

Path defined directed graph vector (*Pgraph*) method for multibody dynamics

Musa Nurullah Yazar¹ · S. Murat Yesiloglu¹

Received: 21 June 2016 / Accepted: 21 September 2017 / Published online: 6 October 2017
© Springer Science+Business Media B.V. 2017

Abstract Dynamical modeling of complex systems may include multiple combinations of serial and tree topologies, as well as closed kinematic chain systems. Simulating of such systems is a costly task not only for run time but also for construction time when a model description and constraints are introduced. This paper presents a systematic framework of a construction time efficient modeling methodology for complex systems by introducing a path defined directed graph vector (*Pgraph*) and the associated methodology based on Linear Graph Theory (LGT). Updating of body coordinate frame vectors which can be a challenge in complex topology systems is easily performed using the proposed method. This technique is especially useful for systems changing topology, such as walking mechanisms where bilateral and unilateral constraints are conditionally embedded. Although this is a general methodology, to be combined with many dynamical modeling algorithms available in the literature, here we demonstrate it using Spatial Operator Algebra (SOA), which is a recursive algorithm based on the Newton–Euler formalism.

Keywords Multibody dynamics · Construction time efficiency · Linear graph theory · Constraint embedding · Complex topology · Serial chain · Tree chain · Closed-chain · Open-chain · Model description · Spatial operator algebra · Newton–Euler formalism · Recursive computational algorithm

1 Introduction

During the mechanical design phase of a multibody system to achieve a desired motion, bodies are interconnected forming different topologies. Complex multibody systems are those that include multiple combinations of serial and tree topologies, as well as closed-chain systems. Simulating of such systems based on high fidelity modeling is a challenging and

✉ M.N. Yazar
mnyazar@itu.edu.tr

S.M. Yesiloglu
yesiloglu@itu.edu.tr

¹ Istanbul Technical University, Istanbul, Turkey

costly task not only for run time but also for construction time when a model description and constraints are introduced. Nevertheless, obtaining accurate yet computationally efficient models of multibody systems is crucial for model-based controller design processes. Without a systematic framework, however, modeling and formulating multibody dynamics can be too tedious depending on the complexity of the system.

Complex topology systems are likely to have a high number of degree of freedom. A number of computationally efficient programs were developed in the last couple of decades to analyze and simulate the dynamics of multibody systems dealing with a large size mass matrix whose inverse is computed in $O(n)$ complexity [1–3, 5, 6, 15, 18]. Almost all of these programs include graph-theoretical approaches to the definition of the system topology. Among them Linear Graph Theory (LGT) has become a fundamental tool for solving complex systems. LGT is initially introduced by Euler to get the solution to the well-known “Königsberg bridge” problem in 1736. Since then, this approach has been widely used for modeling different physical systems. In 1977, Wittenburg [22] introduced a formulation to obtain the equation of motion of a complex multibody system. Later, McPhee et al. [14] stated that Wittenburg formulation can be considered in the scope of LGT. McPhee et al. also combined LGT with virtual work and symbolic computing methods to model a wide range of multibody systems [19–21]. In this regard, we want to emphasize Spatial Operator Algebra (SOA) [7, 10–13, 16, 17] on which this paper is based. Jain has combined SOA with LGT to develop methods for partitioning, aggregating and constraint embedding for multibody systems [8, 9].

All of these studies are done with a particular approach with some focus on optimizing run time. However, the construction time efficiency is one of the major current challenges in multibody dynamics, especially in the case of event-dependent changes in constraints as well as topology. Construction time also includes the effort it takes to set rules for updating of body coordinate frame vectors, which can be error prone.

In terms of describing the connectivity of the bodies, one of the important works done in this area was proposed by Featherstone [4], where undirected graphs describe the connectivity of the bodies by a parent array. Although this method is useful for describing the connectivity of the bodies, it uses also additional two arrays which represent the set of children of related body and the set of joints on the path between related body and root.

In this paper, we propose the *Pgraph* method which is used for describing the connectivity of the bodies and branches based on LGT. Among the significant advantages of this method, obtaining spatial force and spatial velocity distributions as well as updating body coordinate frames in a systematic way should be emphasized. Furthermore, *Pgraph* is used to incorporate the constraint forces into the equation systematically.

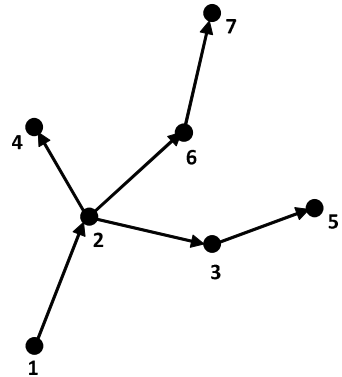
In this paper, we utilize Spatial Operator Algebra (SOA) with a minor difference. Although the original SOA uses an inboard numbering scheme where the numeration of the bodies is done in increasing order from the tip to the base, we use an outboard numbering scheme (from the base to the tip) to be compatible with common literature.

This paper is organized as follows. Section 2 is on the basics of Linear Graph Theory (LGT), Sect. 3 provides the mathematical basis of SOA with LGT. Section 4 presents the *Pgraph* method. In Sect. 5, simulation results are presented. Finally, concluding remarks are given in Sect. 6.

2 Basics of linear graph theory (LGT)

A graph consists of nodes and edges at least one of which is subject to multiple branches as shown in Fig. 1. If the edges have directions, it is called a “directed graph” (digraph), where

Fig. 1 A directed graph with seven nodes



each edge defines parent/child relationship between sequential nodes. A root is the first node and a tip is the last node. Therefore, the root has no parent, while the tip has no child.

Figure 1 demonstrates an example of a digraph with seven nodes. First node is the root node, and fourth, fifth and seventh nodes are the tip nodes of the digraph. The parent node of the j th node is denoted $\rho(j)$. In this example, $\rho(2) = 1, \rho(5) = 3, \rho(4) = 2$ and so on. In the so-called canonical tree structure type digraph, the index of a parent node is always smaller than that of its child node. Two sequential nodes are defined as adjacent if they are connected to each other by an edge. The adjacency matrix is an $n \times n$ matrix that shows the connectivity of the nodes. The element in the i th row and the j th column of the adjacency matrix is 1, if and only if the i th node is the child of the j th node. Otherwise, it is 0.

The adjacency matrix of a digraph, \mathbb{S} , can be expressed as a sum of tensors in the following form:

$$\mathbb{S} = \sum_{j=2}^n \sum_{\kappa \in \rho(j)} \mathbf{e}_j \mathbf{e}_\kappa^T \tag{1}$$

where \mathbf{e}_i is the i th column of the $n \times n$ identity matrix. For the digraph given in Fig. 1, $\mathbf{e}_i (i=1,2,\dots,7)$ vectors are defined as

$$\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \dots, \quad \mathbf{e}_7 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \tag{2}$$

Hence

$$\mathbb{S} = \mathbf{e}_2 \mathbf{e}_1^T + \mathbf{e}_3 \mathbf{e}_2^T + \mathbf{e}_4 \mathbf{e}_2^T + \mathbf{e}_5 \mathbf{e}_3^T + \mathbf{e}_6 \mathbf{e}_2^T + \mathbf{e}_7 \mathbf{e}_6^T, \tag{3}$$

$$\mathbb{S} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \quad (4)$$

The adjacency matrix, \mathbb{S} , contains only one zero row corresponding to the root node and a zero column for each tip node.

