A TOPOLOGY BASED APPROACH FOR EXPLOITING SPARSITY IN MULTIBODY DYNAMICS. JOINT FORMULATION

Dan Negrut[†], Radu Serban[†], Florian A. Potra[‡]

January, 1997

Abstract

In the present paper we are concerned with taking advantage of the problem structure in multibody dynamics simulation when the mechanical system is modeled using a minimal set of generalized coordinates. We show that the inertia matrix associated with any open or closed loop mechanism is positive definite by finding a simple mathematical expression for the quadratic form expressing the kinetic energy in an associated state space. Based on this result an algorithm which efficiently solves for the second time derivatives of the generalized coordinates is presented. Significant speed–ups are expected due both to the no fill–in factorization of the composite inertia matrix technique and to the degree of parallelism attainable with the new algorithm.

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

‡ Department of Mathematics and Computer Science, The University of Iowa, Iowa City, IA 52242

1 Introduction

In terms of the number of variables used to model a rigid multibody system we have the following two extreme approaches:

- the descriptor form, or the cartesian representation, or the body representation, in which the mechanical system is represented using for each body a set of coordinates specifying the position of a particular point on each body, along with a set of parameters specifying the orientation of that body with respect to a global reference frame
- the joint representation, or the recursive formulation, in which the mechanical system is represented in terms of a minimal set of generalized coordinates

As mentioned in [9] each approach has its advantages and drawbacks. The descriptor form is easier to understand, and can be readily formalized. However, when efficiency is an important issue the joint representation of a mechanical system outperforms the alternative descriptor form. In this context, the demands of real time simulation of mechanical systems comprising a large number of subsystems and bodies, like a car, or a tractor semi–trailer, stimulated the use of joint representation (JR) approach as an alternative way of modelling a mechanical system. The JR approach improves efficiency at the cost of a more difficult formalism and implementation of the method.

The improved efficiency of the JR approach is due to 2 factors. First, when a mechanical system is modeled using this formulation the dimension of the problem is smaller. Then, from a mathematical standpoint, the complexity of the problem is reduced. For instance, in the case of open loop mechanisms one does not have to deal with DAEs and the stabilization embedded in most DAE integration algorithms. This latter aspect hurts the efficiency of the cartesian formulation.

The treatment of index 3 DAEs [1] as they arise in multibody dynamics is difficult. The interested reader is referred to [1],[2],[4],[10],[12] for an account of the ideas underlying the treatment of DAEs'. In this context, not having to deal with DAEs, or in the case of closed loop mechanisms dealing with a much smaller number of algebraic equations, is a major advantage the of the JR approach.

In this paper an algorithm that efficiently solves for the generalized coordinate accelerations is presented. We show that taking advantage of the topology of the mechanical system can improve efficiency. The factorization of the composite inertia

matrix (CIM) is performed using a block Cholesky algorithm and it takes advantage of the sparsity pattern of CIM. Furthermore, the sparsity pattern is preserved; no fill–in occurs during factorization. The algorithm is very suitable for parallelization, the factorization progressing independently along the loops of the mechanical system in a similar way in which the integration process does. When a coordinate partitioning type method [12] based on an explicit integration formula is employed in solving the multibody dynamics problem, significant speed–ups are expected.

The paper is organized as follows. In Section 2 we briefly outline the JR approach to modelling multibody systems. The notation and relevant results of [11] will be used throughout this paper. In Section 3 the positive definiteness of CIM for open and closed loop mechanisms is proved and a simple expression is provided for the quadratic form expressing the kinetic energy of the system. The proposed technique of factorizing CIM is presented in Section 4. In Section 5 we show how to take advantage of the parallelism embedded in the algorithm. Results of some numerical experiments aimed at showing the capabilities of the new approach are presented in Section 6. We conclude the paper with some observations and future directions of research.

2 JR Modelling of Multibody Systems

2.1 Basic Concepts

The formalism behind JR modelling of multibody systems is complex. In this section we present a summary of it, focusing on results needed later on in the paper. The interested reader is referred to [11] for a detailed description of the formalism.

The main concept in JR modelling of multibody systems is that a body j is viewed as being located and oriented relative to the inboard body i. Figure 1. shows a pair of connected bodies with general relative rotational and translational motion. The inboard body i is located by the position vector $\mathbf{r}_i$ from the origin of the global xyz coordinate frame to the origin of the body $x_i'y_i'z_i'$ frame. The $x_i'y_i'z_i'$ frame is oriented by an orthogonal transformation matrix $\mathbf{A}_i$, which transforms a vector in the body i reference frame to the global reference frame. A joint reference frame, $x_{ij}''y_{ij}''z_{ij}''$, is defined and fixed on body i at the joint connection point O_{ij}'', which is located by the constant vector $\mathbf{s}_{ij}'$ from the origin of the $x_i'y_i'z_i'$ frame. A vector $\mathbf{d}_{ij}$ is defined from the origin of the joint frame, $x_{ij}''y_{ij}''z_{ij}''$, to the origin O_j', of the $x_j'y_j'z_j'$ frame location of the outboard body. Reference frames for each successive body in the kinematic chain are defined in the same way as those for body i.

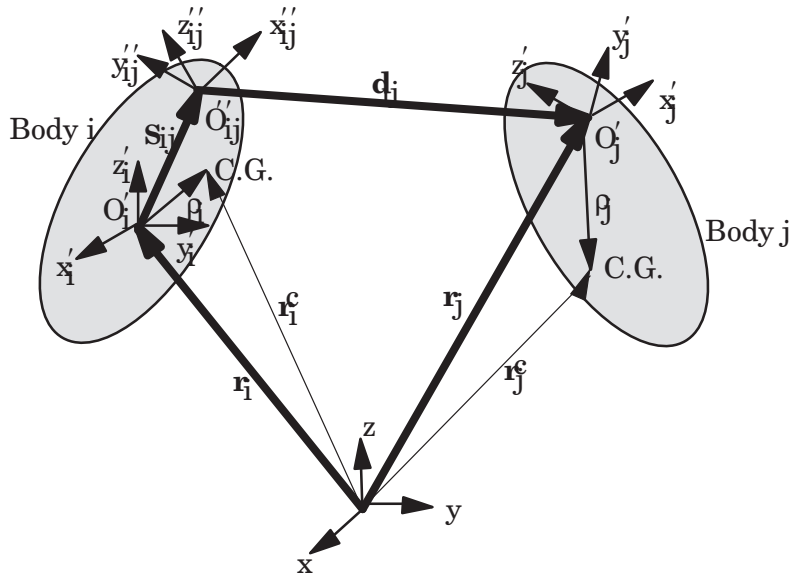


Figure 1 A pair of connected bodies in JR

For each body other than the base body, a body reference frame is defined at the joint connected to the inboard body in the kinematic chain, and joint reference frames are defined at joints that are connected to any outboard bodies. Therefore, the transformation from the inboard body reference frame, $x_i' y_i' z_i'$, to the outboard body reference frame requires only the constant transformation to the joint reference frame, $x_{ij}'' \; y_{ij}'' \; z_{ij}''$, followed by a joint transformation to the reference frame, $x_j' y_j' z_j'$, of body j. The reason for defining the body reference frame at the inboard joint is that every body, except for the base body, has one and only one inboard body in the kinematic chain, while it may have several outboard bodies. Thus, the origin of the body reference frame at the inboard joint is unique.

Once the position of body i is defined, the origin of the body j reference frame is located by the position vector given by

$$\mathbf{r}_j = \mathbf{r}_i + \mathbf{s}_{ij} + \mathbf{d}_{ij} \tag{1}$$

where $\mathbf{d}_{ij} = \mathbf{A}_i \mathbf{C}_{ij} \mathbf{d}_{ij}''(\mathbf{q}_j)$ is a vector from the joint reference frame origin O_{ij}'' on body i to the body reference frame origin O_j' on body j, $\mathbf{q}_j$ is the vector of relative coordinates for the joint and $\mathbf{C}_{ij}$ is the constant orthogonal transformation matrix between the O_{ij}'' and O_i' frames on body i.

The angular velocity of body j can be expressed as

$$\boldsymbol{\omega}_j = \boldsymbol{\omega}_i + \boldsymbol{\omega}_{ij} \tag{2}$$

where $\boldsymbol{\omega}_i$ is the angular velocity of body i, $\boldsymbol{\omega}_j$ is the angular velocity of body j, and $\boldsymbol{\omega}_{ij}$ is the angular velocity of body j relative to body i. The vector $\boldsymbol{\omega}_{ij}$ can be obtained from the relative coordinate velocity, $\dot{\mathbf{q}}_j$, by the equation $\boldsymbol{\omega}_{ij} = \mathbf{H}_j\left(\mathbf{A}_i, \mathbf{q}_j\right)\dot{\mathbf{q}}_j$ where $\mathbf{H}_j\left(\mathbf{A}_i, \mathbf{q}_j\right)$ is a transformation matrix that depends on the orientation of body i and on the relative coordinates, $\mathbf{q}_j$, which are defined for each type of joint.

