A State–Space Based Implicit Integration Algorithm for Differential–Algebraic Equations of Multibody Dynamics

E. J. Haug[†], **D. Negrut**[†], **M. Iancu**[‡]

January 28, 1997

To Appear

Mechanics of Structures and Machines

Abstract. An implicit numerical integration algorithm based on generalized coordinate partitioning is presented for the numerical solution of differential–algebraic equations of motion arising in multibody dynamics. The algorithm employs implicit numerical integration formulas to express independent generalized coordinates and their first time derivative as functions of independent accelerations at discrete integration times. The latter are determined as the solution of discretized equations obtained from state–space, second order ordinary differential equations in the independent coordinates. All dependent variables in the formulation, including Lagrange multipliers, are determined by satisfying the full system of kinematic and kinetic equations of motion. The algorithm is illustrated using the implicit trapezoidal rule to integrate the constrained equations of motion for three stiff mechanical systems with different generalized coordinate dimensions. Results show that the algorithm is robust and has the capability to integrate differential–algebraic equations of motion for stiff multibody dynamic systems.

† Department of Mechanical Engineering, The University of Iowa, Iowa City, IA 52242
‡ Department of Mathematics, The University of Iowa, Iowa City, IA 52242

1 Introduction

Implicit numerical integration methods, specifically the Newmark family of one step methods, are extensively used in solving large scale problems in structural dynamics (Hughes, 1987). The numerical integration method presented in this paper is motivated by the approach used in structural dynamics, specifically formulated to treat the differential–algebraic equations of multibody dynamics. While applications presented in this paper employ the Newmark method, the formulation is suitable for implementation using a broad spectrum of numerical integration methods.

In this paper, $\mathbf{q} = [q_1, q_2, \ldots, q_n]^T$ denotes the vector of generalized coordinates that defines the state of the mechanical system. For rigid bodies, the generalized coordinates may be absolute (Cartesian) coordinates of body reference frames, relative coordinates between bodies, or a combination of both. Joints connecting the bodies of a mechanical system restrict their relative motion and impose constraints on the generalized coordinates. In the most general case, constraints are expressed as algebraic expressions involving generalized coordinates and their first time derivatives. In the algorithm presented in this paper, only the case of holonomic scleronomic constraints is considered. This type of constraint takes the form

$$\Phi(\mathbf{q}) \equiv [\Phi_1(\mathbf{q}), \Phi_2(\mathbf{q}), \ldots, \Phi_m(\mathbf{q})]^T = \mathbf{0} \tag{1}$$

Differentiating Eq. 1 with respect to time yields the kinematic velocity equation,

$$\Phi_{\mathbf{q}}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0} \tag{2}$$

where subscript denotes partial differentiation; i.e., $\Phi_{\mathbf{q}} = \left[\frac{\partial \Phi_i}{\partial q_j}\right]$, and an over dot denotes differentiation with respect to time. Finally, differentiating Eq. 2 with respect to time yields the kinematic acceleration equation,

$$\Phi_{\mathbf{q}}(\mathbf{q})\ddot{\mathbf{q}} = -\left(\Phi_{\mathbf{q}}\dot{\mathbf{q}}\right)_{\mathbf{q}}\dot{\mathbf{q}} \equiv \tau(\mathbf{q}, \dot{\mathbf{q}}) \tag{3}$$

Equations 1 through 3 characterize the admissible motion of the mechanical system.

The configuration of the mechanical system will change in time under the effect of both applied and constraint forces. The Lagrange multiplier form of the constrained equation of motion for the mechanical system is

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^{T}(\mathbf{q})\lambda = \mathbf{Q}^{A}(\mathbf{q}, \dot{\mathbf{q}}, t) \tag{4}$$

where $\mathbf{M}$ is the system mass matrix that may depend on generalized coordinates, λ is the vector of Lagrange multipliers that account for workless constraint forces, and $\mathbf{Q}^{A}$ is the vector of generalized applied forces that may depend on time, generalized coordinates, and their time derivatives.

Equations 1 and 4 comprise a system of differential–algebraic equations (DAE). It is known (Petzold, 1982) that DAE are not ordinary differential equations (ODE). While analytically satisfying Eqs. 1 and 4 assures that Eqs. 2 and 3 are also satisfied, when the problem is solved numerically, this ceases to be the case. In general, the task of obtaining a numerical solution of the DAE of Eqs. 1 and 4 is substantially more difficult and prone to intense numerical computation than one of solving ODE. In particular, the index (Brenan et. al., 1989) of the DAE of multibody dynamics is 3, which makes its numerical treatment even more difficult.

In this paper, an algorithm that implicitly integrates an independent subset of generalized coordinates is presented. It is showed that the proposed algorithm can be applied with good results in numerical integration of index 3 DAE for a double pendulum, a slider–crank, and a seven body mechanism. In Section 2, a

review of the literature summarizes the most important methods currently used for numerical integration of index 3 DAE of multibody dynamics. In Section 3, the proposed algorithm is detailed. Section 4 contains results of numerical experiments carried out to validate the proposed algorithm. Finally, Section 5 presents conclusions.

2 Review of Literature

An early algorithm that allowed for integration of DAE is based on constraint stabilization (Baumgarte, 1972). The problem is reduced to an index 1 formulation, in which all generalized coordinates are integrated. The right side of the kinematic acceleration equation of Eq. 3 is replaced by

$$\bar{\tau} = \tau - 2\alpha\dot{\Phi} - \beta\Phi \tag{5}$$

Compensation terms proportional to the residual in the position and velocity kinematic constraint equations are added to the original right side of the acceleration equation. Ostermeyer (1990) discusses criteria for choosing the coefficients α and β. This process is problematic (Ascher et. al., 1995), and is yet to be solved. While simple to implement and numerically cheap, this method modifies the dynamics of the system being simulated by shifting its poles. Furthermore, integration of all the system variables results in error accumulation, leading to a lack of robustness.

A second class of algorithms is based on so called projection techniques (Eich et. al. 1990). They integrate all generalized coordinates at each time step. Additional multipliers are introduced to account for the requirement that the solution satisfy the constraint equations at the position, velocity, and for some formulations, the acceleration level. Gear et al. (1985) reduce the DAE to an analytically equivalent index 2 problem. Projection is performed only at the velocity level, an extra multiplier μ being introduced to insure that the kinematic velocity

constraint equations are satisfied. This algorithm uses a backward differentiation formula (BDF) to discretize the following form of the equations of motion:

$$\dot{\mathbf{q}} = \mathbf{v} - \Phi_{\mathbf{q}}^{T}(\mathbf{q})\mu \tag{6}$$
$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \mathbf{Q}^{A} - \Phi_{\mathbf{q}}^{T}(\mathbf{q})\lambda$$
$$\Phi(\mathbf{q}) = \mathbf{0}$$
$$\Phi_{\mathbf{q}}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}$$

In a similar approach proposed by Fuhrer and Leimkuhler (1991), the DAE is reduced to index 1 and an additional multiplier η is introduced, along with the requirement that the acceleration kinematic equations are satisfied.

In an analytical framework, all the additional multipliers introduced can be proved to be zero. Under discretization, however, these multipliers assume nonzero values, due to truncation errors of the integration formula being used. They thus coerce the numerical solution to satisfy constraint equations at some, or at all levels. While being robust, these formulations require a great deal of computation, when compared to other techniques. The solution of the index 1 formulation presented by Fuhrer and Leimkuhler (1991), for example, requires solution of $2n + 3m$ nonlinear equations, while the formulation of Gear et al. (1985) involves a nonlinear system of $2n + 2m$ equations.

Starting from the index 1 formulation with additional multipliers, Fuhrer and Leimkuhler (1991) propose an algorithm based on an index 1 formulation with no extra multipliers. All variables are integrated, and constraint equations at the position, velocity, and acceleration level are imposed. Under discretization, an overdetermined set of $2n + 3m$ nonlinear equations in $2n + m$ unknowns must be solved at each integration step. Discretization is carried out using backward differentiation type formula. Because of truncation errors, the equations become inconsistent and can only be solved in a generalized sense. While the so called ssf–solution obtained using a special oblique projection technique is, in the case of

linear constraint equations, equivalent to that obtained by integrating the state–space form using the same discretization formula, this ceases to be the case in general. The method is robust, and is comparable in terms of efficiency to the index 1 formulation with additional multipliers.