3 Spatial operator algebra (SOA) with linear graph theory (LGT)

A full distribution of forces and torques in multibody system can be achieved in a Newton–Euler-based dynamical model. This makes Newton–Euler a better choice when coupling forces and torques matter as in the case of a controller design based on a reaction force. Most importantly, this has to be done in a computationally efficient and systematic way to make the method applicable to complex systems.

Spatial Operator Algebra (SOA) [9, 10] is a computationally efficient recursive algorithm to obtain systematic formulation of the dynamical model of multibody systems which may include serial, tree and closed topologies. SOA is based on Newton–Euler formalism and the full distribution of forces and torques in the multibody system is obtained with sparse operator matrices in a compact form. The algorithm is flexible for topology changes in multibody systems and capable of dealing with constraints. Spatial Kernel Operator (SKOs) and Spatial Propagation Operator (SPOs) represent the distribution of accelerations/forces throughout the bodies of a multibody system. SOA originally uses an inboard numbering scheme where the numeration of the bodies is done in increasing order from the tip to the base. For the sake of being compatible with common literature, we adopt the outboard numbering scheme to SOA as is done in [23].

Let us consider Fig. 2 for representing a multibody system including seven rigid bodies connected by various type of joints in a tree topology structure. Within the scope of

Fig. 2 A multibody system including seven rigid bodies

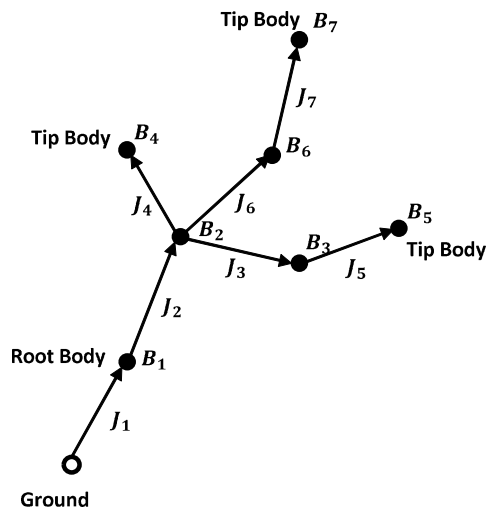
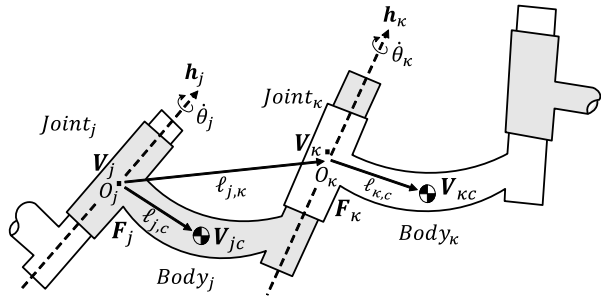


Fig. 3 Defined vectors for sequential bodies in a multibody system



multibody system modeling, nodes and edges are replaced by body and joint notations, respectively. In Fig. 2, the first body (\mathbf{B}_1), which is the one connected to ground by the first joint (\mathbf{J}_1), is called the root body, \mathbf{B}_4 , \mathbf{B}_5 , \mathbf{B}_7 are the last bodies, which have no child body, they are called tip bodies. If \mathbf{B}_2 has more than one child body it is called the separation body.

We use a six-dimensional manifold to represent angular and linear motion of a rigid body. Therefore, a velocity vector in this manifold represents both angular and linear velocities called the “spatial velocity” vector. Within the same manifold the “spatial force” vector is also defined to represent a torque and force pair acting on the same point. The equation of motion of a multibody system can be obtained recursively in a three-sweep algorithm;

- Compute the velocity propagations from the root to tips (outboard).
- Compute the force propagations from the tips to root (inboard).
- Compute the acceleration propagations from the root to tips (outboard).

Figure 3 introduces the defined vectors for sequential bodies in a multibody system:

- \mathbf{h}_j and \mathbf{h}_k are axis of rotation/translation vectors in 3D space,
- O_j and O_k are the coordinate frame centers that are placed on bodies along the joints,
- \mathbf{V}_j and \mathbf{V}_k are spatial velocity vectors at the coordinate frame centers,
- $\mathbf{V}_{j,c}$ and $\mathbf{V}_{k,c}$ are spatial velocity vectors at the mass centers,
- \mathbf{F}_j and \mathbf{F}_k are spatial force vectors at the coordinate frame centers,
- $\ell_{j,k}$, is the link vector between the sequential parent and child body coordinate frames,
- $\ell_{j,c}$ and $\ell_{k,c}$ are the link vectors between the body coordinate frames and body centers.

At this point of the paper, let us introduce a superscript–subscript notation. Let α_c indicate a variable, named α , which belongs to the body c . To express the dimension of a variable, when necessary, we use left-subscript. If a variable α_c is of dimension $n \times m$, then we denote it as ${}_{n \times m}\alpha_c$. If its dimension is $b \times b$, we simply denote it as ${}_b\alpha_c$.

In the presence of an one rotational degree of freedom joint between the sequential bodies, the velocity propagation from the parent to child body can be written as

$$\mathbf{V}_k = \phi_{k,j} \mathbf{V}_j + \mathbf{H}_k \dot{\theta}_k, \tag{5}$$

$${}_6\phi_{k,j} = \begin{bmatrix} \mathbf{3I} & \mathbf{30} \\ -\ell_{j,k} \times & \mathbf{3I} \end{bmatrix}, \quad {}_{6 \times 1}\mathbf{V}_j = \begin{bmatrix} \omega_j \\ \mathbf{v}_j \end{bmatrix}, \quad {}_{6 \times 1}\mathbf{H}_k = \begin{bmatrix} \mathbf{h}_k \\ \mathbf{0} \end{bmatrix}, \tag{6}$$

where $\phi_{k,j}$ is a link propagation matrix. $(\ell_{j,k} \times)$ is the skew symmetric matrix of the link vector between the body coordinate frames. \mathbf{H}_k is the axis of motion vector for a one degree of freedom joint. $\dot{\theta}_k$ is the time derivative of the joint variable. It should be noted that the propagation equation can be written for a q degree of freedom joint, using ${}_{6 \times q}\mathbf{H}_j$. For a

translational joint, all we have to do is to change \mathbf{H}_κ in (5) as

$$\mathbf{H}_\kappa = \begin{bmatrix} \mathbf{0} \\ \mathbf{h}_\kappa \end{bmatrix}. \tag{7}$$

In order to obtain the dynamical model of a multibody system, we need acceleration terms. To get the acceleration propagation between sequential bodies, we need to take the time derivative of the velocity propagation given in (5) and use the $j = \rho(\kappa)$ parent–child relation as shown in Fig. 3. This yields

$$\dot{\mathbf{V}}_\kappa = \phi_{\kappa,j} \dot{\mathbf{V}}_j + \mathbf{H}_\kappa \ddot{\theta}_\kappa + \mathbf{a}_\kappa, \tag{8}$$

where

$$\mathbf{a}_\kappa = \begin{bmatrix} \omega_j \times \omega_\kappa \\ \omega_j \times (\omega_j \times \ell_{j,\kappa}) \end{bmatrix}. \tag{9}$$

