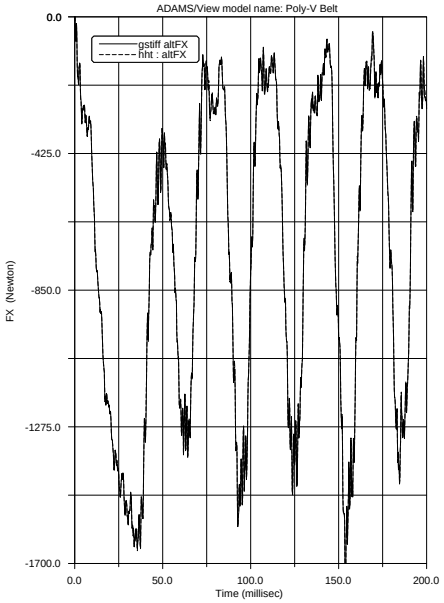

Figure 2: X-comp. of reaction force.

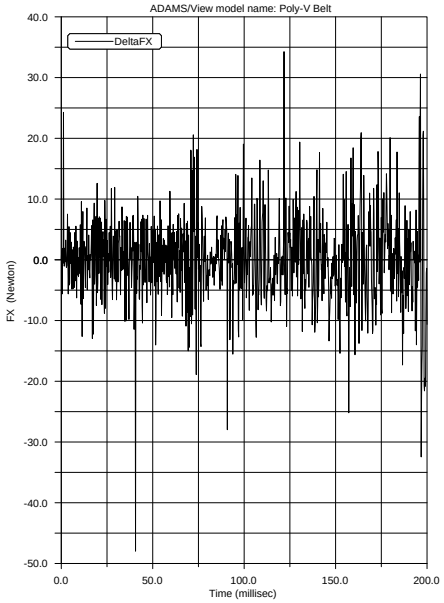

Figure 3: HHT and GSTIFF differences.

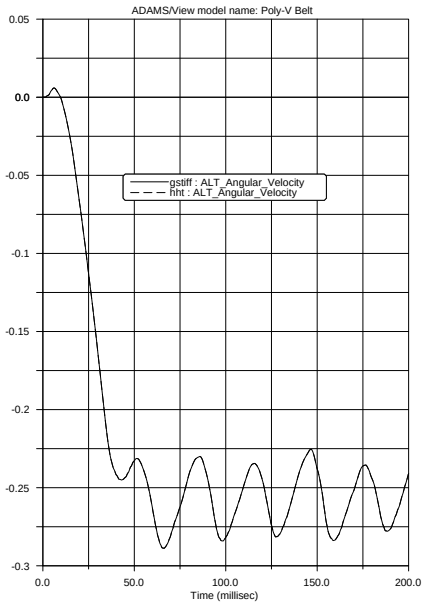

Figure 4: Alternator angular velocity.

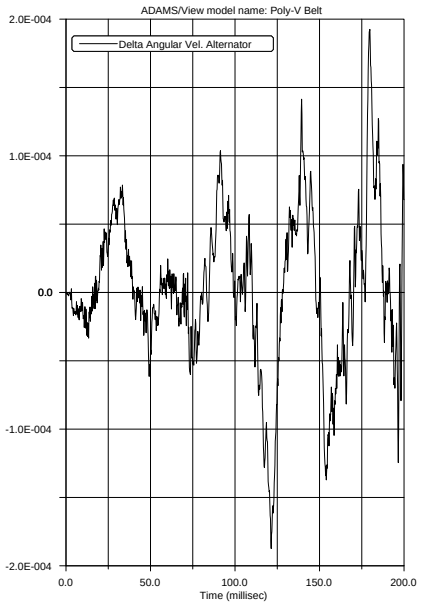

Figure 5: Alternator force difference.

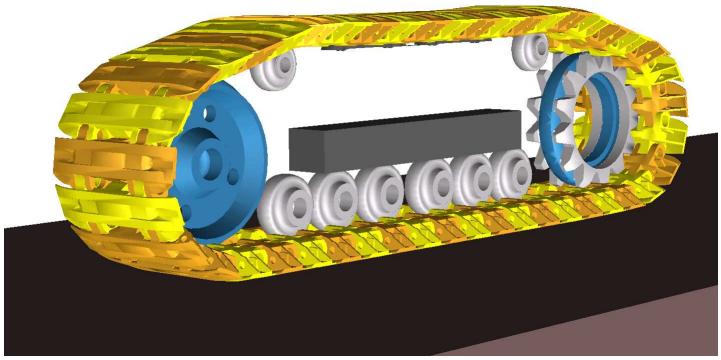

Figure 6: Track subsystem model.

