Springer

Journal of Mechanical Science and Technology

Journal of Mechanical Science and Technology 22 (2008) 714~721

www.springerlink.com/content/1738-494x

A study on numerical solution method for efficient dynamic analysis of constrained multibody systems

Jae-Hwan Lim[1], Hong Jae Yim[1,*], Si-Hyung Lim[1] and Taewon Park[2]

[1]*School of Mechanical and Automotive Engineering, Kookmin University, Jeongneung-dong, Sungbuk-gu, Seoul, Korea*
[2]*Ajou University, San 5, Wonchun-dong, Yeongtong-gu, Suwon, Korea*

(Manuscript Received July 19, 2007; Revised October 17, 2007; Accepted November 28, 2007)

Abstract

A new and efficient computational method for constrained multibody systems is proposed. In the proposed method, local parametrization method is employed to apply the same solution method for position, velocity, and acceleration analyses since the coefficient matrices for each analysis have an identical matrix pattern. The skyline solution method is used to overcome numerical inefficiency when solving large scaled equations. Also, subsystem mpartitioning method is derived systematically to perform parallel processing for real time simulation. To show the numerical accuracy and efficiency of the proposed method, three numerical problems are solved.

Keywords: Method; Multibody systems; Local parametrization method; Skyline solution method; Subsystem partitioning method

1. Introduction

In the design of mechanical systems such as machines, robots, vehicles, and space struct, it is now common to use computer simulations to observe the dynamic responses of the system to be designed [1-3]. Computer-aided design of a mechanical system can reduce the cost and time to construct and test the prototype. Various computational methods and efficient formalisms for mechanical systems have been developed over the past three decades.

Cartesian coordinates and relative coordinates are mainly used for building mathematical modeling. Cartesian coordinate formulations, which yield a maximal set of highly sparse equations, form the basis for highly automated dynamic simulation computer codes [4, 5] that are broadly used in industry. Recursive formulations that build upon the topological relative coordinate foundation [6] have been used to create high-speed dynamic simulation methods [7]. These formulations are applied to commercial programs [8, 9]. The Cartesian formulations are very convenient for deriving equations of motion. The major drawback of the Cartesian coordinate approach is that the dimension of the problem dramatically increases as the number of bodies increases, when compared to the recursive formulations. Hybrid coordinate formulation was proposed to take advantage of both coordinates [10]. Cartesian coordinates are used for deriving equations and relative coordinates are used for performing efficient numerical integration.

A partitioning method was developed to reduce the dimension of the coefficient matrix into *nh*, the number of holonomic constraints, in factorizing it [11]. Lee and Bae proposed a local parametrization method [12] which corrects the drift of the differential variables using an optimization technique. The major advantage of the method is that the coefficient matrices of position, velocity, and acceleration analyses have an identical pattern.

Recently, many researches have tried to speed up computational methods. Parallel computing techniques and machines have been developed to speed up multibody dynamic simulation [13]. Subsystem synthesis methods with independent coordinates have

[*]Corresponding author. Tel.: +82 2 910 4688, Fax.: +82 2 910 4718
E-mail address: hjyim@kookmin.ac.kr
DOI 10.1007/s12206-007-1117-3

been proposed by Song and Kim [14]. In other fields, efficient sparse matrix methods, such as banded-symmetric, bandwidth, and skyline method are used to improve the numerical efficiency in solving a linear system [15-17].

This paper proposes an improved DAE (Differential Algebraic Equation) solution method which employs the subsystem partitioning method, modified local parametrization method, and skyline solution method. The proposed method employs the local parametrization method to apply the same numerical solution method for position, velocity, and acceleration equation since the coefficient matrices for position, velocity, and acceleration analyses have an identical pattern in the local parametrization method. Furthermore, the local parametrization method is modified to diminish the size of the linear equation. The skyline solution method makes the proposed method more efficient by alleviating the burden when solving linear equations. The numerical efficiency and validity for the proposed methods are verified through the numerical examples. Subsystem partitioning method is derived systematically to perform parallel processing for the real time simulation.

2. Local parametrization method

For constrained mechanical systems, the augmented Lagrange equations of motion and constrains for the position, velocity, and acceleration are conventionally written as

$$\mathbf{M}\dot{\mathbf{Y}} + \Phi_{\mathbf{Z}}^{T}\boldsymbol{\lambda} = \mathbf{Q}, \tag{1}$$

$$\Phi = \mathbf{0}, \tag{2}$$

$$\dot{\Phi} = \Phi_{\mathbf{Z}}\mathbf{Y} - \Phi_{t} = \mathbf{0}, \tag{3}$$

$$\ddot{\Phi} = \Phi_{\mathbf{Z}}\dot{\mathbf{Y}} + (\Phi_{\mathbf{Z}}\mathbf{Y})_{\mathbf{Z}}\mathbf{Y} + 2\Phi_{\mathbf{Z}t}\mathbf{Y} + \Phi_{tt} = 0, \tag{4}$$

where $\mathbf{M}$ denotes the mass matrix which depends on the generalized coordinates $\mathbf{Z}$, $\mathbf{Y}$ denotes $\dot{\mathbf{Z}}$, Φ denotes the constraint equations, $\Phi_{\mathbf{Z}}$ denotes the Jacobian matrix of the constraint equations, $\boldsymbol{\lambda}$ denotes the Lagrange multiplier vector corresponding to the Jacobian matrix, and $\mathbf{Q}$ denotes the generalized applied forces.