The velocity of body j can be found by differentiating Equation (1), which yields

$$\dot{\mathbf{r}}_j = \dot{\mathbf{r}}_i + \dot{\mathbf{s}}_{ij} + \dot{\mathbf{d}}_{ij} \tag{3}$$

with $\dot{\mathbf{d}}_{ij} = \frac{\mathrm{d}}{\mathrm{dt}}\left(\mathbf{A}_i\mathbf{C}_{ij}\mathbf{d}'_{ij}\left(\mathbf{q}_j\right)\right) = \tilde{\boldsymbol{\omega}}_i\mathbf{d}_{ij} + \frac{\partial \mathbf{d}_{ij}}{\partial \mathbf{q}_j}\dot{\mathbf{q}}_j$. The tilde ($\tilde{}$) over the vector $\boldsymbol{\omega}_i$ signifies the skew-symmetric operator applied to this vector. After simple manipulations, equations (3) and (2) can be combined in matrix form as

$$\begin{bmatrix} \dot{\mathbf{r}}_j + \tilde{\mathbf{r}}_j\boldsymbol{\omega}_j \\ \boldsymbol{\omega}_j \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{r}}_i + \tilde{\mathbf{r}}_i\boldsymbol{\omega}_i \\ \boldsymbol{\omega}_i \end{bmatrix} + \begin{bmatrix} \dfrac{\partial \mathbf{d}_{ij}}{\partial \mathbf{q}_j} + \tilde{\mathbf{r}}_j\mathbf{H}_j \\ \mathbf{H}_j \end{bmatrix}\dot{\mathbf{q}}_j \tag{4}$$

or, in state–vector notation,

$$\hat{\mathbf{Y}}_j = \hat{\mathbf{Y}}_i + \mathbf{B}_j\dot{\mathbf{q}}_j \tag{5}$$

where the velocity state-vector of body i is defined as $\hat{\mathbf{Y}}_i = \begin{bmatrix} \dot{\mathbf{r}}_i + \tilde{\mathbf{r}}_i\boldsymbol{\omega}_i \\ \boldsymbol{\omega}_i \end{bmatrix}$, and the velocity transformation matrix $\mathbf{B}_j$ between bodies j and i is defined as $\mathbf{B}_j = \begin{bmatrix} \dfrac{\partial \mathbf{d}_{ij}}{\partial \mathbf{q}_j} + \tilde{\mathbf{r}}_j\mathbf{H}_j \\ \mathbf{H}_j \end{bmatrix}$. For the base body this matrix is the identity matrix of appropriate dimension. Finally, the acceleration state-vector of the body j is obtained by differentiating Equation (5) with respect to time, which yields

$$\dot{\hat{\mathbf{Y}}}_j = \dot{\hat{\mathbf{Y}}}_i + \mathbf{B}_j\ddot{\mathbf{q}}_j + \dot{\mathbf{B}}_j\dot{\mathbf{q}}_j \equiv \dot{\hat{\mathbf{Y}}}_i + \mathbf{B}_j\,\ddot{\mathbf{q}}_j + \mathbf{D}_j \tag{6}$$

Once the velocity state-vector $\hat{\mathbf{Y}}_j$ and acceleration state-vector, $\dot{\hat{\mathbf{Y}}}_j$ are obtained, the Cartesian velocity and acceleration, $\mathbf{Y}_j$ and $\dot{\mathbf{Y}}_j$, for body j can be recovered by using the relationships $\mathbf{Y}_j = \mathbf{T}_j\hat{\mathbf{Y}}_j$ and $\dot{\mathbf{Y}}_j = \mathbf{T}_j\dot{\hat{\mathbf{Y}}}_j - \mathbf{R}_j$, with $\mathbf{T}_j$ and $\mathbf{R}_j$ given as follows:

$$\mathbf{T}_j = \begin{bmatrix} \mathbf{I} & -\tilde{\mathbf{r}}_j \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \qquad \mathbf{R}_j = -\dot{\mathbf{T}}_j\hat{\mathbf{Y}}_j = \begin{bmatrix} \tilde{\dot{\mathbf{r}}}_j\boldsymbol{\omega}_j \\ \mathbf{0} \end{bmatrix} \tag{7}$$

In 2.2 we show how the equations of motion for an open loop mechanism are generated. In 2.3 a brief discussion points out the additional requirements imposed when one deals with closed loop mechanisms.

2.2 Equations of Motion for a Tree Structured System

The variational Newton-Euler equation of motion for a rigid body can be expressed in vector–matrix form as

$$\delta\mathbf{Z}^{\mathrm{T}}\left(\mathbf{M}\dot{\mathbf{Y}} - \mathbf{Q}\right) = 0 \tag{8}$$

where the Cartesian acceleration, $\dot{\mathbf{Y}}$, the Cartesian virtual displacements, $\delta\mathbf{Z}$, the modified mass matrix, $\mathbf{M}$, and the generalized force, $\mathbf{Q}$, are defined as follows:

$$\dot{\mathbf{Y}} = \begin{bmatrix} \ddot{\mathbf{r}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} \qquad \delta\mathbf{Z} = \begin{bmatrix} \delta\mathbf{r} \\ \delta\boldsymbol{\pi} \end{bmatrix} \qquad \mathbf{M} = \begin{bmatrix} m\mathbf{I} & -m\tilde{\rho} \\ m\tilde{\rho} & \mathbf{J} \end{bmatrix} \qquad \mathbf{Q} = \begin{bmatrix} \mathbf{F} - m\tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\rho \\ \boldsymbol{\eta} - \tilde{\boldsymbol{\omega}}\mathbf{J}\boldsymbol{\omega} \end{bmatrix}. \tag{9}$$

In (9) ρ is a vector from the body reference frame location, O', to the body center of gravity location. The correspondent state–vector form of (8) is

$$\delta\hat{\mathbf{Z}}^{\mathrm{T}}\left(\hat{\mathbf{M}}\dot{\hat{\mathbf{Y}}} - \hat{\mathbf{Q}}\right) = 0 \tag{10}$$

and with $\mathbf{T}$ and $\mathbf{R}$ as defined in (7), for $\delta\hat{\mathbf{Z}}$, $\hat{\mathbf{M}}$ and $\hat{\mathbf{Q}}$ in (10), we have

$$\delta\hat{\mathbf{Z}} = \mathbf{T}^{-1}\,\delta\mathbf{Z} \qquad \hat{\mathbf{M}} = \mathbf{T}^{\mathrm{T}}\mathbf{M}\mathbf{T} \qquad \hat{\mathbf{Q}} = \mathbf{T}^{\mathrm{T}}(\mathbf{Q} + \mathbf{M}\mathbf{R}) \ . \tag{11}$$

Consider now a multibody system with the tree structure shown in Figure 2. The variational equation of motion for the n–body system is

$$\sum_{i=1}^{p} \delta\hat{\mathbf{Z}}_i^{\mathrm{T}}(\hat{\mathbf{M}}_i\dot{\hat{\mathbf{Y}}}_i - \hat{\mathbf{Q}}_i) + \sum_{i=p+1}^{m} \delta\hat{\mathbf{Z}}_i^{\mathrm{T}}(\hat{\mathbf{M}}_i\dot{\hat{\mathbf{Y}}}_i - \hat{\mathbf{Q}}_i) + \sum_{i=m+1}^{n} \delta\hat{\mathbf{Z}}_i^{\mathrm{T}}(\hat{\mathbf{M}}_i\dot{\hat{\mathbf{Y}}}_i - \hat{\mathbf{Q}}_i)$$

$$\equiv \mathrm{EQ(1)} + \mathrm{EQ(2)} + \mathrm{EQ(3)} = 0 \ . \tag{12}$$

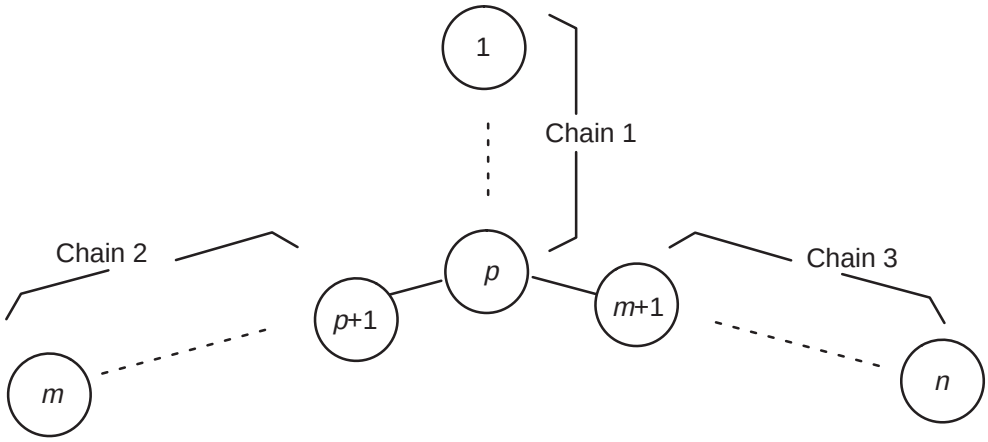


Figure 2 A Tree Structure

The state variation $\delta\hat{\mathbf{Z}}_i$ of body i in chain 2 may is expressed recursively in terms of inboard joint relative coordinate variations $\boldsymbol{\delta}\mathbf{q}_k$ and the state variation $\delta\hat{\mathbf{Z}}_p$ of the junction body as $\delta\hat{\mathbf{Z}}_i = \boldsymbol{\delta}\hat{\mathbf{Z}}_p + \sum_{k=p+1}^{i-1} \mathbf{B}_k \boldsymbol{\delta}\mathbf{q}_k$, while the acceleration state $\dot{\hat{\mathbf{Y}}}_i$ of body i is recursively expressed as $\dot{\hat{\mathbf{Y}}}_i = \dot{\hat{\mathbf{Y}}}_p + \sum_{k=p+1}^{i} \left(\mathbf{B}_k \ddot{\mathbf{q}}_k + \mathbf{D}_k\right)$ with $\mathbf{B}_k$ and $\mathbf{D}_k$ defined in (5) and (6).

