Multibody Syst Dyn (2007) 17:291–319
DOI 10.1007/s11044-007-9044-8

A recursive, numerically stable, and efficient simulation algorithm for serial robots

Ashish Mohan · S. K. Saha

Received: 9 May 2006 / Accepted: 25 January 2007

Abstract Traditionally, the dynamic model, i.e., the equations of motion, of a robotic system is derived from Euler–Lagrange (EL) or Newton–Euler (NE) equations. The EL equations begin with a set of generally independent generalized coordinates, whereas the NE equations are based on the Cartesian coordinates. The NE equations consider various forces and moments on the free body diagram of each link of the robotic system at hand, and, hence, require the calculation of the constrained forces and moments that eventually do not participate in the motion of the coupled system. Hence, the principle of elimination of constraint forces has been proposed in the literature. One such methodology is based on the Decoupled Natural Orthogonal Complement (DeNOC) matrices, reported elsewhere. It is shown in this paper that one can also begin with the EL equations of motion based on the kinetic and potential energies of the system, and use the DeNOC matrices to obtain the independent equations of motion. The advantage of the proposed approach is that a computationally more efficient forward dynamics algorithm for the serial robots having slender rods is obtained, which is numerically stable. The typical six-degree-of-freedom PUMA robot is considered here to illustrate the advantages of the proposed algorithm.

Keywords Simulation · DeNOC matrices · Recursive · Numerical stabile · PUMA

1 Introduction

Comprehensive knowledge and information about the dynamic behavior of a system is essential for its design, simulation, and control. The dynamic analysis of robotic systems requires the derivations of its dynamic model, i.e., the equations of motion. A comprehensive review of various techniques of modeling robotic systems and trends can be found in [1]. A review of some recently published formulations for increasing efficiency of the dynamic simulation of the system is given in [2]. Among the most important criteria for selecting the dynamic

A. Mohan · S. K. Saha (✉)
Mechanical Engineering Department, Indian Institute of Technology Delhi, New Delhi 110016, India
e-mail: saha@mech.iitd.ernet.in

modeling technique are computational efficiency and the numerical stability of the algorithm. It is desired that the order of computations, i.e., number of computational operations required for each integration step size, be linear, i.e., $O(n) - n$ being the degree-of-freedom (DOF) of the serial robot under study. The first $O(n)$ algorithm was developed by Vereshchagin in 1975 [3]. This work marked a significant departure from the traditional formulations requiring $O(n^3)$ complexity. Different researchers have tried to improve the overall computational efficiency through vastly different approaches, namely, based on Newton–Euler equations [4, 5], Lagrangian equations [6], Kane's equations [7–10], and others [11–18]. There is also a parallel algorithm, as reported in [19]. A brief description of different formalizations is provided in Valasek [15], where the methods of direct equations of motion, composite rigid-body inertias, and recursive articulated-body inertias methods are compared. The $O(n)$ algorithms are also proposed by [20–22]. Anderson [21] developed two methods for avoiding formulation singularities as well as a recursive general coupled loop solution that extends the recursive coordinate reduction to the complete set of multibody systems, and has used triangulization, forward substitution, and Gaussian elimination to achieve the efficiency. Chirikjain [22] has used spatial operator algebra to eliminate the constraints equations.

Note that in the study of simulation, numerical stability of the forward dynamics algorithms is an important aspect that should be investigated. The simulation problem consists of two parts: (1) forward dynamics, i.e., the computation of the joint accelerations from the equations of motion of a system at hand for the given actuator forces and torques; and (2) numerical integration, i.e., the computation of the joint velocities and positions from the joint acceleration results obtained in the earlier-referred step (1). The realistic simulation results and the overall simulation speed depend on the numerical stability of the corresponding algorithm. In case the simulation results show instability in a system, it could be due to numerical instability of the algorithm even though the real system is stable. However, the fastest forward dynamics algorithm may not be the best one, especially in conditions close to singularity. Uri et al. [23] have investigated the numerical stability aspects using the formulation stiffness phenomenon. It is shown that a variant of articulated body inertia method is better suited to deal with certain types of ill-conditioning than the composite rigid-body method. Ellis et al. [24] have studied the numerical stability of the forward dynamics algorithms using methods based on energy conservation.

In the present work, an $O(n)$, computationally efficient and numerically stable forward dynamics algorithm is proposed. The algorithm is based on the $\boldsymbol{UDU}^{\mathrm{T}}$ decomposition of the generalized inertia matrix (GIM) associated with the system's dynamic equations of motion, as proposed in [18], where $\boldsymbol{U}$ and $\boldsymbol{D}$ are, respectively, the upper triangular and diagonal matrices. Analytical expression for each scalar element of the matrices, $\boldsymbol{U}$ and $\boldsymbol{D}$, is available due to the use of the decoupled natural orthogonal complement (DeNOC) matrices [17]. Note that an orthogonal complement is the matrix whose columns span the null-space of the matrix of the velocity constraints, and, hence, the premultiplication of its transpose with the unconstrained dynamic equations of motion vanishes the constrained moments and forces. As a result, a set of independent ordinary differential equations (ODE) is obtained, in contrast to the differential algebraic equations (DAE) representing the same system dynamics [25]. The said orthogonal complement is, however, not unique. In some approaches, the orthogonal complement is found using numerical schemes that are computationally intensive, for example, singular value decomposition or eigenvalue computations [26, 27]. Angeles and Lee [28] have obtained the complement for the serial rigid robotic systems naturally from the velocity constraint expressions without any complex computations. Therefore, the complement is called the natural orthogonal complement (NOC). Saha [17] has extended the concept of the NOC by decoupling its representation, called the decoupled natural orthogonal complement

(DeNOC) matrices, and has eventually obtained the independent dynamic equations of motion from which both recursive inverse and forward dynamics algorithms can be derived. Note that the recursive forward dynamics algorithm was not possible with the original form of the NOC. Moreover, the DeNOC-based methodology provides physical interpretations of many terms, e.g., the mass matrix of the composite body, etc. [13, 17, 18]. The DeNOC concept is also successfully used for the modeling of parallel [29], and arbitrary closed-loop systems [30, 31].

In this paper, the DeNOC matrices are used with the EL equations instead with the NE equations, as done earlier in [17, 18, 29, 30]. Such formulation is beneficial because one can treat both rigid and flexible body systems in a unified manner. In this paper, the systems with only rigid bodies are considered. Modeling of systems with rigid-flexible bodies will be reported in a separate communication in order to keep the size of this paper reasonable and not to deviate from the emphasis of this paper, i.e., recursiveness, efficiency, and numerical stability. This paper is organized as follows: In Section 2, some definitions are introduced, followed by the formulation of the dynamic model in Section 3. In Section 4, a recursive $O(n)$ algorithm for the forward dynamics is proposed. The implementational details and the computational complexity of the algorithm are presented in the Section 4.1, and compared with the computational complexity of other algorithms available in the literature. The algorithm is illustrated in Section 4.2 with the help of a spatial 6-DOF PUMA robot. The numerical stability of the proposed algorithm is investigated in Section 5, using the same PUMA robot. The conclusions and discussions are given in Section 6.

2 Some definitions

Figure 1 shows a serial robotic system having a fixed base and n-moving rigid links. The links are numbered #1, ..., #n, which are coupled by n-one-degree-of-freedom kinematic pairs, say, revolute or prismatic, denoted by 1, ..., n. Referring to the motion of the ith link, the following terms are now defined which will be used to derive the dynamic equations of motion of the robotic system at hand:

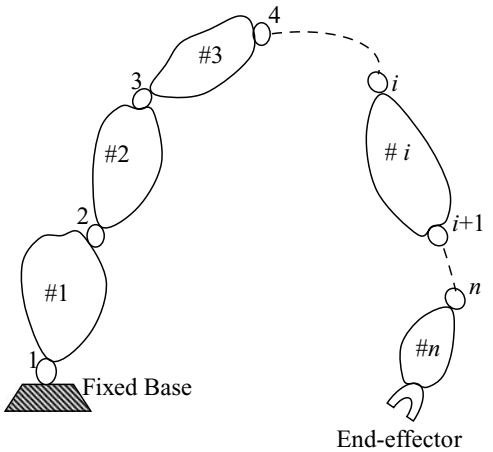


Fig. 1 An n-link serial robotic system

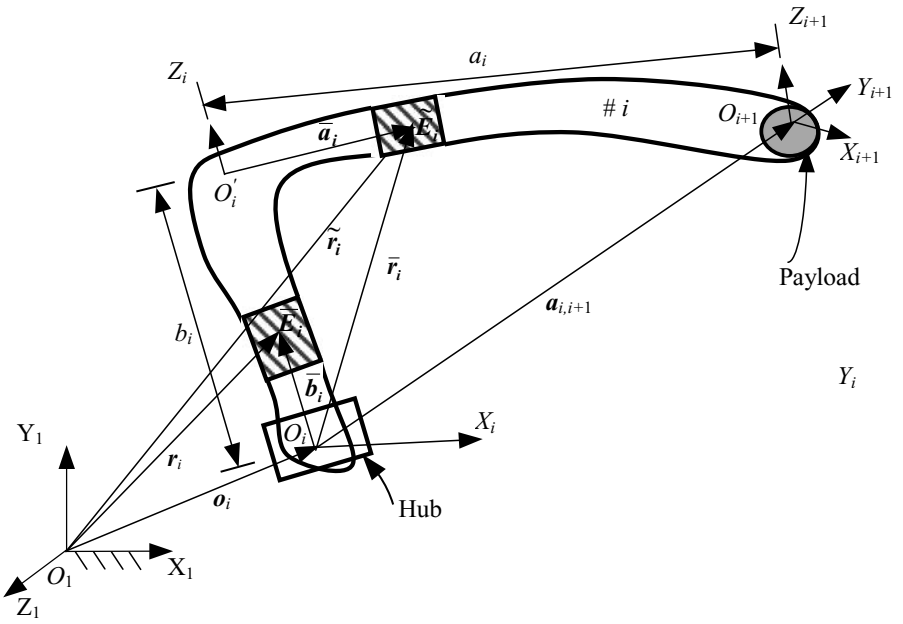


Fig. 2 The *i*th link

- $\boldsymbol{t}_i$ and $\boldsymbol{w}_i$: The 6-dimensional twist and wrench of the *i*th link, respectively, i.e.,

$$\boldsymbol{t}_i \equiv \begin{bmatrix} \boldsymbol{v}_i^{\mathrm{T}} & \boldsymbol{\omega}_i^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}; \quad \boldsymbol{w}_i \equiv \begin{bmatrix} \boldsymbol{f}_i^{\mathrm{T}} & \boldsymbol{n}_i^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{1}$$

  where, $\boldsymbol{v}_i$ and $\boldsymbol{\omega}_i$, are the 3-dimensional vectors of velocity of the origin of the *i*th link, i.e., O_i, of Figure 2, and its angular velocity, respectively. The vectors, $\boldsymbol{f}_i$ and $\boldsymbol{n}_i$, are the 3-dimensional vectors of force at O_i and the moment about O_i of the *i*th link, respectively.
- $\boldsymbol{t}$ and $\boldsymbol{w}$: The $6n$-dimensional vectors of the generalized twist and wrench, respectively, which are defined as

$$\boldsymbol{t} \equiv \begin{bmatrix} \boldsymbol{t}_1^{\mathrm{T}} & \boldsymbol{t}_2^{\mathrm{T}} & \dots & \boldsymbol{t}_n^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}; \quad \boldsymbol{w} \equiv \begin{bmatrix} \boldsymbol{w}_1^{\mathrm{T}} & \boldsymbol{w}_2^{\mathrm{T}} & \dots & \boldsymbol{w}_n^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{2}$$

  where, $\boldsymbol{t}_i$ and $\boldsymbol{w}_i$, for $i = 1, 2, \dots, n$, are given in Equation (1).
- $\dot{\boldsymbol{\theta}}$: The n-dimensional vector of joint rates, i.e.,

$$\dot{\boldsymbol{\theta}} \equiv [\, \dot{\theta}_1 \quad \dot{\theta}_2 \quad \dots \quad \dot{\theta}_n \,]^{\mathrm{T}} \tag{3}$$

  where $\dot{\boldsymbol{\theta}}_i$ is the time-rate of the rotational or translational displacement of the *i*th joint, θ_i, depending on its type, i.e., revolute or prismatic, respectively.

3 Dynamic modeling

In this section, first, the derivation of the decoupled natural orthogonal complement (DeNOC) matrices associated with the velocity constraints of the moving links in a serial-chain robotic system is outlined.

3.1 The DeNOC matrices

Referring to Figures 1 and 2, the steps to obtain the DeNOC matrices [17, 18] are as follows:

(1) The twist of the ith link is expressed in terms of the $(i-1)$st link as

$$\boldsymbol{t}_i = \boldsymbol{A}_{i,i-1}\boldsymbol{t}_{i-1} + \boldsymbol{p}_i\dot{\theta}_i \tag{4}$$

where the 6×6 twist propagation matrix, $\boldsymbol{A}_{i,i-1}$, and the 6-dimensional joint motion propagation vector, $\boldsymbol{p}_i$, are defined as,

$$\boldsymbol{A}_{i,i-1} \equiv \begin{bmatrix} \mathbf{1} & \boldsymbol{a}_{i,i-1}\times\mathbf{1} \\ \boldsymbol{O} & \mathbf{1} \end{bmatrix}; \quad \boldsymbol{p}_i \equiv \begin{bmatrix} \mathbf{0} \\ \boldsymbol{z}_i \end{bmatrix} \text{ for revolute}; \quad \boldsymbol{p}_i \equiv \begin{bmatrix} \boldsymbol{z}_i \\ \mathbf{0} \end{bmatrix} \text{ for prismatic}$$

in which, $\boldsymbol{a}_{i,i-1} \equiv -\boldsymbol{a}_{i-1,i}$, is the position vector of the origin of ith link, i.e., O_i from the $(i-1)$st one, i.e., O_{i-1}. A similarly defined vector, $\boldsymbol{a}_{i,i+1}$, is shown in Figure 2. Moreover, the 3×3 cross-product tensor, $\boldsymbol{a}_{i,i-1}\times\mathbf{1}$, associated with the vector, $\boldsymbol{a}_{i,i-1}$ [30], is defined such that $(\boldsymbol{a}_{i,i-1}\times\mathbf{1})\boldsymbol{x} = \boldsymbol{a}_{i,i-1}\times\boldsymbol{x}$, for any 3-dimensional Cartesian vector $\boldsymbol{x}$. Furthermore, $\boldsymbol{z}_i$ is the 3-dimensional unit vector along the axis of rotation of a revolute joint or along the direction of a prismatic joint, whereas $\boldsymbol{O}$ and $\mathbf{0}$, are, respectively, the 3×3 zero matrix, and the 3-dimensional vector of zeros, and $\mathbf{1}$ is the 3×3 identity matrix.

