Struct Multidisc Optim
DOI 10.1007/s00158-007-0142-2

RESEARCH PAPER

A divide-and-conquer direct differentiation approach for multibody system sensitivity analysis

Rudranarayan M. Mukherjee · Kishor D. Bhalerao · Kurt S. Anderson

Received: 29 October 2006 / Revised: 26 February 2007 / Accepted: 25 March 2007

Abstract In the design and analysis of multibody dynamics systems, sensitivity analysis is a critical tool for good design decisions. Unless efficient algorithms are used, sensitivity analysis can be computationally expensive, and hence, limited in its efficacy. Traditional direct differentiation methods can be computationally expensive with complexity as large as $O(n^4 + n^2m^2 + nm^3)$, where n is the number of generalized coordinates in the system and m is the number of algebraic constraints. In this paper, a direct differentiation divide-and-conquer approach is presented for efficient sensitivity analysis of multibody systems with general topologies. This approach uses a binary tree structure to traverse the topology of the system and recursively generate the sensitivity data in linear and logarithmic complexities for serial and parallel implementations, respectively. This method works concurrently with the forward dynamics problem, and hence, requires minimal data storage. The differentiation required in this algorithm is minimum as compared to traditional methods, and is generated locally on each body as a preprocessing step. The method provides sensitivity values accurately up to integration tolerance and is insensitive to perturbations in design parameter values. This approach is a good alternative to existing methodologies, as it is fairly simple to implement for general topologies and is computationally efficient.

Keywords Multibody dynamics systems · Sensitivity analysis · Direct differentiation · Divide- and-conquer formulation

R. M. Mukherjee (✉) · K. D. Bhalerao · K. S. Anderson
Department of Mechanical, Nuclear and Aerospace Engineering, Rensselaer Polytechnic Institute, 110-8th Street, Troy, NY 12180, USA
e-mail: Rudranarayan.M.Mukherje@jpl.nasa.gov

K. D. Bhalerao
e-mail: bhalek@rpi.edu

K. S. Anderson
e-mail: anderk5@rpi.edu

1 Introduction

Design of multibody dynamics systems is an iterative process and is computationally taxing. Sensitivity analysis is a useful tool that significantly reduces the iterative nature of the design process by helping make intelligent guesses for the design parameters. However, determining sensitivity terms is an involved process given the complexity of governing equations of motions for the simplest of multibody systems. Consequently, sensitivity analysis continues to be an important area of research.

Finite difference approximation is perhaps the most popular and straightforward numerical method for generating sensitivity information. Although successful in many applications, this method suffers from a number of difficulties. These include the sensitivity to parameter perturbation size and system stiffness as discussed in Bestle and Eberhard (1992), Bischof (1996), and Anderson and Hsu (2001). Analytical methods, such as the adjoint variable method, the direct differentiation method, and automatic differentiation (Bischof 1996), do not suffer from the problems faced by the numerical methods. These analytical methods have been used in

Table 1 The nomenclature

Symbol	Meaning
a_i^k	Translational acceleration of handle i on body k
f_i^k	Force on body k through handle i
g	Number of design variables
m	Number of constraint equations for the system
m^k	Mass of body k
n	Number of degrees of freedom in the system
nb	Number of bodies in the system
p	Design parameter
q	Generalized coordinate associated with a joint
$\boldsymbol{q}$	Global column matrix of generalized coordinates for all joints in the system
$r^{k_i k_j}$	Position vectors from point k_i to k_j
$r^{k_0 k_i}\times$	3×3 skew symmetric matrix for cross product of the position vector $r^{k_0 k_i}$
u	Generalized speed at a joint
$\boldsymbol{u}$	Global column matrix of generalized speeds for all joints in the system
$\dot{u}$	Time derivative of generalized speed at a joint
$\dot{\boldsymbol{u}}$	Global column matrix of the derivative of generalized speeds for all joints in the system
A_i^k	Spatial acceleration of body k described at handle i
$\dot{C}$	Time derivative of any quantity C
C^{T}	Transpose of a matrix C
C^{-1}	Inverse of a matrix C
$\mathcal{C}$	Transformation matrix
D^{J^k}	Orthogonal complement matrix of P^{J^k}
F_0^k	Spatial force on body k about its center of mass
F_a^k	Spatial state-dependent forces on body k
F_{ic}^k	Spatial constraint force acting on body k at handle i
$\mathbf{F}_i^k$	Ordered list of measure numbers of the spatial force on body k at handle i
H_i^k	Handle i on body k
I_0^k	Inertia tensor of body k about its center of mass
J	Objective function for sensitivity analysis
J^i	Kinematic joint connecting bodies i and $i-1$
$\boldsymbol{\mathcal{K}}$	Column matrix of state-dependent forces acting on the whole system
L^k	Characteristic dimension of body k
M_0^i	Spatial mass matrix of body k about its center of mass
$\boldsymbol{\mathcal{M}}$	Global mass matrix for the whole system
P^{J^k}	Matrix of free modes of motion at joint J^k
$S^{k_i k_j}$	Spatial matrix for shifting a quantity described at point k_i to its equivalent at point k_j
$\underline{U}$	Identity matrix
$\underline{0}$	Zero matrix
α^k	Angular acceleration of body k
χ	Useful intermediate quantity
ν	Number of degrees of freedom allowed by joint J^k
Π	Useful intermediate quantity
ϕ_{ij}^k	Coupling terms for inertia and state-dependent quantities in sensitivity equation for body k
$\phi_{ij}^{x:z}$	Corresponding terms in sensitivity equations of the assembly of bodies x to z
$\Upsilon_{ij}^{x:z}$	Inertia coupling terms of assembly of bodies x to z
τ_i^k	Torque on body k about handle i
ζ_{ij}^k	Inertia coupling terms of body k
dZ/dp_j	Derivative of any quantity Z with respect to design parameter p_j
$\partial Z/\partial z$	Partial derivative of any quantity Z with respect to another quantity z

sensitivity analysis of multibody dynamics systems, but they too have been known to have some drawbacks.

The adjoint variable method has been presented in Haug and Ehle (1982), Haung et al (1984), Bestle and Seybold (1992), Bestle and Eberhard (1992), and Eberhard (1996), among others. In these methods, a set of adjoint equations is introduced to represent the variations of the state. The advantage of using these methods is that explicit calculation of sensitivity terms is avoided. With this method, for a system modeled as having n generalized coordinates, m algebraic constraints, and g design variables, a total of $2(n+m)+g$ differential equations must be integrated, as discussed in Bestle and Eberhard (1992). First, a record of the complete system state is produced for the time interval of interest by the forward integration of the $n+m$ equations of motion. Using this state information, the sensitivities are then determined from the set of $n+m+g$ adjoint equations, which are integrated backward in time over the same time interval. The use of this method is desirable when the number of design variables is large as compared to the objective functions, particularly when the forward dynamics analysis is being performed in a more traditional [not $O(n)$] manner. If one considers the total cost required of getting these sensitivity terms by the adjoint method, the best one can hope for is $O((g+1)(n+m))$ overall. This is due to the cost associated with each function evaluation in the forward integration of the equations of motion [at best, this is $O(n+m)$ per integration step] and the subsequent cost of each function evaluation in the backward integration of the system of $n+m+g$ adjoint equations. Additionally, the implementation of these methods is complex, and a large amount of data has to be stored for the forward problem. The large number of I/O operations slows down the speed significantly as documented in Chang and Nikravesh (1985) and Pagalday et al (1995). Another source of error is the backward temporal integration necessary for the calculation of adjoint variables. The adaptive nature of the integrators calls for interpolation to obtain all values at matching time steps. Besides this, as indicated in Bestle and Seybold (1992) and Etman (1997), numerical stability for the adjoint variable methods remains an open question.

The direct differentiation methods were proposed in Chang and Nikravesh (1985), Tak (1990), Dias and Pereira (1997), Serban and Haug (1998), and Jain and Rodrigues (2000), among others. These methods are conceptually the easiest to understand. They systematically apply the chain rule of differentiation to obtain analytical expressions for sensitivity terms. The number of integrated equations is roughly equal to the number of state variables times the number of design variables. The major advantages of these methods are higher numerical stability and relative insensitivity of solution accuracy to parameter perturbations. Implementation approaches for direct differentiation vary with different formulations of the equations of motion. Newton–Euler is the most frequently used formulation as found in Chang and Nikravesh (1985) and Serban and Haug (1998). Although the formulation of the sensitivity equations is straightforward, the result is a set of computationally demanding *differential algebraic equations* (DAE). Consequently, the cost of computation of the sensitivity terms depends directly upon the algorithm used for forming and solving the equations of motion.

The analytical methods described above are all capable of calculating the sensitivity derivatives. However, the costs involved in each method can vary greatly. For example, in our system with g design variables, n generalized coordinates, and m independent algebraic constraint equations, the adjoint variable method produces a smaller system of $(n+m+g)$ DAE requiring $O[(n+nm^2+m^3)+(n+m)g]$ operations overall [including required forward integration of the equations of motion using a traditional $O(n)$ scheme], whereas the direct differentiation method involves $(n+m)(g+1)$ DAE.

In this paper, a divide-and-conquer direct differentiation approach (DCA) is presented for efficient sensitivity analysis of multibody systems with general topologies. This method is an efficient direct differentiation scheme. Consequently, it does not require excessive data storage as compared with the adjoint variable method. This is because the sensitivity analysis is carried out concurrently with the solution of the forward dynamics problem. Similarly, there is no need for backward differentiation, and errors due to integration interpolation are eliminated. Further, as the sensitivity information is generated analytically, the method is insensitive to numerical issues of design parameter perturbations. The derivatives required for this approach are limited in number, are mostly temporally invariant, and hence, only need to be evaluated once for a simulation. This approach uses a binary tree structure to traverse the topology of the system and generate the sensitivity data in linear and logarithmic complexities for serial and parallel implementations, respectively. The sensitivity data are accurate to integration error, making this approach a good alternative to existing methodologies, as it is fairly simple to implement for general topologies and is computationally efficient. The methodology presented here is a derived work from (1) divide-and-conquer algorithm (Featherstone 1999a) and (2) the orthogo-

nal complement-based divide-and-conquer algorithm (Mukherjee and Anderson 2007).