Although velocity and acceleration propagations are outboard; from the root body to a tip body, force propagation is inboard; from the tip body towards the root body. Torque propagation from child to parent body can be written as

$$\tau_j = \tau_\kappa + \ell_{j,\kappa} \times \mathbf{f}_\kappa + \ell_{j,c} \times \dot{\mathbf{v}}_j m_j + \frac{d}{dt} (\mathcal{I}_j \omega_j), \tag{10}$$

where the first two terms come from the child body and the last two terms are due to change in the parent body’s linear and angular momentum, respectively. Force propagation from child to parent body can be written as

$$\mathbf{f}_j = \mathbf{f}_\kappa + m_j \frac{d}{dt} (\mathbf{v}_j + \omega_j \times \ell_{j,c}), \tag{11}$$

and (10) and (11) can be written in the matrix form

$$\mathbf{F}_j = \phi_{\kappa,j}^T \mathbf{F}_\kappa + M_j \dot{\mathbf{V}}_j + \mathbf{b}_j, \tag{12}$$

where

$$\mathbf{F}_\kappa = \begin{bmatrix} \tau_\kappa \\ \mathbf{f}_\kappa \end{bmatrix}, \quad M_j = \begin{bmatrix} \mathcal{I}_j & m_j \ell_{j,c} \times \\ -m_j \ell_{j,c} \times & 3 \times 3 \mathbf{I} m_j \end{bmatrix}, \quad \mathbf{b}_j = \begin{bmatrix} \omega_j \times \mathcal{I}_j \omega_j \\ m_j \omega_j \times (\omega_j \times \ell_{j,c}) \end{bmatrix}.$$

Consider the multibody system shown in Fig. 2. In order to apply the propagations systematically on the branches of the digraph, an adjacency matrix is expanded to block the weighted adjacency matrix to include the link propagation matrices. Adjacency matrices have 1 or 0 scalar entries for the sequential nodes in a digraph, in the case of block weighted adjacency matrices, 1 or 0 scalar entries are expanded to a 6×6 dimensional unit matrix or zero matrix and SKO (Spatial Kernel Operator) is obtained by using the following equation:

$$\varepsilon_\phi = \sum_{\kappa=2}^n \sum_{j \in \rho(\kappa)} \mathbf{e}_\kappa \phi_{\kappa,j} \mathbf{e}_j^T, \tag{13}$$

where \mathbf{e}_i is a $6n \times 6$ dimensional matrix which contains all zero matrix entries except for the i th block which is 6×6 dimensional unit matrix. For the system given in Fig. 2, the \mathbf{e}_i

($i = 1, 2, \dots, 7$) matrices are defined as

$$\mathbf{e}_1 = \begin{bmatrix} \mathbf{6I} \\ \mathbf{60} \\ \mathbf{60} \\ \mathbf{60} \\ \mathbf{60} \\ \mathbf{60} \end{bmatrix}, \quad \mathbf{e}_2 = \begin{bmatrix} \mathbf{60} \\ \mathbf{6I} \\ \mathbf{60} \\ \mathbf{60} \\ \mathbf{60} \\ \mathbf{60} \end{bmatrix}, \quad \dots, \quad \mathbf{e}_7 = \begin{bmatrix} \mathbf{60} \\ \mathbf{60} \\ \mathbf{60} \\ \mathbf{60} \\ \mathbf{60} \\ \mathbf{6I} \end{bmatrix}. \tag{14}$$

Therefore

$$\begin{aligned} \boldsymbol{\varepsilon}_\phi &= \mathbf{e}_2\phi_{2,1}\mathbf{e}_1^T + \mathbf{e}_3\phi_{3,2}\mathbf{e}_2^T + \mathbf{e}_4\phi_{4,2}\mathbf{e}_2^T + \mathbf{e}_5\phi_{5,3}\mathbf{e}_3^T + \mathbf{e}_6\phi_{6,2}\mathbf{e}_2^T + \mathbf{e}_7\phi_{7,6}\mathbf{e}_6^T, \tag{15} \\ \boldsymbol{\varepsilon}_\phi &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \phi_{2,1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \phi_{3,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & \phi_{4,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \phi_{5,3} & 0 & 0 & 0 & 0 \\ 0 & \phi_{6,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \phi_{7,6} & 0 \end{bmatrix}. \end{aligned}$$

The SPO matrix which includes n bodies is obtained using the lemma in [17],

$$\Phi = (\mathbf{6nI} - \boldsymbol{\varepsilon}_\phi)^{-1}. \tag{16}$$

Using the $6n \times 6n$ dimensional Φ SPO matrix, all spatial velocities of the bodies in a multi-body system can be obtained:

$$\mathbf{6n} \times \mathbf{1} \mathbf{V} = \Phi \mathbf{H} \dot{\boldsymbol{\theta}}, \tag{17}$$

where

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \vdots \\ \mathbf{V}_n \end{bmatrix}, \quad \mathbf{6n} \times \mathbf{n} \mathbf{H} = \begin{bmatrix} \mathbf{H}_1 & & & \\ & \mathbf{H}_2 & & \\ & & \ddots & \\ & & & \mathbf{H}_n \end{bmatrix}.$$

The propagation matrix Φ_T is employed to calculate the velocities of tip points of multi-body system. For a system which has n_t tip bodies, the propagation matrix Φ_T is of dimension $6n_t \times 6n$. For the system in Fig. 2, the ${}^i\Phi_T$ matrix is given as

$${}^i\Phi_T = \begin{bmatrix} \mathbf{60} & \mathbf{60} & \mathbf{60} & \phi_{i,4} & \phi_{i,5} & \mathbf{60} & \phi_{i,7} \end{bmatrix}. \tag{18}$$

Hence, tip point velocity of an open-chain is

$$\mathbf{V}_T = \Phi_T \Phi \mathbf{H} \dot{\boldsymbol{\theta}} = \mathbf{J} \dot{\boldsymbol{\theta}}, \tag{19}$$

where \mathbf{J} is the Jacobian operator defined as

$$\mathbf{J} = \Phi_T \Phi \mathbf{H}. \tag{20}$$

Spatial accelerations are obtained by taking the derivative of (17);

$$\dot{\mathbf{V}} = \Phi \mathbf{H} \ddot{\boldsymbol{\theta}} + \mathbf{a}. \tag{21}$$

Spatial forces are distributed inboard (from tip towards to root node) as

$$\mathbf{F} = \Phi^T (\mathbf{M}\dot{\mathbf{V}} + \mathbf{b} + \Phi_T^T \mathbf{F}_T), \tag{22}$$

where

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \vdots \\ \mathbf{F}_n \end{bmatrix}, \quad \mathbf{M} = \begin{bmatrix} \mathbf{M}_1 & & & \\ & \mathbf{M}_2 & & \\ & & \ddots & \\ & & & \mathbf{M}_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_n \end{bmatrix}.$$

\mathbf{F}_T is for the $6n_t \times 1$ dimensional external spatial forces that are applied to tip points.

Applied torques or forces are the projection of the body spatial forces along the axis of motion at the joints. This is stated as

$$\tau = \mathbf{H}^T \mathbf{F}. \tag{23}$$

Therefore, the inverse dynamics of a multibody system is obtained:

$$\tau = \mathcal{M}\ddot{\theta} + \mathbf{C} + \mathbf{J}^T \mathbf{F}_T, \tag{24}$$

where

$$\begin{aligned} \mathcal{M} &= \mathbf{H}^T \Phi^T \mathbf{M} \Phi \mathbf{H}, \\ \mathbf{C} &= \mathbf{H}^T \Phi^T (\mathbf{M} \Phi \mathbf{a} + \mathbf{b}). \end{aligned} \tag{25}$$

$\mathcal{M} \in \mathbb{R}^{n \times n}$ is a generalized mass matrix for a multibody system, $\mathbf{C} \in \mathbb{R}^n$ is the velocity dependent vector includes Coriolis and gravity terms.