(2) For $i = 1, \ldots, n$, twist of the ith link is expressed in terms of the twist of the $(i-1)$st link, and consequently in terms of all previous links, as

$$\begin{aligned} \boldsymbol{t}_1 &= \boldsymbol{p}_1\dot{\theta}_1 \\ \boldsymbol{t}_2 &= \boldsymbol{A}_{21}\boldsymbol{t}_1 + \boldsymbol{p}_2\dot{\theta}_2 = \boldsymbol{A}_{21}\boldsymbol{p}_1\dot{\theta}_1 + \boldsymbol{p}_2\dot{\theta}_2 \\ &\vdots \\ \boldsymbol{t}_n &= \boldsymbol{A}_{n,n-1}\boldsymbol{t}_{n-1} + \boldsymbol{p}_n\dot{\theta}_n = \boldsymbol{A}_{n,1}\boldsymbol{p}_1\dot{\theta}_1 + \boldsymbol{A}_{n,2}\boldsymbol{p}_2\dot{\theta}_2 + \cdots + \boldsymbol{A}_{n,n}\boldsymbol{p}_n\dot{\theta}_n. \end{aligned} \tag{5}$$

Note that in the above expressions the following properties of the 6×6 twist propagation matrix, $\boldsymbol{A}_{i,i-1}$, are used [17]

$$\boldsymbol{A}_{i,j} = \boldsymbol{A}_{j,i}; \quad \boldsymbol{A}_{i,i} = \mathbf{1}; \quad \text{and} \quad \boldsymbol{A}_{i,j}^{-1} = \boldsymbol{A}_{j,i}. \tag{6}$$

(3) Next, Equation (5) is rewritten as

$$\boldsymbol{t} = \boldsymbol{N}\dot{\boldsymbol{\theta}} \tag{7}$$

where the $6n$-dimensional vector of the generalized twist $\boldsymbol{t}$ and the n-dimensional vector of joint rates $\dot{\boldsymbol{\theta}}$ are defined in Equations (2) and (3), respectively. The $6n\times n$ matrix

$\boldsymbol{N}$ is the natural orthogonal complement matrix for the n-link serial robot at hand, as introduced by Angeles and Lee [28], i.e.,

$$\boldsymbol{N} \equiv \begin{bmatrix} \boldsymbol{p}_1 & \cdots & \cdots & \boldsymbol{O} \\ \boldsymbol{A}_{21}\boldsymbol{p}_1 & \boldsymbol{p}_2 & \cdots & \boldsymbol{O} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}_{n1}\boldsymbol{p}_1 & \cdots & \boldsymbol{A}_{n,n-1}\boldsymbol{p}_{n-1} & \boldsymbol{p}_n \end{bmatrix} \tag{8a}$$

where $\boldsymbol{O}$ is the 6×6 matrix of zeros. Henceforth, it should be understood as of compatible dimensions based on the expressions where they appear. Note that $\boldsymbol{N}$ can be decoupled as a product of two matrices, namely, the $6n \times 6n$ lower block triangular matrix, $\boldsymbol{N}_{\mathrm{l}}$, and the $6n \times n$ block diagonal matrix, $\boldsymbol{N}_{\mathrm{d}}$ [17] as

$$\boldsymbol{N} \equiv \boldsymbol{N}_{\mathrm{l}}\boldsymbol{N}_{\mathrm{d}} \tag{8b}$$

where $\boldsymbol{N}_{\mathrm{l}}$ and $\boldsymbol{N}_{\mathrm{d}}$ are given by,

$$\boldsymbol{N}_{\mathrm{l}} \equiv \begin{bmatrix} \mathbf{1} & \boldsymbol{O} & \cdots & \boldsymbol{O} \\ \boldsymbol{A}_{21} & \mathbf{1} & \cdots & \boldsymbol{O} \\ \vdots & \vdots & \ddots & \\ \boldsymbol{A}_{n1} & \boldsymbol{A}_{n2} & \cdots & \mathbf{1} \end{bmatrix}; \quad \boldsymbol{N}_{\mathrm{d}} \equiv \begin{bmatrix} \boldsymbol{p}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{p}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{p}_n \end{bmatrix}. \tag{8c}$$

The $6n \times 6n$ lower block triangular matrix $\boldsymbol{N}_{\mathrm{l}}$ and $6n \times n$ block diagonal matrix $\boldsymbol{N}_{\mathrm{d}}$ are the DeNOC matrices for the system at hand, as introduced by Saha in [17]. In Equation (8c), $\mathbf{1}$ denotes the 6×6 identity matrix and $\mathbf{0}$ is the 6-dimensional vector of zeros. Like $\boldsymbol{O}$, henceforth, $\mathbf{1}$ and $\mathbf{0}$ should be understood as of compatible size based on where it appears.

3.2 Dynamic modeling

The dynamic modeling of the rigid robot shown in Figure 1, is now derived using the Euler–Lagrange equations of motion and the DeNOC matrices derived in Section 3.1. The steps are outlined as follows:

(1) Referring to Figure 2, it is assumed that the shape of the ith link is as per the definitions of the Denavit and Hartenberg (DH) parameters, used in [17, 32]. The definitions of DH parameters are also reproduced in Appendix A. The position vectors of the elements, $\bar{E}_i$, $\tilde{E}_i$ and payload of mass m_{pi} on the ith link, namely, $\boldsymbol{r}_i$, $\tilde{\boldsymbol{r}}_i$, and $\boldsymbol{r}_{pi}$ are, respectively, given by,

$$\begin{aligned} \boldsymbol{r}_i &= \boldsymbol{o}_i + \bar{\boldsymbol{b}}_i, \quad \text{where } \bar{\boldsymbol{b}}_i = \bar{b}_i \boldsymbol{z}_i \\ \tilde{\boldsymbol{r}}_i &= \boldsymbol{o}_i + \bar{\boldsymbol{r}}_i, \quad \text{where } \bar{\boldsymbol{r}}_i = b_i \boldsymbol{z}_i + \bar{a}_i \boldsymbol{x}_{i+1}, \quad \text{and} \\ \boldsymbol{r}_{pi} &= \boldsymbol{o}_i + \bar{\boldsymbol{r}}_{pi}, \quad \text{where } \bar{\boldsymbol{r}}_{pi} = b_i \boldsymbol{z}_i + a_i \boldsymbol{x}_{i+1} \end{aligned} \tag{9}$$

where, the nonnegative distance, a_i, and the offset of the axis X_{i+1} from the origin of the ith frame, b_i, are the DH-parameters of the link, as defined in Appendix A, $\bar{b}_i$ is the axial distance of element $\bar{E}_i$ along Z_i from O_i, and $\bar{a}_i$ is the axial distance of element $\tilde{E}_i$ along X_{i+1} from O_i', as shown in Figure 2. Note that $\bar{\boldsymbol{b}}_i$ is the position vector of element $\bar{E}_i$ along Z_i from O_i, whose magnitude is $\bar{b}_i$. The term, $\bar{b}_i$, varies from 0 to b_i, and $\bar{a}_i$ varies from 0 to a_i. Moreover, the unit vectors along Z_i and X_{i+1}-axes are denoted with $\boldsymbol{z}_i$ and $\boldsymbol{x}_{i+1}$, respectively.

(2) The kinetic energy, T_i, for the ith link is then given by

$$T_i = \frac{1}{2}\int_0^{b_i} \rho_i \dot{\boldsymbol{r}}_i^{\mathrm{T}} \dot{\boldsymbol{r}}_i d\bar{b}_i + \frac{1}{2}\int_0^{a_i} \rho_i \dot{\tilde{\boldsymbol{r}}}_i^{\mathrm{T}} \dot{\tilde{\boldsymbol{r}}}_i d\bar{a}_i + \frac{1}{2} m_{pi} \dot{\boldsymbol{r}}_{pi}^{\mathrm{T}} \dot{\boldsymbol{r}}_{pi} + T_{hi} \tag{10}$$

where ρ_i is the mass per unit length of the ith link, and the vectors, $\dot{\boldsymbol{r}}_i$, $\dot{\tilde{\boldsymbol{r}}}_i$, and $\dot{\boldsymbol{r}}_{pi}$ are the velocities of the elements, $\bar{E}_i$, $\tilde{E}_i$ and payload, respectively, which can be written from Equation (9) as

$$\begin{aligned} \dot{\boldsymbol{r}}_i &= \boldsymbol{v}_i + \boldsymbol{\omega}_i \times \bar{\boldsymbol{b}}_i; \\ \dot{\tilde{\boldsymbol{r}}}_i &= \boldsymbol{v}_i + \boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_i + \dot{\boldsymbol{b}}_i \boldsymbol{z}_i; \quad \text{and} \\ \dot{\boldsymbol{r}}_{pi} &= \boldsymbol{v}_i + \boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_{pi} + \dot{\boldsymbol{b}}_i \boldsymbol{z}_i \end{aligned} \tag{11a}$$

where, $\boldsymbol{v}_i$ is substituted for $\dot{\boldsymbol{o}}_i$, i.e, $\boldsymbol{v}_i \equiv \dot{\boldsymbol{o}}_i$. Moreover, $\dot{\tilde{a}}_i = \dot{a}_i = 0$ is used in Equation (11) as the link is rigid. Moreover, $\dot{b}_i$ represents the linear joint rate in the presence of prismatic joint. If the ith joint is revolute, then $\dot{b}_i = 0$. For that, Equation (11a) is modified as

$$\begin{aligned} \dot{\boldsymbol{r}}_i &= \boldsymbol{v}_i + \boldsymbol{\omega}_i \times \bar{\boldsymbol{b}}_i; \\ \dot{\tilde{\boldsymbol{r}}}_i &= \boldsymbol{v}_i + \boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_i; \quad \text{and} \\ \dot{\boldsymbol{r}}_{pi} &= \boldsymbol{v}_i + \boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_{pi} \end{aligned} \tag{11b}$$

in which $\boldsymbol{\omega}_i$ is the angular velocity of the ith link. Finally, the term, T_{hi} is the kinetic energy due to the hub at the joint, which includes the effect of motors/actuators at the joint, and is given by,

$$T_{hi} = \frac{1}{2}\boldsymbol{\omega}_i^{\mathrm{T}} \boldsymbol{I}_{hi} \boldsymbol{\omega}_i. \tag{11c}$$

(3) The Euler–Lagrange (EL) equations of motion for the whole system are then given by [33]

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\theta}_j}\right) - \frac{\partial T}{\partial \theta_j} = \tau_j, \quad \text{for } j = 1, \ldots, n \tag{12}$$

where n is the total number of bodies in the system, which is same as the total number of joints in the serial robotic system (see Figure 1). Moreover, θ_j is the generalized coordinate, and $\tau_j = \tau_j^{\mathrm{E}} + \tau_j^{\mathrm{G}}$, in which τ_j^{E} is the generalized force, corresponding to the jth generalized coordinate, due to external forces and moments on the whole system and

τ_j^{G} is the generalized forces due to gravity. The generalized forces due to gravity, τ_j^{G}, have the following form:

$$\tau_j^{\mathrm{G}} = \frac{\partial V}{\partial \theta_j} \tag{13}$$

where V is the potential energy of the system at hand due to gravity. However, in the proposed algorithm τ_j^{G} need not be calculated using the partial differential of Equation (13). The effect of the τ_j^{G} is included in the algorithm efficiently using the methodology proposed by Walker and Orin [5]. Thus, the terms corresponding to the negative of the acceleration due to gravity are added to the linear velocity component of the twist vector for the first link, i.e., $\boldsymbol{t}_1$. Note that in Equation (12) for a revolute joint, the generalized coordinate, θ_j, represents joint rotation, and for a prismatic joint it should be understood as joint displacement. Now, for the total kinetic energy of the system, $T = \sum_{i=1}^{n} T_i - i$ being the subscript for the bodies – the dynamic equations of motion, as derived in Appendix B, are given as

$$\begin{bmatrix} \left(\dfrac{\partial \boldsymbol{t}_1}{\partial \dot{\theta}_j}\right)^{\mathrm{T}} & \cdots & \left(\dfrac{\partial \boldsymbol{t}_n}{\partial \dot{\theta}_j}\right)^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_1^* \\ \vdots \\ \boldsymbol{w}_n^* \end{bmatrix} = \tau_j, \quad j = 1, \ldots, n \tag{14}$$

where the 6-dimensional vectors, $\partial \boldsymbol{t}_i / \partial \dot{\theta}_j$ and $\boldsymbol{w}_i^*$, for $i, j = 1, \ldots, n$, are defined in Equation (B.11), and reproduced here as

$$\frac{\partial \boldsymbol{t}_i}{\partial \dot{\theta}_j} \equiv \begin{bmatrix} \dfrac{\partial \boldsymbol{v}_i}{\partial \dot{\theta}_j} \\ \dfrac{\partial \boldsymbol{\omega}_i}{\partial \dot{\theta}_j} \end{bmatrix};$$

$$\boldsymbol{w}_i^* \equiv \begin{bmatrix} \int_0^{b_i} \rho_i \ddot{\boldsymbol{r}}_i d\bar{b}_i + \int_0^{a_i} \rho_i \ddot{\bar{\boldsymbol{r}}}_i d\bar{a}_i + m_{pi} \ddot{\boldsymbol{r}}_{pi} \\ \int_0^{b_i} \rho_i \bar{b} (\boldsymbol{z}_i \times \ddot{\boldsymbol{r}}_i) d\bar{b}_i + \int_0^{a_i} \rho_i (\bar{\boldsymbol{r}}_i \times \ddot{\bar{\boldsymbol{r}}}_i) d\bar{a}_i + m_{pi} (\bar{\boldsymbol{r}}_{pi} \times \ddot{\boldsymbol{r}}_{pi}) \\ + (\boldsymbol{I}_{hi} \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \boldsymbol{I}_{hi} \boldsymbol{\omega}_i) \end{bmatrix}. \tag{15}$$

(4) In Equation (15), vector $\boldsymbol{w}_i^*$ can be physically interpreted as the inertia wrench in line with the definition of the wrench defined in Equation (1). It is interesting to note here that the expression of $\boldsymbol{w}_i^*$, Equation (15), has the following representation:

$$\boldsymbol{w}_i^* = \boldsymbol{M}_i \dot{\boldsymbol{t}}_i + \boldsymbol{\gamma}_i \tag{16}$$

where the 6-dimensional vector, $\dot{\boldsymbol{t}}_i \equiv [\dot{\boldsymbol{v}}_i^{\mathrm{T}} \ \dot{\boldsymbol{w}}_i^{\mathrm{T}}]^{\mathrm{T}}$, and the 6×6 mass matrix, $\boldsymbol{M}_i$, and the 6-dimensional vector, $\boldsymbol{\gamma}_i$, are defined by

$$\boldsymbol{M}_i \equiv \int_0^{b_i} \rho_i \begin{bmatrix} \mathbf{1} & -\bar{\boldsymbol{b}}_i \times \mathbf{1} \\ \mathrm{sym} & -\bar{\boldsymbol{b}}_i \times (\bar{\boldsymbol{b}}_i \times \mathbf{1}) \end{bmatrix} d\bar{b}_i + \int_0^{a_i} \rho_i \begin{bmatrix} \mathbf{1} & -\bar{\boldsymbol{r}}_i \times \mathbf{1} \\ \mathrm{sym} & -\bar{\boldsymbol{r}}_i \times (\bar{\boldsymbol{r}}_i \times \mathbf{1}) \end{bmatrix} d\bar{a}_i$$
$$+ m_{pi} \begin{bmatrix} \mathbf{1} & -\bar{\boldsymbol{r}}_{pi} \times \mathbf{1} \\ \mathrm{sym} & -\bar{\boldsymbol{r}}_{pi} \times (\bar{\boldsymbol{r}}_{pi} \times \mathbf{1}) \end{bmatrix} + \begin{bmatrix} \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{I}_{hi} \end{bmatrix} \tag{17a}$$

$$\gamma_i \equiv \int_0^{b_i} \rho_i \begin{bmatrix} \varpi_i \\ \bar{\boldsymbol{b}}_i \times \varpi_i \end{bmatrix} d\bar{b}_i + \int_0^{a_i} \rho_i \begin{bmatrix} \bar{\varpi}_i \\ \bar{\boldsymbol{r}}_i \times \bar{\varpi}_i \end{bmatrix} d\bar{\boldsymbol{a}}_i + \boldsymbol{m}_{pi} \begin{bmatrix} \bar{\varpi}_{pi} \\ \bar{\boldsymbol{r}}_{pi} \times \bar{\varpi}_{pi} \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{\omega}_i \times \boldsymbol{I}_{hi}\boldsymbol{\omega}_i \end{bmatrix} \tag{17b}$$

in which $\varpi_i \equiv \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \bar{\boldsymbol{b}}_i)$, $\bar{\varpi}_i \equiv \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_i)$ and $\bar{\varpi}_{pi} \equiv \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_{pi})$, and "sym" denotes the symmetric elements of the matrix, $\boldsymbol{M}_i$.