Another class of algorithms for solving DAE is based on the state–space reduction (Potra and Rheinboldt, 1991). The DAE is reduced to an equivalent ODE, via parametrization of the constraint manifold. The dimension of the equivalent state–space ODE is significantly reduced to ndof = n − m. The method has the potential of using well established theory and reliable numerical techniques for the solution of ODE. Since constraint equations in multibody dynamics are generally nonlinear, the parametrization can only be determined locally. A computational overhead results each time the parametrization is changed. Nonlinearity also leads to computational effort in retrieving the dependent generalized coordinates through the parametrization. This stage requires the solution of a set of nonlinear equations for which Newton like methods are generally used. This process is carried out once during each integration step for an explicit integration formula.

The choice of constraint parametrization differentiates among the algorithms in this class. The most used parametrization is one based on an independent subset of generalized coordinates of the mechanical system (Wehage, Haug, 1982). The partition of variables into dependent and independent sets is based on **LU** factorization of the constraint Jacobian matrix, and is presented in detail in Section 3. This partition is maintained as long as the subjacobian matrix with respect to the dependent coordinates is not ill conditioned. The method has been used extensively with large scale applications in multibody dynamics and has proved to be reliable and accurate.

Rather than considering a subset of generalized coordinates as parametrization variables, the algorithm presented by Mani et al. (1985) uses linear

combinations of coordinates. The parametrization variables z_i are components of the vector $\mathbf{z} = \mathbf{V}^I\mathbf{q}$, where $\mathbf{q}$ is the vector of generalized coordinates, and $\mathbf{V}^I \in \mathbb{R}^{ndof \times n}$ contains the last ndof rows of the matrix $\mathbf{V}$ in the singular value decomposition (SVD) $\Phi_\mathbf{q} = \mathbf{U}^T\mathbf{D}\mathbf{V}$ of the constraint Jacobian matrix. This parametrization, sometimes called the tangent plane parametrization because $range(\mathbf{V}^{I^T}) = ker(\Phi_\mathbf{q})$, results in a state–space ODE that is numerically well conditioned. However, the condition above only holds for the time instant at which the SVD is carried out. The parametrization is used for several subsequent steps, so in this respect it is not a truly tangent space parametrization.

The differentiable null space method presented by Liang and Lance (1987) reduces the DAE to an equivalent state–space ODE by projecting the equations of motion onto the tangent hyperplane of the manifold. The projection is done before discretization, and Lagrange multipliers are eliminated from the problem. The algorithm requires a set of ndof vectors that span the constraint manifold tangent hyperplane, along with their first time derivatives. The Gram–Schimdt method is used to obtained this information. The algorithm is efficient and robust, the equivalent ODE problem of dimension ndof being well conditioned. The implementation of an implicit formula to integrate the resulting state–space ODE is difficult however, because the Gram–Schmidt process is embedded in the algorithm.

3 Proposed Algorithm

The algorithm proposed in this paper uses a parametrization of the manifold in terms of independent generalized coordinates to reduce the DAE to a state–space ODE in these variables. The solution of the DAE at each time step is obtained by integrating the ODE with an implicit formula, to obtain the independent coordinates as the solution of a system of ndof nonlinear equations. Dependent coordinates and Lagrange multipliers are recovered from kinematic and kinetic equations, all of

which are satisfied at each iteration. This approach is similar to one proposed earlier by Haug and Yen (1992). What differentiates the present algorithm is the strategy for solving systems of nonlinear equations that result from discretization of the state–space ODE. In the method of Haug and Yen (1992), these equations are augmented with kinematic and dependent kinetic equations. Newton iteration is carried out to obtain the solution of the augmented system. The iterative process is started at each time step using first order polynomial extrapolation for all variables in the formulation. In particular, the issue of getting a good starting value for the Lagrange multiplier is difficult and causes trouble. This is one of the factors motivating introduction of the coordinate splitting (CS) technique by Yen and Petzold (1995) for integration of DAE of multibody dynamics. The CS algorithm circumvents the requirement of an initial estimation of Lagrange multipliers by consistently removing them from the numerical formulation.

Finally, Fisette and Vaneghem (1996), proposed an approach similar to that presented here, in which the state–space ODE is discretized using the Newmark family of integrators. At each time step, independent accelerations are obtained via an iterative process. However, the Jacobian matrix of the nonlinear system of algebraic equations obtained after discretization of the state–space ODE neglects a number of terms, as will be pointed out in Subsection 3.2.2, leading to a quasi–Newton numerical method. Numerical experiments carried out in Subsection 4.4 using this quasi–Newton method show degraded numerical effectiveness when compared to the approach presented in this paper.

The development in this section is organized as follows. In Subsection 3.1, the DAE are reduced to a set of state–space ODE in independent generalized coordinates. Subsection 3.2 implements the implicit integration formula for the resulting set of ODE. It contains a description of the framework in which integration is carried out, followed by presentation of a method for generating the

required derivative information. The subsection is concluded with a procedure for solving the discretized equations of motion, along with a summary of linear algebra computations required. Finally, Subsection 3.3 defines the computational flow of the proposed algorithm.

3.1 Reduction of DAE to State–Space ODE

In order to determine a partitioning of the vector $\mathbf{q}$ of generalized coordinates into dependent and independent vectors $\mathbf{u}$ and $\mathbf{v}$, respectively, a set of consistent generalized coordinates $\mathbf{q}_0$; i.e., satisfying Eq.1, is first determined. In this configuration, the constraint Jacobian matrix is evaluated and numerically factored, using the Gauss–Jordan algorithm (Atkinson, 1989),

$$\Phi_{\mathbf{q}}(\mathbf{q}_0) \rightarrow [\Phi_{\mathbf{u}}(\mathbf{q}_0) \mid \Phi_{\mathbf{v}}(\mathbf{q}_0)] \tag{7}$$

The order of appearance of generalized coordinates associated with the columns of the resulting matrix yields a nonsingular subjacobian with respect to $\mathbf{u}$; i.e.,

$$\det(\Phi_{\mathbf{u}}(\mathbf{q}_0)) \neq 0 \tag{8}$$

Having partitioned the generalized coordinates, Eqs. 1 through 4 can be rewritten in the associated partitioned form (Haug, 1989),

$$\mathbf{M}^{\mathbf{vv}}(\mathbf{u},\mathbf{v})\ddot{\mathbf{v}} + \mathbf{M}^{\mathbf{vu}}(\mathbf{u},\mathbf{v})\ddot{\mathbf{u}} + \Phi_{\mathbf{v}}^{\mathrm{T}}(\mathbf{u},\mathbf{v})\lambda = \mathbf{Q}^{\mathbf{v}}(\mathbf{u},\mathbf{v},\dot{\mathbf{u}},\dot{\mathbf{v}}) \tag{9}$$

$$\mathbf{M}^{\mathbf{uv}}(\mathbf{u},\mathbf{v})\ddot{\mathbf{v}} + \mathbf{M}^{\mathbf{uu}}(\mathbf{u},\mathbf{v})\ddot{\mathbf{u}} + \Phi_{\mathbf{u}}^{\mathrm{T}}(\mathbf{u},\mathbf{v})\lambda = \mathbf{Q}^{\mathbf{u}}(\mathbf{u},\mathbf{v},\dot{\mathbf{u}},\dot{\mathbf{v}}) \tag{10}$$

$$\Phi(\mathbf{u},\mathbf{v}) = \mathbf{0} \tag{11}$$

$$\Phi_{\mathbf{u}}(\mathbf{u},\mathbf{v})\dot{\mathbf{u}} + \Phi_{\mathbf{v}}(\mathbf{u},\mathbf{v})\dot{\mathbf{v}} = \mathbf{0} \tag{12}$$

$$\Phi_{\mathbf{u}}(\mathbf{u},\mathbf{v})\ddot{\mathbf{u}} + \Phi_{\mathbf{v}}(\mathbf{u},\mathbf{v})\ddot{\mathbf{v}} = \tau(\mathbf{u},\mathbf{v},\dot{\mathbf{u}},\dot{\mathbf{v}}) \tag{13}$$

The condition of Eq. 8 and the implicit function theorem (Corwin, Szczarba, 1982) guarantee that Eq. 11 can be locally solved for $\mathbf{u}$ as a function of $\mathbf{v}$,

$$\mathbf{u} = \mathbf{g}(\mathbf{v}) \tag{14}$$

where the function $\mathbf{g}(\mathbf{v})$ has as many continuous derivatives as does the constraint function $\Phi(\mathbf{q})$. Thus, at an admissible configuration $\mathbf{q}_0$, there exist neighborhoods $\mathbf{U}_1(\mathbf{v}^0)$ and $\mathbf{U}_2(\mathbf{u}^0)$, and a function $\mathbf{h} : \mathbf{U}_1 \rightarrow \mathbf{U}_2$ such that for any $\mathbf{v} \in \mathbf{U}_1$, Eq. 11 is identically satisfied when $\mathbf{u}$ is given by Eq.14.