2 Sensitivity problem formulations

The objective of sensitivity analysis is to measure the sensitivity of a particular objective function to the change in certain design or control variable value(s). This information is useful in identifying the robustness of a design and tolerances on system performance with respect to variations in design variable values. For multibody dynamics systems, the objective function J is often an explicit function of design variable(s) p as well as state variables q and u, the system generalized coordinates and generalized speed, respectively. The state variables may also be explicit functions of the design variable(s). Further, the system state variable values are themselves dependent on the design variable values currently under consideration. Thus, the sensitivity equation of the objective function J with respect to design variable p can be written as

$$\nabla J = \frac{\partial J}{\partial p} + \sum_{r=1}^{n} \left(\frac{\partial J}{\partial q_r} \frac{dq_r}{dp_j} + \frac{\partial J}{\partial u_r} \frac{du_r}{dp_j} + \frac{\partial J}{\partial \dot{u}_r} \frac{d\dot{u}_r}{dp_j} \right) \tag{1}$$

It is clear from the above equation that the sensitivity analysis requires the generation of the dependencies of the state and state derivatives on the design variable(s). Generating this dependency information can be computationally expensive because the state and state derivative variables are highly coupled for an articulated multibody system. This expense is alleviated somewhat, as there exist the following relations

$$\left.\frac{d\boldsymbol{q}_r}{d\boldsymbol{p}_j}\right|_{t=\tau+dt} = \int_{t=\tau}^{t=\tau+dt} \left.\frac{d\dot{\boldsymbol{q}}_r}{d\boldsymbol{p}_j}\right|_{t=\tau} dt + \left.\frac{d\boldsymbol{q}_r}{d\boldsymbol{p}_j}\right|_{t=\tau} \tag{2}$$

$$\left.\frac{d\boldsymbol{u}_r}{d\boldsymbol{p}_j}\right|_{t=\tau+dt} = \int_{t=\tau}^{t=\tau+dt} \left.\frac{d\dot{\boldsymbol{u}}_r}{d\boldsymbol{p}_j}\right|_{t=\tau} dt + \left.\frac{d\boldsymbol{u}_r}{d\boldsymbol{p}_j}\right|_{t=\tau} \tag{3}$$

Thus, the task reduces to that of finding $d\dot{\boldsymbol{u}}_r/dp_j$ at every instant in the simulation and substituting it back into the above relations to generate the other terms. Now, in the state-space form, the equations of motion of a general multibody system reduce to

$$\boldsymbol{\mathcal{M}}_{n\times n}\dot{\boldsymbol{u}}_{n\times 1} = \boldsymbol{\mathcal{K}}_{n\times 1} \tag{4}$$

The above equation presents a coupled set of n equations where n is the number of degrees of freedom of the system, $\boldsymbol{\mathcal{M}}$ is the generalized mass matrix, $\dot{\boldsymbol{u}}$ is the column matrix of the unknown time derivatives of generalized speeds, and $\boldsymbol{\mathcal{K}}$ is the column matrix of the forces on the system including state-dependent inertia forces. Using a direct differentiation approach, the above equations can be differentiated to generate the desired $\dot{\boldsymbol{u}}$ values as

$$\frac{d\boldsymbol{\mathcal{M}}\dot{\boldsymbol{u}}}{dp_j} = \frac{d\boldsymbol{\mathcal{K}}}{dp_j} \tag{5}$$

$$\Rightarrow [\boldsymbol{\mathcal{M}}_{n\times n}] \frac{d\dot{\boldsymbol{u}}}{dp_j}_{n\times 1} = \frac{\partial \boldsymbol{\mathcal{K}}}{\partial p_j} + \frac{\partial \boldsymbol{\mathcal{K}}}{\partial \boldsymbol{q}} \frac{d\boldsymbol{q}}{dp_j} + \frac{\partial \boldsymbol{\mathcal{K}}}{\partial \boldsymbol{u}} \frac{d\boldsymbol{u}}{dp_j} - \left[\frac{\partial \boldsymbol{\mathcal{M}}}{\partial p_j} + \frac{\partial \boldsymbol{\mathcal{M}}}{\partial \boldsymbol{q}} \frac{d\boldsymbol{q}}{dp_j} + \frac{\partial \boldsymbol{\mathcal{M}}}{\partial \boldsymbol{u}} \frac{d\boldsymbol{u}}{dp_j} \right] \dot{\boldsymbol{u}} \tag{6}$$

The direct method, applied in this manner, incurs large computational expenses in calculating the differentiations, which amount to $O(n^2) - O(n^3)$ complexity. Also, direct decomposition and solution of the above set of n coupled equations amount to $O(n^3)$ complexity. For even small values of n, these costs can become prohibitive. Unless some efficient method is introduced to reduce the cost, generating sensitivity information for multibody systems can quickly become the bottleneck in the design analysis process.

The methodology outlined in this paper reduces the total number of required differentiations and reduces the cost of solving the coupled equations from $O(n^3)$ to $O(n)$ in serial and $O(log(n))$ in parallel implementations. In the next section, the analytical preliminaries required for the method are discussed. After that, a brief overview of the divide-and-conquer scheme is presented. The method for sensitivity analysis is then discussed. Finally, results from numerical simulations using the method are presented.

3 Analytical preliminaries

The position vector $r^{k_0 k_1}$ between any two points (0 and 1) on a representative body k is a function of dimensions L_i and the transformation matrices $\mathcal{C}_i$ between different basis vectors used in the definition of $r^{k_0 k_1}$. The transformation matrices $\mathcal{C}_i$ are functions of the generalized coordinates q_i ($i = 1, 2, ..., n$). Thus, the position vector is an explicit function of dimensions L_i and generalized coordinates q_i. Similarly, the angular and translational velocities obtained from taking time derivatives of the position vector are functions of the dimensions L_i, generalized coordinates q_i, as well as the generalized speeds u_i. Further, the angular and translational accelerations are also kinematic functions of the dimensions, dimensions L_i, generalized coordinates and speeds q_i and u_i, as well as the time derivative of the generalized speeds $\dot{u}_i$. Additionally, the mass of a

body is a function of the density and dimensions of the body. The rotational inertia of the body depends on the mass distribution as well as the position vectors, making it a function of body geometry. These relations are analytical in nature, and thus, the derivatives of the kinematic entities with respect to the design parameter p_j can be obtained analytically and exactly. These lowest level (local) derivatives need only be calculated once as a preprocessing step to the simulation and are temporally invariant. These sets of time-invariant local derivatives, which may often be generated analytically, are going to be referred to as *derivative primitives* and will be treated as known quantities.

For example, consider a multibody system made of nb bodies with the characteristic length of each body expressed as L_i $(i = 1 : nb)$. The derivative primitives of the body-based mass matrix with respect to a specific length L_j chosen as a design parameter are as below.

$$\frac{dM_0^i}{dL_j} = \underline{0} \quad \ldots i \neq j \tag{7}$$

$$\frac{dM_0^i}{dL_j} = \begin{bmatrix} \frac{dI_0^i}{dL_j} & \underline{0} \\ \underline{0} & \frac{dm^i}{dL_j} \end{bmatrix} \quad \ldots i = j \tag{8}$$

In the above equations, the derivatives of the mass and inertia of a body are analytical functions of the length, and hence, can be easily calculated. Further, the derivative primitive dM_0^i/dL_j is local to the body, and there is no coupling with the other bodies in the system.

Although the concept of derivative primitives is explained here through an analytical expression, they need not always be generated analytically. In many cases, multibody systems consist of components with non-standard geometries, and the mass and inertia properties of such components are developed experimentally or from solid modeling (CAD) packages. In such cases, the mass and inertia properties, or their dependence on a design parameter, cannot be expressed analytically. However, in such cases, it would still be possible to generate the derivative primitives through other means. In cases where the components are designed using software packages, the derivative primitives may be generated numerically through a simple finite difference method. Alternately, the derivative primitives may also be developed empirically.

The generation of the derivative primitives is a preprocessing step to the use of this algorithm and needs to be developed only once for the whole simulation. Thus, for the purposes of this algorithm, no distinction is made whether the derivative primitives are developed analytically, numerically, empirically, or using any other methods. The choice of the design parameters and the calculation of the derivative primitives have been previously studied in Serban and Haug (1998) and Hsu and Anderson (2002), and this topic is not pursued further here. Without loss of generality, it is therefore assumed that, for a system of interest, the desired derivative primitives can be independently calculated locally on each body, are temporally invariant, are calculated as a preprocessing step to the simulation, and are henceforth treated as known quantities.

3.1 Brief overview of DCA

In the analytical treatment presented here, direction cosine matrices and transformations between different bases are not shown explicitly. However, appropriate basis transformations have to be taken into account for proper implementation of this algorithm. Additionally, this algorithm uses a redundant set of mixed coordinates, viz. Cartesian coordinates and relative coordinates, throughout the derivation. The set of mixed coordinates offers certain advantages within this formulation and has been used in Kim and VanderPloeg (1986) and Nikravesh (1990) for rigid body dynamics.

Consider two representative bodies, body k and body $k+1$, of any articulated system as shown in Fig. 1. The joint between body k and body $k+1$ is referred to as J^k. Henceforth, any point on the body through which the interactions of the body with the environment can be modeled would be referred to as a handle. The handles on a body can correspond to a joint location, a center of mass, or any desired reference point. The two handles on body k corresponding to the locations H_1^k and H_2^k are associated with joints J^{k-1} and J^k, respectively. Similarly, the two handles on body $k+1$ corresponding to the locations H_1^{k+1} and H_2^{k+1} are associated with joints J^k and J^{k+1}. Further, the acceleration of the handles H_1^k and H_2^k and the constraint forces acting on these points on body k will be denoted by the superscript k and subscripts 1 and 2, respectively.