(5) Writing Equation (14) for all n-links, i.e., $j = 1, \ldots, n$, one obtains

$$\begin{bmatrix} \left(\frac{\partial \boldsymbol{t}_1}{\partial \dot{\theta}_1}\right)^{\mathrm{T}} & \cdots & \left(\frac{\partial \boldsymbol{t}_n}{\partial \dot{\theta}_1}\right)^{\mathrm{T}} \\ \vdots & & \\ \left(\frac{\partial \boldsymbol{t}_1}{\partial \dot{\theta}_n}\right)^{\mathrm{T}} & \cdots & \left(\frac{\partial \boldsymbol{t}_n}{\partial \dot{\theta}_n}\right)^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_1^* \\ \vdots \\ \boldsymbol{w}_n^* \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_n \end{bmatrix} \tag{18}$$

which is expressed as

$$\left(\frac{\partial \boldsymbol{t}}{\partial \dot{\theta}}\right)^{\mathrm{T}} (\boldsymbol{M}\dot{\boldsymbol{t}} + \gamma) = \boldsymbol{\tau}. \tag{19}$$

Note that in Equation (19), the $6n \times 6n$ matrix, $\boldsymbol{M}$, the $6n$-dimensional vector, γ, and the n-dimensional vector, $\boldsymbol{\tau}$, have the following definitions:

$$\boldsymbol{M} \equiv \mathrm{diag}[\, \boldsymbol{M}_1 \ \ldots \ \boldsymbol{M}_n \,]; \quad \gamma \equiv \left[\, \gamma_1^{\mathrm{T}} \ \ldots \ \gamma_n^{\mathrm{T}} \,\right]^{\mathrm{T}}; \quad \boldsymbol{\tau} \equiv [\, \boldsymbol{\tau}_1 \ \ldots \ \boldsymbol{\tau}_n \,]^{\mathrm{T}} \tag{20}$$

whereas the vectors, $\boldsymbol{t}$ and the $\boldsymbol{w}$, are defined in Equation (2). Moreover, the $6n \times n$ matrix, $\frac{\partial \boldsymbol{t}}{\partial \dot{\boldsymbol{\theta}}}$ of Equation (19), is defined as

$$\frac{\partial \boldsymbol{t}}{\partial \dot{\boldsymbol{\theta}}} \equiv \begin{bmatrix} \frac{\partial \boldsymbol{t}_1}{\partial \dot{\theta}_1} & \cdots & \frac{\partial \boldsymbol{t}_1}{\partial \dot{\theta}_n} \\ \vdots & & \vdots \\ \frac{\partial \boldsymbol{t}_n}{\partial \dot{\theta}_1} & \cdots & \frac{\partial \boldsymbol{t}_n}{\partial \dot{\theta}_n} \end{bmatrix} \tag{21}$$

(6) From Equation (7), it is clear that

$$\frac{\partial \boldsymbol{t}}{\partial \dot{\boldsymbol{\theta}}} = \boldsymbol{N}_{\mathrm{l}} \boldsymbol{N}_{\mathrm{d}}. \tag{22}$$

Hence, Equation (19) has the following form:

$$\boldsymbol{N}_{\mathrm{d}}^{\mathrm{T}} \boldsymbol{N}_{\mathrm{l}}^{\mathrm{T}} (\boldsymbol{M}\dot{\boldsymbol{t}} + \gamma) = \boldsymbol{\tau} \tag{23}$$

(7) Differentiating Equation (7), i.e.,

$$\dot{\boldsymbol{t}} = \boldsymbol{N}_{\mathrm{l}} \boldsymbol{N}_{\mathrm{d}} \ddot{\boldsymbol{\theta}} + \boldsymbol{N}_{\mathrm{l}} \dot{\boldsymbol{N}}_{\mathrm{d}} \dot{\boldsymbol{\theta}} + \dot{\boldsymbol{N}}_{\mathrm{l}} \boldsymbol{N}_{\mathrm{d}} \dot{\boldsymbol{\theta}} \tag{24}$$

and substituting the resulting expression, Equation (24), into Equation (23), the independent set of dynamic equations of motion is finally obtained as

$$\boldsymbol{I}\ddot{\boldsymbol{\theta}} = \boldsymbol{\phi} \tag{25}$$

where the $n \times n$ Generalized Inertia Matrix (GIM), $\boldsymbol{I}$, is given by,

$$\boldsymbol{I} \equiv \boldsymbol{N}_{\mathrm{d}}^{\mathrm{T}} \tilde{\boldsymbol{M}} \boldsymbol{N}_{\mathrm{d}}, \quad \text{where} \quad \tilde{\boldsymbol{M}} \equiv \boldsymbol{N}_{\mathrm{l}}^{\mathrm{T}} \boldsymbol{M} \boldsymbol{N}_{\mathrm{l}} \tag{26}$$

and the n-dimensional vector, $\boldsymbol{\phi}$, is as follows:

$$\boldsymbol{\phi} = \boldsymbol{\tau} - \boldsymbol{N}_{\mathrm{d}}^{\mathrm{T}} \boldsymbol{N}_{\mathrm{l}}^{\mathrm{T}} [\boldsymbol{M}(\boldsymbol{N}_{\mathrm{l}} \dot{\boldsymbol{N}}_{\mathrm{d}} + \dot{\boldsymbol{N}}_{\mathrm{l}} \boldsymbol{N}_{\mathrm{d}}) \dot{\boldsymbol{\theta}}]. \tag{27}$$

As shown in [17, 18], the 6×6 block matrix, $\tilde{\boldsymbol{M}}_i$, Equation (26), can be evaluated recursively as

$$\tilde{\boldsymbol{M}}_i \equiv \boldsymbol{M}_i + \boldsymbol{A}_{i+1,i}^{\mathrm{T}} \tilde{\boldsymbol{M}}_{i+1} \boldsymbol{A}_{i,i+1} \tag{28}$$

in which $\tilde{\boldsymbol{M}}_{n+1} \equiv \boldsymbol{O}$, because there is no $(n+1)$th link in the chain. Hence, $\tilde{\boldsymbol{M}}_n \equiv \boldsymbol{M}_n$. Accordingly, the elements of the Generalized Inertia Matrix (GIM), $\boldsymbol{I}$ of Equation (33), can be represented as

$$\boldsymbol{i}_{ij} = \boldsymbol{i}_{ji} \equiv \boldsymbol{p}_i^{\mathrm{T}} \tilde{\boldsymbol{M}}_i \boldsymbol{A}_{ij} \boldsymbol{p}_j \tag{29}$$

for $i = 1, \ldots, n;\ j = 1, \ldots, n$. Equations (25–29) are also reported in [17], where it is derived starting from the NE equations of motion. In this paper, it is shown how the same can be derived starting from the EL equations of motion, Equation (12). The advantage of the present formulation is that one can easily accommodate the flexible links in the system without requiring any modification in the dynamic formulation, e.g., in [17], as the links are treated here as continuous beams. The extension to the flexible systems will be communicated separately.

4 Recursive forward dynamics algorithm

In the previous section, we presented dynamic modeling and the methodology. In this section, the recursive steps to calculate the joint accelerations, $\ddot{\boldsymbol{\theta}}$ from Equation (25), are outlined. The steps are based on the $\boldsymbol{I} \equiv \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^{\mathrm{T}} - \boldsymbol{U}$ and $\boldsymbol{D}$ being the upper and diagonal matrices, respectively – factorization [18].

Step A. Equation (25) is rewritten in terms of the factorized GIM as

$$\boldsymbol{U}\hat{\boldsymbol{\phi}} = \boldsymbol{\phi}, \quad \text{where} \quad \boldsymbol{D}\tilde{\boldsymbol{\phi}} = \hat{\boldsymbol{\phi}} \quad \text{and} \quad \boldsymbol{U}^{\mathrm{T}}\ddot{\boldsymbol{\theta}} = \tilde{\boldsymbol{\phi}}. \tag{30}$$

Step B. Since the $\boldsymbol{I} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^{\mathrm{T}}$ factorization can be done analytically, the expression of each element of the upper triangular matrix $\boldsymbol{U}$ is available. As a result, the analytical expression for each element of the solution for the vector, $\hat{\boldsymbol{\phi}} = \boldsymbol{U}^{-1}\boldsymbol{\phi}$ is given by

$$\hat{\varphi}_i = \varphi_i - \boldsymbol{p}_i^{\mathrm{T}} \boldsymbol{\eta}_{i,i+1} \tag{31}$$

for $i = n-1, \ldots, 1$, where $\hat{\varphi}_n = \varphi_n$, and the 6-dimensional vector, $\boldsymbol{\eta}_{n,n+1}$, is obtained recursively as

$$\boldsymbol{\eta}_{i,i+1} = \boldsymbol{A}_{i+1,i}^{\mathrm{T}} \boldsymbol{\eta}_{i+1}, \quad \text{and} \quad \boldsymbol{\eta}_{i+1} = \boldsymbol{\psi}_{i+1} \hat{\tau}_{i+1} + \boldsymbol{\eta}_{i+1,i+2} \tag{32}$$

in which $\boldsymbol{\eta}_{n,n+1} = \boldsymbol{0}$. The 6×6 twist propagation matrix $\boldsymbol{A}_{i+1,i}$ is defined similar to $\boldsymbol{A}_{i,i-1}$ after Equation (4), whereas the 6-dimensional vector, $\boldsymbol{\psi}_i$, is obtained from

$$\boldsymbol{\psi}_i = \frac{\boldsymbol{A}_{i+1,i} \hat{\boldsymbol{\psi}}_i}{\hat{m}_i}, \quad \text{where} \quad \hat{\boldsymbol{\psi}}_i \equiv \hat{\boldsymbol{M}}_i \boldsymbol{p}_i \quad \text{and} \quad \hat{m}_i \equiv \boldsymbol{p}_i^{\mathrm{T}} \hat{\boldsymbol{\psi}}_i. \tag{33}$$

Note that the 6×6 matrix, $\hat{\boldsymbol{M}}_i$, can also be obtained recursively, as

$$\hat{\boldsymbol{M}}_i = \boldsymbol{M}_i + \boldsymbol{A}_{i+1,i}^{\mathrm{T}} \bar{\boldsymbol{M}}_{i+1} \boldsymbol{A}_{i+1,i}, \quad \text{where} \quad \bar{\boldsymbol{M}}_i \equiv \hat{\boldsymbol{M}}_i - \hat{\boldsymbol{\psi}}_i \boldsymbol{\psi}_i^{\mathrm{T}} \tag{34}$$

for $i = n-1, \ldots, 1$, and $\hat{\boldsymbol{M}}_n \equiv \boldsymbol{M}_n$

Step C. The expression for each element of $\tilde{\boldsymbol{\phi}}$ is now found from, $\tilde{\boldsymbol{\phi}} = \boldsymbol{D}^{-1} \hat{\boldsymbol{\phi}}$, as

$$\tilde{\varphi}_i = \frac{\hat{\varphi}_i}{\hat{m}_i} \tag{35}$$

for $i = 1, \ldots, n$, where $\hat{\varphi}_i$ is available from Step B.

Step D. Finally, the expression for the each element of the joint acceleration vector, $\ddot{\boldsymbol{\theta}} = \boldsymbol{U}^{-1} \tilde{\boldsymbol{\phi}}$, is given by

$$\ddot{\boldsymbol{\theta}}_i = \tilde{\varphi}_i - \boldsymbol{\psi}_i^{\mathrm{T}} \boldsymbol{\mu}_{i,i-1} \tag{36}$$

for $i = 2, \ldots, n$. Note that, $\ddot{\theta}_1 \equiv \tilde{\varphi}_1$ and the 6-dimensional vector, $\boldsymbol{\mu}_{i,i-1}$, is obtained similar to $\boldsymbol{\eta}_{i,i+1}$ as

$$\boldsymbol{\mu}_{i,i-1} \equiv \boldsymbol{A}_{i,i-1} \boldsymbol{\mu}_{i-1}, \quad \text{where} \quad \boldsymbol{\mu}_{i-1} \equiv \boldsymbol{p}_{i-1} \ddot{\theta}_{i-1} + \boldsymbol{\mu}_{i-1,i-2} \quad \text{and} \quad \boldsymbol{\mu}_{10} \equiv \boldsymbol{0}. \tag{37}$$

Using these four steps, the computational complexity of the algorithm is shown next.

4.1 Order (n) algorithm

In this section, computational counts in terms of numbers division/multiplication (M), substraction/addition (A), and the trigonometric operations (T), are performed. In order to compute, the joint accelerations, $\ddot{\theta}$, from Equation (36), the following inputs are needed: For $i = 1, \ldots, n$

(i) Constant Denavit–Hartenberg (DH) parameters [32] of the system under study, i.e., α_i, a_i, and b_i (for revolute joints) or θ_i (for prismatic joints),
(ii) Time history of the variable DH parameter, i.e., θ_i, for the revolute joint, and b_i, for the prismatic joint, and the initial values for the first time derivatives.
(iii) Mass per unit length of each body, ρ_i.

Each step is accompanied with the M, A, T, counts that are given inside {and}, where a vector or a matrix represented in frame i is denoted by $[\bullet]_i$. The calculation sequence is as follows:

1. Find 3-dimensional vector $[\boldsymbol{a}_i]_i$ and $[\boldsymbol{a}_i]_{i+1}$. For $i = 1, \ldots, n$,

$$c\theta_i \equiv \cos\theta_i, \quad s\theta_i \equiv \sin\theta_i \qquad \{2\mathrm{T(n)}\}$$

$$[\boldsymbol{a}_i]_i = [\, a_i c\theta_i \quad a_i s\theta_i \quad b_i \,]^{\mathrm{T}} \qquad \{2\mathrm{M(n)}\}$$

and

$$c\alpha_i \equiv \cos\alpha_i, \quad s\alpha_i \equiv \sin\alpha_i \qquad \{\mathrm{nil}\}$$

$$[\boldsymbol{a}_i]_{i+1} \equiv [\, a_i \quad b_i s\alpha_i \quad b_i c\alpha_i \,]^{\mathrm{T}} \qquad \{2\mathrm{M(n)}\}$$

in which the DH parameters, a_i, b_i, α_i, and θ_i are known from the inputs. Since α_i is constant its "sine" and "cosine" evaluations would be done offline and given earlier as {nil} computations.