With the partition of the generalized coordinate vector induced by Eq. 7, the index 1 system of Eqs. 9 , 10, and 13 is reduced to a set of ODE through a succession of steps, using information provided by Eqs. 12 and 14. Since the coefficient matrix of $\dot{\mathbf{u}}$ in Eq. 12 is nonsingular, $\dot{\mathbf{u}}$ can be determined as a function of $\mathbf{v}$ and $\dot{\mathbf{v}}$, where Eq. 14 is used to eliminate explicit dependence on $\mathbf{u}$. Next, Eq. 13 uniquely determines $\ddot{\mathbf{u}}$ as a function of $\mathbf{v}$, $\dot{\mathbf{v}}$, and $\ddot{\mathbf{v}}$, where results of Eqs. 12 and 14 are substituted. Since the coefficient matrix of λ in Eq.10 is nonsingular, λ can be determined uniquely as a function of $\mathbf{v}$, $\dot{\mathbf{v}}$, and $\ddot{\mathbf{v}}$, using previously derived results. Finally, each of the preceding results is substituted into Eq. 9 to obtain a set of state–space ODE in only the independent generalized coordinates $\mathbf{v}$ (Haug, 1989),

$$\hat{\mathbf{M}}\ddot{\mathbf{v}} = \hat{\mathbf{Q}} \tag{15}$$

where

$$\hat{\mathbf{M}} = \mathbf{M}^{vv} - \mathbf{M}^{vu}\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}} - \Phi_{\mathbf{v}}^{T}(\Phi_{\mathbf{u}}^{-1})^{T}[\mathbf{M}^{uv} - \mathbf{M}^{uu}\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}}]$$

$$\hat{\mathbf{Q}} = \mathbf{Q}^{v} - \mathbf{M}^{vu}\Phi_{\mathbf{u}}^{-1}\tau - \Phi_{\mathbf{v}}^{T}(\Phi_{\mathbf{u}}^{-1})^{T}[\mathbf{Q}^{u} - \mathbf{M}^{uu}\Phi_{\mathbf{u}}^{-1}\tau]$$

3.2 Implicit Integration of the State–Space ODE

3.2.1 General Framework

Using an implicit numerical integration formula to discretize the state–space ODE in Eq. 15 is impractical, because solving the resulting set of nonlinear equations requires derivative information whenever a Newton like method is applied. For the ODE in Eq. 15 these derivatives would be difficult to generate. Using Eq. 9, from which Eq. 15 originates, leads to a consistent and tractable way to

obtain the needed derivative information. Thus, rather than discretizing Eq. 15, an implicit integration formula is used to discretize Eq. 9. The dependent coordinates appearing in this equation, namely $\mathbf{u}$, $\dot{\mathbf{u}}$,$\ddot{\mathbf{u}}$, and λ, are regarded as functions of the independent coordinates and treated accordingly when computing the Jacobian of the resulting nonlinear equations.

No explicit relationship between the dependent and independent coordinates is known. However, for a given set of independent coordinates (positions and velocities), all dependent coordinates can be computed as follows: $\mathbf{u}$ is determined from Eq. 11, and $\dot{\mathbf{u}}$ is determined from Eq. 12. Finally, $\ddot{\mathbf{u}}$, $\dot{\mathbf{v}}$, and λ are simultaneously determined from Eqs. 9, 10, and 13. The procedure for obtaining the derivative information needed is presented in Subsection 3.2.2, after introducing the implicit integrator used to discretize Eq. 9.

A second order integration formula from the Newmark family (Hughes, 1987) is used here to discretize the state–space ODE. The approach, however, is applicable for use with any implicit numerical integration formula. The Newmark family of implicit integrators assumes the form

$$\dot{\mathbf{v}}_{n+1} = \tilde{\dot{\mathbf{v}}}_{n+1} + \gamma h \ddot{\mathbf{v}}_{n+1} \tag{16}$$

$$\mathbf{v}_{n+1} = \tilde{\mathbf{v}}_{n+1} + \beta h^2 \ddot{\mathbf{v}}_{n+1} \tag{17}$$

where h is step size and subscript denotes discrete time step. The predictor terms $\tilde{\dot{\mathbf{v}}}_{n+1}$ and $\tilde{\mathbf{v}}_{n+1}$ contain only past information and are given as

$$\tilde{\dot{\mathbf{v}}}_{n+1} = \dot{\mathbf{v}}_n + (1-\gamma) h \ddot{\mathbf{v}}_n \tag{18}$$

$$\tilde{\mathbf{v}}_{n+1} = \mathbf{v}_n + h\dot{\mathbf{v}}_n + \frac{h^2}{2}(1-2\beta)\ddot{\mathbf{v}}_n \tag{19}$$

The Newmark family is extensively used in structural dynamics, since it is known to have attractive stability properties. It should be noted that the algorithm presented here is not confined to using this class of methods to integrate the state–space ODE.

Regarding Eq. 9 as a set of second order ODE in the independent coordinates $\mathbf{v}$, and substituting $\mathbf{v}_{n+1}$ and $\dot{\mathbf{v}}_{n+1}$ from Eqs. 16 and 17 yields

$$\Psi(\ddot{\mathbf{v}}_{n+1}) \equiv \mathbf{M}^{\mathbf{vv}}_{n+1}\ddot{\mathbf{v}}_{n+1} + \mathbf{M}^{\mathbf{vu}}_{n+1}\ddot{\mathbf{u}}_{n+1} + [\Phi^{T}_{\mathbf{v}}]_{n+1}\lambda_{n+1} - \mathbf{Q}^{\mathbf{v}}_{n+1} = \mathbf{0} \tag{20}$$

This set of nonlinear equations must be solved for the independent accelerations $\ddot{\mathbf{v}}_{n+1}$. In this paper, Newton's method is used for this purpose, requiring the Jacobian $\Psi_{\ddot{\mathbf{v}}}$ where the subscript n + 1 is suppressed for notational convenience.

3.2.2 Generating Derivative Information

Taking the derivative of Eq. 11 with respect to $\ddot{\mathbf{v}}$ results in $\Phi_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} = -\Phi_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}}$. Using Eq. 17 , this reduces to

$$\mathbf{u}_{\ddot{\mathbf{v}}} = -\beta h^2\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}} \equiv \beta h^2\mathbf{H} \tag{21}$$

where the matrix $\mathbf{H}$ is obtained by solving the multiple right side equation $\Phi_{\mathbf{u}}\mathbf{H} = -\Phi_{\mathbf{v}}$. Taking the total derivative of Eq. 12 with respect to $\ddot{\mathbf{v}}$ and using the chain rule of differentiation yields

$$\begin{aligned}\Phi_{\mathbf{u}}\dot{\mathbf{u}}_{\ddot{\mathbf{v}}} &= -\left\{(\Phi_{\mathbf{u}}\dot{\mathbf{u}})_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\Phi_{\mathbf{u}}\dot{\mathbf{u}})_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} + \Phi_{\mathbf{v}}\dot{\mathbf{v}}_{\ddot{\mathbf{v}}} + (\Phi_{\mathbf{v}}\dot{\mathbf{v}})_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\Phi_{\mathbf{v}}\dot{\mathbf{v}})_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}}\right\} \\ &= -\beta h^2\left[(\Phi_{\mathbf{u}}\dot{\mathbf{u}})_{\mathbf{u}}\mathbf{H} + (\Phi_{\mathbf{u}}\dot{\mathbf{u}})_{\mathbf{v}} + (\Phi_{\mathbf{v}}\dot{\mathbf{v}})_{\mathbf{u}}\mathbf{H} + (\Phi_{\mathbf{v}}\dot{\mathbf{v}})_{\mathbf{v}}\right] - \gamma h\Phi_{\mathbf{v}}\end{aligned} \tag{22}$$