In the most general form, the equations of motion for body k using a spatial Newton–Euler formulation can be expressed as

$$\mathcal{M}_0^k \mathcal{A}_0^k = \mathcal{F}_0^k \tag{9}$$

where

$$\mathcal{M}_0^k = \begin{bmatrix} I_0^k & \underline{0} \\ \underline{0} & m^k \end{bmatrix}, \quad \mathcal{A}_0^k = \begin{bmatrix} \alpha_0^k \\ a_0^k \end{bmatrix}, \quad \mathcal{F}_0^k = \begin{bmatrix} \tau_0^k \\ f_0^k \end{bmatrix} \tag{10}$$

In the above equations, the subscript 0 denotes the center of mass of the body, while superscript k indicates that the quantity is associated with representative body k. In (9) and (10), M_0^k is the 6×6 spatial inertia matrix of k defined relative to a reference frame located at the

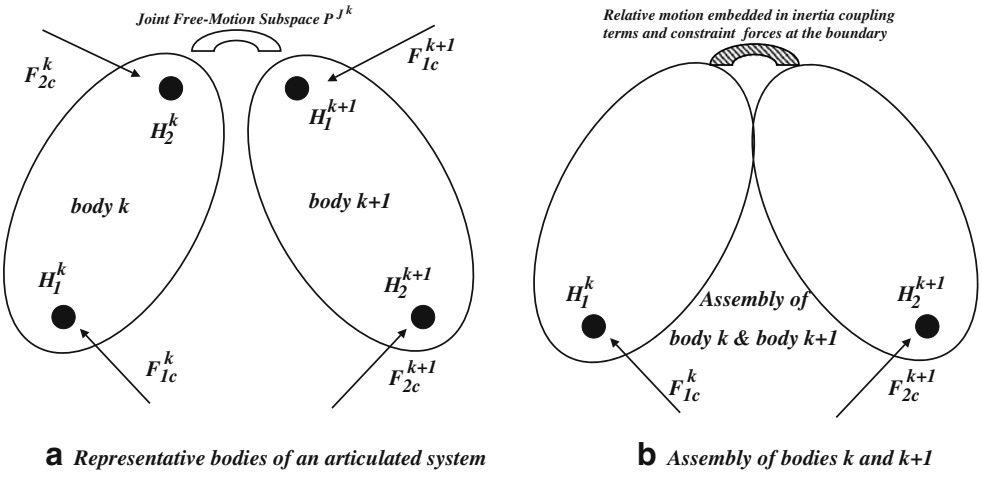

Fig. 1 Representative bodies of a multibody system

point 0. It is composed of the 3×3 inertia matrix of body k about its mass center, the 3×3 diagonal mass matrix m^k in which each diagonal element of the matrix is equal to the mass of body k, and 3×3 zero matrices $\underline{0}$. The quantity A_0^k is the 6×1 spatial acceleration associated with the center of mass of body k in the inertial reference frame N. This matrix is composed of the 3×1 angular acceleration vector α^k of body k in N and the translational acceleration a_0^k of the body k mass center in N. Similarly, F_0^k is the resultant spatial loads matrix associated with all loads (applied and constraint) acting on body k. This matrix is composed of the resultant torque τ_0^k acting on body k with the resultant force f_0^k acting on k with a line of action through the center of mass of k. The total spatial load acting at the center of mass consists of state-dependent active loads such as actuators, loads from potential fields, and constraint forces arising from the joints. These constraint forces depend on the dynamics of the entire system, and hence, introduce coupling between the equations of motion of all bodies in the system. The active forces, on the other hand, are uncoupled and can be calculated independently on each body based on the state of the system. Thus, (9) can be written with $\mathcal{F}_0^k$ expressed explicitly in terms of the known active loads contained in $\mathcal{F}_a^k$ and the unknown constraint loads arising from joints J^k and J^{k+1} as $\mathcal{F}_{1c}^k$ and $\mathcal{F}_{2c}^k$ as

$$\mathcal{M}_0^k \mathcal{A}_0^k = \mathcal{S}^{k_0 k_1} \mathcal{F}_{1c}^k + \mathcal{S}^{k_0 k_2} \mathcal{F}_{2c}^k + \mathcal{F}_a^k \tag{11}$$

Where

$$\mathcal{S}^{k_0 k_1} = \begin{bmatrix} \underline{U} & r_\times^{k_0 k_1} \\ \underline{0} & \underline{U} \end{bmatrix}_{6\times 6} \tag{12}$$

And

$$\mathcal{S}^{k_0 k_2} = \begin{bmatrix} \underline{U} & r_\times^{k_0 k_2} \\ \underline{0} & \underline{U} \end{bmatrix}_{6\times 6} \tag{13}$$

In the following description, a single body is assumed to have only two handles. However, this approach can be easily extended to bodies with multiple handles. The spatial equations of motion for body k can thus be written as

$$\mathcal{A}_1^k = \begin{bmatrix} \boldsymbol{\alpha}^k \\ \mathbf{a}_1^k \end{bmatrix}_{(6\times 1)} = \zeta_{11}^k F_{1c}^k + \zeta_{12}^k F_{2c}^k + \zeta_{13}^k \tag{14}$$

$$\mathcal{A}_2^k = \begin{bmatrix} \boldsymbol{\alpha}^k \\ \mathbf{a}_2^k \end{bmatrix}_{(6\times 1)} = \zeta_{21}^k F_{1c}^k + \zeta_{22}^k F_{2c}^k + \zeta_{23}^k \tag{15}$$

The above equation set is henceforth referred to as the two-handle equations of motion of representative body k. $\mathcal{A}_1^k$ and $\mathcal{A}_2^k$ are then the spatial accelerations of body k for handles H_1^k and H_2^k, respectively. The terms ζ_{ij}^k $(i, j = 1, 2)$ correspond to inertia coupling terms at the joint locations on body k. F_{1c}^k and F_{2c}^k are the unknown spatial constraint loads acting on the body k at the handles H_1^k and H_2^k, respectively, defined as

$$F_{1c}^k = \begin{bmatrix} \boldsymbol{\tau}_{1c}^k \\ \mathbf{f}_{1c}^k \end{bmatrix}_{(6\times 1)} \quad \text{and} \quad F_{2c}^k = \begin{bmatrix} \boldsymbol{\tau}_{2c}^k \\ \mathbf{f}_{2c}^k \end{bmatrix}_{(6\times 1)} \tag{16}$$

with $\boldsymbol{\tau}_{ic\,(3\times 1)}^k$ and $\mathbf{f}_{ic\,(3\times 1)}^k$ $(i = 1, 2)$, representing the constraint torques and constraint forces, respectively, being imposed at handles H_i^k. The active forces at the joint are state-dependent and are treated as known quantities. These are coupled together with the state-dependent inertia forces and expressed as ζ_{ij} $(i = 1, 2;\ \ j = 3)$. Similarly, the two-handle equations of motion for body *k+1* can be written in the form

$$\mathcal{A}_1^{k+1} = \zeta_{11}^{k+1} F_{1c}^{k+1} + \zeta_{12}^{k+1} F_{2c}^{k+1} + \zeta_{13}^{k+1} \tag{17}$$

$$\mathcal{A}_2^{k+1} = \zeta_{21}^{k+1} F_{2c}^{k+1} + \zeta_{22}^{k+1} F_{2c}^{k+1} + \zeta_{23}^{k+1} \tag{18}$$

The spatial accelerations $\mathcal{A}_2^k$ and $\mathcal{A}_1^{k+1}$ occurring at each end of joint J^k are related kinematically by

$$\mathcal{A}_1^{k+1} = \mathcal{A}_2^k + P^{J^k}\dot{u}^{J^k} + \dot{P}^{J^k}u^{J^k} \tag{19}$$

P^{J^k} is the $6 \times \nu^k$ matrix of the *free modes of motion*, permitted by the ν^k degrees of freedom joint J^k with each column of the matrix P^{J^k} being synonymous with the spatial *partial velocities*. $\dot{u}^{J^k}$ is the $\nu^k \times 1$ matrix of the time derivatives of associated generalized speeds.

In the absence of a kinematic joint, two bodies can move with respect to each other through 6 degrees of freedom. So the motion map between the bodies is a rank 6 matrix. A kinematic joint constrains the relative motion between two bodies, allowing only certain degrees of freedom while constraining out the others. Thus, the kinematic joint partitions the six-dimensional relative motion map between two bodies into free modes of motion described by the matrix P^{J^k}, which is of dimension $6 \times \nu^k$ and its orthogonal complement D^{J^k} of dimension $6 \times (6 - \nu^k)$. Columns of P^{J^k} and ν^k are the spatial partial velocities and degrees of freedom for the joint k, respectively. The joint allows relative motion in the space spanned by the columns of the P^{J^k}. The joint cannot support a constraint load in the space spanned by P^{J^k}. However, the constrained degrees of freedom are mapped by the columns of D^{J^k}, and the joint can support constraint loads in the space spanned by D^{J^k}. For example, with a spherical joint, the translational degrees of motion are constrained while the rotational degrees of freedom are maintained. Hence, the corresponding maps may be given by

$$P^{J^k} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad D^{J^k} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{20}$$

From a linear algebra point of view, the joint motion map P^{J^k} can also be interpreted as the $6 \times \nu^k$ matrix that maps the ν^k generalized speeds u at the joint into a 6×1 column matrix of spatial relative velocity across the joint.

It is apparent from their definitions that D^{J^k} and P^{J^k} for any representative kinematic joint between two connected bodies k and $k+1$ satisfy the following orthogonality relations

$$(D^{J^k})^T P^{J^k} = 0 \quad \text{and} \quad (P^{J^k})^T D^{J^k} = 0 \tag{21}$$

The DCA consists of two distinct processes, a hierarchic assembly process and a hierarchic disassembly process. In the hierarchic assembly process, using relationships in (19) and (21), the two-handle equations of motion of two successive bodies are coupled together to form the two-handle equations of motion of the resulting assembly. If we consider the assembly formed from successive bodies k and $k+1$, as shown in Fig. 1, then the assembled two-handle equations are

$$\mathcal{A}_1^k = \Upsilon_{11}^{k:k+1} F_{1c}^k + \Upsilon_{12}^{k:k+1} F_{2c}^{k+1} + \Upsilon_{13}^{k:k+1} \tag{22}$$

$$\mathcal{A}_2^{k+1} = \Upsilon_{21}^{k:k+1} F_{1c}^k + \Upsilon_{22}^{k:k+1} F_{2c}^{k+1} + \Upsilon_{23}^{k:k+1} \tag{23}$$

The two handles of the resulting assembly are H_1^k and H_2^{k+1}, and the constraint loads are those acting on the resulting assembly at those handles as indicated in Fig. 1b. The inertia coupling terms for the resulting assembly, $\Upsilon_{ij}^{k:k+1}$, are calculated using recursive formulae described in Featherstone (1999a) and Mukherjee and Anderson (2007). The assembly process starts from individual body level and moves upward in a binary tree fashion (Fig. 1) until we end up with a single all-encompassing assembly as the root node of the binary tree. The two-handle equations associated with this root node are

$$\mathcal{A}_1^1 = \Upsilon_{11}^{1:nb} F_{1c}^1 + \Upsilon_{12}^{1:nb} F_{2c}^{nb} + \Upsilon_{13}^{1:nb} \tag{24}$$

$$\mathcal{A}_2^{nb} = \Upsilon_{21}^{1:nb} F_{1c}^1 + \Upsilon_{22}^{1:nb} F_{2c}^{nb} + \Upsilon_{23}^{1:nb} \tag{25}$$

where *1:nb* refers to the entire assembled system consisting of *nb* bodies and the two handles correspond to the boundary joints of the articulated system. These two-handle equations of motion can now be solved for general topologies including free-floating chains, chains anchored at one end or at both ends using the methodologies described in Featherstone (1999a) and Mukherjee and Anderson (2007).

