**Mechanical
Sciences**

# Evolution of the DeNOC-based dynamic modelling for multibody systems

**S. K. Saha**[1]**, S. V. Shah**[2]**, and P. V. Nandihal**[1]

[1]Department of Mechanical Engineering, Indian Institute of Technology Delhi, Hauz Khas,
New Delhi 110 016, India
[2]Department of Mechanical Engineering, McGill University, Montreal, Canada

*Correspondence to:* S. K. Saha (saha@mech.iitd.ac.in)

**Abstract.** Dynamic modelling of a multibody system plays very essential role in its analyses. As a result, several methods for dynamic modelling have evolved over the years that allow one to analyse multibody systems in a very efficient manner. One such method of dynamic modelling is based on the concept of the Decoupled Natural Orthogonal Complement (DeNOC) matrices. The DeNOC-based methodology for dynamics modelling, since its introduction in 1995, has been applied to a variety of multibody systems such as serial, parallel, general closed-loop, flexible, legged, cam-follower, and space robots. The methodology has also proven useful for modelling of proteins and hyper-degree-of-freedom systems like ropes, chains, etc. This paper captures the evolution of the DeNOC-based dynamic modelling applied to different type of systems, and its benefits over other existing methodologies. It is shown that the DeNOC-based modelling provides deeper understanding of the dynamics of a multibody system. The power of the DeNOC-based modelling has been illustrated using several numerical examples.

## 1 Introduction

Over the last two decades, applications of multibody dynamics have expanded over the fields of robotics, automobile, aerospace, bio-mechanics, and many others. With continuous development in the above mentioned fields, many complex multibody systems have evolved whose dynamics play a pivotal role in their behaviour. Hence, computer-aided dynamic analysis of multibody systems has been a prime motive to the engineers, as high speed computing facilities are readily available. In order to perform computer-aided dynamic analysis, the actual system is represented with its dynamic model which has the information of its link parameters, joint variables and constraints. The dynamic model is nothing but the equations of motion of the multibody system at hand derived from the physical laws of motions. For a system with fewer links, it is easier to obtain explicit expressions for the equations of motion. However, finding equations of motion for complex systems with many links is not an easy task. Sometimes even with 4 or 5 links, say, a 4-bar mechanism, it is

difficult to find an explicit expression for the system's inertia in terms of its link lengths, masses, and joint angles. Hence, development of the equations of motion is an essential step for the dynamic analysis.

There are several fundamental methods for the formulation of equations of motion (Greenwood, 1988). For example, Newton-Euler (NE) formulation, Euler-Lagrange principle, Gibbs-Appel approach, Kane's method, D'Alembert's principle, and similar others. All the above mentioned approaches when applied to multibody systems have their own advantages and disadvantages. For example, NE approach, which is one of the classical methods for dynamic formulation, uses the concept of "free-body diagrams". For coupled systems, constrained forces (which are meant here to include both forces and moments) along with those applied externally are included in the free-body diagrams. Mathematically, the NE equations of motion lead to three translational equations of motion of the Centre-of-Mass (COM), and three equations determining the rotational motion of the rigid body. The NE equations of any two free bodies are

related through the constraint forces acting at their interface. The constraint forces arise due to the presence of a kinematic pair, e.g., a revolute or a prismatic, between the two neighbouring bodies. For an open-loop multibody system, these constraints along with other unknowns, i.e., the actuating forces can be easily solved recursively. However, for a closed-loop system, the NE equations generally need to be solved simultaneously in order to obtain the driving and constraint forces together. Hence, the use of the NE equations of motion for closed-loop systems is not as efficient as those for open-loop systems.

Euler-Lagrange (EL) formulation is another classical approach which is widely used for dynamic modelling. The EL formulation uses the concept of generalized coordinates instead of Cartesian coordinates. It is based on the minimization of a functional called "Lagrangian" which is nothing but the difference between kinetic energy and potential energy of the system at hand. For open-loop multibody systems, where typically the number of generalized coordinates equals the degree-of-freedom of a system, the constraint forces do not appear in the equations of motion. For closed-loop multibody systems, however, the forces of constraints appear as Lagrange's multipliers.

Kane's formulation (Kane and Levinson, 1983), which is same as the Lagrange's form of D'Alembert's principle, has also been used by many researchers for the development of equations of motion. It is found to be more beneficial than other formulations when used for systems with nonholonomic constraints. Several other methods of dynamic formulations were also proposed in the literature. For example, Khatib (1987) presented the operational-space formulation, whereas Angeles and Lee (1988) presented the natural orthogonal complement (NOC) based approach. Blajer et al. (1994) have also presented an orthogonal complement based formulation for the constrained multibody systems. Park et al. (1995) presented robot dynamics using a Lie group formulation, while Stokes and Brockett (1996) derived the equations of the motion of a kinematic chain using concepts associated with the special Euclidean group. McPhee (1996) showed how to use linear graph theory in multibody system dynamics. Cameron and Book (1997) described a technique based on Boltzmann-Hamel equations to derive dynamic equations of motion. Comprehensive discussion on dynamic formalisms can be found in the seminal text by Roberson and Schwertassek (1988), Schiehlen (1990, 1997), Shabana (2001), and Wittenburg (2008). Recent trends in dynamic formalisms can also be found in the work by Eberhard and Schiehlen (2006).