Concentrating on chain 2, simple manipulations will bring (12) to the following form:

$$\mathrm{EQ(1)} + \mathrm{EQ(3)} + \delta\hat{\mathbf{Z}}_p^{\mathrm{T}}\left(\mathbf{K}_{p+1}\dot{\hat{\mathbf{Y}}}_p + \sum_{k=p+1}^{m} \mathbf{K}_k\mathbf{B}_k\ddot{\mathbf{q}}_k - (\mathbf{L}_{p+1} - \mathbf{K}_{p+1}\mathbf{D}_{p+1})\right)$$
$$+ \sum_{i=p+1}^{m} \delta\mathbf{q}_i^{\mathrm{T}}\left(\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_i\dot{\hat{\mathbf{Y}}}_p + \sum_{k=p+1}^{m} \mathbf{B}_i^{\mathrm{T}}\mathbf{K}_v\mathbf{B}_k\ddot{\mathbf{q}}_k - \mathbf{B}_i^{\mathrm{T}}\left(\mathbf{L}_i - \mathbf{K}_i \sum_{j=p+1}^{i} \mathbf{D}_j\right)\right) = 0 \quad . \tag{13}$$

where the subscript v of $\boldsymbol{K}_v$ is i, if $i \geq k$, or k, if $i < k$. The composite mass and force matrices $\mathbf{K}_i$ and $\mathbf{L}_i$ are recursively obtained as follows:

$$\mathbf{K}_i = \mathbf{K}_{i+1} + \hat{\mathbf{M}}_i \qquad\qquad \mathbf{L}_i = \mathbf{L}_{i+1} - \mathbf{K}_{i+1}\mathbf{D}_{i+1} + \hat{\mathbf{Q}}_i \tag{14}$$

The recursion starts from the last body in the chain, in this case the end body m, and proceeds upwards along the chain toward the base body. For the end body the composite

mass and force matrices $\mathbf{K}_m$ and $\mathbf{L}_m$ are identical to the state–space reduced mass matrix $\hat{\mathbf{M}}_m$ and force matrix $\hat{\mathbf{Q}}_m$ respectively.

For chains 3 and 1 one follows the same steps as for chain 2: express the state variation $\delta\hat{\mathbf{Z}}_i$ of body i in terms of inboard joint relative coordinate variations $\boldsymbol{\delta}\mathbf{q}_k$ and the state variation $\delta\hat{\mathbf{Z}}_p$ for chain 3 and $\delta\hat{\mathbf{Z}}_1$ for chain 1, followed by generating the state–vector accelerations $\dot{\hat{\mathbf{Y}}}_i$ recursively upwards along the chain toward the base body. Notice that for the junction body p the composite mass and force matrices are defined as follows

$$\begin{aligned}\mathbf{K}_p &= \hat{\mathbf{M}}_p + \mathbf{K}_{p+1} + \mathbf{K}_{m+1}\\ \mathbf{L}_p &= \hat{\mathbf{Q}}_p + \left(\mathbf{L}_{p+1} - \mathbf{K}_{p+1}\mathbf{D}_{p+1}\right) + \left(\mathbf{L}_{m+1} - \mathbf{K}_{m+1}\mathbf{D}_{m+1}\right)\end{aligned} \tag{15}$$

After simple manipulations one obtains

$$\begin{aligned}&\delta\hat{\mathbf{Z}}_1^{\mathrm{T}}\left(\mathbf{K}_1\dot{\hat{\mathbf{Y}}}_1 - \mathbf{L}_1 + \sum_{k=2}^{n}\mathbf{K}_k\mathbf{B}_k\ddot{\mathbf{q}}_k\right) + \sum_{i=2}^{n}\boldsymbol{\delta}\mathbf{q}_i^{\mathrm{T}}\left(\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_i\dot{\hat{\mathbf{Y}}}_1 + \sum_{k=2}^{n}\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_v\mathbf{B}_k\ddot{\mathbf{q}}_k - \mathbf{B}_i^{\mathrm{T}}\left(\mathbf{L}_i - \mathbf{K}_i\sum_{j=2}^{i}\mathbf{D}_j\right)\right)\\ &+ \sum_{i=p+1}^{m}\boldsymbol{\delta}\mathbf{q}_i^{\mathrm{T}}\left[\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_i\dot{\hat{\mathbf{Y}}}_1 + \sum_{k=2}^{p}\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_v\mathbf{B}_k\ddot{\mathbf{q}}_k + \sum_{k=p+1}^{m}\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_v\mathbf{B}_k\ddot{\mathbf{q}}_k - \mathbf{B}_i^{\mathrm{T}}\left[\mathbf{L}_i - \mathbf{K}_i\left(\sum_{j=2}^{p}\mathbf{D}_j + \sum_{j=p+1}^{i}\mathbf{D}_j\right)\right]\right]\\ &+ \sum_{i=m+1}^{n}\boldsymbol{\delta}\mathbf{q}_i^{\mathrm{T}}\left[\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_i\dot{\hat{\mathbf{Y}}}_1 + \sum_{k=2}^{p}\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_v\mathbf{B}_k\ddot{\mathbf{q}}_k + \sum_{k=m+1}^{n}\mathbf{B}_i^{\mathrm{T}}\mathbf{K}_v\mathbf{B}_k\ddot{\mathbf{q}}_k - \mathbf{B}_i^{\mathrm{T}}\left[\mathbf{L}_i - \mathbf{K}_i\left(\sum_{j=2}^{p}\mathbf{D}_j + \sum_{j=m+1}^{i}\mathbf{D}_j\right)\right]\right]\\ &= 0\end{aligned} \tag{16}$$

Thus, starting from (12) and recursively expressing the state–vector virtual displacements and accelerations, one ends up with the equivalent equation (16). Equating to zero the expressions multiplying the arbitrary virtual displacements, one obtains the equations of motion of the open loop mechanism in Figure 2. For any tree structured mechanism the equations of motion are obtained following the steps outlined here.

2.3 Equations of Motion for a Closed–Loop System

If the mechanism contains closed loops a number of joints are cut in the process of obtaining a spanning tree. Constraints should therefore be imposed in order to preserve

the behavior of the mechanism. Consequently, the virtual displacements in (16) are no longer arbitrary. They are related through the constraint equations which means that (16) expressed in matrix form as $\boldsymbol{\delta q}^{\mathrm{T}}\left[\overline{\mathbf{M}}\ddot{\mathbf{q}} - \overline{\mathbf{Q}}\right] = \mathbf{0}$ should hold for all virtual displacements satisfying $\boldsymbol{\Phi}_{\mathbf{q}}\,\boldsymbol{\delta q} = \mathbf{0}$. Here it is assumed that the collection of all cut constraint equations for the system are expressed by $\boldsymbol{\Phi}(\mathbf{q}) = \mathbf{0}$. The entries of matrix $\overline{\mathbf{M}}$ will be detailed in (18) after introducing a few concepts from graph theory.

Finally, using the Lagrange multiplier theorem[6] and taking into account the constraint acceleration equations, the equations of motion for closed–loop mechanical systems are obtained as follows.

$$\begin{bmatrix} \overline{\mathbf{M}} & \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}} \\ \boldsymbol{\Phi}_{\mathbf{q}} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \overline{\mathbf{Q}} \\ \boldsymbol{\gamma} \end{Bmatrix} . \tag{17}$$

3 Proving the Positive Definiteness of CIM

Two properties of the composite inertia matrix $\overline{\mathbf{M}}$; i.e., the special structure (or sparsity pattern), and the form of the entries are used to prove its positive definiteness. Both these properties are dictated by the topology of the system as reflected by the equations of motion.

In JR a graph is associated to each mechanism by considering its bodies and joints as being the vertices and the connecting edges of the graph respectively. In addition to the graph concepts introduced in [11], we define a few others needed in the proof of the result.

Two concepts introduced in Section 2 induce a direction in the spanning tree: the inboard–outboard relationship between two neighbor bodies, along with the fact that each body in the spanning tree has one and only one inboard body while having an arbitrary number of outboard bodies. In what follows we refer to the *directed* spanning tree associated with the cut joint mechanism as the *spanning tree* unless otherwise stated.