Using Eqs. 16, 17, and 21, this reduces to

$$\begin{aligned}\dot{\mathbf{u}}_{\ddot{\mathbf{v}}} &= -\beta h^2\left\{\Phi_{\mathbf{u}}^{-1}\left[(\Phi_{\mathbf{u}}\dot{\mathbf{u}})_{\mathbf{u}} + (\Phi_{\mathbf{v}}\dot{\mathbf{v}})_{\mathbf{u}}\right]\mathbf{H} + \Phi_{\mathbf{u}}^{-1}\left[(\Phi_{\mathbf{u}}\dot{\mathbf{u}})_{\mathbf{v}} + (\Phi_{\mathbf{v}}\dot{\mathbf{v}})_{\mathbf{v}}\right]\right\} \\ &\quad - \gamma h\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}} \equiv \beta h^2\mathbf{J} + \gamma h\mathbf{H}\end{aligned} \tag{23}$$

Differentiating Eq. 13 with respect to $\ddot{\mathbf{v}}$ yields

$$\begin{aligned}\Phi_{\mathbf{u}}\ddot{\mathbf{u}}_{\ddot{\mathbf{v}}} &= \tau_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + \tau_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} + \tau_{\dot{\mathbf{u}}}\dot{\mathbf{u}}_{\ddot{\mathbf{v}}} + \tau_{\dot{\mathbf{v}}}\dot{\mathbf{v}}_{\ddot{\mathbf{v}}} - [(\Phi_{\mathbf{u}}\ddot{\mathbf{u}})_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\Phi_{\mathbf{u}}\ddot{\mathbf{u}})_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} + (\Phi_{\mathbf{v}}\ddot{\mathbf{v}})_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} \\ &\quad + (\Phi_{\mathbf{v}}\ddot{\mathbf{v}})_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} + \Phi_{\mathbf{v}}] = \beta h^2[\tau_{\mathbf{u}}\mathbf{H} + \tau_{\mathbf{v}} + \tau_{\dot{\mathbf{u}}}\mathbf{J} - (\Phi_{\mathbf{u}}\ddot{\mathbf{u}})_{\mathbf{u}}\mathbf{H} - (\Phi_{\mathbf{u}}\ddot{\mathbf{u}})_{\mathbf{v}} \\ &\quad - (\Phi_{\mathbf{v}}\ddot{\mathbf{v}})_{\mathbf{u}}\mathbf{H} - (\Phi_{\mathbf{v}}\ddot{\mathbf{v}})_{\mathbf{v}}] + \gamma h(\tau_{\dot{\mathbf{u}}}\mathbf{H} + \tau_{\dot{\mathbf{v}}}) - \Phi_{\mathbf{v}}\end{aligned} \tag{24}$$

$$\begin{aligned}\ddot{\mathbf{u}}_{\ddot{\mathbf{v}}} &= \beta h^2\Phi_{\mathbf{u}}^{-1}\{[\tau_{\mathbf{u}} - (\Phi_{\mathbf{u}}\ddot{\mathbf{u}})_{\mathbf{u}} - (\Phi_{\mathbf{v}}\ddot{\mathbf{v}})_{\mathbf{u}}]\mathbf{H} + \tau_{\mathbf{v}} + \tau_{\dot{\mathbf{u}}}\mathbf{J} - (\Phi_{\mathbf{u}}\ddot{\mathbf{u}})_{\mathbf{v}} - (\Phi_{\mathbf{v}}\ddot{\mathbf{v}})_{\mathbf{v}}\} \\ &\quad + \gamma h\Phi_{\mathbf{u}}^{-1}(\tau_{\dot{\mathbf{u}}}\mathbf{H} + \tau_{\dot{\mathbf{v}}}) + \mathbf{H} \equiv \beta h^2\mathbf{L} + \gamma h\mathbf{N} + \mathbf{H}\end{aligned} \tag{25}$$

When computing the derivative of the dependent accelerations with respect to the independent accelerations in Eq. 25, Fisette and Vaneghem (1996) neglect the contribution of the right side of the acceleration kinematic equation (Eq. 3); i. e., the matrices **N** and **L** in the expression of Eq. 25 for $\ddot{\mathbf{u}}_{\ddot{\mathbf{v}}}$, computation of the Jacobian $\Psi_{\ddot{\mathbf{v}}}$ thus being inexact. The impact of working with this Jacobian is numerically analyzed in Subsection 4.4, from an efficiency standpoint.

Finally, differentiating Eq. 10 with respect to $\ddot{\mathbf{v}}$ yields

$$\begin{aligned}\Phi_{\mathbf{u}}^{\mathrm{T}}\lambda_{\ddot{\mathbf{v}}} &= \mathbf{Q}_{\mathbf{u}}^{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + \mathbf{Q}_{\mathbf{v}}^{\mathbf{u}}\mathbf{v}_{\ddot{\mathbf{v}}} + \mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{u}}\dot{\mathbf{u}}_{\ddot{\mathbf{v}}} + \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{u}}\dot{\mathbf{v}}_{\ddot{\mathbf{v}}} - \{(\Phi_{\mathbf{u}}^{\mathrm{T}}\lambda)_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\Phi_{\mathbf{u}}^{\mathrm{T}}\lambda)_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} + \mathbf{M}^{\mathbf{uv}} \\ &\quad + (\mathbf{M}^{\mathbf{uv}}\ddot{\mathbf{v}})_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\mathbf{M}^{\mathbf{uv}}\ddot{\mathbf{v}})_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} + \mathbf{M}^{\mathbf{uu}}\ddot{\mathbf{u}}_{\ddot{\mathbf{v}}} + (\mathbf{M}^{\mathbf{uu}}\ddot{\mathbf{u}})_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\mathbf{M}^{\mathbf{uu}}\ddot{\mathbf{u}})_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}}\} \\ &= \beta h^2[\mathbf{Q}_{\mathbf{u}}^{\mathbf{u}}\mathbf{H} + \mathbf{Q}_{\mathbf{v}}^{\mathbf{u}} + \mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{u}}\mathbf{J} - (\Phi_{\mathbf{u}}^{\mathrm{T}}\lambda)_{\mathbf{u}}\mathbf{H} - (\Phi_{\mathbf{u}}^{\mathrm{T}}\lambda)_{\mathbf{v}} - (\mathbf{M}^{\mathbf{uv}}\ddot{\mathbf{v}})_{\mathbf{u}}\mathbf{H} - (\mathbf{M}^{\mathbf{uv}}\ddot{\mathbf{v}})_{\mathbf{v}} - \mathbf{M}^{\mathbf{uu}}\mathbf{L} \\ &\quad - (\mathbf{M}^{\mathbf{uu}}\ddot{\mathbf{u}})_{\mathbf{u}}\mathbf{H} - (\mathbf{M}^{\mathbf{uu}}\ddot{\mathbf{u}})_{\mathbf{v}}] + \gamma h(\mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{u}}\mathbf{H} + \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{u}} - \mathbf{M}^{\mathbf{uu}}\mathbf{N}) - \mathbf{M}^{\mathbf{uv}} - \mathbf{M}^{\mathbf{uu}}\mathbf{H}\end{aligned} \tag{26}$$

$$\begin{aligned}\lambda_{\ddot{\mathbf{v}}} &= \beta h^2\Phi_{\mathbf{u}}^{-\mathrm{T}}\{[\mathbf{Q}_{\mathbf{u}}^{\mathbf{u}} - (\Phi_{\mathbf{u}}^{\mathrm{T}}\lambda)_{\mathbf{u}} - (\mathbf{M}^{\mathbf{uv}}\ddot{\mathbf{v}})_{\mathbf{u}} - (\mathbf{M}^{\mathbf{uu}}\ddot{\mathbf{u}})_{\mathbf{u}}]\mathbf{H} + \mathbf{Q}_{\mathbf{v}}^{\mathbf{u}} + \mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{u}}\mathbf{J} \\ &\quad - (\Phi_{\mathbf{u}}^{\mathrm{T}}\lambda)_{\mathbf{v}} - (\mathbf{M}^{\mathbf{uv}}\ddot{\mathbf{v}})_{\mathbf{v}} - (\mathbf{M}^{\mathbf{uu}}\ddot{\mathbf{u}})_{\mathbf{v}} - \mathbf{M}^{\mathbf{uu}}\mathbf{L}\} + \gamma h\Phi_{\mathbf{u}}^{-\mathrm{T}}(\mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{u}}\mathbf{H} + \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{u}} - \mathbf{M}^{\mathbf{uu}}\mathbf{N}) \\ &\quad - \Phi_{\mathbf{u}}^{-\mathrm{T}}(\mathbf{M}^{\mathbf{uv}} + \mathbf{M}^{\mathbf{uu}}\mathbf{H}) \equiv -\Phi_{\mathbf{u}}^{-\mathrm{T}}[\beta h^2\mathbf{R} + \gamma h\mathbf{S} + \mathbf{M}^{\mathbf{uv}} + \mathbf{M}^{\mathbf{uu}}\mathbf{H}]\end{aligned} \tag{27}$$