Local parametrization uses the optimization technique in which integrated generalized coordinates and velocities are projected on the tangent plane of the constraint manifold as in Ref [12]. Considering the design optimization problem to minimize $\left|\mathbf{Z} - \mathbf{Z}^{init.}\right|$, an objective function can be written as

$$F = \frac{1}{2}\left(\mathbf{Z} - \mathbf{Z}^{init.}\right)^{T}\mathbf{M}\left(\mathbf{Z} - \mathbf{Z}^{init.}\right). \tag{5}$$

In Eq. (5) $\mathbf{Z}$ should satisfy the constrained Eq. (2). By the Lagrange multiplier theorem [18], the Lagrange function, L can be written as

$$L = \frac{1}{2}\left(\mathbf{Z} - \mathbf{Z}^{init.}\right)^{T}\mathbf{M}\left(\mathbf{Z} - \mathbf{Z}^{init.}\right) + \boldsymbol{\Phi}^{T}\boldsymbol{\mu}, \tag{6}$$

where $\boldsymbol{\mu}$ is Lagrange multiplier vectors for the constraint Eqs. In the optimum, the following Kuhn-Tucker necessary conditions should be satisfied.

$$\frac{\partial L}{\partial \mathbf{Z}} = \mathbf{M}\left(\mathbf{Z} - \mathbf{Z}^{init.}\right) + \boldsymbol{\Phi}_{\mathbf{Z}}^{T}\boldsymbol{\mu} = \mathbf{0}, \tag{7}$$

$$\frac{\partial L}{\partial \boldsymbol{\mu}} = \boldsymbol{\Phi} = \mathbf{0}. \tag{8}$$

To solve Eqs. (7) and (8), Kuhn-Tucker sufficient conditions are not used, since estimate $\mathbf{Z}^{init.}$ is generally near to the solution of Eq. (8). The Newton-Raphson method is used to solve non-linear Eqs. (7) and (8). The following equation is solved for correction terms $\Delta\mathbf{Z}$ and $\Delta\boldsymbol{\mu}$

$$\begin{bmatrix} \mathbf{M} & \boldsymbol{\Phi}_{\mathbf{Z}}^{T} \\ \boldsymbol{\Phi}_{\mathbf{Z}} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \Delta\mathbf{Z} \\ \Delta\boldsymbol{\mu} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{1} \\ \mathbf{R}_{2} \end{bmatrix}, \tag{9}$$

where

$$\begin{bmatrix} \mathbf{R}_{1} \\ \mathbf{R}_{2} \end{bmatrix} = \begin{bmatrix} -\left\{ \mathbf{M}\left(\mathbf{Z} - \mathbf{Z}^{init.}\right) + \boldsymbol{\Phi}_{\mathbf{Z}}^{T}\boldsymbol{\mu} \right\} \\ -\boldsymbol{\Phi} \end{bmatrix}. \tag{10}$$

$\Delta\mathbf{Z}$ and $\Delta\boldsymbol{\mu}$ are added to $\mathbf{Z}$ and $\boldsymbol{\mu}$ to improve estimates as following

$$\begin{aligned} \mathbf{Z}^{i+1} &= \mathbf{Z}^{i} + \Delta\mathbf{Z}^{i}, \\ \boldsymbol{\mu}^{i+1} &= \boldsymbol{\mu}^{i} + \Delta\boldsymbol{\mu}^{i}. \end{aligned} \tag{11}$$

Similarly, the Eqs. for velocity analysis in matrix and vector form are defined as

$$\begin{bmatrix} \mathbf{M} & \boldsymbol{\Phi}_{\mathbf{Z}}^{T} \\ \boldsymbol{\Phi}_{\mathbf{Z}} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \mathbf{Y} \\ \boldsymbol{\mu}' \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{1}' \\ \mathbf{R}_{2}' \end{bmatrix} \tag{12}$$

where the right-hand side vectors are expressed as

$$\begin{bmatrix} \mathbf{R}_1' \\ \mathbf{R}_2' \end{bmatrix} = \begin{bmatrix} \mathbf{M}\,\mathbf{Y}^{init.} \\ -\mathbf{\Phi}_t \end{bmatrix}, \tag{13}$$

The velocity vector $\mathbf{Y}$ and Lagrange multiplier $\boldsymbol{\mu}'$ can be obtained by solving linear equation (12). From Eqs. (1), (9), and (12), it is known that coefficient matrices of these equations have the identical matrix pattern.

3. Skyline solution method

The computational time for solving linear equations depends on not only the structure of the coefficient matrix but also the size of the coefficient matrix. Therefore, if the size of the coefficient matrix is decreased, the computational efficiency can be increased. In this section an efficient algorithm is proposed to utilize the skyline method. Multiplying Eq. (1) by $\Phi_{\mathbf{Z}}\mathbf{M}^{-1}$ yields

$$\Phi_{\mathbf{Z}}\mathbf{M}^{-1}\mathbf{M}\dot{\mathbf{Y}} + \Phi_{\mathbf{Z}}\mathbf{M}^{-1}\Phi_{\mathbf{Z}}{}^{\mathrm{T}}\boldsymbol{\lambda} = \Phi_{\mathbf{Z}}\mathbf{M}^{-1}\mathbf{Q}. \tag{14}$$