The equation of motion regarding the forward dynamics is obtained using (24).

$$\ddot{\theta} = \mathcal{M}^{-1} (\tau - \mathbf{C} - \mathbf{J}^T \mathbf{F}_T) \tag{26}$$

4 Path defined directed graph vector (*Pgraph*) method

In the previous sections, the general concept of LGT (Linear Graph Theory) and canonical numeration of digraph were explained. Here we need to pay particular attention to (15) which deals with the interconnection of the bodies in a given system. Without a systematic approach, this equation needs to be manually implemented with a particular attention to connectivity of the bodies. On top of this error prone tedious work, then it comes to updating of coordinate frame vectors which may become a challenging task depending on the complexity of the topology of the system. To avoid all of these problems, we introduce a systematic approach under the name of *Pgraph* method. *Pgraph* is an automatic model configuration generator for multibody systems to provide a simplified standard structure for modeling as well as visualization with a significant advantage especially in construction time. Proposed *Pgraph* algorithm is given by the flow chart in Fig. 4, where SBn is the separation body number, TBn is the tip (terminal) body number, SBS is the last-in-first-out (LIFO) type separation body stack, jdof is the joint degree of freedom vector and dof(i) is the degree of freedom between body i and its parent body.

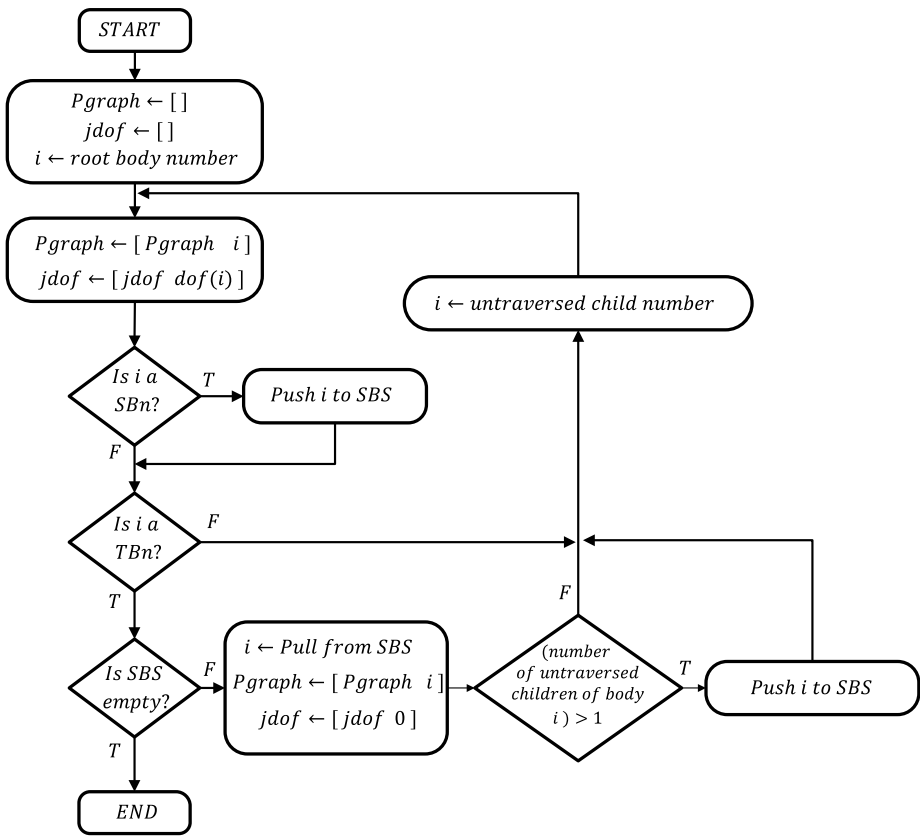


Fig. 4 Flow chart of *Pgraph* algorithm

The flow chart can be summarized in following steps.

1. Create an empty *Pgraph* and *jdof* vectors and assign the root body number to index *i*.
2. Add index number to *Pgraph* and determine the joint DoF between the body *i* and body *i* – 1 (the first element of *jdof* represents the joint DoF between the root body and ground).
3. If body *i* is a separation body then push the index *i* to SBS.
4. If body *i* is a tip body then go to Step 5, otherwise assign an untraversed child body number to index *i*, and go to Step 2.
5. If the SBS is empty then go to Step 7, otherwise pull the index number from SBS and assign it to *i*, add this index value to *Pgraph* and add zero element to *jdof* vector. If separation body *i* has more than one untraversed child body push the index number to SBS back again.
6. Assign an untraversed child body number to index *i*, and go to Step 2.
7. End of the algorithm.

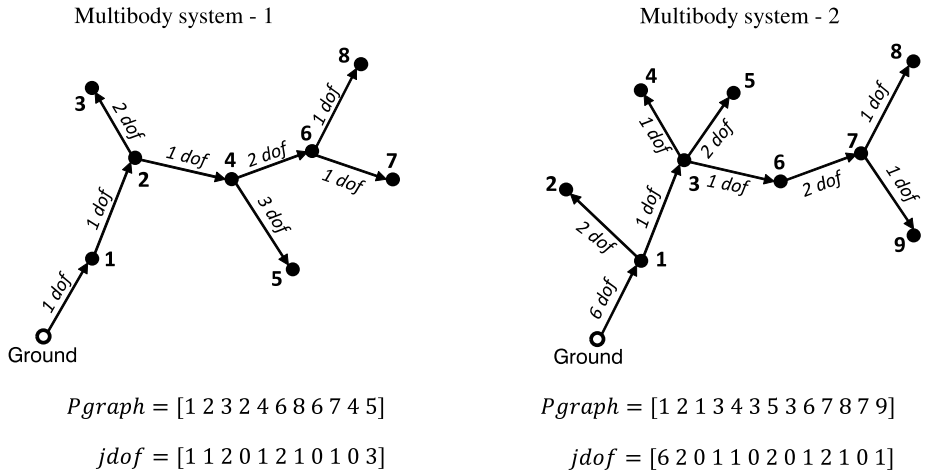


Fig. 5 Example multibody systems to demonstrate *Pgraph* and *jdof* vectors

Pgraph may not be unique for the same multibody system, since one can choose the untraversed child of a separation body in Step 4. In this paper, left child priority has been adopted. Two examples are presented in Fig. 5 to demonstrate *Pgraph* and *jdof* vectors.

In Newton–Euler dynamics, creating a propagation matrix plays a vital role to define the propagation of both spatial velocities and spatial forces in a multibody system. Once *Pgraph* is obtained, (15) can now be constructed systematically as in the flow chart in Fig. 6. This flowchart makes the implementation of the presented multibody dynamics algorithm straightforward for computer programming.