2. Find 3-dimensional unit vectors $[\boldsymbol{x}_{i+1}]_i$ and $[z_i]_i$. For $i = 1, \ldots, n$;

$$[\boldsymbol{x}_{i+1}]_i = \boldsymbol{Q}_i\, [\boldsymbol{x}_{i+1}]_{i+1}, \quad \text{where } [\boldsymbol{x}_{i+1}]_{i+1} = [\,1 \quad 0 \quad 0\,]^{\mathrm{T}}, \quad \text{and } [\boldsymbol{z}_i]_i = [\,0 \quad 0 \quad 1\,]^{\mathrm{T}}. \qquad \{\mathrm{nil}\}$$

The $\boldsymbol{x}_{i+1}$ and z_i are, respectively, the unit vectors along the X_{i+1}- and Z_i-axes, and $[\boldsymbol{Q}_i]_i$ is the 3×3 rotation matrix that defines the orientation of the $(i+1)$st frame with respect to the ith frame, i.e.,

$$\boldsymbol{Q}_i \equiv \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i \\ 0 & s\alpha_i & c\alpha_i \end{bmatrix} \qquad \{4\mathrm{M(n)}\}$$

Since $[\boldsymbol{x}_{i+1}]_{i+1} \equiv [\,1\ 0\ 0\,]^{\mathrm{T}}$, the multiplication of $\boldsymbol{Q}_i$ with it does not involve any computation, as $\boldsymbol{Q}_i\, [\boldsymbol{x}_{i+1}]_{i+1}$ is nothing but the first column of $\boldsymbol{Q}_i$. As a result, $[\boldsymbol{x}_{i+1}]_i$or $[z_i]_i$ does not require any computation.

3. Find the 6×6 mass-matrix, $[\boldsymbol{M}_i]_i$, which is denoted as

$$\boldsymbol{M}_i \equiv \begin{bmatrix} \boldsymbol{M}_{ivv} & \boldsymbol{M}_{ivw} \\ \mathrm{sym} & \boldsymbol{M}_{iww} \end{bmatrix}$$

where $\boldsymbol{M}_{ivv}$, $\boldsymbol{M}_{ivw}$, and $\boldsymbol{M}_{iww}$ are the 3×3 sub-matrices whose expressions are available from Equation (17a) and "sym" denotes the symmetric elements of the matrix. The nomenclature of $\boldsymbol{M}_{ivv}$, $\boldsymbol{M}_{ivw}$, and $\boldsymbol{M}_{iww}$ is done based upon the associated components of the twist vector, $\boldsymbol{v}$ and $\boldsymbol{w}$ (see Equation (2)). For example, $\boldsymbol{M}_{ivv}$ is the mass matrix associated with the linear velocity, $\boldsymbol{v}$. The computations are given, for $i = 1, \ldots, n$, as

$$[\boldsymbol{M}_{ivv}]_i = \rho_i\, (a_i + b_i)\, \mathbf{1} \qquad \{\mathrm{nil}\}$$

where $\rho_i\,(a_i + b_i)$ is constant and need not be counted. Then, the matrix $[\boldsymbol{M}_{ivw}]_i$ is obtained as

$$[\boldsymbol{M}_{ivw}]_i = C_{1i}\,[z_i]_i \times \mathbf{1} + C_{2i}\,[\boldsymbol{x}_{i+1}]_i \times \mathbf{1} \qquad \{2\mathrm{M(n)}\}$$

where $C_{1i} \equiv -\rho_i b_i(b_i/2 + a_i)$ and $C_{2i} \equiv -\rho_i(a_i^2/2)$, which are constants and computed offline.

Hence,

$$[\boldsymbol{M}_{iww}]_i = C_{3i}\,[z_i]_i^{\mathrm{T}}\,[z_i] + C_{4i}\,[\boldsymbol{x}_{i+1}]_i^{\mathrm{T}}\,[\boldsymbol{x}_{i+1}]_i + C_{5i}\big([z_i]_i^{\mathrm{T}}\,[\boldsymbol{x}_{i+1}]_i + [\boldsymbol{x}_{i+1}]_i^{\mathrm{T}}\,[z_i]_i\big) \qquad \{7\mathrm{M(n)}\ 2\mathrm{A(n)}\}$$

in which $C_{3i} \equiv \rho_i b_i^2(b_i/3 + a_i)$, $C_{4i} \equiv \rho_i(a_i^3/3)$ and $C_{5i} \equiv \rho_i a_i^2 b_i$, are also the constants that are computed offline. As an illustration of the computations involved in $[\boldsymbol{M}_{iww}]_i$, the stepwise details of the computations are shown later:

(i) $C_{3i}\,[z_i]_i^{\mathrm{T}}\,[z_i]_i = C_{3i}\begin{bmatrix}1&0&0\\0&1&0\\0&0&0\end{bmatrix} = \begin{bmatrix}C_{3i}&0&0\\0&C_{3i}&0\\0&0&0\end{bmatrix}$: nil

(ii) $C_{4i}\,[\boldsymbol{x}_{i+1}]_i^{\mathrm{T}}\,[\boldsymbol{x}_{i+1}]_i = C_{4i}\begin{bmatrix}s_i^2&-s_ic_i&0\\-s_ic_i&c_i^2&0\\0&0&1\end{bmatrix} = \begin{bmatrix}C_{4i}s_i^2&-C_{4i}s_ic_i&0\\&C_{4i}c_i^2&0\\\mathrm{sym}&&C_{4i}\end{bmatrix}$:

5M

where "sym" stands for symmetric.

(iii) $C_{5i}\left([z_i]_i^{\mathrm{T}}\,[\boldsymbol{x}_{i+1}]_i + [\boldsymbol{x}_{i+1}]_i^{\mathrm{T}}\,[z_i]_i\right) = C_{5i}\left(\begin{bmatrix}0&0&-c_i\\0&0&-s_i\\0&0&0\end{bmatrix} + \begin{bmatrix}0&0&0\\0&0&0\\-c_i&-s_i&0\end{bmatrix}\right)$

$= \begin{bmatrix}0&0&-C_{5i}c_i\\&0&-C_{5i}s_i\\\mathrm{sym}&&0\end{bmatrix}$: 2M

$[\mathbf{M}_{iww}]_i$ is then obtained by summation of (i)–(iii), i.e.,

$$[\boldsymbol{M}_{iww}]_i = \begin{bmatrix}C_{3i}+C_{4i}s_i^2&-C_{4i}s_ic_i&-C_{5i}c_i\\&C_{3i}+C_{4i}c_i^2&-C_{5i}s_i\\\mathrm{sym}&&C_{4i}\end{bmatrix}: \qquad 2\mathrm{A}$$

Thus, the computation of $[\boldsymbol{M}_{iww}]_i$ involves 7M and 2A. Since the terms C_{3i}, C_{4i}, and C_{5i} will be computed offline, no count is reported.

4. Find the n-dimensional vector, ϕ, from Equation (29): For $i = 1, \ldots, n$

$$\phi = \boldsymbol{\tau} - \bar{\boldsymbol{\tau}}, \quad \text{where} \quad \bar{\boldsymbol{\tau}} = \boldsymbol{N}_{\mathrm{d}}^{\mathrm{T}}\boldsymbol{N}_{\mathrm{l}}^{\mathrm{T}}[\boldsymbol{M}(\boldsymbol{N}_{\mathrm{l}}\dot{\boldsymbol{N}}_{\mathrm{d}} + \dot{\boldsymbol{N}}_{\mathrm{l}}\boldsymbol{N}_{\mathrm{d}})\dot{\boldsymbol{\theta}}] \qquad \{1\mathrm{A(n)}\}$$

where the vector, $\bar{\boldsymbol{\tau}}$, is expected to be known, which can be evaluated using any recursive inverse dynamics algorithm [5, 17], etc., while $\ddot{\boldsymbol{\theta}}_i = 0$, and $\boldsymbol{\tau} \equiv [\ \tau_1\ \tau_2\ \ldots\ \tau_n\ ]^{\mathrm{T}}$ is known from input.

5. Find the 6×6 matrix, $[\hat{\boldsymbol{M}}_i]_i$. For the purpose of showing the computational details, $\hat{\boldsymbol{M}}_i$, is written similar to $\boldsymbol{M}_i$ in Step C.3 as

$$\hat{\boldsymbol{M}}_i \equiv \begin{bmatrix} \hat{\boldsymbol{M}}_{ivv} & \hat{\boldsymbol{M}}_{ivw} \\ \text{sym} & \hat{\boldsymbol{M}}_{iww} \end{bmatrix}$$

where the 3×3 submatrices, $\hat{\boldsymbol{M}}_{ivv}$ and $\hat{\boldsymbol{M}}_{iww}$ are also symmetric. For $i = n$

$$[\hat{\boldsymbol{M}}_{nvv}]_n = [\boldsymbol{M}_{nvv}]_n : \text{nil}; [\hat{\boldsymbol{M}}_{nvw}]_n = [\boldsymbol{M}_{nvw}]_n : \text{nil}; [\hat{\boldsymbol{M}}_{nww}]_n = [\boldsymbol{M}_{nww}]_n : \text{nil}$$

For clarity of representation, the computations, $([\hat{\boldsymbol{M}}_{i+1,ww}]_i)[\boldsymbol{a}_i]_i \times \mathbf{1}$, which occur more than once in the computation of $[\hat{\boldsymbol{M}}_i]_i$ are represented as $[\boldsymbol{K}_i]_i$ here. Thus, for $i = n - 1, \ldots, 2$,

$$\begin{aligned}
[\hat{\boldsymbol{M}}_{i+1,ww}]_i \ \equiv\ & \boldsymbol{Q}_i[\hat{\boldsymbol{M}}_{i+1,ww}]_{i+1}\boldsymbol{Q}_i^{\mathrm{T}} : \text{16M17A}; \rightarrow [\boldsymbol{K}_i]_i \equiv -()[\boldsymbol{a}_i]_i \times \mathbf{1} : \text{18M9A}; \\
& \rightarrow ([\boldsymbol{a}_i]_i \times \mathbf{1})() : \text{6M}; \\
& \rightarrow [\hat{\boldsymbol{M}}_{i,ww}]_i = [\boldsymbol{M}_{iww}]_i + \boldsymbol{Q}_i[\bar{\boldsymbol{M}}_{i+1,ww}]_{i+1} + [\bar{\boldsymbol{M}}_{i+1,vw}]_{i+1}([\boldsymbol{a}_i]_{i+1} \times \mathbf{1}) \\
& \quad + ([\boldsymbol{a}_i]_{i+1} \times \mathbf{1})^{\mathrm{T}}[\bar{\boldsymbol{M}}_{i+1,vw}]_{i+1}^{\mathrm{T}}\boldsymbol{Q}_i^{\mathrm{T}} + () : \text{40M56A}; \\
& \rightarrow [\hat{\boldsymbol{M}}_{i,vw}]_i = [\boldsymbol{M}_{ivw}]_i + [\boldsymbol{K}_i]_i : \text{6A}; \\
& \rightarrow [\hat{\boldsymbol{M}}_{i,vv}]_i^i = [\boldsymbol{M}_{ivv}]_i + [\hat{\boldsymbol{M}}_{i+1,ww}]_i : \text{3A} \qquad \{\text{80M91A(n} - 1)\}
\end{aligned}$$

In these steps, the notation, (), represents the expression computed just before it. Moreover, the frame transformation of the symmetric matrices requiring two 3×3 matrix multiplication is done only with 16M 17A [36], instead of straightforward 2(27M 18A) = 54M 36A – 27M 18A are the values corresponding to the two 3×3 matrix multiplication. This is due to the structure of orthogonal matrix $\boldsymbol{Q}_i$, and the symmetric elements of $[\hat{\boldsymbol{M}}_{iww}]_i$. The details are shown in [36]. Similarly, a vector transformation from one to another frame, e.g., $[\boldsymbol{x}]_i = \boldsymbol{Q}_i[\boldsymbol{x}]_{i+1}$, can be done efficiently using only 8M 4A instead of 9M 6A, where $\boldsymbol{x}$ is any 3-dimensional vector.

6. Find the 6-dimensional vector, $[\hat{\boldsymbol{\psi}}_i]_i$, which is expressed as

$$\hat{\boldsymbol{\psi}}_i \equiv \begin{bmatrix} \hat{\boldsymbol{\psi}}_{i1}^{\mathrm{T}} & \hat{\boldsymbol{\psi}}_{i2}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$$

where $\hat{\boldsymbol{\psi}}_{i1}$ and $\hat{\boldsymbol{\psi}}_{i2}$ are the 3-dimensional vectors. Now, for $i = n, \ldots, 1$

$$[\hat{\boldsymbol{\psi}}_{i1}]_i = [\hat{\boldsymbol{M}}_{ivw}]_i[z_i]_i : \text{nil}; [\hat{\boldsymbol{\psi}}_{i2}]_i = [\hat{\boldsymbol{M}}_{iww}]_i[z_i]_i : \text{nil, for revolute joint;} \qquad \{\text{nil}\}$$

$$[\hat{\boldsymbol{\psi}}_{i1}]_i = [\hat{\boldsymbol{M}}_{ivv}]_i[z_i]_i : \text{nil}; [\hat{\boldsymbol{\psi}}_{i2}]_i = [\hat{\boldsymbol{M}}_{ivw}]_i^{\mathrm{T}}[z_i]_i : \text{nil for prismatic joint} \qquad \{\text{nil}\}$$

where $[z_i]_i = [0\ 0\ 1]^{\mathrm{T}}$.

7. Find the scaler $\hat{i}_i$: For $i = n, \ldots, 1$,

$$\hat{i}_i = [z_i]_i^{\mathrm{T}}\,[\hat{\psi}_{i2}]_i \quad \text{for revolute joint;} \quad \hat{i}_i = [z_i]_i^{\mathrm{T}}\,[\hat{\psi}_{i1}]_i \quad \text{for prismatic joint;} \qquad \{\text{nil}\}$$

8. Find the 6-dimensional vector, $[\psi_i]_i$: For $i = n, \ldots, 1$

$$[\psi_{i1}]_i = [\hat{\psi}_{i1}]_i/\hat{\psi}_i : 2\text{M}; \quad [\psi_{i2}]_i = [\hat{\psi}_{i2}]_i/\hat{i}_i : 3\text{M} \qquad \{5\text{M(n)}\}$$

9. Find the 6×6 matrix, $[\bar{\boldsymbol{M}}_i]_i$: A definition similar to $[\hat{\boldsymbol{M}}_i]_i$ is also used here as

$$\bar{\boldsymbol{M}}_i \equiv \begin{bmatrix} \bar{\boldsymbol{M}}_{ivv} & \bar{\boldsymbol{M}}_{ivw} \\ \text{sym} & \bar{\boldsymbol{M}}_{iww} \end{bmatrix}$$

where the 3×3 submatrices, $\bar{\boldsymbol{M}}_{ivv}$ and $\bar{\boldsymbol{M}}_{iww}$, are symmetric.