Eq. (14) can be written as

$$\Phi_{\mathbf{Z}}\mathbf{M}^{-1}\Phi_{\mathbf{Z}}{}^{\mathrm{T}}\boldsymbol{\lambda} = \Phi_{\mathbf{Z}}\mathbf{M}^{-1}\mathbf{Q} - \Phi_{\mathbf{Z}}\dot{\mathbf{Y}}. \tag{15}$$

Using Eq. (4), Eq. (15) can be rewritten as

$$\mathbf{C}\boldsymbol{\lambda} = \mathbf{\Phi}_{\mathbf{Z}}\mathbf{M}^{-1}\mathbf{Q} - \gamma. \tag{16}$$

where

$$\begin{aligned} \mathbf{C} &= \mathbf{\Phi}_{\mathbf{Z}}\mathbf{M}^{-1}\mathbf{\Phi}_{\mathbf{Z}}^{T}, \\ \gamma &= -(\Phi_{\mathbf{Z}}\mathbf{Y})_{\mathbf{Z}}\mathbf{Y} - 2\Phi_{\mathbf{Z}t}\mathbf{Y} - \Phi_{tt}. \end{aligned} \tag{17}$$

Eq. (16) can be solved for $\boldsymbol{\lambda}$ by using a linear equation solver. The coefficient matrix $\mathbf{C}$ is real and symmetric. In the coefficient matrix, the diagonal block sub-matrix for the n-th joint can be written as the following form:

$$\mathbf{C}_{n,n} = \mathbf{\Phi}^{n}_{\mathbf{Z}_j}\mathbf{M}_j^{-1}\mathbf{\Phi}^{n^T}_{\mathbf{Z}_j} + \mathbf{\Phi}^{n}_{\mathbf{Z}_i}\mathbf{M}_i^{-1}\mathbf{\Phi}^{n^T}_{\mathbf{Z}_i}. \tag{18}$$

In Eq. (18), bodies i and j are connected with the n-th joint. $\mathbf{C}_{n,n}$ is the square matrix, and the dimension of $\mathbf{C}_{n,n}$ is the number of the kinematic constraints in the n-th joint. Off-diagonal block sub-matrix has the form

$$\mathbf{C}_{n,m} = \mathbf{\Phi}^{n}_{\mathbf{Z}_k}\mathbf{M}_k^{-1}\left(\mathbf{\Phi}^{m}_{\mathbf{Z}_k}\right)^{T}. \tag{19}$$

In Eq. (19), body k is connected with m-th and n-th joints. Once $\boldsymbol{\lambda}$ is solved from Eq. (16), $\dot{\mathbf{Y}}$ can be solved from Eq. (1) as

$$\dot{\mathbf{Y}} = \mathbf{M}^{-1}\mathbf{Q}^{*} \;\text{ or }\; \dot{\mathbf{Y}}_i = \mathbf{M}_i^{-1}\mathbf{Q}^{*}_{i}, (i = 1, 2, \cdots, n), \tag{20}$$

where n is the number of bodies and $\mathbf{Q}^{*}$ is represented as

$$\mathbf{Q}^{*} = \mathbf{Q} - \mathbf{\Phi}_{\mathbf{Z}}^{T}\,\boldsymbol{\lambda}.$$

The skyline solver can be used to utilize the coefficient matrix in Eq. (18) since $\Phi_{\mathbf{Z}}\mathbf{M}^{-1}\Phi_{\mathbf{Z}}^{T}$ is a real symmetric matrix. To increase numerical efficiency, the joint sequences must be rearranged. Fig. 1 shows the serial 7 link mechanism with pin joints. Fig. 2 shows two different matrix patterns according to the joint arrangement. In the figure, the bold line denotes the skyline. From the figure, it is known that an appropriate arrangement can make the skyline lower. To achieve this purpose, the method proposed by Sloan is employed [19].

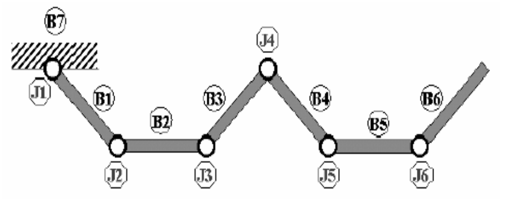

Fig. 1. Serial link mechanism with seven links. Bi(i=1~7) and Jj(j=1~6) represent each body and joint respectively.

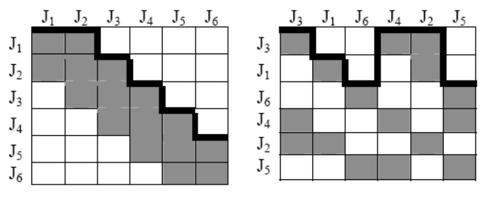

Fig. 2. Pattern comparison of $\Phi_{\mathbf{Z}}\mathbf{M}^{-1}\Phi_{\mathbf{Z}}^{T}$ according to joint sequence. Jj(j=1~6) represents each joint of the serial link mechanism.