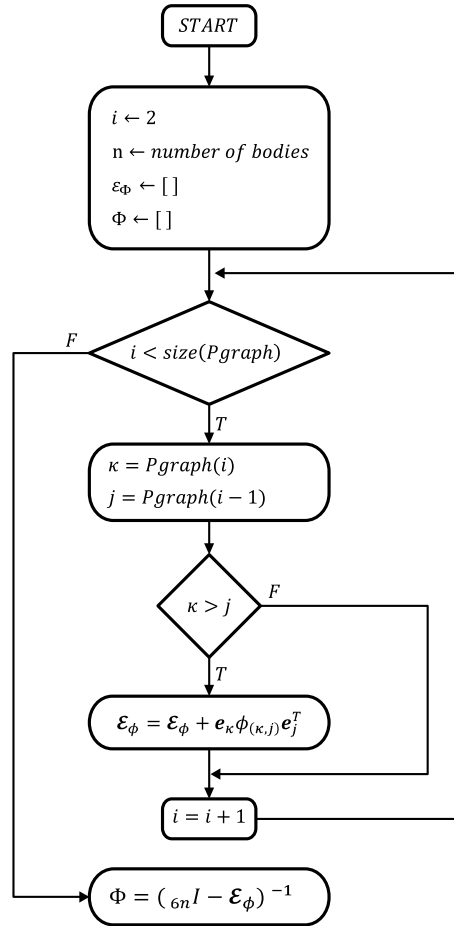
During the run time of a simulation, axis of rotation and link vectors need to be updated. However, updating body coordinate frames of a complex topology system can be a complicated task depending on the complexity of the topology. Another important contribution of the proposed *Pgraph* method is a systematic solution, introduced by the algorithm whose flow chart is given in Fig. 7, to this problem.

Updating body coordinate frame algorithm, shown in Fig. 7, employs the following functions:

- $Rot(\mathbf{h}, \Delta\theta)_\kappa$ is the rotation matrix function based on Rodriguez formulation: $Rot(\mathbf{h}, \Delta\theta)_\kappa = I + (\mathbf{h}_\kappa \times) \sin \Delta\theta_\kappa + (\mathbf{h}_\kappa \times)^2 (1 - \cos \Delta\theta_\kappa)$.
- The $find(Pgraph, Pgraph(\kappa))$ function returns the index of the first place of the *Pgraph*(κ) element in *Pgraph*.
- P is a list structure containing the body indices that were inserted to it. $F(P)$, on the other hand, is a function that returns the body coordinate frame vectors of the bodies whose index numbers are in the P list.

Updating steps are demonstrated on a multibody system as an example in Fig. 8.

Fig. 6 Flowchart for creating SPO matrix



4.1 Model description interface and constraint embedding

When modeling of multibody systems we do not envision people to write the *Pgraph*. Rather, we envision people to fill a simple table and let *Pgraph* and *jdof* vectors be created automatically. After all, the *Pgraph* method is all about multibody system modeling with short construction time.

The name of the body, the name of the parent body, DoF, and the constraint type are the columns of this four column table. But, before this table, the first thing interface asks is the total number of bodies in the model. Once the users enters this number, the four column table gets automatically created with assumptions that the bodies are subject to holonomic constraints, bodies are connected with a single DoF joint in the form of a serial topology and the names of the bodies are sequential numbers starting from 1. Together with this table, digraph representation of the system is displayed on the screen. All the

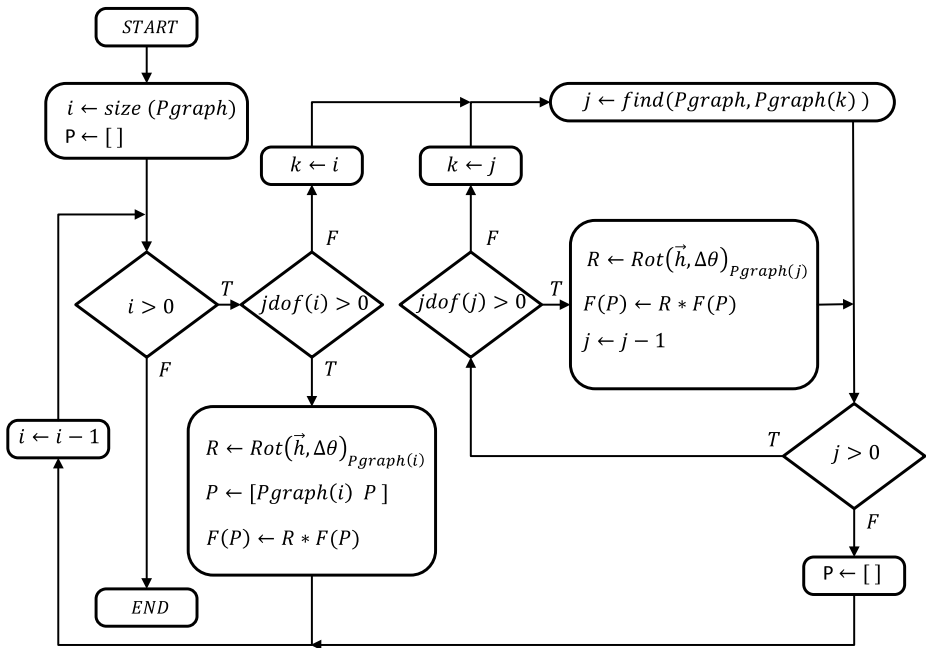


Fig. 7 Flowchart for updating body coordinate frames

link vectors, axes of rotations, masses, and inertias are set at the default values. At this point the user may change the name of the bodies using alphanumeric characters, select a parent body from the body column (the first column) and change the number of DoF. The user can click on the bodies on the digraph representation to open a properties window to set the initial values of the axes of rotations and link vectors, mass and inertia matrix.

This interface is the foreground of the *Pgraph* method. At the background, *Pgraph* method-based multibody modeling software (MMS) will do the actual numbering of the bodies and the construction of the *Pgraph* and *jdof* vectors. Accordingly, MMS will obtain the propagation matrix, Jacobian matrix, generalized mass matrix and others. Updating of body coordinate frames will also be done using the *Pgraph* method.

This interface also provides a separate list so that a user can define the constraints. The last column of the table is updated using this list of constraints. From the digraph representation display, by clicking on the bodies, the user can define conditions on these constraints. As closed kinematic chains, or closed topology systems, are particularly important in multibody dynamics from the force propagation point of view, dealing with such a system, we need to define cut-joints to decompose the closed topology into serial or tree topology system and then impose related constraints. Cut-joint motion constraints can represent both bilateral and unilateral constraints forces. Bilateral constraint forces can be categorized mainly into holonomic and nonholonomic motion constraints and they are constantly involved in the equation of motion. Unilateral constraint forces