3.2.3 Solution of the Discretized Equations of Motion

The integration process progresses by determining the solution $\ddot{\mathbf{v}}_{n+1}$ of Eq. 20 at each time step, using Newton–Raphson iteration. In this subsection, the subscript n + 1 is suppressed for notational convenience. Using results of the preceding subsection, the Jacobian of Ψ in Eq. 20 with respect to $\ddot{\mathbf{v}}$ is

$$\begin{aligned}\Psi_{\ddot{\mathbf{v}}} &= \mathbf{M}^{\mathbf{vv}} + (\mathbf{M}^{\mathbf{vv}}\ddot{\mathbf{v}})_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\mathbf{M}^{\mathbf{vv}}\ddot{\mathbf{v}})_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} + \mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}}_{\ddot{\mathbf{v}}} + (\mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}})_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}})_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} \\ &\quad + \Phi_{\mathbf{v}}^{\mathrm{T}}\lambda_{\ddot{\mathbf{v}}} + (\Phi_{\mathbf{v}}^{\mathrm{T}}\lambda)_{\mathbf{u}}\mathbf{u}_{\ddot{\mathbf{v}}} + (\Phi_{\mathbf{v}}^{\mathrm{T}}\lambda)_{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} - \mathbf{Q}_{\mathbf{u}}^{\mathbf{v}}\mathbf{u}_{\ddot{\mathbf{v}}} - \mathbf{Q}_{\mathbf{v}}^{\mathbf{v}}\mathbf{v}_{\ddot{\mathbf{v}}} - \mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{v}}\dot{\mathbf{u}}_{\ddot{\mathbf{v}}} - \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{v}}\dot{\mathbf{v}}_{\ddot{\mathbf{v}}} \\ &= \mathbf{M}^{\mathbf{vv}} + (\mathbf{M}^{\mathbf{vv}}\ddot{\mathbf{v}})_{\mathbf{u}}\beta h^2\mathbf{H} + (\mathbf{M}^{\mathbf{vv}}\ddot{\mathbf{v}})_{\mathbf{v}}\beta h^2 + \mathbf{M}^{\mathbf{vu}}[\beta h^2\mathbf{L} + \gamma h\mathbf{N} + \mathbf{H}] \\ &\quad + (\mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}})_{\mathbf{u}}\beta h^2\mathbf{H} + (\mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}})_{\mathbf{v}}\beta h^2 + \mathbf{H}^{\mathrm{T}}[\beta h^2\mathbf{R} + \gamma h\mathbf{S} + \mathbf{M}^{\mathbf{uv}} + \mathbf{M}^{\mathbf{uu}}\mathbf{H}] \\ &\quad + (\Phi_{\mathbf{v}}^{\mathrm{T}}\lambda)_{\mathbf{u}}\beta h^2\mathbf{H} + (\Phi_{\mathbf{v}}^{\mathrm{T}}\lambda)_{\mathbf{v}}\beta h^2 - \mathbf{Q}_{\mathbf{u}}^{\mathbf{v}}\beta h^2\mathbf{H} - \mathbf{Q}_{\mathbf{v}}^{\mathbf{v}}\beta h^2 - \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{v}}\gamma h \\ &\quad - \mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{v}}[\beta h^2\mathbf{J} + \gamma h\mathbf{H}]\end{aligned} \tag{28}$$

Collecting terms, this Jacobian is rewritten as

$$\Psi_{\ddot{\mathbf{v}}} = \mathbf{M}^{vv} + \mathbf{M}^{vu}\mathbf{H} + \mathbf{H}^T(\mathbf{M}^{uv} + \mathbf{M}^{uu}\mathbf{H}) + \gamma h(\mathbf{M}^{vu}\mathbf{N} + \mathbf{H}^T\mathbf{S} - \mathbf{Q}^v_{\dot{\mathbf{v}}} - \mathbf{Q}^v_{\dot{\mathbf{u}}}\mathbf{H}) \quad (29)$$
$$+ \beta h^2\{[(\mathbf{M}^{vv}\ddot{\mathbf{v}})_u + (\mathbf{M}^{vu}\ddot{\mathbf{u}})_u]\mathbf{H} + [(\mathbf{M}^{vv}\ddot{\mathbf{v}})_v + (\mathbf{M}^{vu}\ddot{\mathbf{u}})_v] + \mathbf{M}^{vu}\mathbf{L} + \mathbf{H}^T\mathbf{R}$$
$$+ \left(\Phi^T_v\lambda\right)_u\mathbf{H} + \left(\Phi^T_v\lambda\right)_v - \mathbf{Q}^v_u\mathbf{H} - \mathbf{Q}^v_v - \mathbf{Q}^v_{\dot{u}}\mathbf{J}\}$$

Newton–Raphson iteration is carried out in the form

$$\Psi^{(i)}_{\ddot{\mathbf{v}}}\Delta\ddot{\mathbf{v}}^{(i)} = -\Psi^{(i)} \quad (30)$$

$$\ddot{\mathbf{v}}^{(i+1)} = \ddot{\mathbf{v}}^{(i)} + \Delta\ddot{\mathbf{v}}^{(i)}$$

to iteratively solve for $\ddot{\mathbf{v}}$, until convergence criteria are satisfied. Within each iteration, Eqs. 16 and 17 are used to obtain $\mathbf{v}^{i+1}$ and $\dot{\mathbf{v}}^{i+1}$, and Eqs. 10 through 13 are solved for $\mathbf{u}^{i+1}$, $\dot{\mathbf{u}}^{i+1}$, $\ddot{\mathbf{u}}^{i+1}$, and λ^{i+1}. The residual in satisfying Eq. 20 is then evaluated and the process is terminated if error criteria are met. Otherwise, the Jacobian in Eq. 29 is again evaluated and Eqs. 30 are used to obtain a new estimate of the solution $\ddot{\mathbf{v}}$.

3.2.4 Computation of Matrices

Matrix computations required in implementing the foregoing numerical sequence are as follows. First,

$$\Phi_u\mathbf{H} = -\Phi_v \quad (31)$$

is solved for **H**. This process is computationally cheap, once the LU factorization of the matrix Φ_u is available. Then,

$$\Phi_u\mathbf{J} = -\left\{\left[(\Phi_u\dot{\mathbf{u}})_u + (\Phi_v\dot{\mathbf{v}})_u\right]\mathbf{H} + \left[(\Phi_u\dot{\mathbf{u}})_v + (\Phi_v\dot{\mathbf{v}})_v\right]\right\} \quad (32)$$

is solved for **J**,

$$\Phi_u\mathbf{L} = [\tau_u - (\Phi_u\ddot{\mathbf{u}})_u - (\Phi_v\ddot{\mathbf{v}})_u]\mathbf{H} + \tau_v + \tau_{\dot{u}}\mathbf{J} - (\Phi_u\ddot{\mathbf{u}})_v - (\Phi_v\ddot{\mathbf{v}})_v \quad (33)$$

is solved for **L**,

$$\Phi_u\mathbf{N} = \tau_{\dot{u}}\mathbf{H} + \tau_{\dot{v}} \quad (34)$$

is solved for **N**, and

$$\mathbf{R} = -\{[\mathbf{Q}^{u}_{u} - (\Phi^{T}_{u}\lambda)_{u} - (\mathbf{M}^{uv}\dot{\mathbf{v}})_{u} - (\mathbf{M}^{uu}\ddot{\mathbf{u}})_{u}]\mathbf{H} + \mathbf{Q}^{u}_{v} + \mathbf{Q}^{u}_{\dot{u}}\mathbf{J} - (\Phi^{T}_{\dot{u}}\lambda)_{v} - (\mathbf{M}^{uv}\dot{\mathbf{v}})_{v} - (\mathbf{M}^{uu}\ddot{\mathbf{u}})_{v} - \mathbf{M}^{uu}\mathbf{L}\} \tag{35}$$

$$\mathbf{S} = -(\mathbf{Q}^{u}_{\dot{u}}\mathbf{H} + \mathbf{Q}^{u}_{\dot{v}} - \mathbf{M}^{uu}\mathbf{N}) \tag{36}$$

From Eqs. 12 and 13,

$$\dot{\mathbf{u}} = \mathbf{H}\dot{\mathbf{v}} \tag{37}$$

$$\ddot{\mathbf{u}} = \mathbf{H}\ddot{\mathbf{v}} + \Phi_{u}^{-1}\tau = \mathbf{H}\ddot{\mathbf{v}} + \alpha$$

where α is the solution of $\Phi_{u}\alpha = \tau$. From Eq. 10,

$$\Phi^{T}_{u}\lambda = \mathbf{Q}^{u} - \mathbf{M}^{uv}\ddot{\mathbf{v}} - \mathbf{M}^{uu}[\mathbf{H}\ddot{\mathbf{v}} + \alpha] \tag{38}$$

is solved for λ.