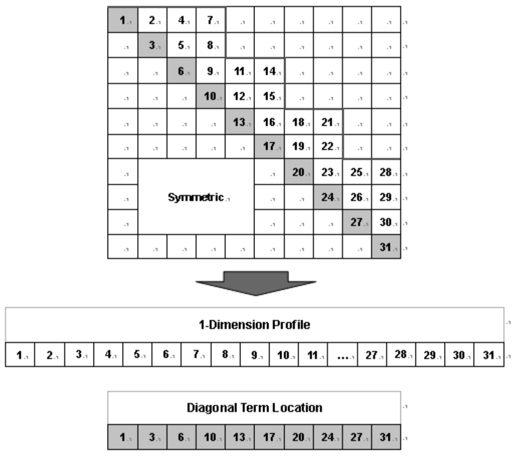

Fig. 3. Storage for the skyline solution method.

Sparse matrix methods such as banded-symmetric and skyline solution method have been introduced in the field of finite element methods. Among them, the skyline solution method is the most efficient for moderately large scaled problems and relatively simple to implement on the computer simulation. The skyline solution method has another merit which requires smaller storage than other methods. Fig. 3 shows the storage for the skyline method. In the figure, the 1-dimension profile array contains the elements below the skyline and the diagonal term location array contains the locations of the diagonal element in the 1-dimension profile array.

Computational implementation of this approach can be carried out as follows.

STEP 0 : Calculate the inverse of each mass matrix $\mathbf{M}_i$ This step should be implemented only once since the mass matrix is constant.

STEP 1 : Calculate $\mathbf{Q}$ and $\boldsymbol{\gamma}$

STEP 2 : Calculate $\Phi_{\mathbf{Z}}\mathbf{M}^{-1}$

STEP 3 : Calculate $\Phi_{\mathbf{Z}}\mathbf{M}^{-1}\Phi_{\mathbf{Z}}^T$

STEP 4 : Calculate $\Phi_{\mathbf{Z}}\mathbf{M}^{-1}\mathbf{Q}$

STEP 5 : Calculate $\Phi_{\mathbf{Z}}\mathbf{M}^{-1}\mathbf{Q}-\boldsymbol{\gamma}$

STEP 6 : Solve $\boldsymbol{\lambda}$ in Eq. (16). The skyline solver is used to solve the equation since the coefficient matrix of the equation is real and symmetric.

STEP 7 : Calculate $\Phi_{\mathbf{q}}^T\boldsymbol{\lambda}$

STEP 8 : Calculate $\mathbf{Q}-\Phi_{\mathbf{q}}^T\boldsymbol{\lambda}$

STEP 9 : Calculate $\dot{\mathbf{Y}}_i=\mathbf{M}_i^{-1}\mathbf{Q}_i^*$

To increase the numerical efficiency, sparse matrix operation techniques are employed in all the steps. For the position and velocity analyses, the same numerical method can be applied since the coefficient matrices are identical.

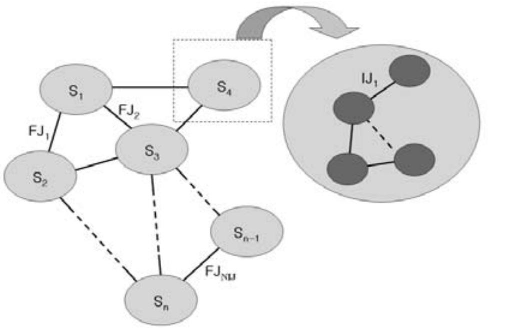

Fig. 4. Mechanical system with n sub-systems.

4. Subsystem partitioning method

Many mechanical systems are too large to be analyzed as a single system. A method to overcome this numerical difficulty is to partition the whole system into smaller units called subsystems. Fig. 4 shows a mechanical system with n sub-systems. In the figure, S_i is the i-th sub-system. For the system with n sub-systems, joints are partitioned into inboard joints IJ, and interface joints FJ. Inboard joints are defined as joints connected with two bodies in a sub-system. Interface joints are defined as joints between adjacent sub-systems. Equations of motion for a constrained system with n sub-systems can be written as

$$\begin{bmatrix} \bar{\mathbf{M}} & \mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJT} \\ \mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\bar{\mathbf{Y}}} \\ \boldsymbol{\lambda}_{FJ} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{Q}} \\ \boldsymbol{\gamma}_{FJ} \end{bmatrix}, \tag{21}$$

where

$$\bar{\mathbf{M}} = \begin{bmatrix} \bar{\mathbf{M}}_{S1} & 0 & \cdots & 0 \\ 0 & \bar{\mathbf{M}}_{S2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{\mathbf{M}}_{Sn} \end{bmatrix}, \tag{22}$$

$$\bar{\mathbf{M}}_{Si} = \begin{bmatrix} \mathbf{M}_{Si} & \mathbf{\Phi}_{\mathbf{Z}_{Si}}^{IJ\ T} \\ \mathbf{\Phi}_{\mathbf{Z}_{Si}}^{IJ} & \mathbf{0} \end{bmatrix}, \tag{23}$$

$$\dot{\bar{\mathbf{Y}}} = \begin{bmatrix} \dot{\bar{\mathbf{Y}}}_{S1}^T & \dot{\bar{\mathbf{Y}}}_{S2}^T & \cdots & \dot{\bar{\mathbf{Y}}}_{Sn}^T \end{bmatrix}^T, \tag{24}$$

$$\dot{\bar{\mathbf{Y}}}_{Si}^T = \begin{bmatrix} \dot{\mathbf{Y}}_{Si}^T & \boldsymbol{\lambda}_{Si}^T \end{bmatrix}^T, \quad (i=1,2,\cdots,n), \tag{25}$$