(a) For $i = n$, $[\bar{\boldsymbol{M}}_n]_n$ $\{16\text{M}14\text{A}\}$

$$[\hat{\psi}_{n1}]_n[\psi_{n1}]_n^{\mathrm{T}} : 4\text{M}; \rightarrow [\bar{\boldsymbol{M}}_{n11}]_n = [\hat{\boldsymbol{M}}_{nvv}]_n - () : 2\text{A};$$
$$[\hat{\psi}_{n2}]_n[\psi_{n1}]_n^{\mathrm{T}} : 6\text{M}; \rightarrow [\bar{\boldsymbol{M}}_{nwv}]_n = [\hat{\boldsymbol{M}}_{nwv}]_n - () : 6\text{A};$$
$$[\hat{\psi}_{n2}]_n[\psi_{n2}]_n^{\mathrm{T}} : 6\text{M}; \rightarrow [\bar{\boldsymbol{M}}_{nww}]_n = [\hat{\boldsymbol{M}}_{nww}]_n - () : 6\text{A}$$

(b) For $i = n-1, \ldots, 2$, $[\bar{\boldsymbol{M}}_i]_i$ $\{16\text{M}18\text{A}(\text{n}-1)\}$

$$[\hat{\psi}_{i1}]_i[\psi_{i1}]_i^{\mathrm{T}} : 4\text{M}; \rightarrow [\bar{\boldsymbol{M}}_{ivv}]_i = [\hat{\boldsymbol{M}}_{ivv}]_i - () : 6\text{A}$$
$$[\hat{\psi}_{i2}]_i[\psi_{i1}]_i^{\mathrm{T}} : 6\text{M}; \rightarrow [\bar{\boldsymbol{M}}_{iwv}]_i = [\hat{\boldsymbol{M}}_{iwv}]_i - () : 6\text{A};$$
$$[\hat{\psi}_{i2}]_i[\psi_{i2}]_i^{\mathrm{T}} : 6\text{M}; \rightarrow [\bar{\boldsymbol{M}}_{iww}]_i = [\hat{\boldsymbol{M}}_{iww}]_i - () : 6\text{A}$$

10. Find the scalar $\hat{\varphi}_i$:

(a) For $i = n$, $\hat{\varphi}_n = \varphi_n$ $\{\text{nil}\}$
(b) For $i = n-1, \ldots, 1$: $\{14\text{M}8\text{A}(n-1)\}$

$$[\bar{\boldsymbol{a}}_i]_{i+1} = [\boldsymbol{a}_i]_{i+1} \times [\boldsymbol{\eta}_{i+1,1}]_{i+1} : 6\text{M}; [\tilde{\boldsymbol{a}}_i]_i = \boldsymbol{Q}_i([\boldsymbol{\eta}_{i+1,2}]_{i+1} + [\bar{\boldsymbol{a}}_i]_{i+1}) : 8\text{M}7\text{A};$$
$$\hat{\varphi}_n = \varphi_n - [z_i]_i^{\mathrm{T}}\,[\tilde{\boldsymbol{a}}_i]_i : 1\text{A}$$

11. Find the 6-dimensional vector $[\boldsymbol{\eta}_i]_i$. The vector $\boldsymbol{\eta}_i$ is defined as

$$\boldsymbol{\eta}_i \equiv [\,\boldsymbol{\eta}_{i1}^{\mathrm{T}} \quad \boldsymbol{\eta}_{i2}^{\mathrm{T}}\,]^{\mathrm{T}}$$

where $\boldsymbol{\eta}_{i1}$ and $\boldsymbol{\eta}_{i2}$ are the 3-dimensional vectors containing first and last three scalar elements of η_i respectively.

(a) For $i = n$, $[\boldsymbol{\eta}_n]_n = \hat{\varphi}_n[\psi_n]_n$ $\{6\text{M}\}$

(b) For $i = n-1, \ldots, 1$, $[\boldsymbol{\eta}_i]_i$ {14M10A$(n-1)$}

$$[\boldsymbol{\eta}_{i(b)}]_i = [\psi_{i1}]_i \hat{\varphi}_i + \boldsymbol{Q}_i [\boldsymbol{\eta}_{i+1,1}]_{i+1} : 11\text{M}7\text{A}; [\boldsymbol{\eta}_{i2}]_i = [\psi_{i2}]_i \hat{\varphi}_i + [\tilde{\boldsymbol{a}}_i]_i : 3\text{M}\,3\text{A}$$

12. Find $\tilde{\varphi}$: For $i = 1, \ldots, n$

$$\tilde{\varphi}_i = \frac{\hat{\varphi}_i}{\hat{m}_i} : 1\text{M} \qquad \{1\text{M(n)}\}$$

13. Find $\ddot{\theta}$:

(a) For $i = 1$, $\ddot{\theta}_1 = \tilde{\varphi}_1$ {nil}
(b) For $i = 2, \ldots, n$, {4M6A$(n-1)$}

$$[\psi_{i2}]_i^{\mathrm{T}} [\bar{\boldsymbol{\mu}}_{i2}]_i : 2\text{M}, 2\text{A}; \rightarrow [\psi_{i1}]_i^{\mathrm{T}} [\bar{\boldsymbol{\mu}}_{i1}]_i + () : 2\text{M}, 3\text{A}; \rightarrow \ddot{\theta}_i = \tilde{\varphi}_i - () : 1\text{A}$$

14. Find the 6-dimensional vector $[\bar{\boldsymbol{\mu}}_i]_i$ as

$$\bar{\boldsymbol{\mu}}_i \equiv \begin{bmatrix} \bar{\boldsymbol{\mu}}_{i1}^{\mathrm{T}} & \bar{\boldsymbol{\mu}}_{i2}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$$

where $\bar{\boldsymbol{\mu}}_{i1}$ and $\bar{\boldsymbol{\mu}}_{i2}$ are the 3-dimensional vectors containing first and last three elements of $\bar{\boldsymbol{\mu}}_i$, respectively.

(a) For $i = 1$, $[\bar{\boldsymbol{\mu}}_1]_1 = 0$ {nil}
(b) For $i = 1, \ldots, n-1$ {22M14A$(n-1)$}

$$-[\boldsymbol{a}_{i-1}]_{i-1} \times [\bar{\boldsymbol{\mu}}_{i-1,2}]_{i-1} : 6\text{M}3\text{A}; \rightarrow [\bar{\boldsymbol{\mu}}_{i1}]_i = \boldsymbol{Q}_{i-1}^{\mathrm{T}} \{[\boldsymbol{\mu}_{i-1,1}]_{i-1} + ()\} : 8\text{M}7\text{A};$$
$$[\bar{\boldsymbol{\mu}}_{i2}]_i = \boldsymbol{Q}_{i-1}^{\mathrm{T}} [\boldsymbol{\mu}_{i-1,2}]_{i-1} : 8\text{M}4\text{A}$$

15. The 6-dimensional vector $[\boldsymbol{\mu}_i]_i$. The vector, $\boldsymbol{\mu}_i$, is defined as

$$\boldsymbol{\mu}_i \equiv \begin{bmatrix} \boldsymbol{\mu}_{i1}^{\mathrm{T}} & \boldsymbol{\mu}_{i2}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \qquad \{\text{nil}\}$$

(a) $[\boldsymbol{\mu}_1]_1$: For $i = 1$,

$$\left[\boldsymbol{\mu}_{11}^{\mathrm{T}}\right]_1 = \mathbf{0} : \text{nil}; \left[\boldsymbol{\mu}_{12}^{\mathrm{T}}\right]_1 = \ddot{\theta}_1 [z_1]_1 : \qquad \text{nil}$$

(b) $[\boldsymbol{\mu}_i]_i$: For $i = 1$,

$$\left[\boldsymbol{\mu}_{i1}^{\mathrm{T}}\right]_i = [\bar{\boldsymbol{\mu}}_{i1}]_i : \text{nil}; \left[\boldsymbol{\mu}_{i2}^{\mathrm{T}}\right]_i = \ddot{\theta}_i [z_i]_i + [\bar{\boldsymbol{\mu}}_{i2}]_i : \qquad \text{nil}$$

The total complexity of the proposed algorithm for the forward dynamics algorithm is finally obtained by adding the values on the right inside the {and} as (173n–128)M, (150n–133)A and 2nT. From Table 1, it is clear that the proposed forward dynamics algorithm is of $O(n)$ and is the most efficient. The efficiency is mainly achieved due to the assumption of robot links as slender rods rather than of a very general shape, as done in [18] and others. Such assumption is quite practical as it closely resembles real robot links used in the industries. The

Table 1 Computational complexities in forward dynamics of rigid robots

Algorithm	Multiplication	Additions	n^*	n^+
Proposed	$173\,n$–128	$150\,n$–133	10 (1602 M)	12 (1948 M 1667 A)
Saha [17]	$191n$–284	$187\,n$–325	10 (1626 M)	14 (2390 M 2293 A)
Featherstone [13]	$199\,n$–198	$174\,n$–173	12 (2190 M)	14 (2588 M 2263 A)
Valasek [15]	$226\,n$–343	$206\,n$–345	13 (2595 M)	15 (3047 M 2745A)
Brandle et al. [15]	$250\,n$–222	$220\,n$–198	14 (3278 M)	16 (3778 M 3322A)
Walker and Orin (as implemented by Featherstone)	$\frac{1}{6}n^3 + \frac{23}{2}n^2 + \frac{115}{3}n - 47$	$\frac{1}{6}n^3 + 7n^2 + \frac{233}{6}n - 46$	10 (1653 M)	12 (2357M 1716 A)

n^*: Number of links for which the $O(n)$ algorithm benefits over $O(n^3)$ one when only multiplications are considered
n^+: Number of links for which the $O(n)$ algorithm benefits over $O(n^3)$ one when both multiplications and additions are considered

Table 2 The DH parameters, link masses, and hub inertias of the 6-DOF PUMA robot

Link	a_i (m)	b_i (m)	α_i (rad)	θ_i (rad)	m_i (Kg)	$I_{zz}{}^*$ (Kg m^2)
1	0	0.20	$-\pi/2$	θ_1	10.521	0
2	0.432	0.149	0	θ_2	15.7612	0
3	0.02	0	$\pi/2$	θ_3	8.767	0
4	0	0.432	$-\pi/2$	θ_4	0	0.181
5	0	0	$\pi/2$	θ_5	0	0.0735
6	0	0.056	0	θ_6	0	0.0141

$^*I_{zz}$ is the principle moment of inertia of the hub about the axis of joint

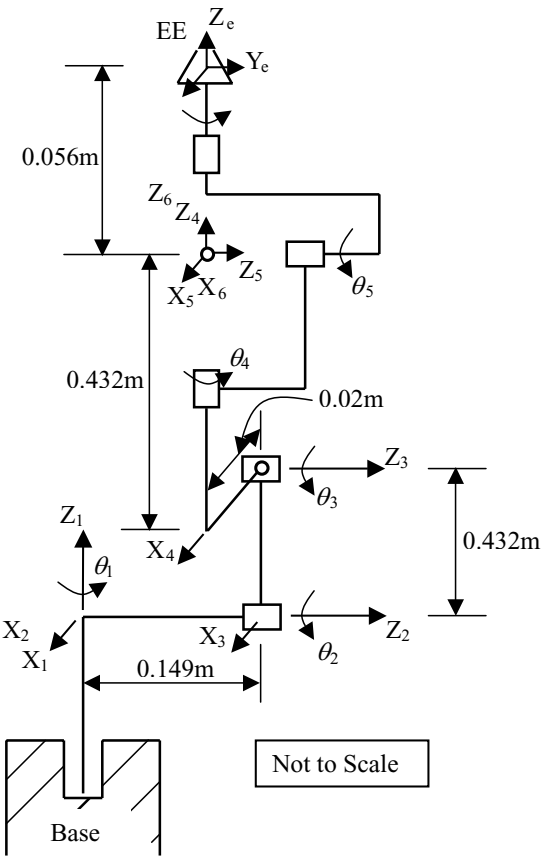
assumption avoids matrix multiplications associated with the general 3×3 inertia tensors of the links.

4.2 Simulation

In order to validate the proposed forward dynamics algorithm, simulation of a spatial 6-degree of freedom (DOF) PUMA robot, as shown in Figure 3 is performed. The DH-parameters of the robot, along with the mass of each link and hub inertias at each joint are shown in Table 2. Note that joints 4–6 are merely a wrist attached at the tip of a 3-dof PUMA robot. Kinematically the wrist is represented as shown in Figure 3. Moreover, I_{h} is the hub inertia on the ith links. The physical values of the link parameters are taken from the literature [34] to verify the results, where the masses of last three links are taken as zeros as they are negligible compared to the first three links. Forced simulation is performed where the joint torques, τ of Equation (23), are applied, which is calculated from an inverse dynamics algorithm to follow the trajectory, given as follows:

$$\theta_i = \left[\frac{\pi}{T}t - \frac{1}{2}\sin\left(\frac{2\pi}{T}t\right)\right] \tag{38}$$

Fig. 3 PUMA 6-DOF robot



while $\theta_i(0) = 0$ and $\theta_i(T) = \pi$ and $T = 10$. Figures 4(a–f) show the torques applied at the joints to get the desired trajectory of Equation (38). The initial configuration is taken as, $\theta_1 = \theta_2 = \theta_3 = \theta_4 = \theta_5 = \theta_6 = 0$. The algorithm used for the numerical integration is "ode45" of MATLAB [35], which is based on the Runga–Kutta method [25]. The step size is taken as 0.001, whereas the tolerance is 10^{-8}. Figures 5(a–f) show the forced simulation results where the simulated joint angles are compared with the desired joint trajectory, i.e., Equation (38). The simulated angles for joints 1, 2, and 4 match exactly with the desired ones, whereas for joint 3, 5, and 6 they match upto 5.5 s, beyond which the deviation occurs. The reason for such daviation is caused by the zero-eigenvalue as explained in [29].