Equation 20 then becomes

$$[\mathbf{M}^{vv} + \mathbf{M}^{vu}\mathbf{H} + \mathbf{H}^{T}(\mathbf{M}^{uv} + \mathbf{M}^{uu}\mathbf{H})]\ddot{\mathbf{v}} = \mathbf{Q}^{v} - \mathbf{M}^{vu}\alpha + \mathbf{H}^{T}(\mathbf{Q}^{u} - \mathbf{M}^{uu}\alpha) \tag{39}$$

It is known (Haug, 1989) that the coefficient matrix on the left is positive definite, for systems in which kinetic energy is positive for nonzero kinematically admissible velocities.

3.3 Computational Algorithm

The iterative computational algorithm defined in the preceding sections is summarized as follows:

Predictor Stage

(a) Equations 16 and 17 are used to estimate $\mathbf{v}$ and $\dot{\mathbf{v}}$, with $\beta = \gamma = 0$
(b) Equation 11 is solved iteratively for $\mathbf{u}$
(c) Equation 31 is used to obtain $\mathbf{H}$
(d) Equations 37 are used to obtain $\dot{\mathbf{u}}$ and $\ddot{\mathbf{u}}$
(e) Equation 39 is solved for $\ddot{\mathbf{v}}$

Corrector Stage

(1) Given $\ddot{\mathbf{v}}$, $\mathbf{v}$ and $\dot{\mathbf{v}}$ are found from Eqs. 16 and 17

(2) Equation 11 is solved iteratively for $\mathbf{u}$

(3) Equation 31 is used to obtain $\mathbf{H}$

(4) Equations 37 are used to obtain $\dot{\mathbf{u}}$ and $\ddot{\mathbf{u}}$

(5) Equation 38 is used to obtain λ

(6) Ψ is evaluated using Eq. 20. Stop if convergence criteria are met. Otherwise continue.

(7) $\Psi_{\ddot{\mathbf{v}}}$ is found from Eq. 29, using matrices obtained from Eqs. 32 through 36

(8) Equation 30 is solved for $\Delta\ddot{\mathbf{v}}$ and a new $\ddot{\mathbf{v}}$ is obtained

(9) Return to Step 1

Note that Step 2 of the corrector stage yields the $\mathbf{LU}$ factorization of matrix $\Phi_{\mathbf{u}}$. Steps 2 through 7 use this $\mathbf{LU}$ factorization to repetitively solve matrix equations of the form $\Phi_{\mathbf{u}}\mathbf{X} = \mathbf{Y}$.

4 Numerical Experiments

The foregoing algorithm is implemented with a constant time step, second order, Newmark formula (Hughes, 1987) obtained from Eqs. 16 and 17 with $\gamma = \frac{1}{2}$ and $\beta = \frac{1}{4}$. The focus of numerical experiments is on robustness of the algorithm, rather than efficiency. A set of three test problems is considered, including a double pendulum, a slider–crank, and the seven body mechanism that is used in the literature (Schiehlen, 1990) as a challenging test problem.

4.1 Double Pendulum Example

The double pendulum shown in Fig. 6 is a two body, two degree of freedom planar mechanical system. The mechanism is modeled using Cartesian coordinates; i.e., the x and y coordinates of the centers of mass and angle θ that defines the orientation of the local centroidal reference frames with respect to the global reference frame. A large amount of stiffness is added to the system by means of two rotational spring–damper–actuators (RSDA). The parameters of the problem are provided in Table 1, in SI units.

L_1	m_1	k_1	c_1	L_2	m_2	k_2	c_2
1.0							

Table 1 Parameters for the Double Pendulum

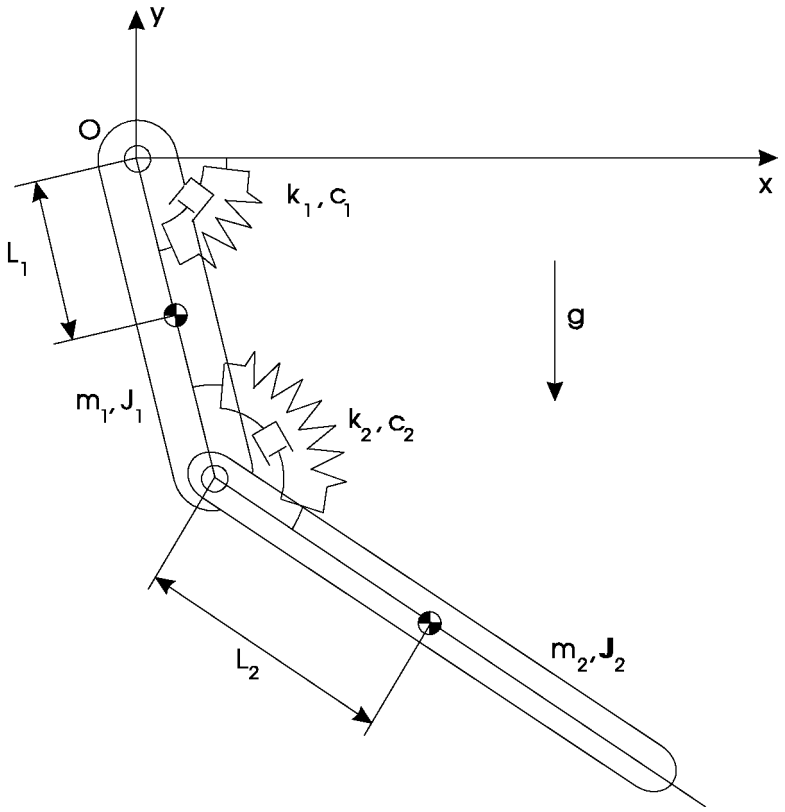

Figure 6 Double Pendulum Mechanism

Initial conditions for the problem are given in Table 2. The first row contains position information and the second contains velocity information.

Body 1			**Body 2**		
x coordinate	*y coordinate*	*θ coordinate*	*x coordinate*	*y coordinate*	*θ coordinate*
0.0	0.0	0.0	0.0	0.0	10.0

Table 2 Initial Conditions for the Double Pendulum

The problem is also solved using a second order, constant time step, explicit integrator, namely the Adams–Bashfort–Moulton predictor–corrector method.

A set of numerical tests is performed, to compare results obtained with the explicit algorithm and the proposed implicit formulation. When a simulation is run, for each time step h, the following information is generated:

(1) Sup–Norm of the error and the time instant at which it occurred

(2) 2–Norm of the error

(3) CPU time in which the integrator completes a full simulation [0,T]

The reference, or exact, solution is generated using a fourth order Runge–Kutta method with very small time step. The length of the simulation is $T = 0.1$ seconds. With nstep integration steps, such that for a given step size h, $(\text{nstep} - 1)h < T \leq \text{nstep} \cdot h$, the norms computed are

$$\text{2–Norm} = \sqrt{\sum_{i=1}^{\text{nstep}} e_i^2}$$

$$\text{Sup–Norm} = \max e_i, \quad 1 \leq i \leq \text{nstep}$$

where, for a step size h, the error in the variable p at time $t_i = i \cdot h$ is defined as $e_i = |p_i^{\text{ref}} - p_i|$. Finally, if k is such that $e_k \geq e_i$ for $1 \leq i \leq \text{nstep}$, then if $p_k^{\text{ref}} \neq 0$,

$$\text{Rel–Err} = 100 \times \frac{\text{Sup–Norm}}{|p_k^{\text{ref}}|}$$

The largest step size attained with each of the integrators is $h_{\text{IMPL}}^{\max}$=0.02 for the implicit trapezoidal integrator (IMPL), and $h_{\text{EXPL}}^{\max}$=0.00012 for the Adams–Bashfort–Moulton (EXPL) predictor–corrector method.

4.2 Slider–Crank Example

The slider–crank mechanism in Fig. 7 is a two body, two degree of freedom planar mechanical system. The mechanism is modeled using Cartesian coordinates.