The hierarchic disassembly process begins with the solution of the two-handle equations of motion of the root node. From this solution, the spatial accelerations and constraint forces acting on the two handles of the single assembly are known. The spatial accelerations and constraint forces generated by solving the two-handle equations of an assembly are identically the values of the spatial accelerations and constraint forces on one handle on each of the two constituent assemblies. From these known quantities, the two-handle equations of motion of the constituent assemblies can be easily solved for the spatial constraint force and acceleration at the connecting joint. Thus, the hierarchic disassembly process continues till all the unknown quantities in all the subassemblies are completely known.

3.2 Sensitivity formulation

From the previous section, the two-handle equations of motion of a body k can be written as follows

$$\mathcal{A}_1^k = \zeta_{11}^k \mathcal{F}_1^k + \zeta_{12}^k \mathcal{F}_2^k + \zeta_{13}^k \tag{26}$$

$$\mathcal{A}_2^k = \zeta_{21}^k \mathcal{F}_1^k + \zeta_{22}^k \mathcal{F}_2^k + \zeta_{23}^k \tag{27}$$

In the above equations, the matrices ζ_{11}^k and ζ_{22}^k are symmetric positive definite (SPD), and the matrices ζ_{12}^k and ζ_{21}^k are transposes of each other. The matrices ζ_{13}^k and ζ_{23}^k include the terms that are state-dependent (for, e.g., the active forces) and can be calculated independently on each body. Similar expressions for body $k+1$ can be written as

$$\mathcal{A}_1^{k+1} = \zeta_{11}^{k+1} \mathcal{F}_1^{k+1} + \zeta_{12}^{k+1} \mathcal{F}_2^{k+1} + \zeta_{13}^{k+1} \tag{28}$$

$$\mathcal{A}_2^{k+1} = \zeta_{21}^{k+1} \mathcal{F}_1^{k+1} + \zeta_{22}^{k+1} \mathcal{F}_2^{k+1} + \zeta_{23}^{k+1} \tag{29}$$

Let p_j represent any design variable with respect to which the sensitivity of the dynamic system's objective function J is to be calculated. For a dynamic system, p_j may be a mass or inertia value, geometric constant such as length, radius, or active force among others. Differentiating (26) and (27) with respect to p_j, the following equations are derived.

$$\frac{d\mathcal{A}_1^k}{dp_j} = \frac{d\zeta_{11}^k}{dp_j}\mathcal{F}_1^k + \zeta_{11}^k \frac{d\mathcal{F}_1^k}{dp_j} + \frac{d\zeta_{12}^k}{dp_j}\mathcal{F}_2^k + \zeta_{12}^k \frac{d\mathcal{F}_2^k}{dp_j} + \frac{d\zeta_{13}^k}{dp_j} \tag{30}$$

$$\frac{d\mathcal{A}_2^k}{dp_j} = \frac{d\zeta_{21}^k}{dp_j}\mathcal{F}_1^k + \zeta_{21}^k \frac{d\mathcal{F}_1^k}{dp_j} + \frac{d\zeta_{22}^k}{dp_j}\mathcal{F}_2^k + \zeta_{22}^k \frac{d\mathcal{F}_2^k}{dp_j} + \frac{d\zeta_{23}^k}{dp_j} \tag{31}$$

With

$$\frac{d(\mathcal{M}_0^k)^{-1}}{dp_j} = -(\mathcal{M}_0^k)^{-1} \frac{d\mathcal{M}_0^k}{dp_j} (\mathcal{M}_0^k)^{-1} \tag{32}$$

In the above equations, the terms $d\zeta_{ij}^k/dp_j$ can be easily obtained from $d(\mathcal{M}_0^k)/dp_j$ and $d\mathcal{S}^{k_0 k_i}/dp_j$ locally on each body, as there is no coupling in these terms from other bodies in the system. These terms are zero when p_j is a design variable that is not local to the body k. The other terms, viz. $d\mathcal{F}_i^k/dp_j$ and $d\mathcal{A}_i^k/dp_j$ $(i = 1:2)$, cannot be calculated locally on each body, as these depend on the coupling between different bodies in the system. Further, by solving the equations of motion (26) and (27) at any instant, the terms $\mathcal{F}_i^k$ $(i = 1:2)$ are generated. Thus, having solved the equations of motion at any instant, (30) and (31) reduce to the following form with the only unknowns at each body being the terms $d\mathcal{F}_i^k/dp_j$ and $d\mathcal{A}_i^k/dp_j$ $(i = 1:2)$.

$$\frac{d\mathcal{A}_1^k}{dp_j} = \Phi_{11}^k \frac{d\mathcal{F}_1^k}{dp_j} + \Phi_{12}^k \frac{d\mathcal{F}_2^k}{dp_j} + \Phi_{13}^k \tag{33}$$

$$\frac{d\mathcal{A}_2^k}{dp_j} = \Phi_{21}^k \frac{d\mathcal{F}_1^k}{dp_j} + \Phi_{22}^k \frac{d\mathcal{F}_2^k}{dp_j} + \Phi_{23}^k \tag{34}$$

where

$$\Phi_{13}^k = \frac{d\zeta_{11}^k}{dp_j}\mathcal{F}_1^k + \frac{d\zeta_{12}^k}{dp_j}\mathcal{F}_2^k + \frac{d\zeta_{13}^k}{dp_j} \tag{35}$$

and

$$\Phi_{23}^k = \frac{d\zeta_{21}^k}{dp_j}\mathcal{F}_1^k + \frac{d\zeta_{22}^k}{dp_j}\mathcal{F}_2^k + \frac{d\zeta_{23}^k}{dp_j} \tag{36}$$

with

$$\Phi_{ij}^k = \frac{d\zeta_{ij}^k}{dp_j} \quad \text{for i , j = 1:2} \tag{37}$$

The corresponding equations for body $k+1$ can be expressed as

$$\frac{d\mathcal{A}_1^{k+1}}{dp_j} = \Phi_{11}^{k+1} \frac{d\mathcal{F}_1^{k+1}}{dp_j} + \Phi_{12}^{k+1} \frac{d\mathcal{F}_2^{k+1}}{dp_j} + \Phi_{13}^{k+1} \tag{38}$$

$$\frac{d\mathcal{A}_2^{k+1}}{dp_j} = \Phi_{21}^{k+1} \frac{d\mathcal{F}_1^{k+1}}{dp_j} + \Phi_{22}^{k+1} \frac{d\mathcal{F}_2^{k+1}}{dp_j} + \Phi_{23}^{k+1} \tag{39}$$

Thus, (26) and (27) and (33) and (34) are in the same analytical form, albeit with different quantities in the equations. Further, (33) and (34) are obtained in the desired form only if (26) and (27) have already been solved. Thus, the basic procedure is: (1) solve the dynamics equations of motion to generate the values of the constraint forces at each body; (2) substitute the values for the constraint forces in (30) and (31) to generate (33) and (34). These are now the sensitivity equations of each body that need to be solved.

4 Two-handle generalized inertia

The relative acceleration between two joint locations on successive bodies k and $k+1$ can be expressed in terms of the free modes of motion matrix P^{J^k} and the generalized speeds at the joint u^{J^k} as

$$\mathcal{A}_1^{k+1} - \mathcal{A}_2^{k+1} = P^{J^k}\dot{u}^{J^k} + \dot{P}^{J^k}u^{J^k} \tag{40}$$

Differentiating the above equation with respect to parameter p_j,

$$\frac{d\mathcal{A}_1^{k+1}}{dp_j} - \frac{d\mathcal{A}_2^{k}}{dp_j} = P^{J^k}\frac{d\dot{u}^{J^k}}{dp_j} + \underbrace{\frac{dP^{J^k}}{dp_j}\dot{u}^{J^k} + \dot{P}^{J^k}\frac{du^{J^k}}{dp_j} + \frac{d\dot{P}^{J^k}}{dp_j}u^{J^k}}_{Locally\ Generated} \tag{41}$$

$$\Rightarrow \quad \frac{d\mathcal{A}_1^{k+1}}{dp_j} - \frac{d\mathcal{A}_2^{k}}{dp_j} = P^{J^k}\frac{d\dot{u}^{J^k}}{dp_j} + \Pi \tag{42}$$

where

$$\Pi = \frac{dP^{J^k}}{dp_j}\dot{u}^{J^k} + \frac{d\dot{P}^{J^k}}{dp_j}u^{J^k} + \dot{P}^{J^k}\frac{du^{J^k}}{dp_j} \tag{43}$$

In the above equations, locally generated terms depend only on the state sensitivities and can be treated as known quantities. Further, from Newton's third law, the loads acting through joint J^k as seen by bodies k and $k+1$ are equal and opposite.

$$\mathcal{F}_2^k = -\mathcal{F}_1^{k+1} \quad \Rightarrow \quad \frac{d\mathcal{F}_2^k}{dp_j} = -\frac{d\mathcal{F}_1^{k+1}}{dp_j} \tag{44}$$

Substituting the expressions for $d\mathcal{A}_2^k/dp_j$ and $d\mathcal{A}_1^{k+1}/dp_j$ from (34), (35), (36), (37), (38), and (39) into (41), respectively, and using the relationships (21), the following expressions can be derived:

$$\frac{d\mathcal{A}_1^{k+1}}{dp_j} - \frac{d\mathcal{A}_2^{k}}{dp_j} = \Phi_{11}^{k+1}\frac{d\mathcal{F}_1^{k+1}}{dp_j} + \Phi_{12}^{k+1}\frac{d\mathcal{F}_2^{k+1}}{dp_j} + \Phi_{13}^{k+1} - \Phi_{21}^{k}\frac{d\mathcal{F}_1^{k}}{dp_j} - \Phi_{22}^{k}\frac{d\mathcal{F}_2^{k}}{dp_j} - \Phi_{23}^{k} \tag{45}$$

$$\Rightarrow \left[\Phi_{11}^{k+1}+\Phi_{22}^{k}\right]\frac{d\mathcal{F}_1^{k+1}}{dp_j} = \left[\Phi_{21}^{k}\frac{d\mathcal{F}_1^{k}}{dp_j} - \Phi_{12}^{k+1}\frac{d\mathcal{F}_2^{k+1}}{dp_j} + \Phi_{23}^{k} - \Phi_{13}^{k+1} + P^{J^k}\frac{d\dot{u}}{dp_j} + \Pi\right] \tag{46}$$