A 14-second simulation was run on a Windows XP Dell Precision M530 machine, with 2 GB ECC RAM, and 2.8 GHz HyperThread Xeon CPUs. A first set of results was obtained with the default version of the ADAMS/Solver. The key integrator settings were as follows: GSTIFF integrator, stabilized index 2 (SI2) DAE formulation [19], ERROR=1E-2, KMAX=1 (to reflect the 3D contact-induced discontinuous nature of the simulation), MAXIT=10. The HHT integrator was run in the beta version of the 2005 release, with the following key settings: ERROR=1E-5, MAXIT=10, DAE formulation was index 3 [27, 6]. Note the difference between the ERROR settings in the two cases. This is explained by the different ways in which the error is quantified by the two integrators. In this context, the SI2-GSTIFF integrator has a much stricter interpretation of the user-set ERROR. In has been noticed that in order to obtain qualitatively comparable results, the HHT ERROR setting should be two orders of magnitude more stringent than that of the SI2-GSTIFF integrator. To be on the safe side, for this model the HHT integrator was run with an even tighter error setting, ERROR=1.E-5, which actually is the HHT default setting for this attribute.

The speedup obtained when using the HHT integrator was more than fivefold: it took 1,713 seconds for the simulation to complete when using the HHT integrator, while it took 8,988 seconds for the GSTIFF integrator to finish the same simulation. The quality of the new results is very good. The accelerations are always the most likely to show differences. Differences can only rarely be noticed in velocity results, while the quality of the position level results is almost always very good in both integrators. For comparison, in Fig. 7 the acceleration and velocity results are displayed for track number 8. The results match overall very well everywhere with the exception of some spikes that are explained by the sensitivity of the simulation with respect to the large number of contact forces present in the model.

A model with flexible bodies

The model in Fig. 8 is an all-terrain vehicle (ATV) with a flexible frame. Beneath the ATV is a 4-post shaker device, which is used to simulate a durability event represented by a simple-harmonic translational displacement constraint at each of the four posts (pitch-type excitation, of approximately 2Hz).

The frame component in Fig. 8 is an MSC.ADAMS flexible-body modal representation [12] that was created with the MSC.Patran finite element package; it contains 134 modes (from 56.7 Hz to 13.1 kHz). The purpose of this simulation is to recover the von Mises stresses in the flexible body and identify critical stress locations in the frame.

The steering system has a motion constraint that applies a rotational displacement function, causing the front wheels to turn left to right in a sinusoidal fashion. The tires of the vehicle interact with the shaker by means of three dimensional solid-to-solid contact forces. A rider weight of 230 pounds has been approximated with lumped masses and distributed between the steering column assembly and frame as 30 pounds and 200 pounds, respectively. Remaining parts in the model that are attached to the frame, such as the engine, are modeled as lumped masses and are connected by using fixed joints.