5 Numerical stability and CPU time

The computational complexity of the forward dynamics algorithm proposed in Section 4.1 is shown to be better than other $O(n)$ algorithms reported in the literature. However, the computational complexity of the forward dynamics cannot alone be considered for the overall efficiency of the motion integration since the numerical stability characteristics of the

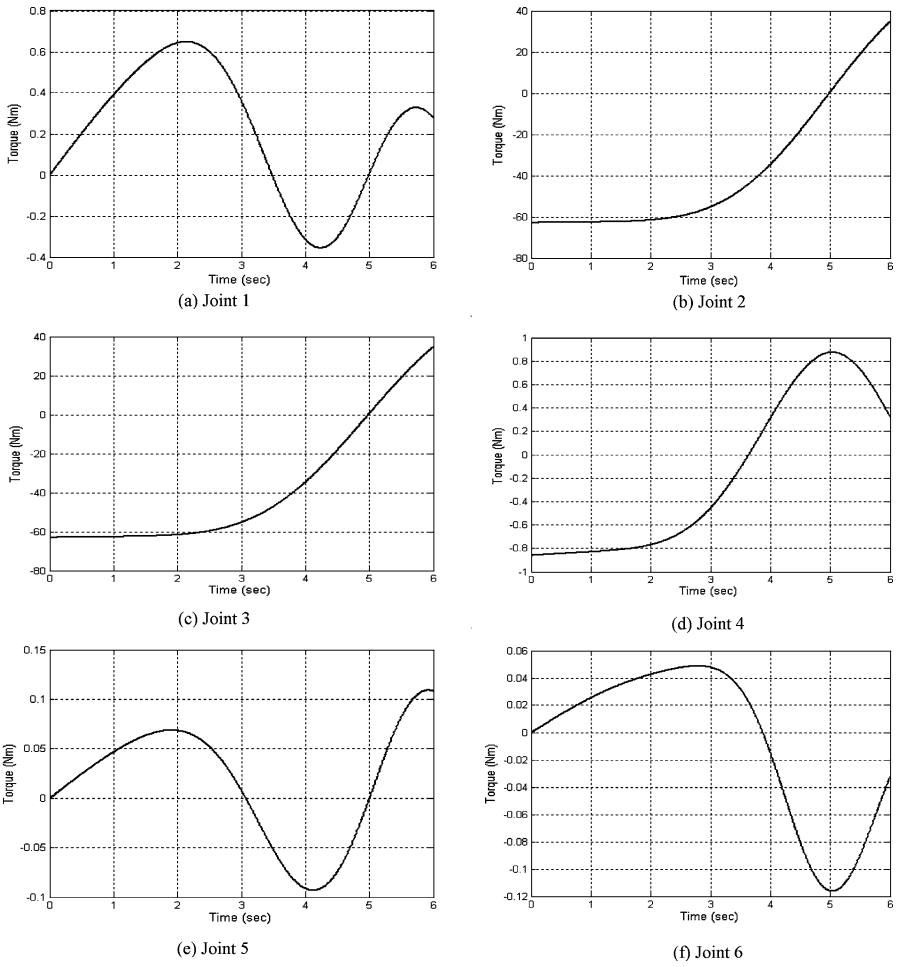


Fig. 4 Input joint torques for forced simulation

forward dynamics algorithm play an important role [23]. Hence, the numerical stability of the proposed algorithm and the associated CPU time are studied. The results are compared with those obtained using a nonrecursive algorithm while all other calculations strategies are kept same. In the nonrecursive algorithm, the GIM, $\boldsymbol{I}$ of Equation (25), is first obtained numerically using Equation (29) before it is factorized numerically using the Cholesky decomposition [36], followed by the solution of the joint accelerations, $\ddot{\boldsymbol{\theta}}$, using forward and backward substitutions. The algorithm is known to have $O(n^3)$ complexities [30]. The calculations of ϕ are carried out exactly in a manner done for the proposed recursive algorithm. Hence, the effect of recursive and nonrecursive algorithms becomes explicit. In the case of nonrecursive algorithm, the simulated results match with the desired results for a lesser duration of time as compared to the recursive algorithm. The error between the desired and the simulated results from the nonrecursive and the proposed recursive algorithms are plotted in Figures 6(a–f). The reason for the error in the simulated results from the reference results is the difference in the numerical stability of the algorithms used in respective

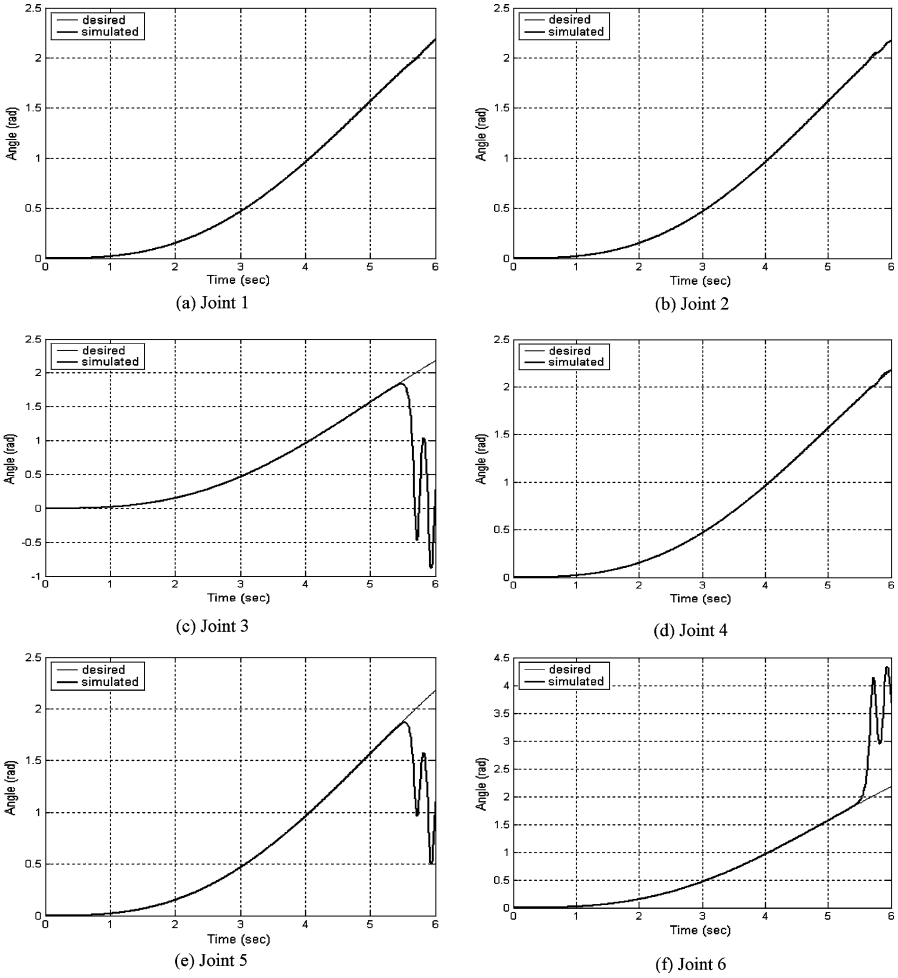


Fig. 5 Forced simulation for given torques

simulations. It is clear that the proposed algorithm gives accurate results for all joints upto 5.2 s as compared to the nonrecursive algorithm (about 1.5 s). To investigate the build-up of the errors, the nature of the joint accelerations, $\ddot{\theta}_i$, for $i = 1, \ldots, 6$, is obtained using both the algorithms. The joint accelerations for the recursive algorithm are smoother for a longer duration of time (about 5.5 s) compared to those obtained from the nonrecursive algorithm as shown in Figures 7(a–b) for joint 3. Similar behaviors were obtained for the other joints also. Power balance of the system, i.e., the desired input power obtained as, $\pi = \sum_{i=1}^{n} \tau_i \dot{\theta}_i - \tau_i$ and $\dot{\theta}_i$ being the torque obtained from the inverse dynamics algorithm and the desired joint rates, respectively, is compared with those obtained from the simulated results, namely, $\tilde{\pi} = \sum_{i=1}^{n} \tau_i \dot{\tilde{\theta}}_i - \dot{\tilde{\theta}}_i$ being the simulated joint rate, is also checked. Since no loss due to friction and damping is assumed, desired and simulated powers should be same. However, from Figure 8, it is evident that the simulated power deviates from the input power. The deviations in both the recursive and nonrecursive schemes are attributed

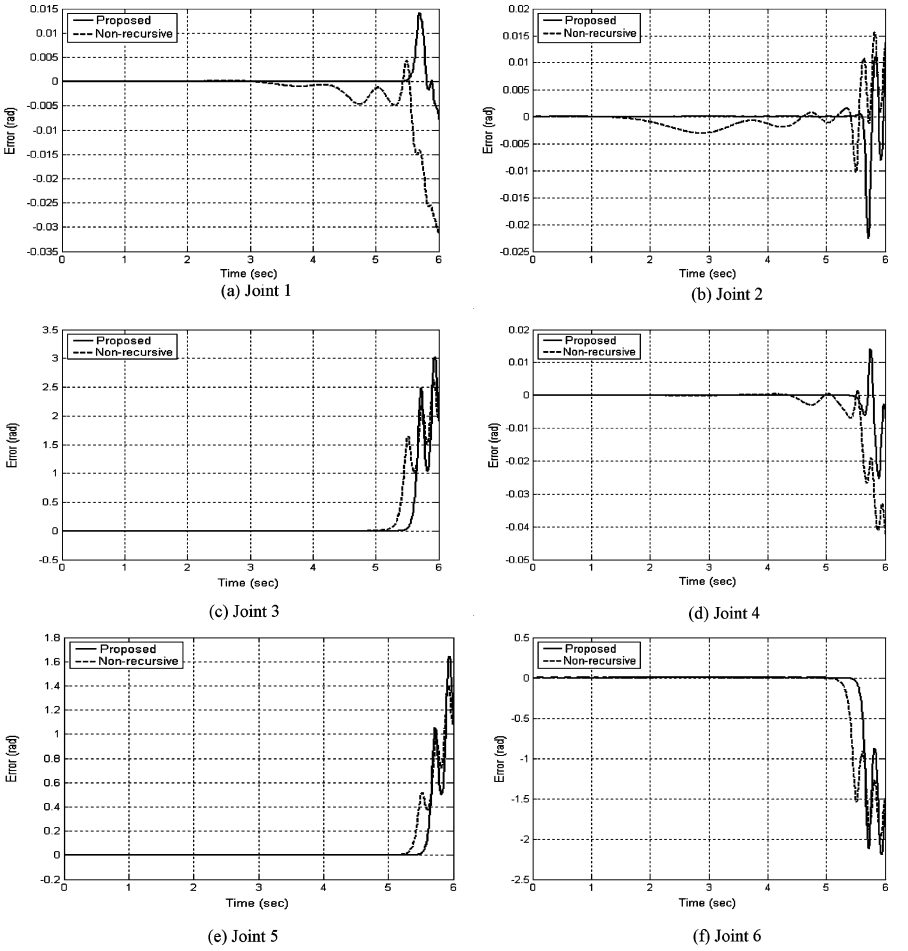
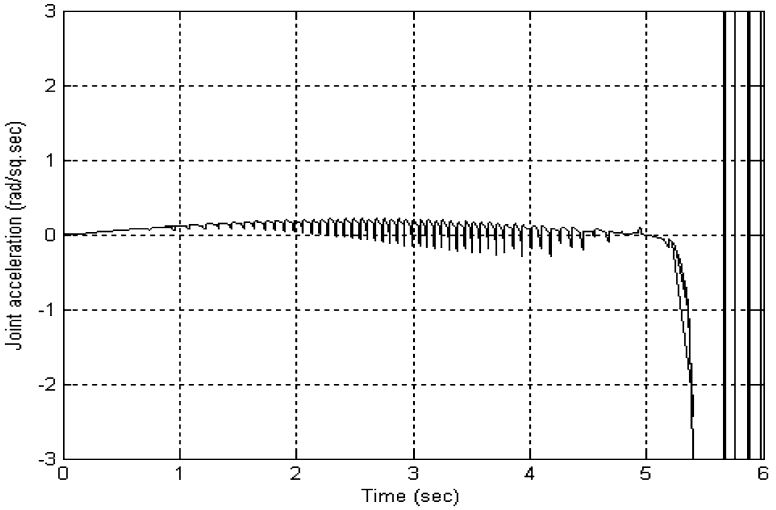


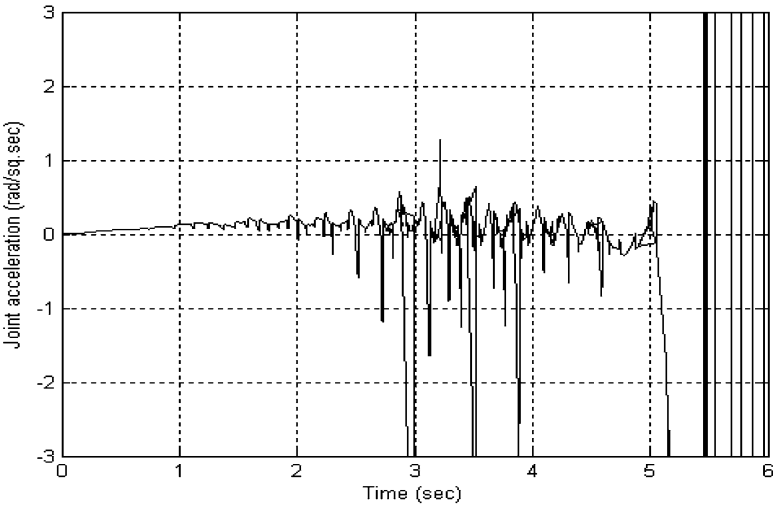
Fig. 6 Error between the desired and simulated values

to the numerical error build-ups in the forward dynamics solution and in the subsequent numerical integrations. Note that the recursive algorithm provides power match for longer duration compared to the nonrecursive one. This is attributed to the numerical stability of the recursive algorithm which stems from the fact that there is no explicit inversion of the GIM, $\boldsymbol{I}$ of Equation (25). Hence, any error amplification in the solution of the linear algebraic equations, Equation (25) particularly, for the ill-conditioned GIM, $\boldsymbol{I}$, is completely avoided.

In order to investigate the efficiency of the proposed algorithm, the CPU times taken by the recursive and the nonrecursive algorithms for the forced simulation of the 6R-PUMA robot arm is done. The simulation times, after which the simulated joint angles deviate from the desired values are considered unacceptable, namely 5.4 s for recursive algorithm and 5.1 s for nonrecursive one, are shown in Table 3. The CPU times were obtained using "tic" and "toc" commands of the MATLAB at the beginning and the end of the program for the simulation duration of 6 s. It is clear from Figure 9 that the proposed $O(n)$ recursive algorithm



(a) Recursive



(b) Non-recursive

Fig. 7 Joint accelerations at joint 3 (step size of integration = 0.001)

is efficient than its nonrecursive counterpart. This is due to the fact that the joint accelerations, $\ddot{\theta}_i$ for $i = 1, \ldots, 6$, obtained for the recursive algorithm are smoother for a longer duration of time than the nonrecursive one, as shown in Figures 7(a–f). As a result, the number of iterations required in the numerical integration are less, hence, requiring less CPU time. These results are also consistent with the power plots shown in Figure 8. In order to see the effect of the step sizes on the CPU time taken by the recursive and the nonrecursive algorithms, step-sizes of 0.1, 0.01, 0.001, and 0.0001 are used in "ode45." No significant effect was observed.