We say that body j is *descendent* of body i if there is a path in the spanning tree from body i to body j. *Family* i, denoted by $\mathbf{F}(i)$, is the set of all descendents of body i. If we add body i to $\mathbf{F}(i)$ we denote the new collection of bodies by $\mathbf{F}[i]$. The sequence of bodies on the unique path starting from body i and ending at body j is called *subchain* and denoted by $\mathbf{d}[i,j]$. If body i is left out from this sequence we denote the subchain by $\mathbf{d}(i,j]$. The sub–chains $\mathbf{d}[i,j)$ and $\mathbf{d}(i,j)$ are defined in similarly. The subchain starting at the base body and ending at body i is denoted by $\mathbf{d}[i]$. Notice that a subchain inherits the

order from the spanning tree, while a family doesn't. By concatenating subchain $\mathbf{d}[i]$ to family $\mathbf{F}(i)$ one obtains the *subtree* $\mathbf{c}(i)$ which will be ordered from the base body down to body i. If b is the base body, the subtree $\mathbf{c}(b)$ represents the entire spanning tree and for convenience it will be denoted simply by $\mathbf{S}$. Finally, we say body i is a *leaf* of the spanning tree if it has no outboard bodies.

Lemma 1. Let $\mathbf{x}_w$ be matrix or vector quantity associated with body w, and $\mathbf{y}_s$ be a matrix or vector quantity associated to body s such that the multiplication $\mathbf{x}_w\mathbf{y}_s$ is well defined. Then

$$\sum_{s\in\mathbf{F}(r)}\sum_{w\in\mathbf{F}[s]}\mathbf{x}_w\mathbf{y}_s = \sum_{w\in\mathbf{F}(r)}\sum_{s\in\mathbf{d}(r,w]}\mathbf{x}_w\mathbf{y}_s \ ,$$

where r is an arbitrary body of the spanning tree.

The proof of this result is based on induction and will be skipped. Based on *Lemma 1*, by taking r to be a fictitious inboard body of the base body we have the following

Corollary 1. With $\mathbf{y}_s{}^{\mathrm{T}}$ and $\mathbf{x}_w$ vectors of appropriate dimensions associated with bodies s and w respectively, the following holds:

$$\sum_{s\in\mathbf{S}}\sum_{w\in\mathbf{F}[s]}\mathbf{y}_s^{\mathrm{T}}\mathbf{x}_w = \sum_{w\in\mathbf{S}}\sum_{s\in\mathbf{d}[w]}\mathbf{y}_s^{\mathrm{T}}\mathbf{x}_w \ .$$

In general the unknowns related to body u occupy position i in the global vector of unknowns. We define a permutation $\mathbf{p}$ which for each body gives its position in the global vector of unknowns; $\mathbf{p}(u) = i$. If u and v are two bodies of the spanning tree, with $i = \mathbf{p}(u)$ and $j = \mathbf{p}(v)$ the block entry (i,j) of CIM is given as

$$\overline{\mathbf{M}}[i,j] = \begin{cases} \mathbf{B}_u^{\mathrm{T}}\mathbf{K}_u\mathbf{B}_v & \text{if } v \in \mathbf{d}[u] \\ \mathbf{B}_u^{\mathrm{T}}\mathbf{K}_v\mathbf{B}_v & \text{if } v \in \mathbf{F}(u) \\ \mathbf{0} & \text{if } v \notin \mathbf{c}(u) \end{cases} \tag{18}$$

The matrices $\mathbf{B}$ and $\mathbf{K}$ are as defined in 2.1 and 2.2 respectively. Notice that rather than dealing with scalar entries, $\overline{\mathbf{M}}$ is defined in terms of blocks. The number of rows and columns of block $\overline{\mathbf{M}}[i,j]$ is equal to the dimension of $\mathbf{q}_i$ and $\mathbf{q}_j$ respectively. In general, if N is the number of bodies in the system and n_i is the dimension of the generalized vector $\mathbf{q}_i$, $\mathbf{q}_i \in \mathbb{R}^{\mathrm{n}_i}$, then $\overline{\mathbf{M}} \in \mathbb{R}^{n\times n}$, with $\mathrm{n} = \sum_{i=1}^{\mathrm{N}} \mathrm{n}_i$

Theorem. The composite inertia matrix $\overline{\mathbf{M}}$ as defined in (18) is positive definite.

Proof. Defining $\mathbf{v} = \left[\mathbf{v}_1^{\mathrm{T}}, \mathbf{v}_2^{\mathrm{T}}, .., \mathbf{v}_n^{\mathrm{T}}\right]^{\mathrm{T}}$, we show that $\frac{1}{2}\mathbf{v}^{\mathrm{T}}\overline{\mathbf{M}}\mathbf{v} \geq 0$, equality being obtained only if $\mathbf{v} = \mathbf{0}$. Considering the vector $\mathbf{z} = \overline{\mathbf{M}}\mathbf{v}$ and denoting $\mathbf{y}_k \equiv \mathbf{B}_k\mathbf{v}_k$ we have

$$\mathbf{z}(r) = \sum_{s\in\mathbf{d}[r]} \mathbf{B}_r^{\mathrm{T}}\mathbf{K}_r\mathbf{y}_s + \sum_{s\in\mathbf{F}(r)} \mathbf{B}_r^{\mathrm{T}}\mathbf{K}_s\mathbf{y}_s \tag{19}$$

$$= \sum_{s\in\mathbf{d}[r]} \mathbf{B}_r^{\mathrm{T}} \sum_{w\in\mathbf{F}[r]} \hat{\mathbf{M}}_w\mathbf{y}_s + \sum_{s\in\mathbf{F}(r)} \mathbf{B}_r^{\mathrm{T}} \sum_{w\in\mathbf{F}[s]} \hat{\mathbf{M}}_w\mathbf{y}_s \ .$$

Using the result of *Lemma 1* and defining $\mathbf{u}_w \equiv \sum_{s\in\mathbf{d}[w]} \mathbf{y}_s$ we further have

$$\mathbf{z}(r) = \sum_{w\in\mathbf{F}[r]} \sum_{s\in\mathbf{d}[r]} \mathbf{B}_r^T\hat{\mathbf{M}}_w\mathbf{y}_s + \sum_{w\in\mathbf{F}(r)} \sum_{s\in\mathbf{d}(r,w]} \mathbf{B}_r^T\hat{\mathbf{M}}_w\mathbf{y}_s \tag{20}$$

$$= \sum_{s\in\mathbf{d}[r]} \mathbf{B}_r^T\hat{\mathbf{M}}_r\mathbf{y}_s + \sum_{w\in\mathbf{F}(r)} \sum_{s\in\mathbf{d}[w]} \mathbf{B}_r^T\hat{\mathbf{M}}_w\mathbf{y}_s = \sum_{w\in\mathbf{F}[r]} \sum_{s\in\mathbf{d}[w]} \mathbf{B}_r^T\hat{\mathbf{M}}_w\mathbf{y}_s$$

$$= \sum_{w\in\mathbf{F}[r]} \mathbf{B}_r^T\hat{\mathbf{M}}_w \sum_{s\in\mathbf{d}[w]} \mathbf{y}_s = \mathbf{B}_r^T \sum_{w\in\mathbf{F}[r]} \hat{\mathbf{M}}_w\mathbf{u}_w \ .$$

Finally, by using the result of *Corollary 1*,

$$\frac{1}{2}\mathbf{v}^{\mathrm{T}}\overline{\mathbf{M}}\mathbf{v} = \frac{1}{2}\sum_{r\in\mathbf{S}} \mathbf{y}_r^T \sum_{w\in\mathbf{F}[r]} \hat{\mathbf{M}}_w\mathbf{u}_w = \frac{1}{2}\sum_{r\in\mathbf{S}} \sum_{w\in\mathbf{F}[r]} \mathbf{y}_r^T\hat{\mathbf{M}}_w\mathbf{u}_w = \frac{1}{2}\sum_{r\in\mathbf{S}} \sum_{w\in\mathbf{F}[r]} \mathbf{y}_r^T\hat{\mathbf{M}}_w\mathbf{u}_w \tag{21}$$

$$= \frac{1}{2}\sum_{w\in\mathbf{S}} \sum_{r\in\mathbf{d}[w]} \mathbf{y}_r^T\hat{\mathbf{M}}_w\mathbf{u}_w = \frac{1}{2}\sum_{w\in\mathbf{S}} \mathbf{u}_w^{\mathrm{T}}\hat{\mathbf{M}}_w\mathbf{u}_w \equiv \frac{1}{2}\sum_{w\in\mathbf{S}} \| \mathbf{u}_w \|^2_{\hat{\mathbf{M}}_w} \ .$$

In (21), $\| \cdot \|_{\hat{\mathbf{M}}_w}$ defines a norm since $\hat{\mathbf{M}}_w$ is positive definite. To see this, notice first that the state–vector reduced matrix $\hat{\mathbf{M}} = \mathbf{T}^{\mathrm{T}}\mathbf{M}\mathbf{T} = \mathbf{T}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{M}_c\mathbf{H}\mathbf{T}$, with $\mathbf{M}$ and $\mathbf{T}$ defined as (9) and (7) respectively, and $\mathbf{M}_c$ and $\mathbf{H}$ given by

$$\mathbf{M}_c = \begin{bmatrix} m\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_c \end{bmatrix} \qquad , \qquad \mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \tilde{\rho} & \mathbf{I} \end{bmatrix} . \tag{22}$$

$\mathbf{J}_c$ is the body inertia matrix with respect to a reference frame located at the centroid of the body and parallel with the body reference frame as defined in Section 2. Therefore, both $\mathbf{J}_c$ and $\mathbf{M}_c$ are positive definite. Since matrices $\mathbf{H}$ and $\mathbf{T}$ are nonsingular we conclude that $\hat{\mathbf{M}}$ is positive definite.