At the tip of body 2, a translational spring–damper–actuator (TSDA) element is attached. The parameters of the problem are provided in Table 3, in SI units.

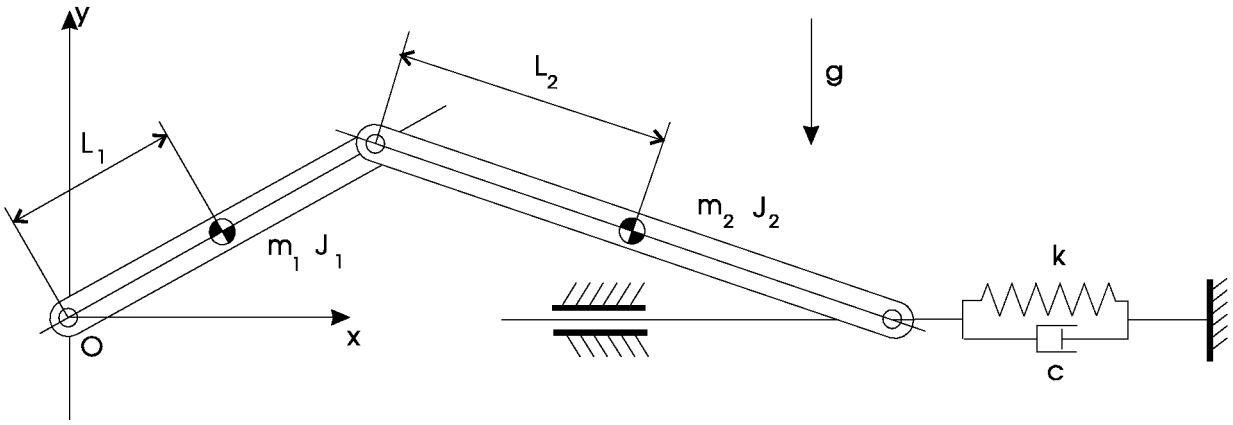

Figure 7 Slider–Crank Mechanism

L_1	m_1	L_2	m_2	k	c
0.3	3.0	0.6	9.0		

Table 3 Parameters for the Slider–Crank

Initial conditions for the problem are given in Table 4. The first row contains position information and the second contains velocity information. The length of the simulation is T=0.1 seconds.

Body 1			**Body 2**		
x coordinate	*y coordinate*	*θ coordinate*	*x coordinate*	*y coordinate*	*θ coordinate*
0.0	0.3	π/2	0.5196152	0.3	3π/2
0.0	0.0	0.0	0.0	0.0	0.0

Table 4 Initial Conditions for the Slider–Crank

4.3 Seven Body Mechanism Example

Numerical experiments are performed using ”Andrew's squeezing mechanism”, which has become a classical test problem in the literature, through the work of Schiehlen (1990) and Hairer and Wanner (1991). It consists of 7 rigid bodies connected by joints, without friction, and undergoes planar motion. A driving torque is applied to one of the bodies, while a stiff spring is connected to a different body in the system. The mechanism exhibits very large accelerations, limiting integrators to small step sizes to stably simulate the motion. However, in order that this problem serve the goal of testing the implicit integrator, a damper was attached to the spring connecting body 3 and ground. The original value of the spring stiffness was modified to k=30,000, while the damping coefficient was chosen c=100,000.

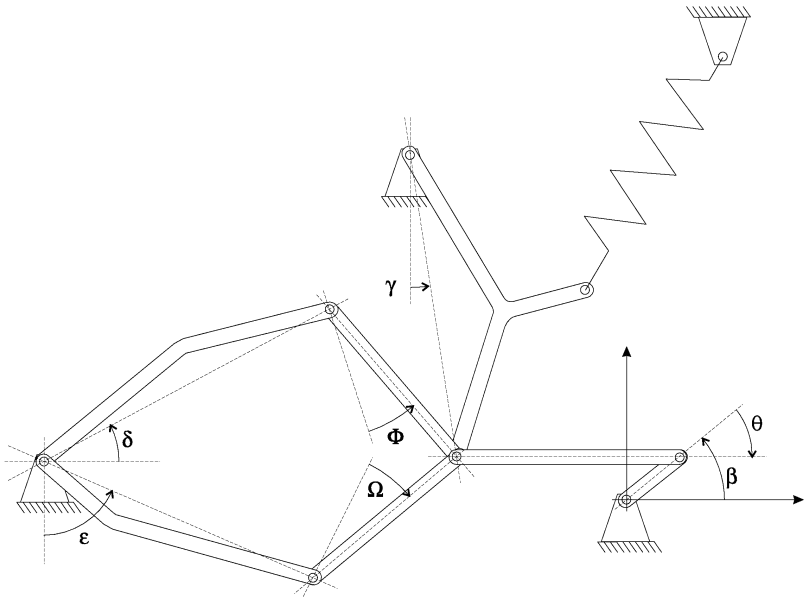

Figure 8 Seven Body Mechanism

Cartesian coordinates are used to model the mechanism. For this problem, the vector of generalized coordinates is of dimension 21, and 20 constraints must be

satisfied. Thus, the mechanism has one degree of freedom. Parameters defining the problem (inertia properties, initial conditions, and driving forces) are as given by Hairer and Wanner (1991). The interval of simulation considered is [0, 0.035] seconds.

4.4 Comparison of Implicit Algorithms

In order to see how the algorithm of Fisette and Vaneghem (1996) compares with the algorithm presented here, a set of numerical experiments is performed. The method of Fisette and Vaneghem (1996) is implemented with constant step size and is denoted in what follows by FV. The test problems used for comparison are the three problems studied in the foregoing.

The difference between the Jacobian $\Psi_{\ddot{v}}$ of Eq. 29, and the Jacobian $\Pi_{\ddot{v}}$ of the Fisette and Vaneghem algorithm is

$$\begin{aligned}\Psi_{\ddot{v}} - \Pi_{\ddot{v}} = {} & (\mathbf{M}^{vu} + \mathbf{H}^T\mathbf{M}^{uu})(\gamma h\mathbf{N} + \beta h^2\mathbf{L}) \\ & + \beta h^2[(\Phi_v^T\lambda)_v + (\Phi_v^T\lambda)_u\mathbf{H} + \mathbf{H}^T(\Phi_u^T\lambda)_v + \mathbf{H}^T(\Phi_u^T\lambda)_u\mathbf{H}]\end{aligned} \tag{40}$$

It should be noted that the algorithm FV produces accurate results, since the integration of the state–space ODE is correct. It is the numerical solution process that is degraded, because of the iteration matrix considered in FV. As a result, the overall performance of the algorithm is affected. To illustrate this, the largest integration step size, and the number of iterations required to solve the algebraic equations obtained upon discretization of the state–space ODE are compared for the two algorithms.

Table 5 shows the largest step sizes achieved with the algorithms IMPL and FV. The step sizes allowed by IMPL are larger than those taken by FV by a factor of 1.3–18, depending of the problem being solved. It is apparent from Table 5 that for problems in which kinematic aspects prevail; i.e., the model is subject to many

constraint equations, as in the case of the Seven Body mechanism, the influence of the terms neglected in FV have a larger impact on performance of the algorithm.

Model	IMPL	FV
Double Pendulum	0.015	0.0089
Slider–Crank	0.0649	0.0086
Seven Body	0.0018	0.0001

Table 5 Time Step Comparison

Finally, in Figs. 9, 10, and 11 the solid lines show the number of Newton iterations required in determining the independent accelerations in IMPL, while the dashed lines give the number of Newton iterations for FV. Note that IMPL requires half or less iterations.