Table 3 Acceptable simulation duration

Angle	Recursive (s)	Nonrecursive (s)
θ_1	5.4	5.1
θ_2	5.6	5.1
θ_3	5.4	4.8
θ_4	5.4	5.1
θ_5	5.4	5.1
θ_6	5.4	4.8

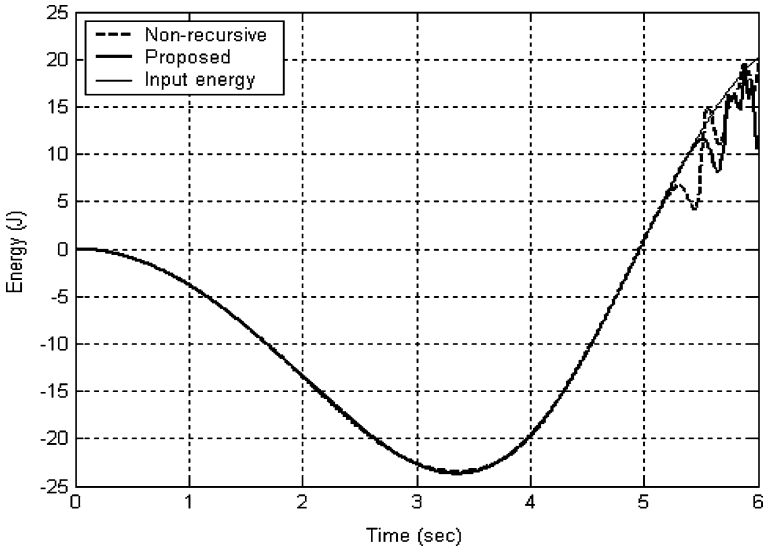


Fig. 8 Comparison of desired and simulated powers

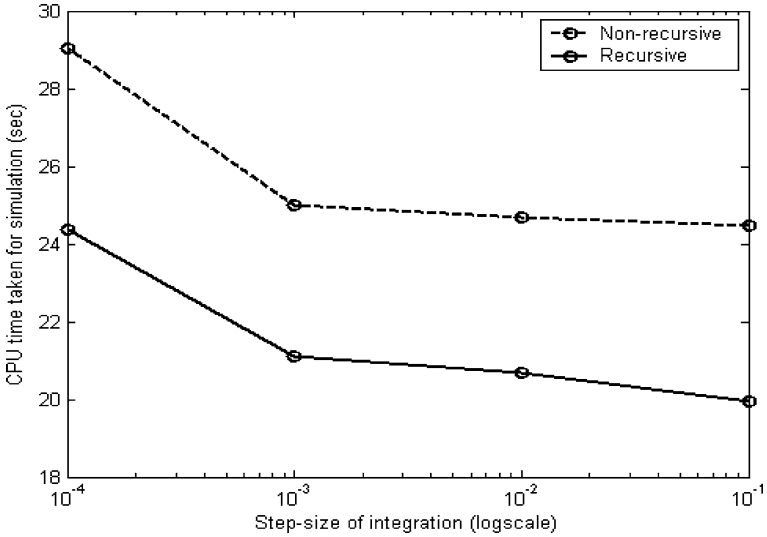


Fig. 9 Comparison of CPU times

6 Conclusions

An alternative dynamic modeling approach for the serial-chain robot with rigid links, namely, based on the Euler–Lagrange equations of motion and the decoupled natural orthogonal complement (DeNOC) matrices, is proposed that leads to a recursive forward dynamics and simulation algorithm which is not only numerically stable but efficient too. The recursiveness was obtained due to the UDU^{T} decomposition of the generalized inertia matrix, as proposed in [18]. In earlier publications, the numerical stability and efficiency aspects were not reported. Hence, one of the contributions of this paper is the stability and efficiency study and the establishment of such simulation algorithm allowing realistic and real-time simulation for the serial rigid robotic system. The other contributions of this paper are: (1) simplification of the dynamic algorithm based on the assumption of link shapes as bent slender beams, which is realistic in most practical robot architectures; (2) Evaluation of the computational complexity of the proposed algorithm; (3) Stability analysis for a 6-DOF PUMA robot which is, as per the authors knowledge, reported for the first time in the literature; (4) Treatment of the links as continuum system. Hence, it can be easily extended to the robots with flexible links also, which will be reported in future as a separate communication.

Appendix A: Denavit and Hartenberg parameters

The Denavit and Hartenberg (DH) parameters [32] is a systematic method to define the relative position and orientation of the consecutive links in a multibody robotic system. The definitions help in computing the coordinate transformations between the frames attached to the links. Note, however, that the DH parameters can be assigned differently for the same system. The DH parameters that are used in this thesis are explained next. Referring to Figure 1, the serial robot manipulator under study consists of $(n+1)$ bodies or links, namely, the fixed base and the bodies numbered as #1, ..., #n. All the bodies are coupled by n joints, numbered as $1, \ldots, n$. As shown in Figure 1, the ith joint couples the $(i-1)$st link; for the $(i+1)$st frame, i.e., X_{i+1}, Y_{i+1}, Z_{i+1}, it is clearly shown in Figure 2 that it is attached to the ith.

Now, for the first n frames, the DH parameters are defined according to the following rules: Referring to Figure A.1

1. Z_i is the axis of the ith joint. Its positive direction can be chosen arbitrarily.
2. X_i is defined as the common perpendicular to Z_{i-1} and Z_i, directed from the former to latter. The origin of the ith frame, O_i, is the point where X_i intersects Z_i. If these two axes intersect, the positive direction of X_i is chosen arbitrarily. And the origin, O_i, coincides with the origin of the $(i-1)$st frame, i.e., O_{i-1}.
3. The distance between Z_i and Z_{i+1} is defined as a_i, which is a nonnegative number.
4. The Z_i coordinate of the intersection of the X_{i+1} axis with Z_i, which is shown in Figure A.1 as the distance between O_i and O_i' is defined as b_i. This can be either positive or negative. For a prismatic joint, b_i is a variable.
5. The angle between Z_i and Z_{i+1} is defined as α_i, and is measured about the positive direction of X_{i+1}.
6. The angle between X_i and X_{i+1} is defined as θ_i, and is measured about the positive direction of Z_i. For a revolute joint, θ_i is a variable.

Since no $(n+1)$st link exists, these definitions do not apply to the $(n+1)$st frame and its axes can be chosen at will.

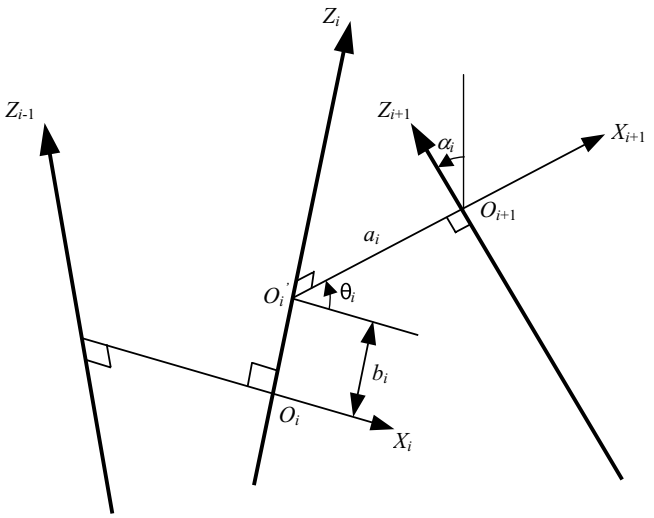


Fig. A.1 Definition of DH parameters

Appendix B: Derivation of Equation (14)

Using the expressions for the total kinetic energy, $T = \sum_{i=1}^{n} T_i$, where T_i is given by Equation (10), partial differentiations with respect to the jth generalized speed and coordinate, i.e., $\dot{\theta}_j$ and θ_j, respectively, as required in Equation (12) are obtained as

$$\frac{\partial T}{\partial \dot{\theta}_j} = \sum_{i=1}^{n} \left[\int_0^{b_i} \rho_i \dot{\boldsymbol{r}}_i^{\mathrm{T}} \frac{\partial \dot{r}_i}{\partial \dot{\theta}_j} \boldsymbol{d} \bar{b}_i + \int_0^{a_i} \rho_i \dot{\tilde{\boldsymbol{r}}}_i^{\mathrm{T}} \frac{\partial \dot{\tilde{\boldsymbol{r}}}_i}{\partial \dot{\theta}_j} \boldsymbol{d} \bar{a}_i + m_{pi} \dot{\boldsymbol{r}}_{pi}^{\mathrm{T}} \frac{\partial \dot{\boldsymbol{r}}_{pi}}{\partial \dot{\theta}_j} + (\boldsymbol{I}_{hi} \boldsymbol{\omega}_i)^{\mathrm{T}} \frac{\partial \boldsymbol{\omega}_i}{\partial \dot{\theta}_j} \right] \tag{B.1a}$$

$$\frac{\partial T}{\partial \theta_j} = \sum_{i=1}^{n} \left[\int_0^{b_i} \rho_i \dot{\boldsymbol{r}}_i^{\mathrm{T}} \frac{\partial \dot{\boldsymbol{r}}_i}{\partial \theta_j} \boldsymbol{d} \bar{b}_i + \int_0^{a_i} \rho_i \dot{\tilde{\boldsymbol{r}}}_i^{\mathrm{T}} \frac{\partial \dot{\tilde{\boldsymbol{r}}}_i}{\partial \theta_j} \boldsymbol{d} \bar{a}_i + m_{pi} \dot{\boldsymbol{r}}_{pi}^{\mathrm{T}} \frac{\partial \dot{r}_{pi}}{\partial \theta_j} + (\boldsymbol{I}_{hi} \boldsymbol{\omega}_i)^{\mathrm{T}} \frac{\partial \boldsymbol{\omega}_i}{\partial \theta_j} \right]. \tag{B.1b}$$

Next, $\frac{d}{dt}(\frac{\partial T}{\partial \dot{\theta}_j})$, is derived from Equation (B.1a), as

$$\begin{aligned} \frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\theta}_j}\right) = \sum_{i=1}^{n} \Bigg[& \int_0^{b_i} \rho_i \left\{ \ddot{\boldsymbol{r}}_i^{\mathrm{T}} \frac{\partial \dot{\boldsymbol{r}}_i}{\partial \dot{\theta}_j} + \dot{\boldsymbol{r}}_i^{\mathrm{T}} \frac{d}{dt}\left(\frac{\partial \dot{\boldsymbol{r}}_i}{\partial \dot{\theta}_j}\right) \right\} d\bar{b}_i + \int_0^{a_i} \rho_i \left\{ \ddot{\tilde{\boldsymbol{r}}}_i^{\mathrm{T}} \frac{\partial \dot{\tilde{\boldsymbol{r}}}_i}{\partial \dot{\theta}_j} + \dot{\tilde{\boldsymbol{r}}}_i^{\mathrm{T}} \frac{d}{dt}\left(\frac{\partial \dot{\tilde{\boldsymbol{r}}}_i}{\partial \dot{\theta}_j}\right) \right\} d\bar{a}_i \\ & + m_{pi} \left\{ \ddot{\boldsymbol{r}}_{pi}^{\mathrm{T}} \frac{\partial \dot{\boldsymbol{r}}_{pi}}{\partial \dot{\theta}_j} + \dot{\boldsymbol{r}}_{pi}^{\mathrm{T}} \frac{d}{dt}\left(\frac{\partial \dot{\boldsymbol{r}}_{pi}}{\partial \dot{\theta}_j}\right) \right\} \\ & + \left\{ (\boldsymbol{I}_{hi} \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \boldsymbol{I}_{hi} \boldsymbol{\omega}_i)^{\mathrm{T}} \frac{\partial \boldsymbol{\omega}_i}{\partial \dot{\theta}_j} + (\boldsymbol{I}_{hi} \boldsymbol{\omega}_i)^{\mathrm{T}} \frac{d}{dt}\left(\frac{\partial \boldsymbol{\omega}_i}{\partial \dot{\theta}_j}\right) \right\} \Bigg]. \end{aligned} \tag{B.2}$$

As shown in [15] and others, it is evident that

$$\frac{\partial \dot{\boldsymbol{r}}_i}{\partial \dot{\theta}_j} = \frac{\partial \boldsymbol{r}_i}{\partial \theta_j}; \quad \frac{\partial \dot{\tilde{\boldsymbol{r}}}_i}{\partial \dot{\theta}_j} = \frac{\partial \tilde{\boldsymbol{r}}_i}{\partial \theta_j} \quad \text{and} \quad \frac{\partial \dot{\boldsymbol{r}}_{pi}}{\partial \dot{\theta}_j} = \frac{\partial \boldsymbol{r}_{pi}}{\partial \theta_j} \tag{B.3}$$

Hence, the parts of the 2nd, 4th, and 6th terms on the right-hand side of Equation (B.2) are given by

$$\begin{aligned} &\frac{d}{dt}\left(\frac{\partial \dot{\boldsymbol{r}}_i}{\partial \dot{\theta}_j}\right) = \frac{d}{dt}\left(\frac{\partial \boldsymbol{r}_i}{\partial \theta_j}\right) = \frac{\partial \dot{\boldsymbol{r}}_i}{\partial \theta_j}; \quad \frac{d}{dt}\left(\frac{\partial \dot{\tilde{\boldsymbol{r}}}_i}{\partial \dot{\theta}_j}\right) = \frac{d}{dt}\left(\frac{\partial \tilde{\boldsymbol{r}}_i}{\partial \theta_j}\right) = \frac{\partial \dot{\tilde{\boldsymbol{r}}}_i}{\partial \theta_j} \quad \text{and} \\ &\frac{d}{dt}\left(\frac{\partial \dot{\boldsymbol{r}}_{pi}}{\partial \dot{\theta}_j}\right) = \frac{d}{dt}\left(\frac{\partial \boldsymbol{r}_{pi}}{\partial \theta_j}\right) = \frac{\partial \dot{\boldsymbol{r}}_{pi}}{\partial \theta_j}. \end{aligned} \tag{B.4}$$

Similar to Equation (B.4) it can also be shown that

$$\frac{d}{dt}\left(\frac{\partial \boldsymbol{\omega}_i}{\partial \dot{\theta}_j}\right) = \frac{\partial \boldsymbol{\omega}_i}{\partial \theta_j} \tag{B.5}$$

Substituting Equations (B.4) and (B.5) into Equation (B.2), and using the resulting expressions, along with Equation (B.1b), the left-hand side of Equation (12) yields

$$\sum_{i=1}^{n}\left[\int_0^{b_i} \rho_i \ddot{\boldsymbol{r}}_i^{\mathrm{T}} \frac{\partial \dot{\boldsymbol{r}}_i}{\partial \dot{\theta}_j} d\bar{b}_i + \int_0^{a_i} \rho_i \ddot{\tilde{\boldsymbol{r}}}_i^{\mathrm{T}} \frac{\partial \dot{\tilde{\boldsymbol{r}}}_i}{\partial \dot{\theta}_j} d\bar{a}_i + m_{pi} \ddot{\boldsymbol{r}}_{pi}^{\mathrm{T}} \frac{\partial \dot{\boldsymbol{r}}_{pi}}{\partial \dot{\theta}_j} + (\boldsymbol{I}_{hi}\dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \boldsymbol{I}_{hi}\boldsymbol{\omega}_i)\frac{\partial \boldsymbol{\omega}_i}{\partial \dot{\theta}_j}\right] = \tau_j \tag{B.6}$$

Next, $\ddot{\boldsymbol{r}}_i$, $\ddot{\tilde{\boldsymbol{r}}}_i$, and $\ddot{\boldsymbol{r}}_{pi}$, are obtained from the time differentiation of expressions of Equation (11a) as,

$$\ddot{\boldsymbol{r}}_i = \dot{\boldsymbol{v}}_i + \dot{\boldsymbol{\omega}}_i \times \bar{\boldsymbol{b}}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \bar{\boldsymbol{b}}_i) \tag{B.7a}$$

$$\ddot{\tilde{\boldsymbol{r}}}_i = \dot{\boldsymbol{v}}_i + \dot{\boldsymbol{\omega}}_i \times \bar{\boldsymbol{r}}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_i) \tag{B.7b}$$

$$\ddot{\boldsymbol{r}}_{pi} = \dot{\boldsymbol{v}}_i + \dot{\boldsymbol{\omega}}_i \times \bar{\boldsymbol{r}}_{pi} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_{pi}). \tag{B.7c}$$