The last sum in (21) will be equal to zero only when $\mathbf{u}_w = \mathbf{0},\ \forall w \in \mathbf{S}$. If this is true, then $\mathbf{y}_w = \mathbf{0},\ \forall w \in \mathbf{S}$, or equivalently $\mathbf{B}_w\mathbf{v}_w = \mathbf{0},\ \forall w \in \mathbf{S}$. Since proper modelling in JR requires $\mathbf{B}_w$ to be full column rank we conclude that $\mathbf{v}_w = \mathbf{0},\ \forall w \in \mathbf{S}$. □

Taking $\mathbf{v} = \left[\mathbf{v}_1^T, \mathbf{v}_2^T, .., \mathbf{v}_n^T\right]^T \equiv \left[\dot{\mathbf{q}}_{\mathbf{b}(1)}^T, \dot{\mathbf{q}}_{\mathbf{b}(2)}^T, .., \dot{\mathbf{q}}_{\mathbf{b}(n)}^T\right]^T$, we define the quadratic form $\frac{1}{2}\mathbf{v}^T\overline{\mathbf{M}}\mathbf{v}$ as representing the kinetic energy of the tree–structured mechanism in the state–vector space. Notice that $\mathbf{u}_w$ defined above represents the state–vector velocity of body w, i.e., the vector denoted by $\hat{\mathbf{Y}}_w$ in Section 2. Hence (21) says that the kinetic energy of the system in the state–vector space is equal to the sum of the kinetic energy in the state–vector space of each body in the system. Based on (11) and the fact that $\mathbf{Y}_w = \mathbf{T}_w\hat{\mathbf{Y}}_w$ follows that

Corollary 2. The kinetic energy of each body in the system is the same as its kinetic energy in the state–vector space. Therefore the kinetic energy of the tree–structured mechanism has the same property.

4 Factoring the Augmented Matrix

Since the composite inertia matrix $\overline{\mathbf{M}}$ is positive definite the algorithm of choice for factoring this matrix is block Cholesky. In the most general case one has to factor the coefficient matrix of (17). We refer to this matrix as the augmented matrix and we denote it by **A**.

In the case of a tree–structured mechanism, the augmented matrix is identical to the composite inertia matrix. Block Cholesky factorization of the matrix **A** will result in a decomposition $\mathbf{L}\mathbf{L}^T$ whose implementation is detailed later on.

Cholesky factorization is not directly applicable when closed loops are present in the mechanical system. In this instance the augmented matrix is not positive definite due to the presence of the Jacobian of the constraint equations. The augmented matrix will be therefore factorized as follows:

$$\mathbf{A} \equiv \begin{bmatrix} \overline{\mathbf{M}} & \mathbf{\Phi}_\mathbf{q}^T \\ \mathbf{\Phi}_\mathbf{q} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{L} & \mathbf{0} \\ \mathbf{T}^T & \mathbf{I}_m \end{bmatrix} \begin{bmatrix} \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & -\mathbf{T}^T\mathbf{T} \end{bmatrix} \begin{bmatrix} \mathbf{L}^T & \mathbf{T} \\ \mathbf{0} & \mathbf{I}_m \end{bmatrix} \tag{23}$$

In (23), **L** is obtained from the Cholesky factorization of $\overline{\mathbf{M}} = \mathbf{L}\mathbf{L}^T$, while $\mathbf{T} \in \mathbb{R}^{n\times m}$ is the solution of the matrix equation

$$\mathbf{L}\mathbf{T} = \mathbf{\Phi}_\mathbf{q}^T \tag{24}$$

Finally, the matrix $\mathbf{T}^{\mathrm{T}}\mathbf{T}$ is positive definite since the Jacobian of the constraint equations is assumed to have full row rank. In what follows we concentrate on taking advantage of the topology of the problem when performing the factorization of the augmented matrix.

4.1 Factoring CIM

An important feature of the JR formulation is that sparsity is not lost, and furthermore it is structured. Qualitatively, the sparsity is dictated by the topology of the mechanism, and in the case of closed loop mechanisms also by the cut joints. The major problem with direct algorithms for sparse systems is that to a certain extent the sparsity is lost during factorization. The proposed algorithm preserves the sparsity pattern of CIM; i.e., no fill–in occurs during the Cholesky factorization. For this to happen, one might have to reorder the unknown accelerations $\ddot{\mathbf{q}}_i$ in the vector of unknowns $\ddot{\mathbf{q}} = \left[\ddot{\mathbf{q}}_1^{\mathrm{T}}, \ddot{\mathbf{q}}_2^{\mathrm{T}}, .., \ddot{\mathbf{q}}_n^{\mathrm{T}}\right]^{\mathrm{T}}$ to obtain $\ddot{\mathbf{q}}_{new} = \left[\ddot{\mathbf{q}}_{b(1)}^{\mathrm{T}}, \ddot{\mathbf{q}}_{b(2)}^{\mathrm{T}}, .., \ddot{\mathbf{q}}_{b(n)}^{\mathrm{T}}\right]^{\mathrm{T}} = \mathbf{P}\ddot{\mathbf{q}}$, where $\mathbf{P}$ is a permutation matrix. If, after reordering, body u is assigned position i in $\ddot{\mathbf{q}}_{new}$, we define a permutation array $\mathbf{b}$ for which $\mathbf{b}(i) = u$. The inverse permutation $\mathbf{p}(u) = i$ was defined in Section 3.

Lemma 2. Let u and v be two bodies in the spanning tree associated with a mechanical system. No fill–in occurs during the block Cholesky factorization of CIM if the following holds:

$$\text{If } u \in \mathbf{F}(v) \text{ then } \mathbf{p}(u) < \mathbf{p}(v) .$$

Proof. The symmetry and the block structure of the composite inertia matrix are used throughout the proof. $\overline{\mathbf{M}}$ is factored using a block–oriented Cholesky factorization and we show that blockwise, $\mathbf{L}[k,j] = \mathbf{0}$ if $\overline{\mathbf{M}}[k,j] = \mathbf{0}$. For this, notice first that $\mathbf{L}[1,1]$ is determined from $\mathbf{L}[1,1]\mathbf{L}^{\mathrm{T}}[1,1] = \overline{\mathbf{M}}[1,1]$ and due to the positive definiteness of $\overline{\mathbf{M}}$ it is not identically zero. $\mathbf{L}[k,1]$ is the solution of the matrix equation $\mathbf{L}[1,1]\mathbf{L}^{\mathrm{T}}[k,1] = \overline{\mathbf{M}}[k,1]$, $1 < k \leq n$, and clearly $\mathbf{L}[k,1] = \mathbf{0}$ when $\overline{\mathbf{M}}[k,1] = \mathbf{0}$.

Suppose that up to column $j-1$ the conclusion of *Lemma 2* holds. The positive definiteness of $\overline{\mathbf{M}}$ implies that $\mathbf{L}[j,j]$ obtained from $\mathbf{L}[j,j]\mathbf{L}^{\mathrm{T}}[j,j] = \overline{\mathbf{M}}[j,j] - \sum_{i=1}^{j-1} \mathbf{L}[j,i]\mathbf{L}^{\mathrm{T}}[j,i]$, is not identically zero. For $k > j$, $\mathbf{L}[k,j]$ is the solution of the matrix equation $\mathbf{L}[j,j]\mathbf{L}^{\mathrm{T}}[k,j] = \overline{\mathbf{M}}[k,j] - \sum_{i=1}^{j-1} \mathbf{L}[j,i]\mathbf{L}^{\mathrm{T}}[k,i]$.

Let $\overline{\mathbf{M}}[k,j] = \mathbf{0}$ and suppose there is an i, $i < j < k$, such that $\mathbf{L}[j,i]\mathbf{L}^{\mathrm{T}}[k,i] \neq \mathbf{0}$. Then $\overline{\mathbf{M}}[i,j] \neq \mathbf{0}$ and $\overline{\mathbf{M}}[i,k] \neq \mathbf{0}$. Let bodies u, v, w be such that $\mathbf{p}(u) = i$, $\mathbf{p}(v) = j$ and $\mathbf{p}(w) = k$. Because of the way in which precedence was defined in the vector of unknowns, this implies that $v \in \mathbf{d}[u]$ and $w \in \mathbf{d}[u]$. But then, since $j < k$, we also have $w \in \mathbf{d}[v]$ and therefore $\overline{\mathbf{M}}[k,j] \neq \mathbf{0}$. This is a contradiction and thus, no such i exists. Then $\mathbf{L}[j,j]\mathbf{L}^{\mathrm{T}}[k,j] = \mathbf{0}$ and therefore $\mathbf{L}[k,j] = \mathbf{0}$.□

The reordering induced by the permutation array $\mathbf{r}$ is not unique. We do not provide an algorithm for computing the permutation array $\mathbf{r}$ but developing one is straightforward. Notice that a transformation of the form $\overline{\mathbf{M}}_{new} = \mathbf{P}\overline{\mathbf{M}}\mathbf{P}^{\mathrm{T}}$ is to be applied to the original CIM too. Since the reordering holds for the whole simulation, an initial renumbering of the mechanism joints would take care of this aspect. In this case no permutations are needed to ensure no fill–in factorization of the CIM.