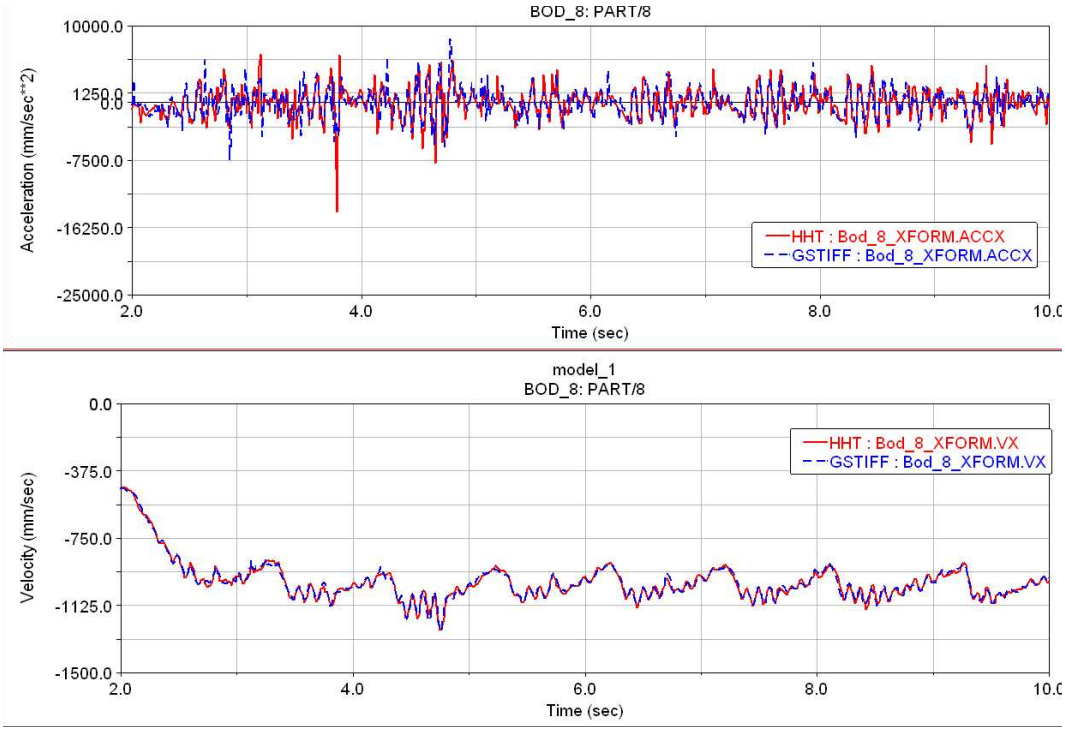

Figure 7: Acceleration and velocity of track 8.

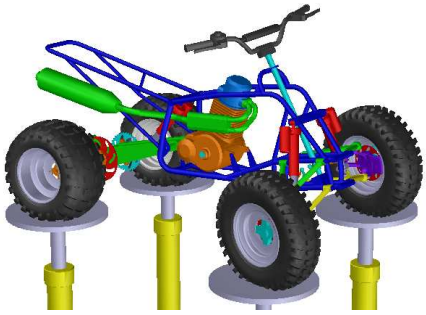

Figure 8: All-terrain vehicle (ATV).

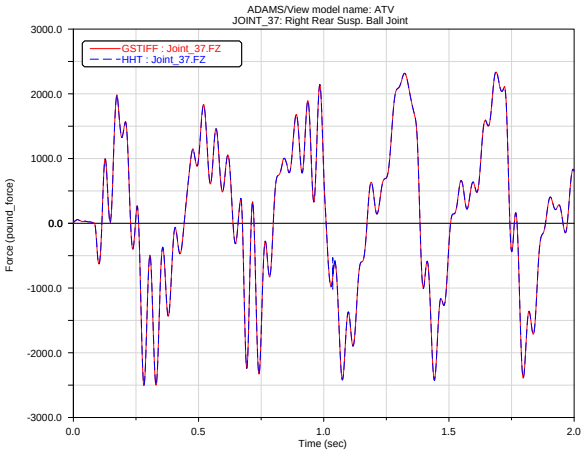

Figure 9: Comparison of vertical reaction force.

Table 1: Top Ten von Mises Hot Spots for Flexible Frame

GSTIFF				HHT				Stress Difference
No.	Point	Stress (lbf/inch^2)	Time (sec)	No.	Point	Stress (lbf/inch^2)	Time (sec)	GSTIFF vs. HHT
1	23956	300738	1.104	1	23956	300768	1.104	0.01%
2	26654	204503	0.746	2	26654	204274	0.746	-0.11%
3	33918	204075	0.206	3	33918	204191	0.206	0.06%
4	46577	200688	0.746	4	46577	200354	0.746	-0.17%
5	34560	198106	0.211	5	34560	197887	0.211	-0.11%
6	46580	193729	0.326	6	46580	192929	0.326	-0.41%
7	30060	190386	0.692	7	30060	189911	0.692	-0.25%
8	24704	173882	0.281	8	24704	173355	0.281	-0.30%
9	36156	171990	1.446	9	36156	172746	1.446	0.44%
10	23007	170191	1.100	10	23007	170375	1.100	0.11%

Altogether there are 28 moving parts, 43 joints, 5 motions, and 1 flexible body, leading to a total number of 532 equations for the HHT integrator and 927 equations for the Index-3 GSTIFF (I3) formulation. The MSC.ADAMS modeling units used for this experiment were pound_force, pound_mass, inch, and second. The duration of the simulation was 2.0 seconds, which allows the ATV to move up and down several times. The GSTIFF integrator was run with ERROR=1e-4, while HHT was run with ERROR=1e-5. Considering the typical use of a simulation like this, namely in durability (fatigue) analysis, a maximum time step (HMAX=5e-4) was specified for accurate post-processing hot-spot stress-recovery.