Substituting $\dot{\boldsymbol{r}}_i, \dot{\tilde{\boldsymbol{r}}}_i$, and $\dot{\boldsymbol{r}}_{pi}$ from Equation (11b) into Equation (B.6), one obtains

$$\begin{aligned} \sum_{i=1}^{n}\Bigg[\int_0^{b_i} \rho_i \left\{\ddot{\boldsymbol{r}}_i^{\mathrm{T}} \frac{\partial \boldsymbol{v}_i}{\partial \dot{\theta}_j} + \ddot{\boldsymbol{r}}_i^{\mathrm{T}} \frac{\partial (\boldsymbol{\omega}_i \times \bar{b}_i \boldsymbol{z}_i)}{\partial \dot{\theta}_j}\right\} d\bar{b}_i + \int_0^{a_i} \rho_i \left\{\ddot{\tilde{\boldsymbol{r}}}_i^{\mathrm{T}} \frac{\partial \boldsymbol{v}_i}{\partial \dot{\theta}_j} + \ddot{\tilde{\boldsymbol{r}}}_i^{\mathrm{T}} \frac{\partial [\boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_i]}{\partial \dot{\theta}_j}\right\} d\bar{\boldsymbol{a}}_i \\ + m_{pi}\left\{\ddot{\boldsymbol{r}}_{pi}^{\mathrm{T}} \frac{\partial \boldsymbol{v}_i}{\partial \dot{\theta}_j} + \ddot{\boldsymbol{r}}_{pi}^{\mathrm{T}} \frac{\partial [\boldsymbol{\omega}_i \times \bar{\boldsymbol{r}}_{pi}]}{\partial \dot{\theta}_j}\right\} + (\boldsymbol{I}_{hi}\dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \boldsymbol{I}_{hi}\boldsymbol{\omega}_i)\frac{\partial \boldsymbol{\omega}_i}{\partial \dot{\theta}_j}\Bigg] = \tau_j. \end{aligned} \tag{B.8}$$

Using the vector triple product rule (Strang, 1980), $\boldsymbol{a}^{\mathrm{T}}(\boldsymbol{b}\times\boldsymbol{c}) = (\boldsymbol{c}\times\boldsymbol{a})^{\mathrm{T}}b - \boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ are any 3-dimensional Cartesian vectors – one can show that

$$\ddot{\boldsymbol{r}}_i^{\mathrm{T}}\frac{\partial(\boldsymbol{\omega}_i\times\bar{b}_i\boldsymbol{z}_i)}{\partial\dot{\theta}_j} = \bar{b}_i\ddot{\boldsymbol{r}}_i^{\mathrm{T}}\left[\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}\times\boldsymbol{z}_i + \boldsymbol{\omega}_i\times\frac{\partial\boldsymbol{z}_i}{\partial\dot{\theta}_j}\right] = \bar{b}_i(\boldsymbol{z}_i\times\ddot{\boldsymbol{r}}_i)^{\mathrm{T}}\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j} \tag{B.9a}$$

$$\ddot{\bar{\boldsymbol{r}}}_i^{\mathrm{T}}\frac{\partial(\boldsymbol{\omega}_i\times\bar{\boldsymbol{r}}_i)}{\partial\dot{\theta}_j} = \ddot{\bar{\boldsymbol{r}}}_i^{\mathrm{T}}\left[\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}\times\bar{\boldsymbol{r}}_i + \boldsymbol{\omega}_i\times\frac{\partial\bar{\boldsymbol{r}}_i}{\partial\dot{\theta}_j}\right] = [\bar{\boldsymbol{r}}_i\times\ddot{\bar{\boldsymbol{r}}}_i]^{\mathrm{T}}\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j} \tag{B.9b}$$

$$\ddot{\boldsymbol{r}}_{pi}^{\mathrm{T}}\frac{\partial\left(\boldsymbol{\omega}_i\times\bar{\boldsymbol{r}}_{pi}\right)}{\partial\dot{\theta}_j} = \ddot{\boldsymbol{r}}_{pi}^{\mathrm{T}}\left[\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}\times\bar{\boldsymbol{r}}_{pi} + \boldsymbol{\omega}_i\times\frac{\partial\bar{\boldsymbol{r}}_{pi}}{\partial\dot{\theta}_j}\right] = (\bar{\boldsymbol{r}}_{pi}\times\ddot{\boldsymbol{r}}_{pi})^{\mathrm{T}}\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}. \tag{B.9c}$$

In Equations (B.9a–c), $\partial z_i/\partial\dot{\theta}_j = 0$, $\partial\bar{\boldsymbol{r}}_i/\partial\dot{\theta}_j = 0$, and $\partial\bar{\boldsymbol{r}}_{pi}/\partial\dot{\theta}_j = 0$ as $\boldsymbol{z}_i$ $\bar{\boldsymbol{r}}_i$ and $\bar{\boldsymbol{r}}_{pi}$ are functions of θ_j's only, not $\dot{\theta}_j$'s. Hence, using Equations (B.9a–c), Equation (B.8) is rewritten as

$$\sum_{i=1}^{n}\left[\int_0^{b_i}\rho_i\left\{\ddot{\boldsymbol{r}}_i^{\mathrm{T}}\frac{\partial\boldsymbol{v}_i}{\partial\dot{\theta}_j} + \bar{b}_i(\boldsymbol{z}_i\times\ddot{\boldsymbol{r}}_i)^{\mathrm{T}}\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}\right\}d\bar{b}_i + \int_0^{a_i}\rho_i\left\{\ddot{\bar{\boldsymbol{r}}}_i^{\mathrm{T}}\frac{\partial\boldsymbol{v}_i}{\partial\dot{\theta}_j} + (\bar{\boldsymbol{r}}_i\times\ddot{\bar{\boldsymbol{r}}}_i)^{\mathrm{T}}\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}\right\}d\bar{a}_i\right.$$
$$\left. + m_{pi}\left\{\ddot{\boldsymbol{r}}_{pi}^{\mathrm{T}}\frac{\partial\boldsymbol{v}_i}{\partial\dot{\theta}_j} + (\bar{\boldsymbol{r}}_{pi}\times\ddot{\boldsymbol{r}}_{pi})^{\mathrm{T}}\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}\right\} + (\boldsymbol{I}_{hi}\dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i\times\boldsymbol{I}_{hi}\boldsymbol{\omega}_i)\frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}\right] = \tau_j. \tag{B.10}$$

Now, introducing the following definitions

$$\frac{\partial\boldsymbol{t}_i}{\partial\dot{\theta}_j} \equiv \begin{bmatrix}\frac{\partial\boldsymbol{v}_i}{\partial\dot{\theta}_j}\\ \frac{\partial\boldsymbol{\omega}_i}{\partial\dot{\theta}_j}\end{bmatrix};$$

$$\boldsymbol{w}_i^* \equiv \begin{bmatrix}\int_0^{b_i}\rho_i\ddot{\boldsymbol{r}}_i d\bar{b}_i + \int_0^{a_i}\rho_i\ddot{\bar{\boldsymbol{r}}}_i d\bar{a}_i + m_{pi}\ddot{\boldsymbol{r}}_{pi}\\ \int_0^{b_i}\rho_i\bar{b}(\boldsymbol{z}_i\times\ddot{\boldsymbol{r}}_i)d\bar{b}_i + \int_0^{a_i}\rho_i[\bar{\boldsymbol{r}}_i\times\ddot{\bar{\boldsymbol{r}}}_i]\,d\bar{a}_i + m_{pi}[\bar{\boldsymbol{r}}_{pi}\times\ddot{\boldsymbol{r}}_{pi}] + (\boldsymbol{I}_{hi}\dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i\times I_{hi}\boldsymbol{\omega}_i)\end{bmatrix} \tag{B.11}$$

Equation (B.10) is rewritten in compact form as

$$\begin{bmatrix}\left(\frac{\partial\boldsymbol{t}_1}{\partial\dot{\theta}_j}\right)^{\mathrm{T}} & \cdots & \left(\frac{\partial\boldsymbol{t}_n}{\partial\dot{\theta}_j}\right)^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\boldsymbol{w}_1^*\\ \vdots\\ \boldsymbol{w}_n^*\end{bmatrix} = \tau_j \tag{B.12}$$

which is the desired equation of motion to obtain the generalized force corresponding to the jth generalized coordinate.

Acknowledgements The research work reported in this paper is carried out under the partial financial aid from the DST, Government of India (SR/S3/RM/46/2002) which is duly acknowledged.

References

1. Schiehlen, W.O.: Multibody system dynamics: roots and perspectives. Multibody Syst. Dyn. **1**, 149–188 (1997)
2. Garcia, J.: Improved dynamic formulations for the dynamic simulation of multibody systems. Available: http://mat21.etsii.upm.es/mbs/matlabcode/Mbs3dv1/papers/2003GarciaDeJalonEtAl.pdf
3. Vereshchagin, A.F.: Gauss principle of least constraint for modeling the dynamics of automatic manipulators using a digital computer. Sov. Phys. – Dokl. **20**(1), 33–34 (1975)
4. Armstrong, W.W.: Recursive solution of the equations of motion of an n-link manipulator. In: Proceedings of the Fifth World Congress on the Theory of Machines and Mechanisms, Montreal, Canada, vol. 2, pp. 1342–1346 (1979)
5. Walker, M.W., Orin, D.E.: Efficient dynamic computer simulation of robotic mechanisms. ASME J. Dyn. Syst. Meas. Control **104**, 205–211 (1982)
6. Hollerbach, J.M.: A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity. IEEE Trans. Syst. Man Cybern. **SMC-10**(11), 730–736 (1980)
7. Anderson, K.S.: An order-N formulation for motion simulation of general constrained multi-rigidbody systems. Comput. Struct. **43**(3), 565–572 (1992)
8. Rosenthal, D.E., Sherman, M.A.: High performance multibody simulations via symbolic equation manipulation and Kane's method. J. Astronaut. Sci. **34**(3), 223–239 (1986)
9. Rosenthal, D.E.: An order N formulation for robotic systems. J. Astronaut. Sci. **38**(4), 511–529 (1990)
10. Banerjee, A.K.: Block-diagonal equations for multibody elastodynamics with geometric stiffness and constraints. J. Guid. Control Dyn. **16**(6), 1092–1100 (1993)
11. Bae, D.S., Haug, E.J.: A recursive formation for constrained mechanical systems dynamics: part I, open loop systems. Mech. Struct. Mach. **15**(3), 359–382 (1987)
12. Bae, D.S., Haug, E.J.: A recursive formation for constrained mechanical systems dynamics: part II, closed loop systems. Mech. Struct. Mach. **15**(4), 481–506 (1987)
13. Featherstone, R.: The calculation of robotic dynamics using articulated body inertias. Int. J. Robot. Res. **2**(1), 13–30 (1983)
14. Featherstone, R.: Robotic Dynamics Algorithms. Kluwer Academic, Dordrecht, The Netherlands (1987)
15. Stejskal, V., Valasek, M.: Kinematics and Dynamics of Machinery. M. Dekkar, New York (1996)
16. Jain, A.: Unified formulation of dynamics for serial rigid multibody systems. J. Guid. Control Dyn. **14**(3), 531–542 (1991)
17. Saha, S.K.: Dynamic modeling of serial multibody systems using decoupled natural orthogonal complement matrices. ASME J. Appl. Mech. **29**(2), 986–996 (1999)
18. Saha, S.K.: A decomposition of manipulator inertia matrix. IEEE Trans. Robot. Autom. **13**(2), 301–304 (1997)
19. Fijany, A., Inna, S., D'Eleuterios, G.M.T.: Parallel $O(\log N)$ algorithms for the computation of manipulator forward dynamics. In: Proceedings of the IEEE ICRA, San Diego, CA, USA, pp. 1547–1553 (1994)
20. Naudet, J., Lefeber, D.: General formulation of an efficient recursive algorithm based on canonical momenta for forward dynamics of closed-loop multibody systems. In: Proceedigns of the XI DINAME, Ouro Preto, Brazil (2005)
21. Lee, K., Chirikjain, S.G.: A new perspective on $O(n)$ mass-matrix inversion for serial revolute manipulators. In: Proceedings of IEEE ICRA, Barcelona, Spain, pp. 4733–4737 (2005)
22. Critchley, J.S., Anderson, K.S.: A generalized recursive coordinate reduction method for multibody system dynamics. Multibody Syst. Dyn. **9**, 185–212 (2003)
23. Ascher, U.M., Pai, D.K., Cloutier, B.P.: Forward dynamics, elimination methods, and formulation stiffness in robot simulation. Int. J. Robot. Res. **16**(6), 747–758 (1997)
24. Ellis, R.E., Ismaeil, O.M., Carmichael, I.H.: Numerical stability of forward-dynamics algorithms. In: Proceedings of IEEE Conference on Robotics and Automation, Nice, France, pp. 305–311 (1992)
25. Nikravesh P.E.: Computer-Aided Analysis of Mechanical Systems. Prentice-Hall, Englewood Cliffs, NJ (1988)
26. Wehage R.A., Haug, E.J.: Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems. ASME J. Mech. Des. **104**, 247–255 (1982)
27. Kamman, J.W., Huston, R.L.: Dynamics of constrained multibody systems. ASME J. Appl. Mech. **51**, 899–903 (1984)
28. Angeles, J., Lee, S.K.: The formulation of dynamical equations of holonomic mechanical systems using a natural orthogonal complement. ASME J. Appl. Mech. **55**, 243–244 (1988)
29. Saha, S.K., Schiehlen, W.O.: Recursive kinematics and dynamics for closed loop multibody systems. Int. J. Mech. Struct. Mach. **29**(2), 143–175 (2001)

30. Khan, W.A., Krovi, V.N., Saha, S.K., Angeles, J.: Modular and recursive kinematics and dynamics for parallel manipulators. Multibody Syst. Dyn. **14**, 419–455 (2005)
31. Chaudhary, H., Saha, S.K.: Constraint force formulation for closed-loop multibody systems. Trans. ASME J. Mech. Des. (in press)
32. Denavit, J., Hartenberg, R.S.: A kinematic notation for lower-pair mechanisms based on matrices. ASME J. Appl. Mech. **77**, 445–450 (1955)
33. Meirovitch, L.: Analytical Methods in Vibrations. Macmillan, New York (1967)
34. Pratap, R.: MATLAB 6: A Quick Introduction for Scientists and Engineers. Oxford University Press, New York (2002)
35. Strang, G.: Linear Algebra and its Applications. Harcourt, Brace, Jovanovich, Florida (1988)
36. Bhangale, P.: Dynamics Based Modeling, Computational Complexity and Architecture Selection of Robot Manipulators. Ph.D. Thesis, Indian Institute of Technology, New Delhi (2004)
37. Luh, J.Y.S., Walker, M.W., Paul, R.P.: On-line computational scheme for mechanical manipulators. Trans. ASME J. Dyn. Syst. Meas. Control **102**, 64–76 (1980)