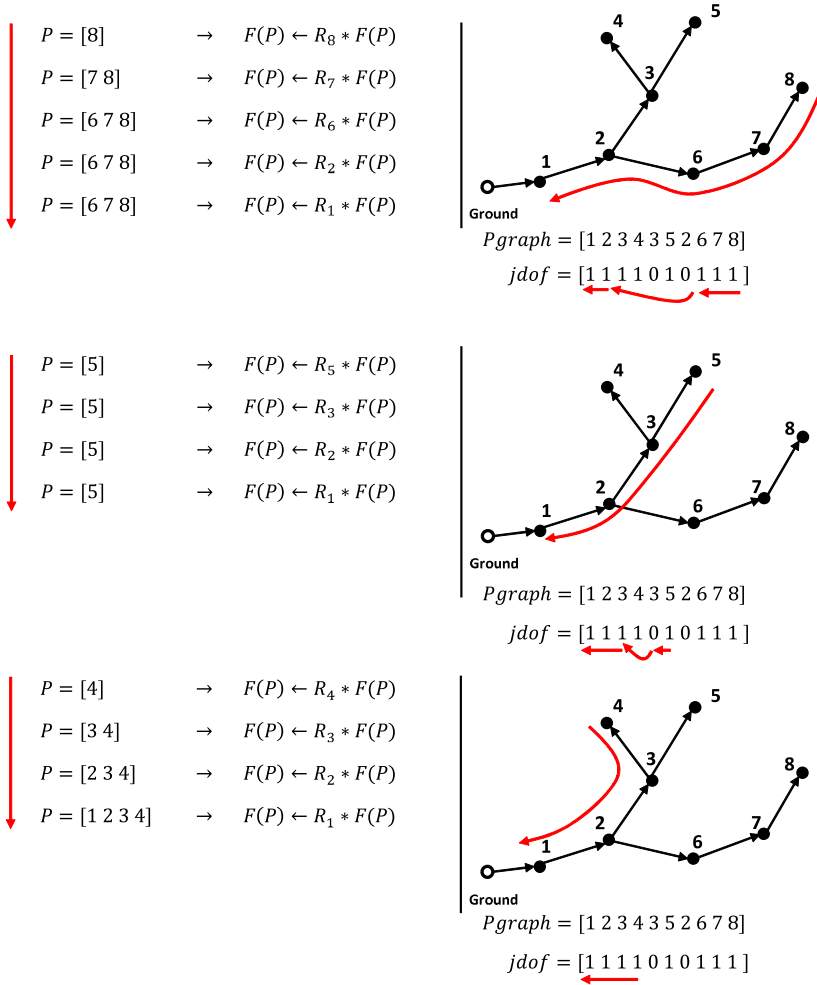


Fig. 8 Flowchart for updating body coordinate frames

can occur in the case of body to body contact, body to ground contact or joint limits and these constraint forces are intermittently involved in the equation of motion. The main approaches for handling such constraints are soft contact, penalty and complementarity methods.

In the next section, simulations are performed for a cheetah-like multibody system including open and closed topology subsystems depending on the gait cycle, which is divided into swing, weight acceptance and stance phases. During the swing phase of the gait cycle, the multibody system includes an open topology in the form of serial and tree structures. During the weight acceptance phase when the tip bodies are contacted with ground, Coulomb friction and impact forces are introduced until the linear velocity at the contact point becomes zero. Next is the stance phase where multibody system becomes of closed topology due to the existence of closed kinematic chain, and ground contact forces are computed using holonomic constraints. Since a sticky type ground is not considered here, it is assumed that the ground cannot pull the foot. In other words, ground contact forces can only

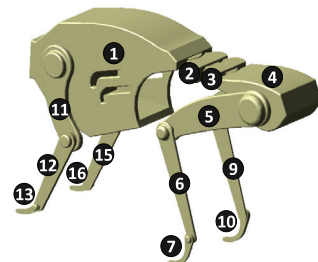
act in one direction, hence, they are unilateral. On the other hand, this force is only valid in the, so called, ground friction cone. This means that in the case the applied force from foot to ground is outside of the reversed of this cone, the ground friction force will remain on the border of the cone, causing them not to be equal and opposite to each other. Consequently, the foot will slide on the ground.

5 Simulations

Figure 9 shows the P_{graph} and j_{dof} vectors on 3D model of the quadruped and the parts of the model are given in Table 1.

As a case study, only the stance phase of the gait cycle of quadruped is considered so that simulation results can be compared with ADAMS software to verify. Simulation results of the other phases are not suitable for comparing since ADAMS uses graphical contact detection models, which is not in the scope of this paper. Constraint forces during the stance phase of the gait cycle are defined for the tip points of the quadruped to keep them at zero

Fig. 9 3D CAD model of quadruped



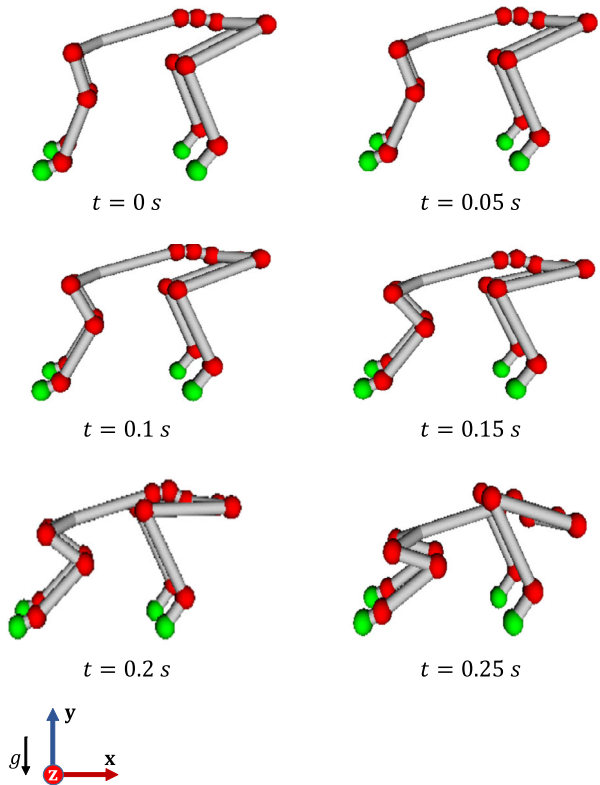
$$p_{graph} = [1\ 2\ 3\ 4\ 5\ 6\ 7\ 4\ 8\ 9\ 10\ 1\ 11\ 12\ 13\ 1\ 14\ 15\ 16]$$

$$j_{dof} = [6\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 1]$$

Table 1 Quadruped parts

Body number	Parts	Body number	Parts
1	Torso	11	Left shoulder
2	Spine	12	Left forearm
3	Spine	13	Left front paw
4	Hip	14	Right shoulder
5	Left thigh	15	Right forearm
6	Left hind leg	16	Right front paw
7	Left back paw		
8	Right thigh		
9	Right hind leg		
10	Right paw		

Fig. 10 Simulation frames (0–0.25 s)



velocity by the following equation:

$$\begin{aligned} \mathbf{V}_T &= J\dot{\theta} = \mathbf{0}, \\ \dot{\mathbf{V}}_T &= J\ddot{\theta} + \dot{J}\dot{\theta} = \mathbf{0}. \end{aligned} \tag{27}$$

Multiplying the forward dynamics equation with J leads to

$$J\ddot{\theta} = JM^{-1}(\mathcal{T} - \mathbf{C} - J^T \mathbf{F}_T). \tag{28}$$

Plugging (28) into (27), the constraint forces can be solved:

$$\begin{aligned} 0 &= JM^{-1}(\mathcal{T} - \mathbf{C} - J^T \mathbf{F}_T) + \dot{J}\dot{\theta}, \\ JM^{-1}J^T \mathbf{F}_T &= JM^{-1}(\mathcal{T} - \mathbf{C}) + \dot{J}\dot{\theta}, \\ \mathbf{F}_T &= \Omega^{-1}(JM^{-1}(\mathcal{T} - \mathbf{C}) + \dot{J}\dot{\theta}), \end{aligned} \tag{29}$$

where

$$\Omega = JM^{-1}J^T.$$

The 3D CAD model of the quadruped robot shown in Fig. 9 is formed in the solid commercial package CATIA v5 and it is imported to ADAMS software for analyzing and comparing