Pre-multiplying (46) by $(D^{J^k})^T$ and calling on the orthogonality condition between D^{J^k} and P^{J^k},

$$\begin{aligned}(D^{J^k})^T\left[\Phi_{11}^{k+1}+\Phi_{22}^{k}\right]\frac{d\mathcal{F}_1^{k+1}}{dp_j} &= (D^{J^k})^T\left[\Phi_{21}^{k}\frac{d\mathcal{F}_1^{k}}{dp_j} + \Phi_{23}^{k} - \Phi_{13}^{k+1} - \Phi_{12}^{k+1}\frac{d\mathcal{F}_2^{k+1}}{dp_j} + \Pi\right] \\ &\quad + \underbrace{(D^{J^k})^T P^{J^k}}_{0}\frac{d\dot{u}}{dp_j}\end{aligned} \tag{47}$$

From the definition of the orthogonal complement of joint motion subspace, the constraint force $\mathcal{F}_1^{k+1}$ can be expressed in terms of the measure numbers of the constraint torques and constraint forces as

$$\mathcal{F}_1^{k+1} = D^{J^k}\mathbf{F}_1^{k+1} \tag{48}$$

$$\Rightarrow \quad \frac{d\mathcal{F}_1^{k+1}}{dp_j} = \frac{dD^{J^k}}{dp_j}\mathbf{F}_1^{k+1} + D^{J^k}\frac{d\mathbf{F}_1^{k+1}}{dp_j} \tag{49}$$

where the constraint force and constraint moment measure numbers $\tilde{f_{1c}}^{k+1}$ and $\tilde{\tau_{1c}}^{k+1}$, respectively, are represented as

$$\mathbf{F}_1^{k+1} = \begin{bmatrix}\tilde{\tau_{1c}}^{k+1} \\ \tilde{f_{1c}}^{k+1}\end{bmatrix} \tag{50}$$

Substituting this into (47), an expression for $d\mathcal{F}_1^{k+1}/dp_j$ can be derived as below:

$$\begin{aligned}D^{J^k\,T}\left[\Phi_{11}^{k+1}+\Phi_{22}^{k}\right]\left(\frac{dD^{J^k}}{dp_j}\mathbf{F}_1^{k+1} + D^{J^k}\frac{d\mathbf{F}_1^{k+1}}{dp_j}\right) \\ = D^{J^k\,T}\left[\Phi_{21}^{k}\frac{d\mathcal{F}_1^{k}}{dp_j} + \Phi_{23}^{k} - \Phi_{13}^{k+1} - \Phi_{12}^{k+1}\frac{d\mathcal{F}_2^{k+1}}{dp_j} + \Pi\right]\end{aligned} \tag{51}$$

$$\Rightarrow \frac{d\mathcal{F}_1^{k+1}}{dp_j} = \mathcal{X}\left[\Phi_{21}^{k}\frac{d\mathcal{F}_1^{k}}{dp_j} + \Phi_{23}^{k} - \Phi_{12}^{k+1}\frac{d\mathcal{F}_2^{k+1}}{dp_j} - \Phi_{13}^{k+1}\right] - \frac{dD^{J^k}}{dp_j}\mathbf{F}_1^{k+1} \tag{52}$$

$$\text{where } \mathcal{X} = D^{J^k}([D^{J^k}]^T[\Phi_{22}^{k} + \Phi_{11}^{k+1}]D^{J^k})^{-1}[D^{J^k}]^T \tag{53}$$

Substituting this expression for $d\mathcal{F}_1^{k+1}/dp_j$ and $d\mathcal{F}_2^{k}/dp_j$ into (33), (34), (35), (36), (37), (38), and (39), the following are obtained:

$$\frac{d\mathcal{A}_1^k}{dp_j} = \Phi_{11}^{k:k+1}\frac{d\mathcal{F}_1^k}{dp_j} + \Phi_{12}^{k:k+1}\frac{d\mathcal{F}_2^{k+1}}{dp_j} + \Phi_{13}^{k:k+1} \tag{54}$$

$$\frac{d\mathcal{A}_2^{k+1}}{dp_j} = \Phi_{21}^{k:k+1}\frac{d\mathcal{F}_1^k}{dp_j} + \Phi_{22}^{k:k+1}\frac{d\mathcal{F}_2^{k+1}}{dp_j} + \Phi_{23}^{k:k+1} \tag{55}$$

In substituting the expression for the derivative of the constraint load at the common joint, the equations of the derivatives of the spatial accelerations of the two bodies are coupled together to form the corresponding equations of the resulting assembly. In the resulting assembly, the two joints that connect the assembly to its parent and child bodies (or assemblies) are J_1^k and J_2^{k+1}. The $\Phi_{ij}^{k:k+1}$ are the inertia coupling terms of the assembly of bodies k and $k+1$ given by

$$\Phi_{11}^{k:k+1} = \left[\Phi_{11}^k - \Phi_{12}^k \mathcal{X} \Phi_{21}^k\right] \tag{56}$$

$$\Phi_{12}^{k:k+1} = \left[\Phi_{12}^k \mathcal{X} \Phi_{12}^{k+1}\right] \tag{57}$$

$$\Phi_{13}^{k:k+1} = \left[\Phi_{13}^k - \Phi_{12}^k \mathcal{X} \Phi_{13}^{k+1}\right] \tag{58}$$

$$\Phi_{21}^{k:k+1} = \left[\Phi_{21}^{k+1} \mathcal{X} \Phi_{21}^k\right] \tag{59}$$

$$\Phi_{22}^{k:k+1} = \left[\Phi_{22}^{k+1} - \Phi_{21}^{k+1} \mathcal{X} \Phi_{12}^{k+1}\right] \tag{60}$$

$$\Phi_{23}^{k:k+1} = \left[\Phi_{23}^{k+1} + \Phi_{21}^{k+1} \mathcal{X} \Phi_{23}^{k+1}\right] \tag{61}$$

5 Hierarchic assembly–disassembly

In the previous section, a set of recursive formulae were derived that may be used to couple together the sensitivity equations of two adjacent bodies to form the corresponding equations of the resulting assembly. In the associated manipulations, the two bodies are coupled together to form an assembly by expressing the derivative of the intermediate (common) joint constraint load with respect to the design variable in terms of the corresponding derivatives of the constraint forces at the other two handles. This process can now be repeated for all bodies in the system where the equations of two successive bodies or assemblies are coupled together using the recursive formulae to obtain the corresponding equations of the resulting assembly. This process works hierarchically exploiting the same structure as that of a binary tree.

This process begins at the level of individual bodies of the system. Adjacent bodies of the system are hierarchically assembled to construct a binary tree as shown in Fig. 2. Individual bodies that make up the system form the leaf nodes of the binary tree. The sensitivity equations of motion of a pair of bodies are coupled together using the recursive set of formulae (56), (57), (58), (59), (60), and (61) to form the corresponding equations of the resulting assembly. The resulting assembly now corresponds to a node of the next level in the binary tree. Working along the binary tree in this hierarchic assembly process, only a single assembly is left at the root node of the binary tree. The root node corresponds to the two-handle representation of the entire articulated system modeled as a single assembly. The sensitivity equations of this root node can be expressed as

$$\frac{d\mathcal{A}_1^1}{dp_j} = \Phi_{11}^{1:nb}\frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{12}^{1:nb}\frac{d\mathcal{F}_2^{nb}}{dp_j} + \Phi_{13}^{1:nb} \tag{62}$$

$$\frac{d\mathcal{A}_2^{nb}}{dp_j} = \Phi_{21}^{1:nb}\frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{22}^{1:nb}\frac{d\mathcal{F}_2^{nb}}{dp_j} + \Phi_{23}^{1:nb} \tag{63}$$

Here, the superscript *1:nb* is used to denote the whole system being represented as a single entity as the root node of the binary tree. In this case, the handles 1 and 2 of this entity are the boundary joints of the articulated system. Similarly, the derivatives of the spatial constraint loads are those of the spatial constraint loads arising from the interaction of the system with its boundaries. The above equations represent two sets of equations in terms of four sets of unknowns, i.e., the derivatives of the spatial accelerations at the boundary joints $d\mathcal{A}_1^1/dp_j$ and $d\mathcal{A}_2^{nb}/dp_j$ and the derivatives of the corresponding constraint loads and $d\mathcal{F}_1^1/dp_j$ and $d\mathcal{F}_2^{nb}/dp_j$. Consider the three following scenarios that may arise for a system.

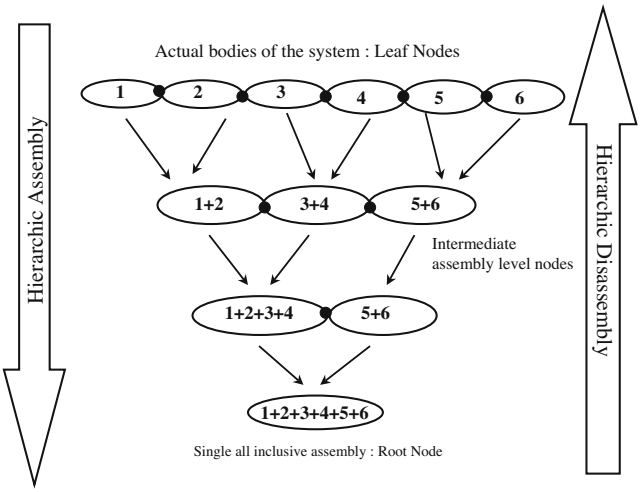

Fig. 2 The hierarchic assembly and disassembly process using binary tree structure

5.1 Free floating

This case corresponds to a system that is free floating, i.e., there are no kinematic joints connecting the system to the inertial frame. In the absence of any kinematic joints at either boundary, there are no constraint forces that can act on the system at the boundaries. In this case, in (62) and (63), the constraint loads terms are all zero, and hence, their derivatives are always zero. From this, the derivatives of the spatial accelerations can be easily solved as

$$\frac{d\mathcal{A}_1^1}{dp_j} = \Phi_{13}^{1:nb} \quad \text{and} \quad \frac{d\mathcal{A}_2^{nb}}{dp_j} = \Phi_{23}^{1:nb} \tag{64}$$