$$\mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ} = \begin{bmatrix} \mathbf{\Phi}_{\mathbf{Z}_{S1}}^{FJ} & \mathbf{0} & \mathbf{\Phi}_{\mathbf{Z}_{S2}}^{FJ} & \mathbf{0} & \cdots & \mathbf{\Phi}_{\mathbf{Z}_{Sn}}^{FJ} & \mathbf{0} \end{bmatrix}, \tag{26}$$

$$\bar{\mathbf{Q}} = \left[\bar{\mathbf{Q}}_{S1}^T \quad \bar{\mathbf{Q}}_{S2}^T \quad \cdots \quad \bar{\mathbf{Q}}_{Sn}^T\right]^T, \tag{27}$$

$$\bar{\mathbf{Q}}_{Si} = \left[\mathbf{Q}_{Si}^T \quad \gamma_{Si}^{IJ\,T}\right]^T, \quad (i = 1,2,\cdots,n). \tag{28}$$

In Eqs. (21)-(28), subscript *FJ* denotes the interface joint between sub-systems and *IJ* denotes the inboard joint.

The first equation in Eq. (21), $\dot{\bar{\mathbf{Y}}}$, is represented as

$$\dot{\bar{\mathbf{Y}}} = \bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}}^* \ or \ \dot{\bar{\mathbf{Y}}}_{Si} = \bar{\mathbf{M}}_{Si}^{-1}\bar{\mathbf{Q}}_{Si}^* \quad (i = 1,2,\cdots,n), \tag{29}$$

where

$$\bar{\mathbf{Q}}^* = \bar{\mathbf{Q}} - \mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ\,T}\boldsymbol{\lambda}_{FJ}. \tag{30}$$

Substitution of Eq. (29) into the second equation in Eq. (21) gives the following equation.

$$\Phi_{\bar{\mathbf{Z}}}^{*FJ}\bar{\mathbf{M}}^{-1}\left(\Phi_{\bar{\mathbf{Z}}}^{*FJ}\right)^T\boldsymbol{\lambda}_{FJ} = \Phi_{\bar{\mathbf{Z}}}^{*FJ}\bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}} - \gamma_{FJ}. \tag{31}$$

Eq. (9) can be re-written to solve $\Delta\mathbf{Z}$ and $\Delta\boldsymbol{\mu}$ as

$$\begin{bmatrix} \bar{\mathbf{M}} & \left(\mathbf{\Phi}_{\bar{\mathbf{Z}}^*}^{*FJ}\right)^T \\ \mathbf{\Phi}_{\bar{\mathbf{Z}}^*}^{*FJ} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\bar{\mathbf{Z}} \\ \Delta\boldsymbol{\mu}_{IJ} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{R}}_1 \\ \bar{\mathbf{R}}_2 \end{bmatrix}, \tag{32}$$

where

$$\bar{\mathbf{Z}} = \left[\bar{\mathbf{Z}}_{S1}^T \quad \bar{\mathbf{Z}}_{S2}^T \quad \cdots \quad \bar{\mathbf{Z}}_{Sn}^T\right]^T, \tag{33}$$

$$\bar{\mathbf{Z}}_{Si} = \left[\mathbf{Z}_{Si}^T \quad \left(\boldsymbol{\mu}_{Si}^{IJ}\right)^T\right]^T, \tag{34}$$

$$\bar{\mathbf{R}}_1 = \left[\left(\bar{\mathbf{R}}_1\right)_{S1}^T \quad \left(\bar{\mathbf{R}}_1\right)_{S2}^T \quad \cdots \quad \left(\bar{\mathbf{R}}_1\right)_{Sn}^T\right]^T, \tag{35}$$

$$\left(\bar{\mathbf{R}}_1\right)_{Si} = \begin{bmatrix} -\left\{\mathbf{M}_{Si}\left(\mathbf{Z}_{Si} - \mathbf{Z}_{Si}^{init.}\right) + \left(\mathbf{\Phi}_{\mathbf{Z}_{Si}}^{IJ}\right)^T\boldsymbol{\mu}_{IJ}\right\} \\ -\mathbf{\Phi}_{Si}^{IJ} \end{bmatrix}, \tag{36}$$

$$\mathbf{R}_2 = -\mathbf{\Phi}^{JF}. \tag{37}$$

Eq. (12) can be re-written to solve for $\bar{\mathbf{Y}}$, $\boldsymbol{\mu}'_{FJ}$. Writing these conditions in matrix and vector form, we obtain

$$\begin{bmatrix} \bar{\mathbf{M}} & \left(\mathbf{\Phi}_{\bar{\mathbf{Z}}^*}^{*FJ}\right)^T \\ \mathbf{\Phi}_{\bar{\mathbf{Z}}^*}^{*FJ} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \bar{\mathbf{Y}} \\ \boldsymbol{\mu}'_{FJ} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{R}}'_1 \\ \bar{\mathbf{R}}'_2 \end{bmatrix}, \tag{38}$$

where

$$\bar{\mathbf{Y}} = \left[\bar{\mathbf{Y}}_{S1}^T \quad \bar{\mathbf{Y}}_{S2}^T \quad \cdots \quad \bar{\mathbf{Y}}_{Sn}^T\right]^T, \tag{39}$$

$$\bar{\mathbf{Y}}_{Si} = \left[\mathbf{Y}_{Si}^T \quad \left(\boldsymbol{\mu}'^{IJ}_{Si}\right)^T\right]^T, \tag{40}$$