4.2 The Closed–Loop Case

In the case of a closed–loop mechanical system, a set of M joints connecting bodies in the system are cut to obtain a spanning tree of the mechanism. A set of constraint equations $\mathbf{\Phi}^{(1)}$, $\mathbf{\Phi}^{(2)}$, .. $\mathbf{\Phi}^{(M)}$, are imposed to account for the cut joints. In Section 2.3 the collection of constraint equations was denoted by $\mathbf{\Phi}^{\mathrm{T}} = \left[\mathbf{\Phi}^{(1)^{\mathrm{T}}}\mathbf{\Phi}^{(2)^{\mathrm{T}}}... \mathbf{\Phi}^{(M)^{\mathrm{T}}}\right] = \mathbf{0}$ with $\mathbf{\Phi} \in \mathbb{R}^m$, $m = \sum_{i=1}^{M} m_i$, $\mathbf{\Phi}^{(i)} \in \mathbb{R}^{m_i}$. With a columnwise partition of $\mathbf{T}$ in (23), one has to solve for $\mathbf{T}^{(j)} \in \mathbb{R}^{n \times m_j}$ in $\mathbf{L}\mathbf{T}^{(j)} = \mathbf{\Phi}_{\mathbf{q}}^{(j)^{\mathrm{T}}}$ for $1 \leq j \leq M$. With the precedence in the vector of unknowns induced by *Lemma 2*, the following result singles out the zeros of the matrix $\mathbf{T}$ in (24):

Lemma 3. Let constraint j account for the cut joint between bodies k and l and let $i = \mathbf{p}(u)$, with $u \notin \mathbf{d}[k] \cup \mathbf{d}[l]$. Then $\mathbf{T}^{(j)}[i] = \mathbf{0}$.

Proof. Let i be the smallest integer which satisfies the hypothesis of the lemma and yet violates the conclusion.

Row i of $\mathbf{L}\mathbf{T}^{(j)} = \mathbf{\Phi}_{\mathbf{q}}^{(j)^{\mathrm{T}}}$ reads $\sum_{v=1}^{i-1} \mathbf{L}[i,v]\mathbf{T}^{(j)}[v] + \mathbf{L}[i,i]\mathbf{T}^{(j)}[i] = \mathbf{\Phi}_{\mathbf{q}u}^{(j)}$. Since $u \notin \mathbf{d}[k] \cup \mathbf{d}[l]$, we have $\mathbf{\Phi}_{\mathbf{q}u}^{(j)} = \mathbf{0}$ and therefore i cannot be 1. Suppose there exists v, $1 \leq v < i$, such that $\mathbf{L}[i,v]\mathbf{T}^{(j)}[v] \neq \mathbf{0}$. With the precedence induced by *Lemma 2*, and w defined by $v = \mathbf{b}(w)$, we have that $w \in \mathbf{F}(u)$ and therefore $u \in \mathbf{d}[w]$. Finally, since

$\mathbf{T}^{(j)}[v] \neq \mathbf{0}$ and $v < i$ we have $w \in \mathbf{d}[k] \cup \mathbf{d}[l]$. But then $u \in \mathbf{d}[k] \cup \mathbf{d}[l]$, which is a contradiction.□

4.3 Algorithm

Based on the decomposition (23) we show in the algorithm below the steps for solving the augmented system (17). The ordering of the elements of the unknown vector as induced by *Lemma 2* is assumed:

Algorithm for computing the accelerations and the Lagrange Multipliers in JR formulation:

Step 0	Set $\mathbf{y}_n = \mathbf{0}$, $\mathbf{y}_n \in \mathbb{R}^n$.
Step 1	Factor $\overline{\mathbf{M}} = \mathbf{L}\mathbf{L}^{\mathrm{T}}$.
Step 2	Solve $\mathbf{L}\mathbf{z}_n = \mathbf{Q}^{\mathrm{A}}$, $\mathbf{z}_n \in \mathbb{R}^n$.
	IF (closed–loop) then:
Step 3.1	Solve $\mathbf{L}\mathbf{T} = \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}$, $\mathbf{T} \in \mathbb{R}^{n \times m}$;
Step 3.2	Set $\mathbf{z}_m = \mathbf{T}^{\mathrm{T}}\mathbf{z}_n - \boldsymbol{\gamma}$;
Step 3.3	Compute $\mathbf{T}^{\mathrm{T}}\mathbf{T}$;
Step 3.4	Solve $\mathbf{T}^{\mathrm{T}}\mathbf{T}\boldsymbol{\lambda} = \mathbf{z}_m$, $\boldsymbol{\lambda} \in \mathbb{R}^m$;
Step 3.5	Set $\mathbf{y}_n = \mathbf{T}\boldsymbol{\lambda}$;
	endIF
Step 4	Solve $\mathbf{L}^{\mathrm{T}}\ddot{\mathbf{q}} = \mathbf{z}_n - \mathbf{y}_n$.

Remarks.

1. During Step 1 one factors the CIM using block Cholesky with no fill–in. Therefore the sparsity pattern for $\mathbf{L}$ is known beforehand.
2. Solving for $\mathbf{z}_n$ requires only to forward substitution. Without taking sparsity into account in Step 2, row i of $\mathbf{L}\mathbf{z}_n = \mathbf{Q}^{\mathrm{A}}$ reads $\sum_{v=1}^{i-1} \mathbf{L}[i,v]\mathbf{z}_n[v] + \mathbf{L}[i,i]\mathbf{z}_n[i] = \mathbf{Q}^{\mathrm{A}}[i]$. A reduced number of operations results if in the light of *Lemma 2* one equivalently expresses row i as $\sum_{k \in \mathbf{F}(\mathbf{r}(i))} \mathbf{L}[i,\mathbf{b}(k)]\mathbf{z}_n[\mathbf{b}(k)] + \mathbf{L}[i,i]\mathbf{z}_n[i] = \mathbf{Q}^{\mathrm{A}}[i]$ when solving for $\mathbf{z}_n[i]$. The same remark holds when backward substitution retrieves the accelerations $\ddot{\mathbf{q}}$ during Step 4.

3. Using the result of *Lemma 3*, one can further take advantage of the problem structure when computing the matrix $\mathbf{T}$ during Step 3.1. Basically, as with remark 1, some entries of this matrix are known beforehand to be zero and need not be computed. Sparsity can be further exploited when performing any matrix–matrix, or matrix–vector multiplications involving $\mathbf{T}$ in Step 3.
4. From *Lemma 3*, the coefficient matrix in $\mathbf{T}^{\mathrm{T}}\mathbf{T}\boldsymbol{\lambda} = \mathbf{z}_m$ is dense. The $m \times m$ coefficient matrix is positive definite (see Section 4) and it is factored using Cholesky decomposition.

In general, it is not possible to give an operation count for this algorithm. The number of operations will depend on the topology of the particular mechanical system one deals with. We expect our algorithm to require fewer operations than gaussian elimination to solve (17). In 6 we present a comparison in terms of number of operations and CPU time for several alternatives when dealing with a vehicle model.

5 Taking Advantage of Parallelism

Throughout this Section we assume that the precedence in the vector of unknowns is as in *Lemma 2*. Before stating the main result of this Section, we define two sets. Let j be an integer such that $1 \leq j \leq \mathrm{N}$, where N is the dimension of the CIM. Define

$$\mathbf{D}(j) \equiv \{\, i \mid 1 \leq i < j \ \text{ and } \ \mathbf{b}(i) \in \mathbf{F}(\mathbf{b}(j))\}$$

$$\mathbf{U}(j) \equiv \{\, i \mid j \leq i \leq N \ \text{ and } \ \mathbf{b}(i) \in \mathbf{d}[\mathbf{b}(j)]\}$$

Lemma 4. During the Cholesky factorization of the CIM, for any j, $1 \leq j \leq \mathrm{N}$, and $k \in \mathbf{U}(j)$, $\mathbf{L}[k,j]$ can be computed provided for each $i \in \mathbf{D}(j)$, $\mathbf{L}[l,i]$ is available for $l \in \mathbf{U}(i)$.

Proof. Let v be the body in the system for which $\mathbf{p}(v) = j$. We first show that the conclusion holds if j corresponds to a leaf v. Then we will show that it holds for any j.