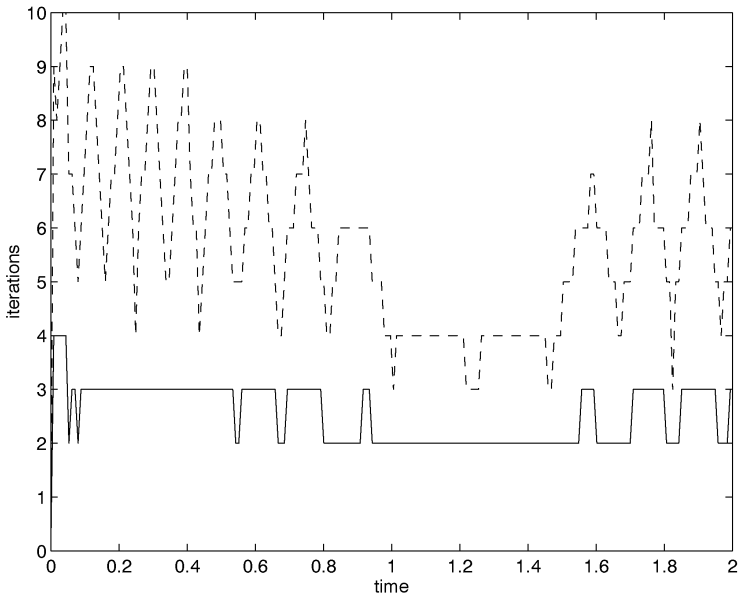

Figure 9 Iterations for Double Pendulum, h=0.0089

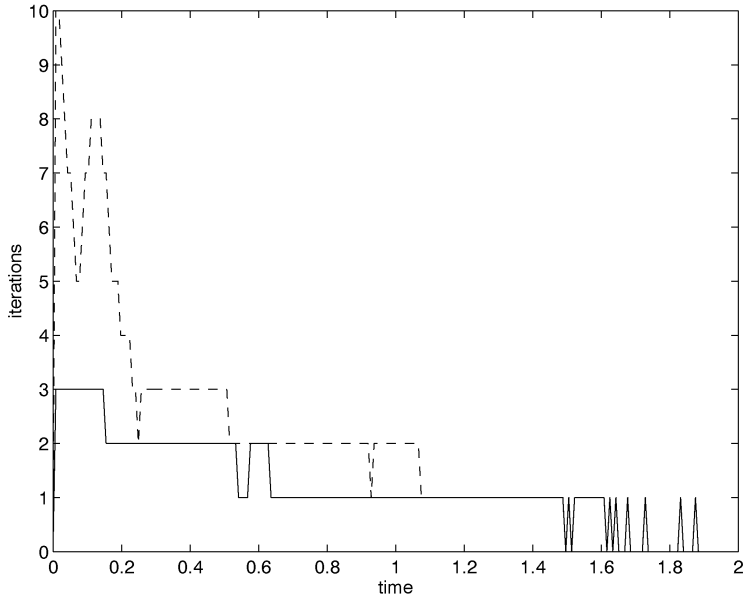

Figure 10 Iterations for Slider–Crank, h=0.0086

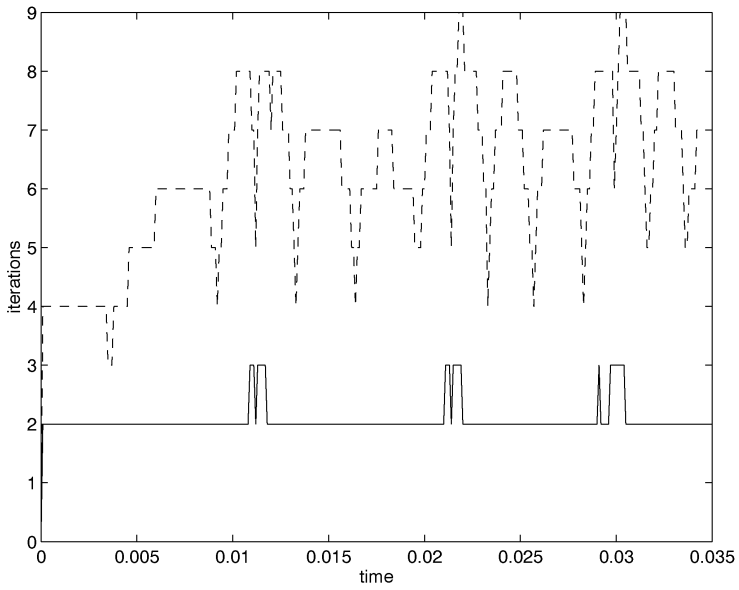

Figure 11 Iterations for Seven Body Mechanism, h=0.0001

5 Conclusions

The proposed algorithm, based on generalized coordinate partitioning method, reduces index 3 differential–algebraic equations of multibody dynamics to a set of second order, state–space ordinary differential equations, integrated using an implicit integration formula. A systematic technique to generate the required derivative information, using kinetic and kinematic equations, is presented. Three mechanical system models are used to validate the proposed formulation against well established explicit integrators. The algorithm is robust and is shown to accurately integrate the differential–algebraic equations of multibody dynamics for stiff mechanical systems. The constant order, constant step size implementation used in this paper limits the efficiency of the algorithm. Much as in the case of ordinary differential equations, efficiency will be significantly improved by using variable time step implicit integrators with error control.

Acknowledgment

This research was supported by the US Army Tank Automotive Command (TACOM) through the Automotive Research Center (Department of Defense contract number DAAE07–94–C–R094).

References

Ascher, U., M., Chin, H., Petzold, L., R., Reich, S., 1995, “Stabilization of Constrained Mechanical Systems with DAEs and Invariant Manifolds”, *Mech. Struct. & Mach.*, 23(2), pp. 135–157

Atkinson, K. E., 1989, An introduction to Numerical Analysis, Second Edition, John Wiley&Sons

Baumgarte, J., 1972, “Stabilization of constraints and integrals of motion in dynamical system”, *Comp. Meth. in Appl. Mech. and Eng.* 1, pp. 1–16

Brenan, K. E., Campbell, S. L., Petzold, L. R., 1989, The Numerical Solution of Initial Value Problems in Ordinary Differential–Algebraic Equations, North Holland Publishing Co.

Corwin, L. J., Szczarba, R. H., 1982, Multivariable Calculus, Marcel Dekker, New York

Eich, E., Fuhrer, C., Leimkuhler, B. J., Reich, S., 1990, “Stabilization and Projection Methods for Multibody Dynamics”, *Tech. Report*, Helsinki University of Technology, Finland

Fisette, P., Vaneghem B., 1996, ”Numerical integration of multibody dynamic equations using the coordinate partitioning method in an implicit Newmark scheme”, *Comput. Methods Appl. Mech. Engrg.*, 135, pp.85–105

Fuhrer, C., Leihmkuhler, B., J., 1991, “Numerical Solution of Differential–Algebraic Equations for Constrained Mechanical Motion”, *Numerische Mathematik*, vol. 59, pp. 55–69

Gear, C. W., Gupta, G., K., Leimkuhler, B., J., 1985, “Automatic Integration of the Euler–Lagrange Equations with Constraints”, *J. Comp. Appl. Math.*, vol.12&13, pp. 77–90

Hairer, E., Wanner, G., 1991, Solving Ordinary Differential Equations II: Stiff and Differential–Algebraic Problems, Springer–Verlag, Berlin

Haug, E. J., 1989, Computer Aided Kinematics and Dynamics of Mechanical Systems Volume I: Basic Methods, Allyn–Bacon, Boston

Haug, E. J., Yen, J., 1992, "Implicit Numerical Integration of Constrained Equation of Motion Via Generalized Coordinate Partitioning", *J. Mechanical Design*, (114), pp. 296–304

Hughes, T., J., 1987, The Finite Element Method, Prentice–Hall, Englewood Cliffs, New Jersey

Liang, C. G., Lance, G. M., 1987, "A differentiable Null Space Method for Constrained Dynamic Analysis," *J. of Mechanisms, Transmissions, and Automation in Design*, Vol.109, pp. 405–411

Mani, N. K., Haug E. J., Atkinson, K. E., 1985, "Application of Singular Value Decomposition for Analysis of Mechanical System Dynamics", *J. of Mechanisms, Transmissions, and Automation in Design*, Vol.107, pp. 82–87

Ostermeyer, G., P., 1990, "Baumgarte stabilization for differential–algebraic equations," NATO Advanced Research Workshop on Real Time Simulation, E.J. Haug and R. Deyo, Ed., Springer–Verlag Berlin Heidelberg

Petzold, L. R., 1982, "Differential /Algebraic Equations Are Not ODE's," *SIAM Journal of Scientific and Statistical Computing*, Vol. 3, No. 3, pp. 367–384

Potra, F., Rheinboldt, W. C., 1991, "On the numerical integration for Euler–Lagrange equations via tangent space parametrization", *Mech. Struct. Mach.*, Vol. 19, No. 1

Schiehlen, W., 1990, Multibody Handbook, New York, Berlin, Heidelberg, Springer

Wehage, R. A., Haug, E. J.,1982, "Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems," *J. Mech. Design*, Vol. 134, pp. 247–255

Yen, J., Petzold, L. R., 1995, ”On the Numerical Solution of Constrained Multibody Dynamic System, AHPCRC TRT 94–038, Univ. of Minnesota