$$\bar{\mathbf{R}}'_1 = \left[\left(\bar{\mathbf{R}}'_1\right)_{S1}^T \quad \left(\bar{\mathbf{R}}'_1\right)_{S2}^T \quad \cdots \quad \left(\bar{\mathbf{R}}'_1\right)_{Sn}^T\right]^T, \tag{41}$$

$$\left(\mathbf{R}'_1\right)_{Si} = \begin{bmatrix} \mathbf{M}_{Si}\mathbf{Y}_{Si}^{init.} \\ -\left(\mathbf{\Phi}_{Si}^{IJ}\right)_t \end{bmatrix}, \tag{42}$$

$$\mathbf{R}_2 = -\mathbf{\Phi}^{FJ}. \tag{43}$$

Computational implementation can the proposed substructure partitioning method may be carried out as follows.

STEP 0 : Calculate $\bar{\mathbf{M}}_{Si}^{-1}$ using skyline solution method

STEP 1 : Calculate $\bar{\mathbf{Q}}$ and γ_{FJ}

STEP 2 : Calculate $\mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ}\bar{\mathbf{M}}^{-1}$

STEP 3 : Calculate $\mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ}\bar{\mathbf{M}}^{-1}\left(\mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ}\right)^T$

STEP 4 : Calculate $\mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ}\bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}}$

STEP 5 : Calculate $\mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ}\bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}} - \gamma_{FJ}$

STEP 6 : Solve $\boldsymbol{\lambda}_{FJ}$ in Eq. (31). The skyline solver is used to solve the Eq. since the coefficient matrix of the Eq. is real and symmetric.

STEP 7 : Calculate $\left(\mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ}\right)^T\boldsymbol{\lambda}_{FJ}$

STEP 8 : Calculate $\bar{\mathbf{Q}} - \left(\mathbf{\Phi}_{\bar{\mathbf{Z}}}^{*FJ}\right)^T\boldsymbol{\lambda}_{FJ}$

STEP 9 : Calculate $\dot{\bar{\mathbf{Y}}}_{Si} = \bar{\mathbf{M}}_{Si}^{-1}\bar{\mathbf{Q}}_{Si}^*$

Velocity and position analyses can be carried out in the same manner of acceleration analysis since the coefficients matrices have identical patterns.

5. Numerical examples

To demonstrate the efficiency and validity of the local parametrization method with the skyline solver, numerical examples are presented in this section. Open loop and closed loop mechanical systems are solved to validate the method. A commercial program is used to show the numerical accuracy of the proposed method. Numerical efficiency of the proposed method is compared to that of the conventional local parametrization method. The comparison for numerical efficiency is made for the serial link mechanisms. The computational times for the problems are compared as the number of links and joints increases.

The validity of subsystem partitioning method is verified by comparing numerical results of the commercial program and proposed method. For numerical analysis, numerical parameters are shown in Table 1. In the table, NR error denotes the error tolerance of the Newton-Raphson method for the position analysis.

Transient analysis is performed for the serial seven link mechanism shown in Fig. 5. Fig. 6 shows the angular acceleration of the 7th link. In the figure, "COMMERCIAL PROGRAM", "PROPOSED LOCAL PARAMETRIZATION", and "CONVENTIONAL LOCAL PARAMETRIZATION" denote numerical results by commercial program, proposed local parametrization method, and conventional local parametrization method. From the numerical results, it is known that the result of the proposed local parametrization method is almost identical to those of the other methods. To show numerical efficiency of the proposed local parametrization method, integration time for the proposed method is compared with that of the conventional methods. Table 2 shows the comparison of computational times. From the numerical results, it is known that proposed method is more efficient than the conventional method.

Table 1. Numerical parameters for dynamic analysis.

Integrator	integration error	NR error	reporting step
RKF45	1.0E-4	1.0E-3	5.0E-2 (sec)

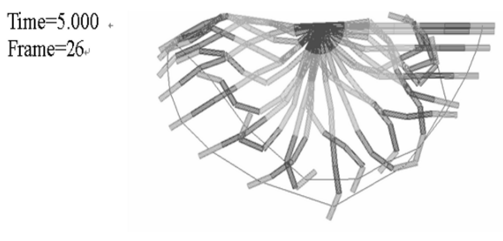

Fig. 5. Configuration of the serial link mechanism at each time step.

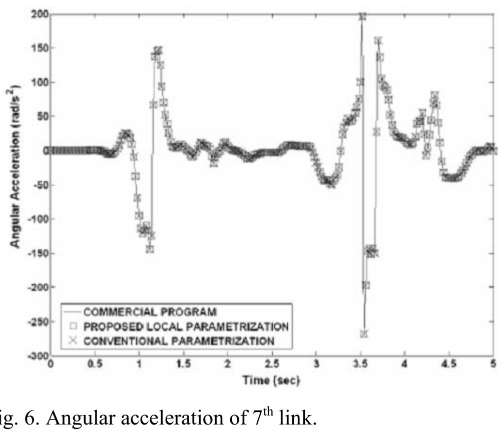

Fig. 6. Angular acceleration of 7th link.