The simulation CPU time for GSTIFF was 367.08 seconds, while HHT completed the simulation in 122.47 seconds. The simulation was run by using MSC.ADAMS 2005 on a Windows 2000 laptop computer with a single 2.20 GHz Pentium 4 CPU and 1GB RAM.

There is excellent agreement in location of critical stresses as presented in Table 1; the difference in peak stress between the two sets of results is less than 0.5%. The Z-component of the reaction force in the spherical joint that connects the right half of the rear suspension component to the frame is presented in Fig. 9. The plot shows good correlation between the force results obtained with the GSTIFF and HHT integrators. Figure 10 presents the time variation of the angular velocity of the engine assembly, and is an indicator of the severity of the ATV pitching behavior. Figure 11 confirms that the difference between the angular velocity computed with GSTIFF and HHT integrators is less than 1.6%.

CONCLUSIONS

The HHT method used in structural dynamics and later introduced in the context of multibody dynamics simulation in [9] serves as the starting point of an algorithm implemented and validated in an industrial strength mechanical system simulation package. Strategies for corrector stopping criteria, error estimation, and step-size control were presented in detail. A set of real-life numerical experiments indicate that simulations are at least two to three times faster when compared with the default BDF-based integrator used in ADAMS [18, 29]. An explanation for the improved performance is based on three key observations. (1) The most time consuming part of simulation is the computation of the Jacobian associated with the nonlinear discretization system. The proposed algorithm contains heuristics to reduce as much as possible the number of Jacobian evaluations. Unlike the BDF integrator employed by GSTIFF, in which terms of the integration Jacobian can become disproportionately large as a result of a scaling by the inverse of the step-size, the proposed integrator employs a different approach where certain values are multiplied (never divided) by the step-size prior to populating the Jacobian. As long as the step-size does not

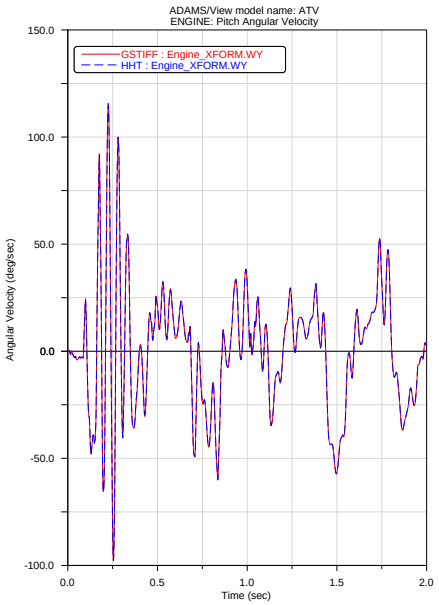

Figure 10: Engine pitch angular velocity.

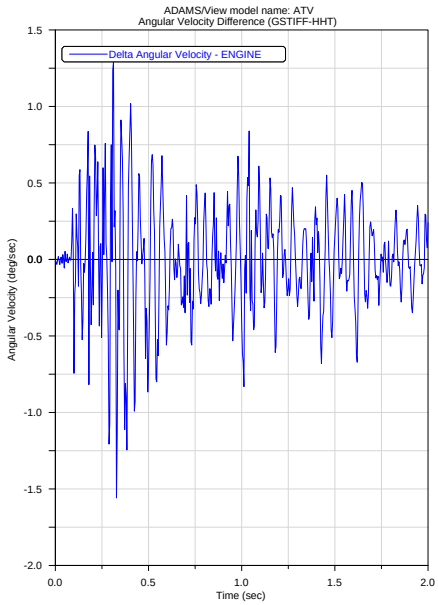

Figure 11: Angular velocity difference.