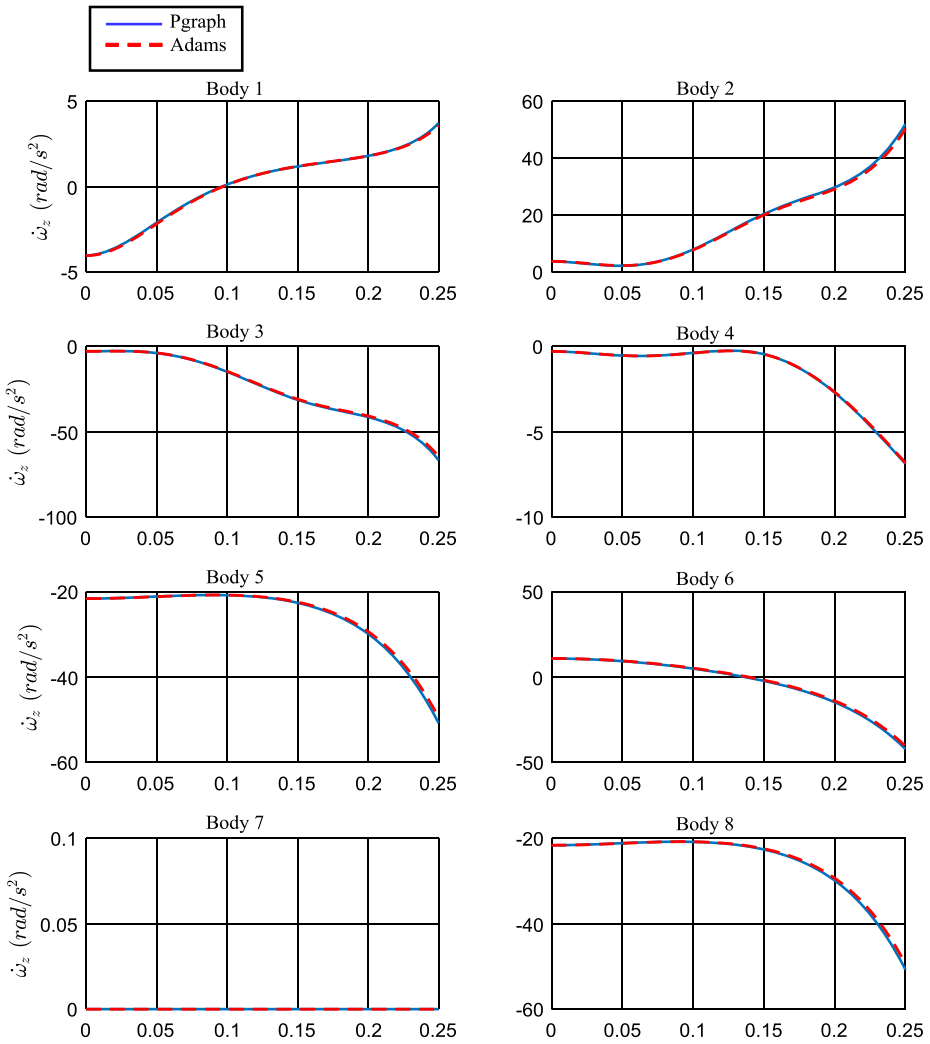


Fig. 11 Angular accelerations of the bodies around z direction (Body 0–Body 8)

the dynamical simulation results. The maximum difference between the two results (the one obtained using ADAMS and the one obtained using *Pgraph*-based MMS) is less than 4.0% as shown in Figs. 11 and 12.

Simulation results including tip point forces are obtained for 0.25 seconds. Angular accelerations of the bodies around z direction are shown in Figs. 11 and 12, comparing results obtained using ADAMS and *Pgraph*-based multibody modeling software (MMS). Figure 10 displays the pictorial representation of the configuration of the system in 50 ms intervals. Tip point forces for the left back paw are given in Fig. 13.

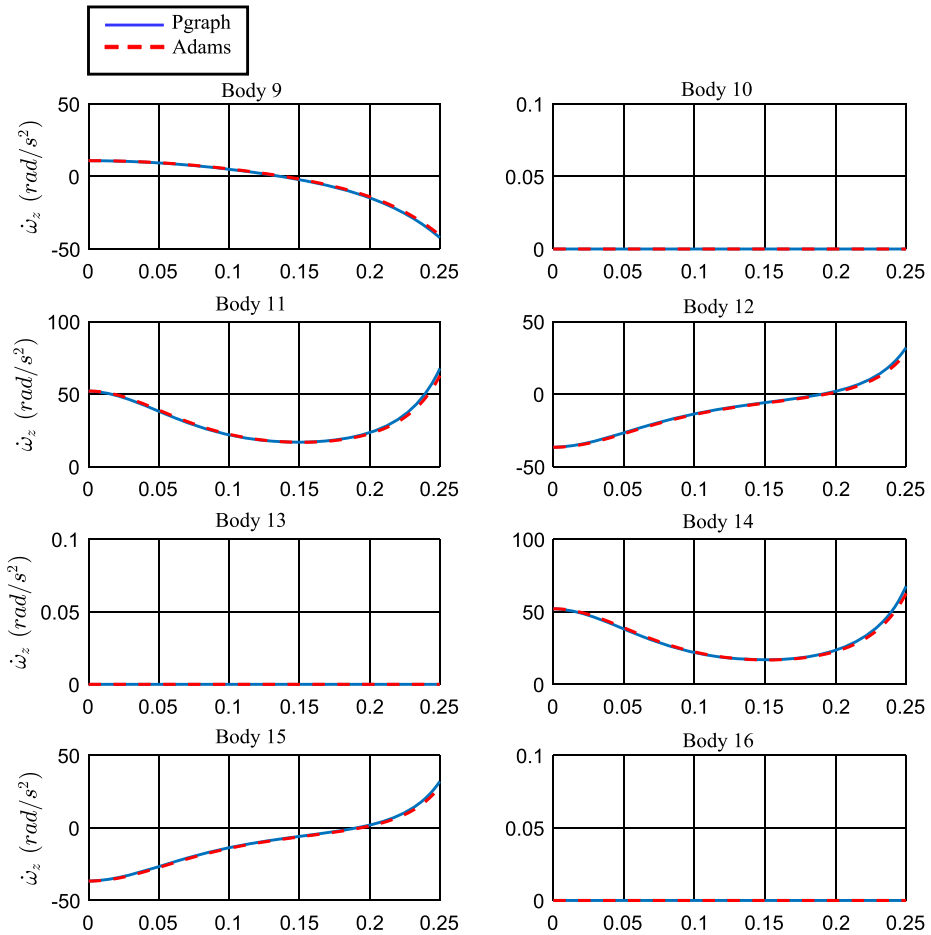


Fig. 12 Angular accelerations of the bodies around *z* direction (*Body 9–Body 16*)

6 Conclusions

The presented method of a path defined digraph vector (*Pgraph*) is a novel approach for the construction time efficient dynamical modeling of complex systems which may include multiple combinations of serial and tree topologies, as well as closed kinematic chain systems. Propagation matrix for spatial velocity and spatial force that plays a pivotal role in Newton–Euler-based multibody modeling can be obtained conveniently using the *Pgraph* method. Similarly, systematic updating of body coordinate frame vectors, which can be a challenge in complex topology systems, is easily performed using the proposed method. Combining this technique with a model description interface and constraint embedding, the construction time of complex multibody system modeling is greatly reduced. This technique is especially useful for systems changing topology, such as a walking mechanisms where bilateral and unilateral constraints are conditionally embedded.