Fig. 7 shows the configuration of the four-bar mechanism at each time step. Numerical analysis is performed for the four-bar mechanism with one closed loop. To show the numerical accuracy of the proposed local parametrization method, accelerations along the x and y directions of follower are compared as shown in Figs. 8 and 9. The results by the proposed method agree with the results of the other methods. To check the numerical efficiency of the proposed method, the integration time for the proposed method is compared with that of the conventional method. Table 3 shows the comparison of the dimension of the factorized matrix and solving time. From the results, it is known that the skyline solution method is more efficient that the dense linear solution method for the closed loop system.

Table 2. Numerical efficiency for open-loop systems.

Method	Size of coefficient matrix	Solving Time	Time ratio
Proposed	nc=14	390(msec)	1.00
Conventional	3nbd+nc= 21+14	1160(msec)	2.97

nc : Number of holonomic constraints
nbd : Number of Bodies (ground body is excluded)

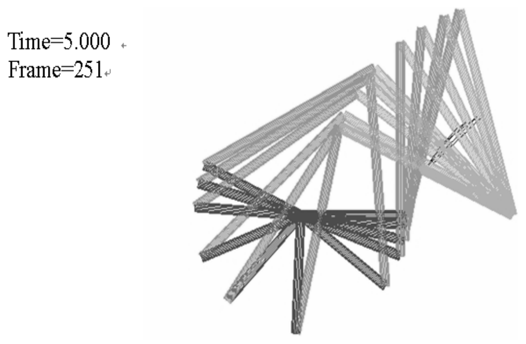

Fig. 7. Configuration of four-bar mechanism at each time step.

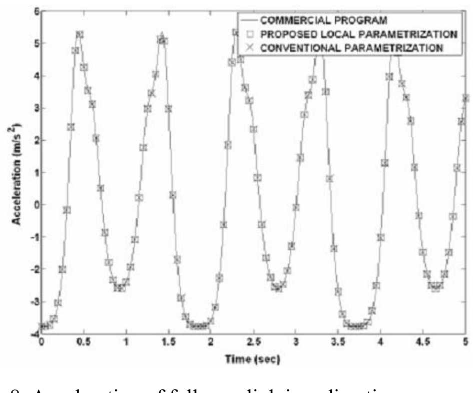

Fig. 8. Acceleration of follower link in x direction.

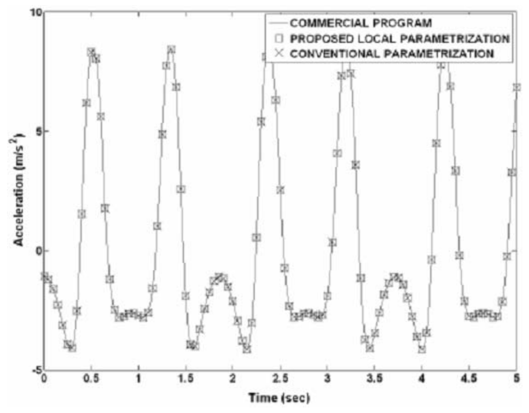

Fig. 9. Acceleration of follower link in y direction.

Table 3. Numerical efficiency for close-loop systems.

Method	size of coefficient matrix	Solving time	Time ratio
Proposed	nc=8	110 (msec)	1
Conventional	3nbd+nc=9 + 8	240 (msec)	2.18

nc : Number of holonomic constraints
nbd : Number of Bodies (ground body is excluded)

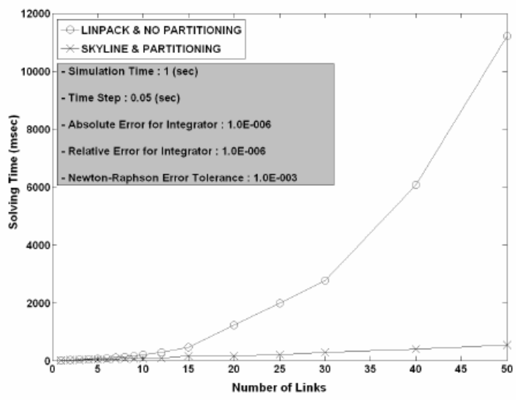

Fig. 10. Numerical efficiency according to the number of links.

To show the numerical efficiency of the skyline solution method, transient analyses for serial link mechanism are performed. All the links are connected by revolute joints. Fig. 10 shows the computing time for dynamic analyses according to the number of links. In the figure, the red circles show the computational times for the proposed method with skyline solution method and blue asterisks show those for the conventional local parametrization method. From the numerical results, the proposed method is more efficient than the conventional method as the number of links increases.

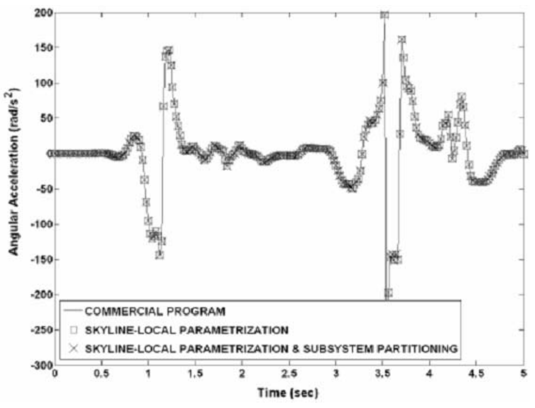

Fig. 11. Angular acceleration of the 7th Link.