significantly change over several consecutive time steps, this approach better supports the recycling of the Jacobian. (2) When compared with the BDF Jacobian, the HHT Jacobian is numerically better conditioned, thereby leading to more reliable corrections in the Newton-like iterative approach for large problems. Typically, this results in a smaller number of corrector iterations. This desirable attribute is further enhanced by the fact that since certain partial derivatives are scaled by the step-size h, or by h^2 prior to populating the Jacobian, small errors in these partial derivatives are going to have a less negative effect on the overall quality of the Jacobian. (3) On one hand, the BDF formulas of order higher than one contain regions of instability in the left plane. The higher the order, the smaller the region of stability. On the other hand, BDF intrinsically is designed to maximize the integration order/step-size. Because of these two conflicting attributes, particularly for models that are mechanically stiff (models with stiff springs, flexible bodies, etc., that lead to systems with large eigenvalues close to the imaginary axis) an order/step-size choice often lands the BDF integrator outside the stability region [20]. These integration time-steps typically end up being rejected, and smaller step-sizes are required to advance the simulation. This is a non issue with the HHT method, which is a fixed low-order method with good stability properties in the whole left plane.

It should be pointed out that there are situations when BDF-type formulas are going to work significantly faster. These are the cases where BDF can sustain a high integration order throughout the simulation. If the model simulated allows BDF to work at order 5 or 6, the HHT method cannot produce a solution of similar quality in comparable CPU time because of the low-order integration formulas employed. However, this scenario is not very common, because most real-life large models contain discontinuities or stiff mechanical components that typically limit the BDF integration order to 1 or 2. As seen in the numerical experiments presented, in these cases the HHT-based algorithm has proved to be very competitive.

The accuracy of the results is good, occasional spikes in accelerations and reaction forces being explained by the use of a variable step integration algorithm for the solution of an index 3 DAE problem, an operation that is conjectured in [7] to further reduce the order of an already low-order method. Quantitatively, the simulation results can be improved by decreasing the user-specified integration error; qualitatively, the results could be improved by using a Runge-Kutta method as proposed in [24], using the generalization of the HHT method as proposed in [11], or reducing the index of the problem in an approach similar to the one proposed in [19], an alternative that is currently under investigation [25].

ACKNOWLEDGMENTS

The authors thank Dipl.-Ing. Holger Haut of the University of Aachen, Germany, for providing images, results plots, and timing results for the track subsystem simulation, and Andrei Schaffer of MSC.Software for his suggestions. The authors thank an anonymous reviewer for pointing out additional reference material for the concept of DAE index.

References

[1] K.S. Anderson and M. Oghbaei. A dynamics simulation of multibody systems using a new state-time methodology. *Multibody Systems Dynamics*, 14(1):61–80, 2005.

[2] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia, PA, 1998.

[3] K. E. Atkinson. *An Introduction to Numerical Analysis*. John Wiley & Sons Inc., New York, second edition, 1989.

[4] O. A. Bauchau, C. L. Bottasso, and L. Trainelli. Robust integration schemes for flexible multibody systems. *Computer Methods in Applied Mechanics and Engineering*, 192:395 – 420, 2003.

[5] J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 1:1–16, 1972.

[6] K. Brenan and B. E. Engquist. Backward differentiation approximations of nonlinear differential/algebraic systems. *Mathematics of Computation*, 51(184):659–676, 1988.

[7] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. North-Holland, New York, 1989.

[8] O. Bruls, P. Duysinx, and J. Golinval. A unified finite element framework for the dynamic analysis of controlled flexible mechanisms. In *Proceeding of Multibody Dynamics ECCOMAS Thematic Conference*, 2005.

[9] A. Cardona and M. Geradin. Time integration of the equation of motion in mechanical analysis. *Computer and Structures*, 33:881–820, 1989.

[10] A. Cardona and M. Geradin. Numerical integration of second order differential-algebraic systems in flexible mechanics dynamics. In M. F. O. S. Pereira and J. A. C. Ambrosio, editors, *Computer-Aided Analysis of Rigid and Flexible Mechanical Systems*. Kluwer Academic Publishers, 1994.

[11] J. Chung and G. M. Hulbert. A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-α method. *Transactions of ASME, Journal of Applied Mechanics*, 60(2):371–375, 1993.

[12] R. R. Craig and M. C. C. Bampton. Coupling of substructures for dynamics analyses. *AIAA Journal*, 6(7):1313–1319, 1968.