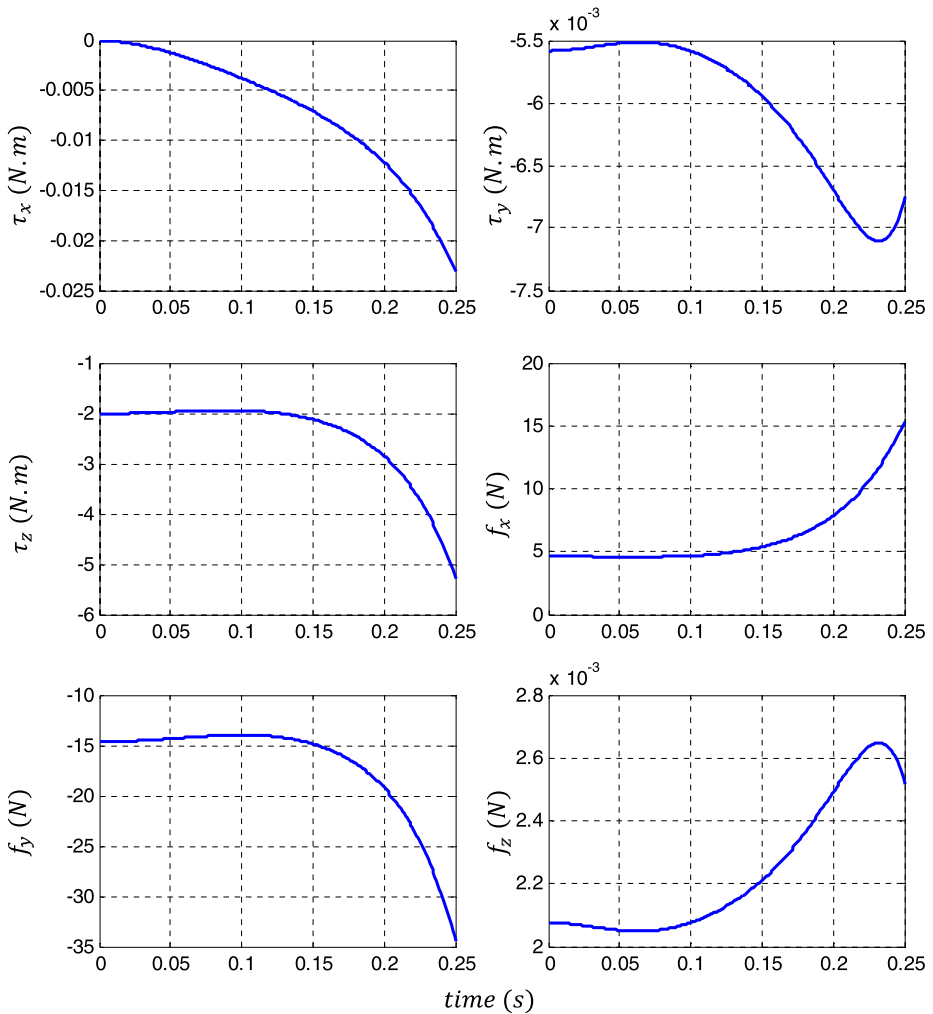


Fig. 13 Tip point reaction forces from the ground on left back paw

References

1. Anderson, K.: An order n formulation for the motion simulation of general multi-rigid-body constrained systems. *Comput. Struct.* **43**(3), 565–579 (1992)
2. Baraff, D.: Linear-time dynamics using lagrange multipliers. In: *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 137–146. ACM, New York (1996)
3. Featherstone, R.: The calculation of robot dynamics using articulated-body inertias. *Int. J. Robot. Res.* **2**(1), 13–30 (1983)
4. Featherstone, R.: A beginner's guide to 6-d vectors (part 2) [tutorial]. *IEEE Robot. Autom. Mag.* **4**(17), 88–99 (2010)
5. Featherstone, R., Orin, D.: Robot dynamics: equations and algorithms. In: *ICRA*, pp. 826–834 (2000)
6. Fijany, A., Featherstone, R.: A new factorization of the mass matrix for optimal serial and parallel calculation of multibody dynamics. *Multibody Syst. Dyn.* **29**(2), 169–187 (2013)
7. Jain, A.: Unified formulation of dynamics for serial rigid multibody systems. *J. Guid. Control Dyn.* **14**(3), 531–542 (1991)
8. Jain, A.: *Robot and Multibody Dynamics: Analysis and Algorithms*. Springer, New York (2010)

9. Jain, A.: Graph theoretic foundations of multibody dynamics. *Multibody Syst. Dyn.* **26**(3), 335–365 (2011)
10. Jain, A.: Multibody graph transformations and analysis. *Nonlinear Dyn.* **67**(4), 2779–2797 (2012)
11. Jain, A., Rodriguez, G.: Recursive flexible multibody system dynamics using spatial operators. *J. Guid. Control Dyn.* **15**(6), 1453–1466 (1992)
12. Jain, A., Rodriguez, G.: An analysis of the kinematics and dynamics of underactuated manipulators. *IEEE Trans. Robot. Autom.* **9**(4), 411–422 (1993)
13. Jain, A., Rodriguez, G.: Linearization of manipulator dynamics using spatial operators. *IEEE Trans. Syst. Man Cybern.* **23**(1), 239–248 (1993)
14. McPhee, J.J., Ishac, M.G., Andrews, G.C.: Wittenburg's formulation of multibody dynamics equations from a graph-theoretic perspective. *Mech. Mach. Theory* **31**(2), 201–213 (1996)
15. Murray, R.M., Li, Z., Sastry, S.S., Sastry, S.S.: *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Boca Raton (1994)
16. Richard, M., Bouazara, M., Therien, J.: Analysis of multibody systems with flexible plates using variational graph-theoretic methods. *Multibody Syst. Dyn.* **25**(1), 43–63 (2011)
17. Rodriguez, G., Jain, A., Kreutz-Delgado, K.: A spatial operator algebra for manipulator modeling and control. *Int. J. Robot. Res.* **10**(4), 371–381 (1991)
18. Rodriguez, G., Jain, A., Kreutz-Delgado, K.: Spatial operator algebra for multibody system dynamics. *J. Astronaut. Sci.* **40**(1), 27–50 (1992)
19. Schmitke, C., Morency, K., McPhee, J.: Using graph theory and symbolic computing to generate efficient models for multi-body vehicle dynamics. *J. Multi-Body Dyn.* **222**(4), 339–352 (2008)
20. Shi, P., McPhee, J.: Dynamics of flexible multibody systems using virtual work and linear graph theory. *Multibody Syst. Dyn.* **4**(4), 355–381 (2000)
21. Shi, P., McPhee, J.: Symbolic programming of a graph-theoretic approach to flexible multibody dynamics. *Mech. Struct. Mach.* **30**, 123–154 (2002)
22. Wittenburg, J.: *Dynamics of Systems of Rigid Bodies*, vol. 33. Springer, New York (1977)
23. Yesiloglu, S.M., Temeltas, H.: Dynamical modeling of cooperating underactuated manipulators for space manipulation. *Adv. Robot.* **24**(3), 325–341 (2010)