One step of the Cholesky factorization algorithm can be carried out in the following sequence: solve first for $\mathbf{L}[j,j]$ in (25), and then for $\mathbf{L}[k,j]$, $j < k \leq \mathrm{N}$ in (26). In what follows we concentrate on computing the summation in the RHS of (25) and (26).

$$\mathbf{L}[j,j]\mathbf{L}^{\mathrm{T}}[j,j] = \overline{\mathbf{M}}[j,j] - \sum_{i=1}^{j-1} \mathbf{L}[j,i]\mathbf{L}^{\mathrm{T}}[j,i] \tag{25}$$

$$\mathbf{L}[j,j]\mathbf{L}^{\mathrm{T}}[k,j] = \overline{\mathbf{M}}[k,j] - \sum_{i=1}^{j-1} \mathbf{L}[j,i]\mathbf{L}^{\mathrm{T}}[k,i] \tag{26}$$

Let j in $\mathbf{p}(v) = j$ be such that v is a leaf. Then $\mathbf{D}(j) = \{\emptyset\}$. For $1 \leq i < j$ let u be such that $\mathbf{p}(u) = i$. One has then that $u \notin \mathbf{F}(v)$ because otherwise $\mathbf{D}(j) \neq \{\emptyset\}$. Likewise $u \notin \mathbf{d}[v]$ since $\mathbf{p}(u) < \mathbf{p}(v)$, and this would violate the precedence induced by *Lemma 2*. Therefore $u \notin \mathbf{c}[v]$ and with the definition of $\overline{\mathbf{M}}[j,i]$ as given in (18) along with the result of *Lemma 2* one is left with $\mathbf{L}[j,j]$ being the solution of $\mathbf{L}[j,j]\mathbf{L}^{\mathrm{T}}[j,j] = \overline{\mathbf{M}}[j,j]$. Furthermore, since $\mathbf{L}[j,i] = \mathbf{0}$ for $1 \leq i < j$, $\mathbf{L}[k,j]$ is the solution of $\mathbf{L}[j,j]\mathbf{L}^{\mathrm{T}}[k,j] = \overline{\mathbf{M}}[k,j]$, $j < k \leq \mathrm{N}$. Thus for the case of j corresponding to a leaf of the tree one can compute $\mathbf{L}[k,j]$ for $k \in \mathbf{U}(j)$.

During the last step of this proof let j be such that $\mathbf{D}(j) \neq \{\emptyset\}$, i.e. v in $\mathbf{p}(v) = j$ is not a leaf of the tree. We assume that for each $i \in \mathbf{D}(j)$, $\mathbf{L}[l,i]$ are available for $l \in \mathbf{U}(i)$. We show that $\mathbf{L}[k,j]$ can be computed for $k \in \mathbf{U}(j)$.

With i being the index in the RHS summation in (25), let u be such that $\mathbf{p}(u) = i$. There are two alternatives: $u \notin \mathbf{F}(v)$ or $u \in \mathbf{F}(v)$. In the first case $u \notin \mathbf{d}[v]$ because otherwise the precedence induced by *Lemma 2* is violated. Then $u \notin \mathbf{c}[v]$ and again with the definition of $\overline{\mathbf{M}}[j,i]$ as given in (18) along with the result of *Lemma 2* one has that $\mathbf{L}[j,i] = \mathbf{0}$. On the other hand if $u \in \mathbf{F}(v)$ then $j \in \mathbf{U}(i)$ and therefore $\mathbf{L}[j,i]$ is known. Consequently, one can evaluate the summation in the RHS of (25) and obtain the value $\mathbf{L}[j,j]$.

Finally, to compute $\mathbf{L}[k,j]$ for $k \in \mathbf{U}(j)$ one needs to evaluate the summation in the RHS of (26). It was shown above that $\mathbf{L}[j,i]$ is either identically zero (and in this case $\mathbf{L}[k,i]$ needs not be evaluated) or known (when $j \in \mathbf{U}(i)$). In this latter situation one needs the value of $\mathbf{L}[k,i]$. If $j \in \mathbf{U}(i)$, since $k \in \mathbf{U}(j)$ we have that $k \in \mathbf{U}(i)$ as well, and consequently $\mathbf{L}[k,i]$ is known and the summation can be evaluated.□

When the Cholesky factorization progresses in the sequence described by (25) and (26) the process moves columnwise. Each column is filled starting from the diagonal element and proceeding down to the last row of the matrix. *Lemma 4* states that based on a certain amount of information, some of the entries of column j, namely $\mathbf{L}[k,j]$ with $k \in \mathbf{U}(j)$ can be computed. It is easy to show that all the other entries are identically zero. For when $k \notin \mathbf{U}(j)$ and $j < k \leq \mathrm{N}$, an argument following the proof above based on (18) and *Lemma 2* shows that $\mathbf{L}[k,j] = \mathbf{0}$.

The results in this section give a practical way in which CIM can be factorized: column j can be computed once the columns $i = \mathbf{p}(u)$ corresponding to all $u \in \mathbf{F}(v)$ have been computed. In other words, the factorization progresses independently upwards

from the tree–end bodies towards the root of the tree. We conclude these remarks in the following corollary based on the result of *Lemma 4*.

Corollary 3. Let u and v be two bodies in the spanning tree associated with a mechanical system and $u \notin \mathbf{F}(v)$ and $v \notin \mathbf{F}(u)$. Then the columns $k = \mathbf{p}(w)$ for $w \in \mathbf{F}(u)$ and $l = \mathbf{p}(t)$ for $t \in \mathbf{F}(v)$ of the matrix $\mathbf{L}$ in the Cholesky factorization of CIM can be computed in parallel.

Note that there is another stage in the algorithm presented in Section 4 that can be easily parallelized; any forward or backward substitution involving the matrix $\mathbf{L}$ can be parallelized. The result is similar to the one in *Lemma 4* and will be skipped here.

6 Numerical Experiments

In this section four different methods for solving the augmented system will be compared. The comparison is made in terms of CPU time, and for gaussian elimination and the proposed algorithm in terms of number of operations as well.

The first method analyzed is based on gaussian elimination and it is denoted by *Gauss*. The method *Symmetric* is based on a $\mathbf{PAP}^{\mathrm{T}} = \mathbf{LDL}^{\mathrm{T}}$ decomposition of the symmetric augmented matrix $\mathbf{A}$, where $\mathbf{D}$ is a diagonal matrix with blocks of dimension 1×1 or 2×2. The method *Harwell* solves the augmented system by using linear algebra subroutines from the Harwell library. The last method is the one described in Section 4 of this paper.

The numerical experiments were performed on the 14–body model of the Army's High Mobility Multipurpose Wheeled Vehicle (HMMWV) [8]. Figure 3 shows the graph associated with the mechanical system. R stands for revolute joint, T for translational joint, S for spherical joint, and D for distance constraint.

Body:

1	Chassis	2	Right front upper control arm
3	Right front wheel spindle	4	Left front upper control arm
5	Left front wheel spindle	6	Right rear upper control arm
7	Right rear wheel spindle	8	Left rear upper control arm
9	Left rear wheel spindle	10	Rack
11	Right front lower control arm	12	Left front lower control arm
13	Right rear lower control arm	14	Left rear lower control arm

For this problem the coefficient matrix in (17) has dimension 43. 16 constraints account for the cut joints; the vector of generalized coordinates has dimension 27. The vehicle model has 11 degrees of freedom.

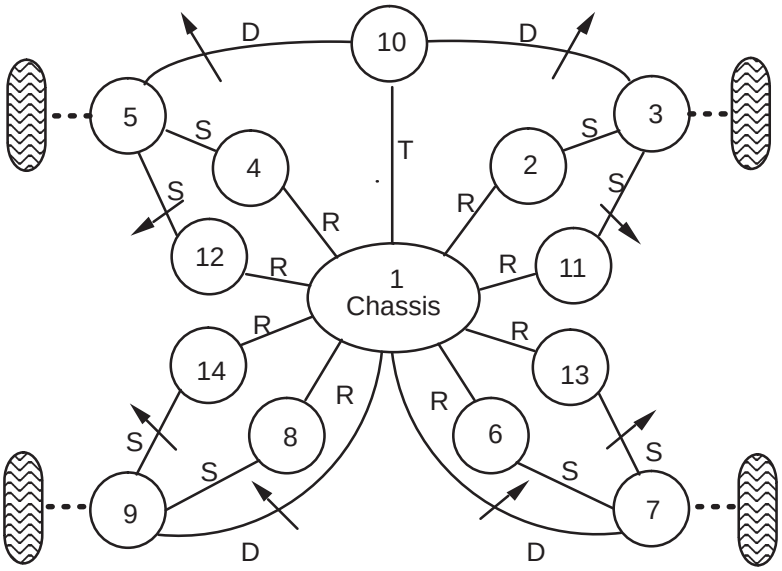


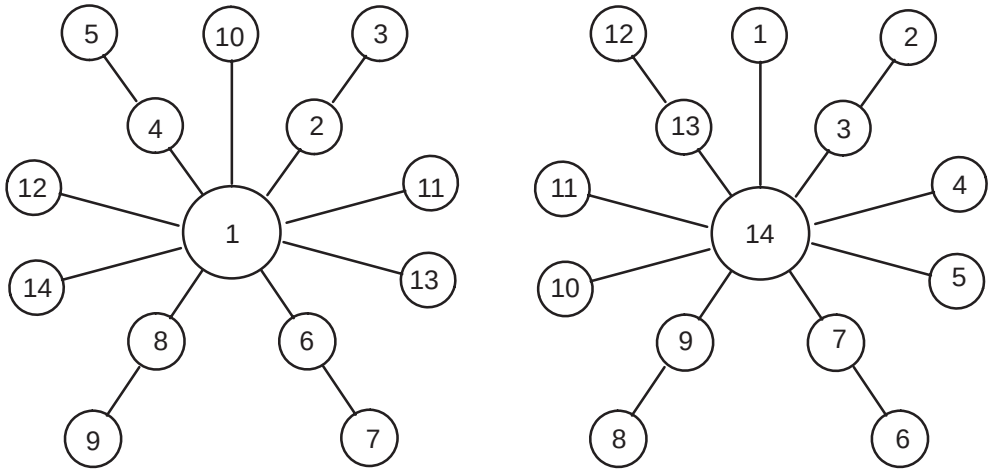
Figure 3 Topological Graph for HMMWV 14–Body Model

The constraints marked with an arrow were cut to obtain in Figure 4 (a), the spanning tree associated with the mechanism.