5.2 Anchored at one end by kinematic joint

In this case, the system is connected to the inertial frame by a kinematic joint at one end while the other end is free floating. For such a system, there is no constraint load acting at the free end, and in (62) and (63), the term $d\mathcal{F}_2^{nb}/dp_j = 0$, and hence, its derivative is also always zero. However, at the other end, the system will experience a constraint load because of the presence of the kinematic joint and its derivative needs to be accounted for. The equations in this case reduce to

$$\frac{d\mathcal{A}_1^1}{dp_j} = \Phi_{11}^{1:nb}\frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{13}^{1:nb} \tag{65}$$

$$\frac{d\mathcal{A}_2^{nb}}{dp_j} = \Phi_{21}^{1:nb}\frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{23}^{1:nb} \tag{66}$$

From the definition of the kinematic joint and its joint motion map, there exist the following kinematic relations:

$$\frac{d\mathcal{A}_1^1}{dp_j} = P^1\frac{d\dot{u}^1}{dp_j} + \underbrace{\frac{dP^1}{dp_j}\dot{u}^1 + \frac{d(\dot{P}^1u^1)}{dp_j}}_{\text{Locally generated}} \tag{67}$$

$$\Rightarrow \quad \frac{d\mathcal{A}_1^1}{dp_j}P^1\frac{d\dot{u}^1}{dp_j} + \Pi \tag{68}$$

where

$$\Pi = \frac{dP^1}{dp_j}\dot{u}^1 + \frac{d(\dot{P}^1u^1)}{dp_j} \tag{69}$$

Further, from the definition of the orthogonal complement of the joint motion map, the constraint load at the handle can be expressed as

$$\mathcal{F}_1^1 = D^1\mathbf{F}_1^1 \quad \Rightarrow \quad \frac{d\mathcal{F}_1^1}{dp_j} = \frac{dD^1}{dp_j}\mathbf{F}_1^1 + D^1\frac{d\mathbf{F}_1^1}{dp_j} \tag{70}$$

Substituting (67), (68), (69), and (70) into (65) and (66), the following are derived.

$$P^1\frac{d\dot{u}^1}{dp_j} + \Pi = \Phi_{11}^{1:nb}\left[\frac{dD^1}{dp_j}\mathbf{F}_1^1 + D^1\frac{d\mathbf{F}_1^1}{dp_j}\right] + \Phi_{13}^{1:nb} \tag{71}$$

$$\Rightarrow \; P^1\frac{d\dot{u}^1}{dp_j} = \Phi_{11}^{1:nb}\left[\frac{dD^1}{dp_j}\mathbf{F}_1^1 + D^1\frac{d\mathbf{F}_1^1}{dp_j}\right] + \Phi_{13}^{1:nb} - \Pi \tag{72}$$

Using the orthogonality relation between P^1 and D^1, the derivative of the generalized speed at the joint as well as that of the constraint load can be solved from (72) as

$$\underbrace{D^{1\,T}P^1}_{0}\frac{d\dot{u}^1}{dp_j} = D^{1\,T}\Phi_{11}^{1:nb}D^1\frac{d\mathbf{F}_1^1}{dp_j} + D^{1\,T}\left[\Phi_{11}^{1:nb}\tfrac{dD^1}{dp_j}\mathbf{F}_1^1 + \Phi_{13}^{1:nb} - \Pi\right] \tag{73}$$

$$\Rightarrow \frac{d\mathcal{F}_1^1}{dp_j} = -D^1\left[(D^1)^T\Phi_{11}^{1:nb}D^1\right]^{-1}(D^1)^T \times \left[\Phi_{11}^{1:nb}\tfrac{dD^1}{dp_j}\mathbf{F}_1^1 + \Phi_{13}^{1:nb} - \Pi\right] \tag{74}$$

$$\Rightarrow \frac{du^1}{dp_j} = P^1\left[(P^1)^T(\Phi_{11}^{1:nb})^{-1}P^1\right]^{-1}(P^1)^T \times \left[\Phi_{11}^{1:nb}\tfrac{dD^1}{dp_j}\mathbf{F}_1^1 + \Phi_{13}^{1:nb} - \Pi\right] \tag{75}$$

Substituting (74) and (75) into (65) and (66), the derivatives of the boundary spatial accelerations, i.e., $d\mathcal{A}_1^1/dp_j$ and $d\mathcal{A}_1^1/dp_j$, can be easily calculated.

5.3 Anchored at both ends by kinematic joints

In this case, the system is connected to the inertial frame by a kinematic joint at both ends, and the system reduces to a kinematically closed-loop topology. For such a system, there are constraint loads acting at both ends due to the kinematic joints. In this case, the sensitivity equations for the system remain

$$\frac{d\mathcal{A}_1^1}{dp_j} = \Phi_{11}^{1:nb}\frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{12}^{1:nb}\frac{d\mathcal{F}_2^{nb}}{dp_j} + \Phi_{13}^{1:nb} \tag{76}$$

$$\frac{d\mathcal{A}_2^{nb}}{dp_j} = \Phi_{21}^{1:nb}\frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{22}^{1:nb}\frac{d\mathcal{F}_2^{nb}}{dp_j} + \Phi_{23}^{1:nb} \tag{77}$$

Similar to the previous situation, the following kinematic relations exist between the boundary joints and their joint motion maps.

$$\frac{d\mathcal{A}_1^1}{dp_j} = P^1 \frac{d\dot{u}^1}{dp_j} + \underbrace{\frac{dP^1}{dp_j}\dot{u}^1 + \frac{d(\dot{P}^1 u^1)}{dp_j}}_{\text{Locally generated}}$$

and

$$\frac{d\mathcal{A}_2^{nb}}{dp_j} = P^{nb} \frac{d\dot{u}^{nb}}{dp_j} + \underbrace{\frac{dP^{nb}}{dp_j}\dot{u}^{nb} + \frac{d(\dot{P}^{nb} u^{nb})}{dp_j}}_{\text{Locally generated}} \tag{78}$$

$$\Rightarrow \quad \frac{d\mathcal{A}_1^1}{dp_j} = P^1 \frac{d\dot{u}^1}{dp_j} + \Pi_1$$

and

$$\frac{d\mathcal{A}_2^{nb}}{dp_j} = P^{nb} \frac{d\dot{u}^{nb}}{dp_j} + \Pi_2 \tag{79}$$

where

$$\Pi_1 = \frac{dP^1}{dp_j}\dot{u}^1 + \frac{d(\dot{P}^1 u^1)}{dp_j}$$

and

$$\Pi_2 = \frac{dP^{nb}}{dp_j}\dot{u}^{nb} + \frac{d(\dot{P}^{nb} u^{nb})}{dp_j} \tag{80}$$

Further, from the definition of the orthogonal complement of the joint motion map, the constraint load at the handle can be expressed as

$$\mathcal{F}_1^1 = D^1 \mathbf{F}_1^1 \;\Rightarrow\; \frac{d\mathcal{F}_1^1}{dp_j} = \frac{dD^1}{dp_j}\mathbf{F}_1^1 + D^1 \frac{d\mathbf{F}_1^1}{dp_j} \tag{81}$$

$$\mathcal{F}_2^{nb} = D^{nb} \mathbf{F}_2^{nb} \;\Rightarrow\; \frac{d\mathcal{F}_2^{nb}}{dp_j} = \frac{dD^{nb}}{dp_j}\mathbf{F}_2^{nb} + D^{nb} \frac{d\mathbf{F}_2^{nb}}{dp_j} \tag{82}$$

Substituting (79) into (76) and (77) and absorbing the terms Π_i into the $\Phi_{i3}^{1:nb}$ term ($i = 1:2$), one obtains

$$P^1 \frac{d\dot{u}^1}{dp_j} = \Phi_{11}^{1:nb} \frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{12}^{1:nb} \frac{d\mathcal{F}_2^{nb}}{dp_j} + \Phi_{13}^{1:nb} \tag{83}$$

$$P^{nb} \frac{d\dot{u}^{nb}}{dp_j} = \Phi_{21}^{1:nb} \frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{22}^{1:nb} \frac{d\mathcal{F}_2^{nb}}{dp_j} + \Phi_{23}^{1:nb} \tag{84}$$

Multiplying the above equations by $(D^1)^T$ and $(D^{nb})^T$, respectively, and calling on the orthogonality relation, the following are obtained.

$$\overbrace{(D^1)^T P^1}^{0} \frac{du}{dp_j}^1 = (D^1)^T \Big[\Phi_{11}^{1:nb} \frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{12}^{1:nb} \frac{d\mathcal{F}_2^{nb}}{dp_j} + \Phi_{13}^{1:nb} \Big] = 0 \tag{85}$$

$$\underbrace{(D^{nb})^T P^{nb}}_{0} \frac{du^{nb}}{dp_j} = (D^{nb})^T [\Phi_{21}^{1:nb} \frac{d\mathcal{F}_1^1}{dp_j} + \Phi_{22}^{1:nb} \frac{d\mathcal{F}_2^{nb}}{dp_j} + \Phi_{23}^{1:nb}] = 0 \tag{86}$$

Substituting the expressions for the derivatives of the constraint loads from (81) and (82) into (85) and (86), one obtains

$$(D^1)^T \Phi_{11}^{1:nb} D^1 \frac{d\mathbf{F}_1^1}{dp_j} + (D^1)^T \Phi_{12}^{1:nb} D^{nb} \frac{d\mathbf{F}_2^{nb}}{dp_j} + (D^{nb})^T \Big[\Phi_{11}^{1:nb} \frac{dD^1}{dp_j}\mathbf{F}_1^1 + \Phi_{12}^{1:nb} \frac{dD^{nb}}{dp_j}\mathbf{F}_2^{nb} + \Phi_{13}^{1:nb} \Big] = 0 \tag{87}$$

$$(D^{nb})^T \Phi_{21}^{1:nb} D^1 \frac{d\mathbf{F}_1^1}{dp_j} + (D^{nb})^T \Phi_{22}^{1:nb} D^{nb} \frac{d\mathbf{F}_2^{nb}}{dp_j} + (D^{nb})^T \Big[\Phi_{21}^{1:nb} \frac{dD^1}{dp_j}\mathbf{F}_1^1 + \Phi_{22}^{1:nb} \frac{dD^{nb}}{dp_j}\mathbf{F}_2^{nb} + \Phi_{23}^{1:nb} \Big] = 0 \tag{88}$$

In these equations, $(D^1)^T \Phi_{11}^{1:nb} D^1$ and $(D^{nb})^T \Phi_{22}^{1:nb} D^{nb}$ are SPD matrices, and there is no problem associated with their inversion. For notational convenience, the above equations can be represented compactly in matrix form as

$$\begin{bmatrix} \chi_{11} & \chi_{12} \\ \chi_{21} & \chi_{22} \end{bmatrix} \begin{bmatrix} d\mathbf{F}_1^1/dp_j \\ d\mathbf{F}_2^{nb}/dp_j \end{bmatrix} = - \begin{bmatrix} \chi_{13} \\ \chi_{23} \end{bmatrix} \tag{89}$$

where the corresponding χ_{ij} can be derived from the above equation. The matrix in (89) is also SPD with $\chi_{12} = \chi_{21}^T$. The above set of equations can now be easily solved. Having solved the above equations for the values of $d\mathbf{F}_1^1/dp_j$ and $d\mathbf{F}_2^{nb}/dp_j$, the corresponding expression for $d\mathcal{F}_1^1/dp_j$ and $d\mathcal{F}_2^{nb}/dp_j$ can be obtained from (81) and (82). At this point, the derivatives of both constraint loads on the boundary joints are known. Consequently, (76) and (77) of the root node can be solved to obtain the derivatives of the spatial accelerations $d\mathcal{A}_1^1/dp_j$ and $d\mathcal{A}_1^1/dp_j$ at the corresponding joints.