[13] J. Cuadrado, D. Dopico, M. N. Naya, and M. Gonzalez. Penalty, semi-recursive and hybrid methods for MBS Real-Time dynamics in the context of structural integrators. *Multibody Systems Dynamics*, (12):117–132, 2004.

[14] J. Garcia de Jalon and E. Bayo. *Kinematic and Dynamic Simulation of Multibody Systems. The Real-Time Challenge*. Springer-Verlag, Berlin, 11994.

[15] J. Dennis and R. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1983.

[16] E. Eich-Sollner and C. Fuhrer. *Numerical Methods in Multibody Dynamics*. Teubner-Verlag, Stuttgart, 1998.

[17] B. Gavrea, D. Negrut, and F. Potra. The Newmark integration method for simulation of multibody systems: analytical considerations (IMECE 2005-81770). In *Proceedings of the International Mechanical Engineering Congress and Exposition, Orlando, Florida*. ASME, 2005.

[18] C. W. Gear. *Numerical Initial Value Problems of Ordinary Differential Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1971.

[19] C. W. Gear, G. Gupta, and B. Leimkuhler. Automatic integration of the Euler-Lagrange equations with constraints. *J. Comp. Appl. Math.*, 12:77–90, 1985.

[20] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations*, volume II of *Computational Mathematics*. Springer-Verlag, 1991.

[21] E. J. Haug. *Computer-Aided Kinematics and Dynamics of Mechanical Systems*. Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[22] H. M. Hilber, T. J. R. Hughes, and R. L. Taylor. Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Eng. and Struct. Dynamics*, 5:283–292, 1977.

[23] T. J. R. Hughes. *Finite Element Method - Linear Static and Dynamic Finite Element Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[24] L. O. Jay. Structure preservation for constrained dynamics with super partitioned additive Runge-Kutta methods. *SIAM J. Sci. Comput.*, 20:416–446, 1998.

[25] L. O. Jay and D. Negrut. On an extension of the HHT method for index 3 differential algebraic equations. *in preparation*, 2006.

[26] P. Kunkel and V. Mehrmann. *Differential algebraic equations. Analysis and Numerical Solution*. European Math. Soc. Textbooks in Mathematics, 2006.

[27] C. Lötstedt and L. Petzold. Numerical solution of nonlinear differential equations with algebraic constraints I: Convergence results for backward differentiation formulas. *Mathematics of Computation*, 174:491–516, 46.

[28] C. Lubich, U. Nowak, U. Pohle, and C. Engstler. MEXX - numerical software for the integration of constrained mechanical multibody systems. *Mechanics of Structures and Machines*, 23:473–495, 1995.

[29] MSCsoftware. ADAMS User Manual. Also available online at http://www.mscsoftware.com, 2005.

[30] N. M. Newmark. A method of computation for structural dynamics. *Journal of the Engineering Mechanics Division, ASCE*, pages 67–94, 1959.

[31] B. Owren and H. Simonsen. Alternative integration methods for problems in structural dynamics. *Structural Dynamics. Comput. Meth. in Appl. Mech. and Eng.*, 122:1–10, 1995.

[32] L. A. Pars. *A Treatise on Analytical Dynamics*. John Wiley & Sons, New York, 1965.

[33] L. R. Petzold. Differential-algebraic equations are not ODE's. *SIAM J. Sci., Stat. Comput.*, 3(3), 1982.

[34] F. A. Potra. Implementation of linear multistep methods for solving constrained equations of motion. *SIAM. Numer. Anal.*, 30(3), 1993.

[35] M. Schaub and B. Simeon. Automatic h-scaling for the efficient time integration of stiff mechanical systems. *Multibody System Dynamics*, 8:329–345, 2002.

[36] A. A. Shabana. *Computational Dynamics*. John Wiley & Sons, 1994.

[37] G. Le Vey. Differential algebraic equations - a new look at the index. Technical report, Institut de Recherche en Informatique et Systemes Aleatoires, 1994.

[38] J. Yen, L. Petzold, and S. Raha. A time integration algorithm for flexible mechanism dynamics: The DAE α-method. Technical Report TR96-024, Dept. of Comp. Sci., University of Minnesota, 1996.

[39] O.C. Zienkiewicz and Y.M. Xie. A simple error estimator and adaptive time-stepping procedure for dynamic analysis. *Earthquake Engineering and Structural Dynamics*, 20:871–887, 1991.