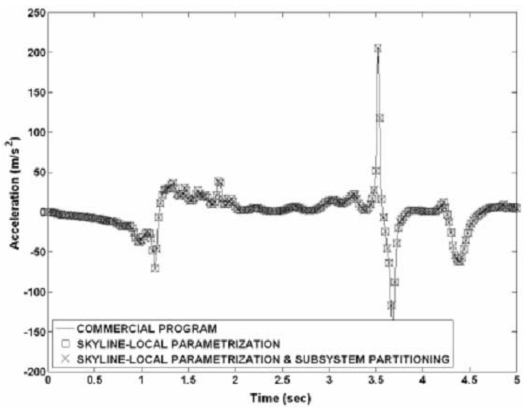

Fig. 12. Acceleration of the 7th link in x direction.

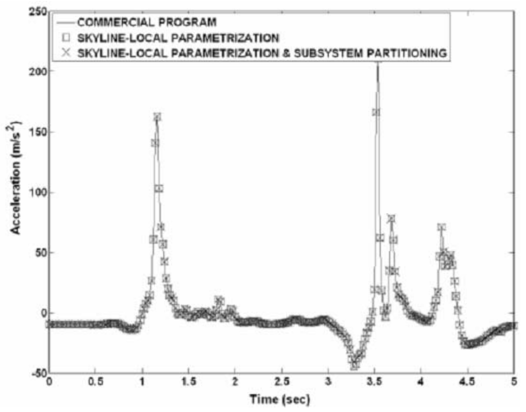

Fig. 13. Acceleration of the 7th link in y direction.

A serial link mechanism is considered to show the validity of the subsystem partitioning method with the local parametrization and skyline solution methods. The mechanical system considered in this section is partitioned into two subsystems. Each subsystem includes three revolute joints and three bodies except for a grounded body. J4 shown in figure 1 is the interface joint between two subsystems. Figs. 10, 11 and 12 show numerical results. From the results, the subsystem partitioning method yields accurate numerical results.

6. Conclusion

This paper proposes an improved DAE solution method which employs the subsystem partitioning method, modified local parametrization method, and skyline solution method.

The proposed method employs the local parametrization method to apply the same numerical solution method for position, velocity, and acceleration equations. Furthermore, local the parametrization method is modified to diminish the size of linear equations. Skyline solution method makes the proposed method more efficient by alleviating the burden when solving linear equations. The numerical efficiency and validity for the proposed methods are verified through the numerical examples. Subsystem partitioning method is derived systematically to perform parallel processing for the real time simulation.

In future work, hybrid coordinates will be employed to perform numerical integration more efficiently. In the hybrid coordinate methods, Cartesian coordinates are used for ease of deriving equations, and relative coordinates are used for efficient numerical integration [11]. Parallel processing technique will be employed to take advantage of the subsystem partitioning method.

Acknowledgments

This work was supported by Seoul Research & Business Development Program (Grant 10583) and the research program 2007 of Kookmin University in Korea.

References

[1] E. J. Haug, Computer aided kinematics and dynamics of mechanical systems, Volume I: Basic Methods, Allyn and Bacon, (1989) 199-280.

[2] E. J. Haug, 1992, Intermediate Dynamics, Prentice-Hall, 335-345.

[3] P. E. Nikravesh, 1988, Computer-Aided Analysis of Mechanical Systems, Prentice-Hall, 6-14.

[4] ADAMS User Manual, 2002, Ver. 12.0, MDI, USA.

[5] DADS Reference Manual, 1996, rev. 8.5, CADSI Inc.

[6] S. S. Kim and M. J. Vanderploeg, 1984, A state space formulation for multibody dynamic systems subject to control, University of Iowa, Iowa City, Iowa, Dec.

[7] D. S. Bae and E. J. Haug, 1986, A recursive formulation for constrained mechanical system dynamics, Technical Report 86-3, University of Iowa, Iowa City, Iowa, Feb.

[8] RecurDyn/Theoretical Background, http://www.functionbay.co.kr.

[9] SIMPACK/What's SIMPACK, http://www.simpack.co.kr.

[10] J. H. Park, Dynamic analysis of constrained multibody systems using hybrid coordinates, ACMD (2004) 361-367.

[11] R. Serban, D. Negrut, F. A. Potra and E. J. Haug, Topology based linear solver for exploiting sparsity in multibody dynamics, (1995).

[12] S. H. Lee, A geometric kinematic and dynamic analysis of multibody systems, Master Dissertation, Hanyang University, Ansan, (1993).

[13] G. Quaranta, P. Masarati and P. Mantegazza, Speeding up multibody analysis by parallel computing, CAPI 2000, Sep. (2000).

[14] K. J. Song and S. S. Kim, Subsystem synthesis methods with independent coordinates for multibody dynamics systems, *J. of KSME*, (2003) 1724-1729.

[15] D. L. Logan, A First Course in the Finite Element Method, BROOKS/COLE, (2002) 643-646.

[16] T. J. R. Hughes, The finite element method, linear static and dynamic finite element analysis, Prentice-Hall, (1987) 631-636.

[17] L. O. Chua and P. M. Lin, Computer-Aided Analysis of Electronic Circuits: Algorithm and Computational Techniques, Prentice-Hall, (1975).

[18] J. S. Arora, 1989, Introduction to Optimum Design, McGraw-Hill.

[19] S. W. Sloan, An algorithm for profile and wave front reduction of sparse matrices, *International Journal for Numerical Methods in Engineering*, 23 (4) (1986) 239-251.