Thus, in all three cases, the derivatives of the spatial accelerations and the constraint loads at the boundary joints can be calculated. This initiates the hierarchic disassembly process. The derivatives of the spatial accelerations and the constraint loads generated by solving the sensitivity equations of an assembly are identically the values of the derivatives of the spatial accelerations and the constraint loads on one handle on each of the two constituent assemblies. From these known quantities, the sensitivity equations of the constituent assemblies can be solved to obtain the derivatives of the spatial accelerations and that of the constraint loads at the connecting joint. For example, for a representative assembly made from body k and body $k+1$, the sensitivity equations are given by (54) and (55). On solving these equations, the quantities $d\mathcal{A}_1^k/dp_j$, $d\mathcal{A}_2^{k+1}/dp_j$, $d\mathcal{F}_1^k/dp_j$, and $d\mathcal{F}_2^{k+1}/dp_j$ are generated. These quantities are then substituted into the sensitivity equations of the constituent sub-assemblies body k and body $k+1$. Thus, knowing the values of $d\mathcal{A}_1^k/dp_j$ and $d\mathcal{F}_1^k/dp_j$, (33) and (34) can be solved, while from $d\mathcal{A}_2^{k+1}/dp_j$ and $d\mathcal{F}_2^{k+1}/dp_j$, (38) and (39) can also be solved. This process is repeated in a hierarchic disassembly of the binary tree where the known derivatives of the boundary conditions are used to solve the sensitivity equations of the immediate sub-assemblies, until derivatives of the spatial acceleration and constraint forces on all bodies in the system are calculated.

6 Discussion

In the previous section, the method for calculating the sensitivities of multibody dynamics systems in the three general topologies is described. These topologies include free-floating kinematic chains, kinematic chains anchored at one end, and topologies with kinematically closed loops. The sensitivity analysis is based on the forward dynamics algorithms (Featherstone 1999a; Mukherjee and Anderson 2007) and gains from the inherent capabilities of these methods. The calculation of the sensitivities is simplified by exploiting the topology of the system and the fundamental idea of the joint free modes of motion map and its orthogonal complement. The local derivatives used in the algorithm are temporally invariant and exact. These are generated once during a preprocessing step and introduced into the algorithm as an input. This algorithm works in six sweeps of the system, traversing the system topology like a binary tree. The first four sweeps are associated with formulating and solving the equations of motion for the forward dynamics problem. The next two sweeps are associated with the sensitivity analysis and correspond to the hierarchic assembly and the hierarchic disassembly processes, respectively. These last two sweeps may additionally be performed concurrently with the final two sweeps of the forward dynamics formulation if the concurrent formulation is preferred. The concurrent formulation is an implementation detail with a minor manipulation of the equations and is not described in this paper. The concurrent formulation would reduce the number of sweeps of the topology from six to four where the sensitivity analysis could be performed in lock-step with the forward dynamics problem.

Modeling systems in kinematically closed-loop topologies has traditionally been an interesting problem in multibody dynamics because of the presence of explicit loop closure constraints. Along with the dynamics equations of motion of an equivalent unconstrained system, traditional methods maintain the constraints through an extra set of algebraic equations, which are used to either (1) reduce out excessive degrees of freedom producing a minimum dimension system of equations, or (2) augment the equations of motion producing a larger dimension system of equations involving redundant state variables. This gives rise to two primary problems: (1) the saddle point problem originating from constraint equations becoming numerically dependent and (2) the accumulation of integration errors leading to significant drift in constraint satisfaction. Because most methods for sensitivity analysis of multibody dynamics systems are based on the forward dynamics method, the sensitivity analysis of these systems suffers from the same drawbacks as the forward dynamics methods in handling kinematically closed loops.

However, in the method described in this paper, the loop closure constraint is modeled using a different approach. Instead of using explicit constraint equations, the constraints are implicitly imposed by describing the topology of the system through the relative coordinates and the use of a set of redundant generalized coordinates to enforce the loop closure constraint. The redundant set of generalized coordinates maintains the definition of the kinematic joint that converts an unconstrained system into a constrained system in a loop configuration. The presence of an extra kinematic joint introduces an additional orthogonality relation between the map of the free modes of motion, the joint, and its orthogonal complement. This allows for the loop closure constraint to be implicitly imposed. Further, the use of a redundant set of generalized coordinates always maintains the correct dimensionality of the system equations, making this algorithm free from rank deficiency issues as all matrices to be inverted are SPD. This ensures that the algorithm can robustly handle

what would normally be considered singular configurations. The manner in which this method avoids singularities appears similar, in some regards, to that of Euler parameters. With Euler parameters, one deals with a redundant four-member set of generalized coordinates (parameters) for the global and nonsingular description of general spatial rotation. The constraints between these four coordinates are implicitly enforced. If the constraints were explicitly used to reduce out the extra generalized coordinate (parameter), the representation again may become singular. In the same manner, this algorithm enforces the constraints implicitly, thereby, avoiding singularities.

The problem of constraint violation is not completely eliminated in this algorithm because the constraints are imposed at the acceleration level. However, performance of this algorithm on sample test cases as discussed in Mukherjee and Anderson (2007) indicates that the method is able to maintain the constraint violation growth at a minimum rate and perform comparable to (or better than) methods with velocity level constraint imposition. The imposition of the constraints at the velocity and position level within the framework of this algorithm continues to be a research endeavor.

7 Computational complexity and parallel aspects

In the previous sections, the sensitivity analysis for any given multibody system is explained using six sweeps traversing the topology of the system. However, as explained before, the implementation can use four sweeps where the sensitivity information is generated concurrently with the solution of the forward dynamics problem. While the concurrent implementation can further improve the computational efficiency, for ease of understanding, the discussion is limited to the six-sweep process.

The generation of the derivative primitives is dependent on the system under consideration and the design parameters. Consequently, the derivative primitives can be calculated numerically, analytically, or empirically. The generation of the derivative primitives is a preprocessing step and needs to be done once (potentially for an entire family of simulations). Hence, it is not considered in the computational complexity of this algorithm. Once the derivative primitives have been developed, the algorithm proceeds in the same way as the forward dynamics problem. The forward dynamics problem is based on the divide-and-conquer scheme and has been detailed in Featherstone (1999a) and Mukherjee and Anderson (2007). The sensitivity analysis method described in this paper cannot function independent of the forward dynamics problem, as the sensitivity analysis requires the determination of accelerations and the constraint forces before calculating the sensitivities. Thus, a large number of the entities used in the sensitivity analysis are already developed for the forward dynamics problem. Consequently, the first two sweeps of the method are analogous to the first two sweeps of the forward dynamics problem.

The additional cost incurred in the sensitivity analysis is associated with the coupling of the sensitivity equations of successive bodies using the hierarchic assembly process and then the subsequent solution of these equations using the hierarchic disassembly process. In serial implementation, the hierarchic assembly and disassembly processes work recursively and solve the sensitivity problem in linear or $O(n)$ complexity, where n is the number of degrees of freedom of the system. The cost associated with the recursive solution in serial implementation has been studied in detail in Hsu and Anderson (2002) and is similarly applicable to this algorithm.

For parallel implementation, this algorithm is processor and time-optimal, solving the sensitivity problem in $O(log(n))$ complexity in the presence of n processors, where n now is the number of bodies in the system. In the presence of n processors, each body of the system is mapped onto a different processor where the sensitivity problem for that body is formulated. The mapping of the bodies onto the processors is developed in a binary tree representation as shown in Fig. 2. The hierarchic assembly–disassembly process then works via two traversals of the binary tree. These traversals work exactly in the same fashion as for the forward dynamics problem (Featherstone 1999b) and are achieved in $O(log(n))$ complexity. This highly parallel aspect of the algorithm arises from the nature of the sensitivity equations for individual bodies or assemblies. As discussed previously, the equations are cast in the *two-handle* form where the problem is posed in terms of the interactions of the body or an assembly with its boundaries by expressing the internal unknowns as functions of boundary unknowns. This allows the assembly–disassembly process to proceed in an order-independent, concurrent form, facilitating high parallel efficiency.

In the presence of a modest number of processors, the system is divided into assemblies equal to the number of processors available. Within each processor, the equations of the corresponding assembly can be formulated in linear complexity to express the problem in terms of the boundary unknowns of the assembly, similar to the serial implementation. The calculation of the boundary unknowns proceeds using the binary