In [8], the strategy for solving the system (17) is based on gaussian elimination. Table 1 shows the number of operations for the gaussian elimination. **Fact** stands for factorization, **FS** for forward substitution, **BS** for back substitution. The number of additions **A**, multiplications **M**, divisions **D** and square roots **SQ** was counted at each stage of *Gauss*.

Gauss	**Fact**	**FS**	**BS**	**Total**
A	25585	903	903	27391
M	25585	903	903	27391
D	903	0	43	946
SQ	0	0	0	0

Table 1 Operation count for gaussian elimination



(a) Initial Numbering (b) Renumbering used by the algorithm

Figure 4 Spanning Tree for HMMWV 14–Body Model

Table 2 provides the operation counts for algorithm *Alg–S*. This implementation of the proposed algorithm uses topology information to take advantage of sparsity. In order to preserve the sparsity pattern of CIM, the joints of the system were renumbered as in Figure 4 (b).

Alg–S	**1**	**2**	**3.1**	**3.2**	**3.3**	**3.4**	**3.5**	**4**	**Total**
A	975	165	708	172	834	680	145	192	3871
M	975	165	804	172	1135	680	162	165	4258
D	165	27	174	0	0	120	0	27	513
SQ	27	0	0	0	0	16	0	0	43

Table 2 Operation count for *Alg–S*

The number of additions, multiplications and divisions for *Alg-S* is clearly smaller than for *Gauss*. As implemented for this test problem, it required the calculation of 43 square roots when performing the two Cholesky factorizations, while *Gauss* required none. Square root calculations can be eliminated using an $\mathbf{LDL}^T$ approach; this alternatives remains to be investigated.

One advantage of *Alg-S* over the gaussian elimination family of solvers is that no pivoting is involved. Gaussian elimination needs pivoting in order to ensure numerical stability. *Alg-S* gets around this by employing Cholesky factorizations twice, which is known to be numerically stable [3]. A disadvantage of the new algorithm is the data

accessing pattern and number of vector touches (sparsity–related overhead). While for gaussian elimination the fashion in which data is manipulated is intuitive, it is not straightforward for the proposed algorithm. It is difficult to asses to what extent this will affect the overall performance. It depends on the particular mechanical system being modeled, and the way the algorithm is coded. An ideal implementation of *Alg–S* would be a no–loop, hardcoded, problem dependent version, generated during the preprocessing stage of the simulation. This would require a program that based on topology information codes the algorithm of Section 4.3 for each mechanical system.

We made an attempt to evaluate the sparsity–related overhead for this test problem. For this, the same steps of the algorithm were to be followed but all the sparsity–related book–keeping present in *Alg–S* was eliminated by using dense–matrix operations. BLAS 2 and 3 operations along with dense Cholesky factorization are at the core of *Alg–D[ense]*. One does not need to renumber the joints of the mechanical system nor keep track of the topology information. The operation counts for this implementation of the algorithm is provided in Table 3.

Alg–D	**1**	**2**	**3.1**	**3.2**	**3.3**	**3.4**	**3.5**	**4**	**Total**
A	3276	351	5616	432	3536	2280	405	378	15914
M	3276	351	5616	432	3672	2280	432	351	16410
D	351	27	432	0	0	152	0	27	989
SQ	27	0	0	0	0	16	0	0	43

Table 3 Operation counts for *Alg–D*

Table 4 list the CPU times in microseconds for the methods discussed above. The times correspond to one solving of the augmented system (17). For comparison reasons we also include in this table the timing results obtained when using the Harwell solver and the algorithm *Symmetric*.

Gauss	*Symmetric*	*Harwell*	*Alg–S*	*Alg–D*
2336	2022	3532	1201	1179

Table 4 CPU times for different methods

It turns out that *Alg–D* performs better than all the others. The dimension of the test problem is too small for any benefit to show up as a result of taking into account the

topology information in *Alg-S*. The sparsity–related overhead for this test problem hurts the overall efficiency of the algorithm. We emphasize that the numerical implementation behind *Alg-S* could be further improved.

In *Alg-S* and *Alg-D* the positive definite systems were solved using the driver **dposv**. In *Gauss*, *Symmetric* and *Harwell* the routines used were **dgesv**, **dsysv**, and the triple **ma28ad**/**ma28bd**/**ma28cd** respectively. With the exception of the triple **ma28..** taken from an older HARWELL public domain library, the other routines were available in LAPACK[7]. The newer and faster routine **ma47** designed in HARWELL[5] to solve large sparse symmetric systems was unavailable to the authors at the time of this study. It certainly arises as a possible alternative for larger problems.

For the sake of completeness, Table 5 presents a detailed profile of the algorithms *Alg-S* and *Alg-D*, listing CPU times in microseconds for each step of the implementations.

	1	2	3.1	3.2	3.3	3.4	3.5	4	Total
Alg–S	300	59	246	50	276	159	57	54	1201
Alg–D	289	31	347	27	276	147	31	31	1179

Table 5

The results above show that for this case, the advantage of using sparsity fades when dealing with forward/backward eliminations with a sparse coefficient matrix. It is only when sparsity information is used for the coefficient and right–hand matrices in step 3.1 (computing the **T** matrix) that *Alg-S* gains an edge over *Alg-D*. As mentioned before this suggests that *Alg-S* can be further improved. Swapping between the two algorithms is not likely to result in any benefit because of the different storage techniques.

7 Conclusions

Current vehicle models contain generalized coordinates numbering in the tens or at most hundreds. For this dimension range we developed an algorithm for solving the system of linear equations providing for the Lagrange multipliers and the set of accelerations. The algorithm was implemented in two versions: *Alg-D*, the dense matrix implementation, and *Alg-S* the sparse version. The first one turned out to be slightly more efficient for a 14–body model of the Army's High Mobility Multipurpose Wheeled Vehicle (HMMWV).

For this test problem the two methods performed better than the gaussian elimination in a ratio of approx. 1:2. Gaussian elimination is the method currently implemented in NADSdyna [8].

For larger models it is expected that the sparsity will become an important factor. An improved version of *Alg–S*, or the new routine **ma47** from Harwell used in *Alg–D* is expected to increase the speed–up. Taking into account the data in Table 1 and Table 2, we expect the 1:2 ratio above could be further improved if a no–loop, hardcoded version of *Alg–S* that takes full advantage of sparsity is implemented. This would be done by a preprocessing tool that based on topology information codes the proposed algorithm.

Finally, the parallelism inherent in the joint formulation might be an additional speed–up factor for large problems. Additional numerical experiments with larger models should be conducted to asses the dimension range in which the parallelism results in better efficiency.

Acknowledgment

The authors would like to thank Jim Cramer for diligently reading the manuscript of this paper and for the comments he made.

References

[1] Brenan, K.E., Campbell S.L., and Petzold L.R., 1989, Numerical Solution of Initial–Value Problems in Differential–Algebraic Equations, Elsevier

[2] Eich, E., Fuhrer, C., Leimkuhler, B., and Reich, S., 1990, "Stabilization and Projection Methods for Multibody Dynamics," Research Report A281, Helsinki University of Technology, Institute of Mathematics, Otakaari 1, SF–02150 Espoo, Finland

[3] Golub, G.H. and Van Loan, C. F., 1989, Matrix Computation, John Hopkins University Press, Baltimore

[4] Hairer, E., Wanner, G., 1996, Solving Ordinary Differential Equations II: Stiff and Differential–Algebraic Problems, Springer–Verlag, Berlin

[5] *Harwell Subroutine Library - Specifications*, 1995, AEA Technology, Harwell Laboratory, Oxfordshire, England

[6] Haug, E.J., 1989, Computer–Aided Kinematics and Dynamics of Mechanical Systems, Volume I: Basic Methods, Allyn and Bacon, Needham, Massachusets

[7] LAPACK Users' Guide, 1992, SIAM, Philadelphia

[8] NADS Vehicle Dynamics Software, Vol.2, Release 4, August 1995, CCAD, The University of Iowa, Iowa City, Iowa

[9] Serban, R., Negrut, D., Potra, F. A., and Haug, E.J., 1997, "A Topology Based Approach for Exploiting Sparsity in Multibody Dynamics in Cartesian Formulation", to appear in Mech. of Struc. and Mach., Vol. 25, No. 3

[10] Potra, F. A., 1993, "Numerical Methods for Differential–Algebraic Equation with Application to Real–Time Simulation of Mechanical Systems", Zeitschrift fur Angewandte Mathematik und Mechanik (ZAMM), 74, 3 (1994), pp. 177–187

[11] Tsai, F. F., 1983, "Automated Methods for High Speed Simulation of Multibody Dynamic System", Ph.D. Thesis, The University of Iowa

[12] Wehage, R.A. and Haug E.J., 1982, "Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems," *ASME J. of Mechanical Design*, Vol. 104, No. 1, pp 247