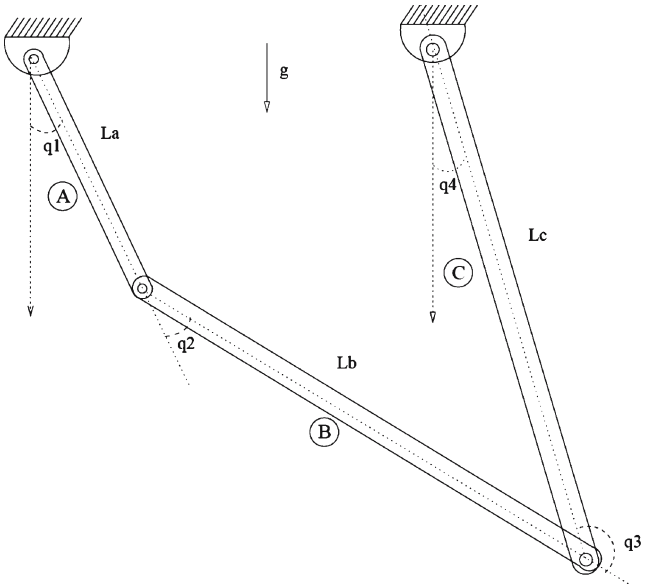

Fig. 3 Four-bar: closed-loop mechanism

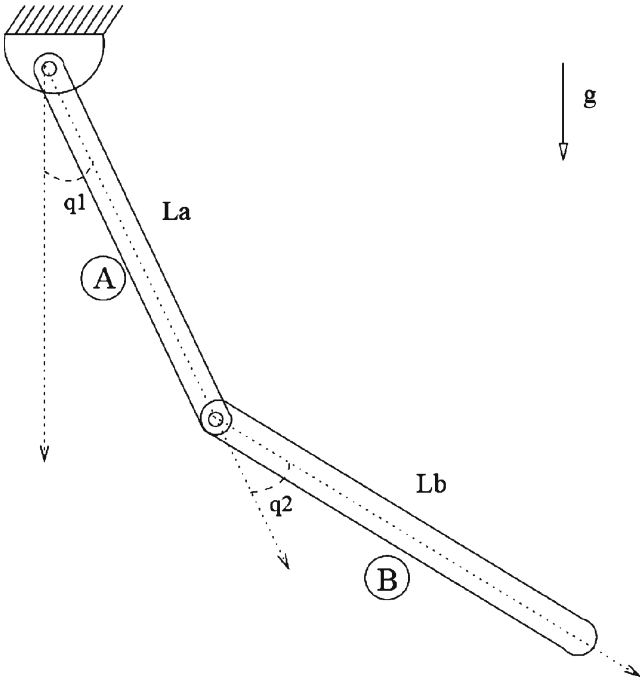

Fig. 5 Double pendulum: serial chain

tree representation of the system, where the leaf nodes now correspond to the assemblies instead of the physical bodies. This is achieved in logarithmic complexity. On completing this, the boundary unknowns of every assembly in the system are known. Using these values, the sensitivity equations of the bodies in each assembly can be solved in linear complexity using the recursive process as in the serial implementation. This linear complexity process is carried out concurrently for all assemblies. The number of bodies in the different assemblies and the actual computational gains would depend on different parameters such as load balancing, multiprocessor architecture, the method for inter-processor communications, and costs, among implementation-dependent aspects.

8 Numerical examples

The sensitivity values generated using the algorithm described above are verified by simulating multibody system test cases previously presented in literature. Results from two test cases are presented here. These test cases were chosen to demonstrate the ability of the algorithm to accurately generate sensitivity information for serial chain systems as well as systems with a kinematically closed loop.

The first test case is a kinematically closed-loop system consisting of three rigid links A, B, and C con-

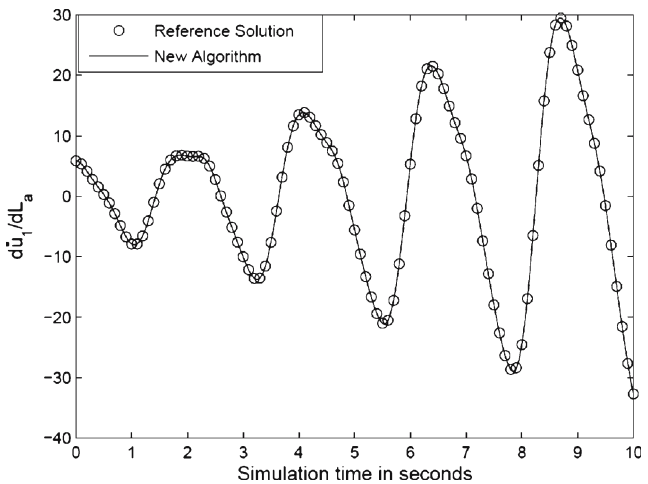

Fig. 4 Sensitivity comparison with reference solution for $d\dot{u}_1/dL_a$

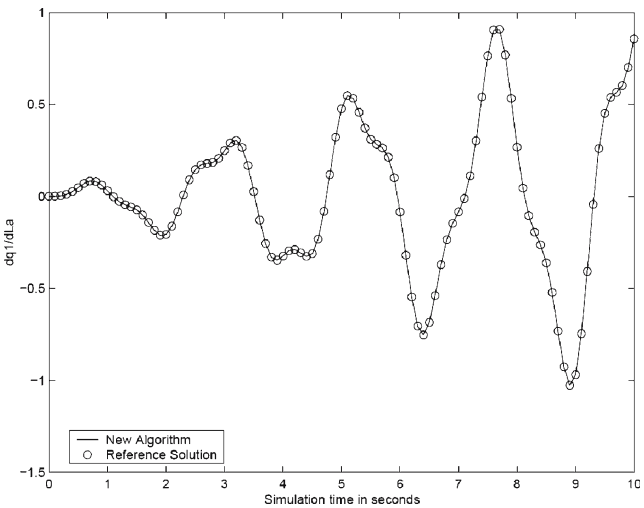

Fig. 6 Sensitivity comparison with reference solution for dq_1/dL_a

nected by revolute joints to form a four-bar mechanism as shown in Fig. 3. The system is released from rest with the initial configuration of $q_1 = 30$, $q_2 = 44.4775$, and $q_4 = 45.5225$ degrees, respectively, under the effect of gravity. The length and mass of the links in the system are $L_a = 1\,m$, $L_b = L_c = 2\,m$, $m_a = 10$, $m_b = m_c = 20\,kg$, and gravity = 9.81. The sensitivity of $\dot{u}_1$ with respect to L_a $(d\dot{u}_1/dL_a)$ is calculated for a 10-s simulation and shown in Fig. 4.

The next system considered is a double pendulum moving under the effect of gravity as shown in Fig. 5. The system parameters are $L_a = L_b = 1\,m$, $M_a = M_b = 1\,m, q_1 = q_2 = \pi/30$ radians, and gravity = 9.81 m/s^2. The mechanism is released from rest. For this system, the sensitivity of q_1 with respect to L_a (dq_1/dL_a) is calculated, and the results are shown in Fig. 6.

In both cases, the results obtained are compared against established results obtained from the direct differential method as described in Anderson and Hsu (2004). The results obtained using the new algorithm are in excellent agreement with the reference solutions. The results clearly demonstrate the ability of the algorithm to provide sensitivity values accurate up to integration tolerance. For the system in closed-loop configuration, the method does not suffer from excessive constraint violations.

9 Conclusions

In this paper, a new efficient method is presented for sensitivity analysis of multibody systems. The method uses a direct differentiation approach and implements it in a divide-and-conquer scheme. The method maps the topology of the system to a binary tree and generates the sensitivity information using several traversals of this binary tree. The computational complexity of this algorithm is expected to be linear and logarithmic in serial and parallel implementations, respectively. The method works in tandem with the forward dynamics problem. Consequently, there is no excessive data storage and no backward integration in this scheme. Thus, the method does not suffer from numerical issues associated with perturbations in design variables. The method is robust and does not suffer from numerical dependency issues associated with singular configurations. The low computational cost of the algorithm, its simplicity in implementation, applicability for serial and closed-loop systems, insensitivity to numerical and data storage issues, and accuracy of results make this algorithm a useful tool in sensitivity analysis for multibody dynamics systems.

Acknowledgements This work was funded by the NSF NIRT Grant Number 0303902. The authors thank the funding agency for their support.

References

Anderson KS, Hsu YH (2001) Low operational order analytic sensitivity analysis for tree-type multibody dynamic systems. J Guidance Control Dyn 24(6):1133–1143

Anderson KS, Hsu YH (2004) 'Order-(n+m)' direct differentiation determination of design sensitivity for constrained multibody dynamic systems. Struct Multidisc Optim 26(3–4):171–182

Bestle D, Eberhard P (1992) Analysing and optimizing multibody systems. Struct Machines 20:67–92

Bestle D, Seybold J (1992) Sensitivity analysis of constrained optimization in dynamic systems. Archive Appl Mech 62:181–190

Bischof CH (1996) On the automatic differentiation of computer programs and an application to multibody systems. In: Proceedings of the IUTAM symposium on optimization of mechanical systems, pp 41–48

Chang CO, Nikravesh PE (1985) Optimal design of mechanical systems with constaint violation stabilization method. J Mech Trans Autom Des 107:493–498

Dias J, Pereira M (1997) Sensitivity analysis of rigidflexible multibody systems. Multibody Syst Dyn 1(3):303–322

Eberhard P (1996) Analysis and optimization of complex multibody systems using advanced sensitivity methods. Math Mech 76:40–43

Etman L (1997) Optimization of multibody systems using approximation concepts. PhD thesis, Technische Universiteit Eindhoven, The Netherlands

Featherstone R (1999a) A divide-and-conquer articulated body algorithm for parallel $O(\log(n))$ calculation of rigid body dynamics. Part 1: Basic algorithm. Int J Robot Res 18(9):867–875

Featherstone R (1999b) A divide-and-conquer articulated body algorithm for parallel $O(\log(n))$ calculation of rigid body dynamics. Part 2: Trees, loops, and accuracy. Int J Robot Res 18(9):876–892

Haug E, Ehle PH (1982) Second-order design sensitivity analysis of mechanical system dynamics. Int J Numer Methods Eng 18:1699–1717

Haug E, Wehage RA, Mani NK (1984) Design sensitivity analysis of large-scaled constrained dynamic mechanicsl systems. Trans ASME 106:156–162

Hsu YH, Anderson KS (2002) Recursive sensitivity analysis for constrained multi-rigid-body dynamic systems design optimization. Struct Multidisc Optim 24(4):312–324

Jain A, Rodrigues G (2000) Sensitivity analysis of multibody systems using spatial operators. In: Proceedings of the international conference on method and models in automation and robotics (MMAR 2000), Miedzyzdroje, Poland

Kim SS, VanderPloeg MJ (1986) Generalized and efficient method for dynamic analysis of mechanical systems using velocity transforms. J Mech Trans Autom Des 108(2):176–182

Mukherjee R, Anderson KS (2007) An orthogonal complement based divide-and-conquer algorithm for constrained multibody systems. Nonlinear Dyn 48(1-2):199–215

Nikravesh PE (1990) Systematic reduction of multibody equations to a minimal set. Int J Non Linear Mech 25(2-3):143–151

Pagalday J, Aranburu I, Avello A, Jalon JD (1995) Multibody dynamics optimization by direct differentiation methods using object oriented programming. In: Proceedings of the IUTAM symposium on optimization of mechanical systems, Stuttgart, Germany, pp 213–220

Serban R, Haug EJ (1998) Kinematic and kinetics derivatives for multibody system analyses. Mech Struct Machines 26(2):145–173

Tak T (1990) A recursive approach to design sensitivity analysis of multibody systems using direct differentiation. PhD thesis, University of Iowa, Iowa City