## 1.1    Natural Orthogonal Complement (NOC)

It is pointed out here that the Newton-Euler (NE) equations of motion are still found to be popular in the literature of dynamic modelling and analyses. However, it requires solution of the constraint forces which do not play any role in the motion of a system. Hence, extra calculations are required in motion studies. To avoid such extra calculations, there are formulations proposed in the literature where the equations of motion in the Euler-Lagrange (EL) form are obtained from the NE equations. Huston and Passerello (1974) were first to introduce a computer oriented method to reduce the dimension of the unconstrained NE equations by eliminating the constraint forces. Later, Kim and Vanderploeg (1986) derived the equations of motion in terms of relative joint coordinates from Cartesian coordinates through the use of velocity transformation matrix. Velocity transformation matrix relates linear and angular velocities of the links with joint velocities. It is worth noting here that the vector of constraint forces is orthogonal to the columns of the velocity transformation matrix. More precisely, the columns of the velocity transformation matrix span the nullspace of the matrix of velocity constraints. Hence, the said velocity transformation matrix is also referred to as an "orthogonal complement matrix". The phrase "orthogonal complement" was first coined by Hemami and Weimer (1981) for the modelling of nonholonomic systems. Orthogonal complements are not unique. In some approaches, it was obtained numerically, e.g., using singular value decomposition or treating it as an eigen value problem (Wehage and Haug, 1982; Kamman and Huston, 1984, Mani et al., 1985), which are computationally inefficient.

Alternatively, Angeles and Lee (1988) presented a methodology where they derived an orthogonal complement naturally from the velocity constraints. Hence, the name Natural Orthogonal Complement (NOC) was attached to their methodology. The NOC matrix, when combined with the NE equations of motion, leads to the minimal-order constrained dynamic equations of motion by eliminating the constraint forces. This facilitates the representation of the equations of motion in Kane's form that is suitable for recursive computation in inverse dynamics or in the EL form that is suitable for forward dynamics and integration. Later, Angeles and Ma (1988), Cyril (1988), Angeles et al. (1989), and Saha and Angeles (1991) showed the effectiveness of the use of the NOC matrix while applied to systems with holonomic and nonholonomic constraints.

## 1.2    The Decoupled NOC (DeNOC)

Subsequently, Saha (1995, 1997) presented the decoupled form of the NOC for the serial multibody systems. The two resulting block matrices, namely, an upper block triangular and a block diagonal matrices, are referred to as the Decoupled NOC (DeNOC) matrices. In contrast to the NOC, the DeNOC matrices allow one to recursively obtain the analytical expressions of the vectors and matrices appearing in the equations of motion (Saha, 1999a). This in turn helps to analytically decompose the Generalized Inertia Matrix (GIM) arising out of the constrained equations of motion of the system at hand, allowing one to obtain analytical inverse of the

GIM (Saha, 1999b) and a recursive algorithm for forward dynamics (Saha, 2003). Later, Saha and Schiehlen (2001) showed the power of the DeNOC matrices in obtaining recursive algorithms for the dynamics analyses of closed-loop parallel systems. Subsequently, Khan et al. (2005) illustrated the effectiveness of the DeNOC-based methodology in modelling parallel manipulators. Inspired by the concept of the DeNOC matrices, Dimitrov (2005) used a similar method for dynamic analysis, trajectory planning, and control of space robots. Garcia de Jalon et al. (2005) have also derived matrices which they have pointed out to be similar to the DeNOC matrices of Saha (1995, 1997). The DeNOC matrices have also found an application in the architecture design of a manipulator through its dynamic model simplifications (Saha et al., 2006). More recently, Chaudhary and Saha (2007) have applied the concept of the DeNOC matrices for the dynamic analyses of general closed-loop systems. They have also introduced the concepts like "determinate" and "indeterminate" subsystems which helped to achieve subsystem-level recursions for the inverse dynamics of a general closed-loop system. Systems with closed-loops which are used in automobile steering systems were analyzed by Hanzaki et al. (2009), whereas fuel injection pumps of diesel engines with rolling contacts were analyzed by Sundarranan et al. (2012). Extending the concept of the DeNOC matrices to other type of systems, Mohan and Saha (2007) showed how to derive the DeNOC matrices for a rigid-flexible multibody system. The methodology not only provided efficient dynamic algorithms but also produced numerically stable results. Very recently, Shah et al. (2012a) introduced a concept of "kinematic module" to a tree-type multibody system and derived module-level DeNOC matrices, which provided macroscopic purview of the multibody systems. Moreover, intra- and inter-modular recursive algorithms were derived for the analyses and control of legged robots (Shah, 2011; Shah et al., 2013). It was shown that the concept of Euler-angle-joints (EAJs) (Shah et al., 2012b) coupled with the module-level DeNOC matrices provided very efficient dynamic algorithms for the multibody system consisting of multiple branches and multiple-degrees-of-freedom joints. The algorithms have been implemented in a free software called ReDySim (acronym for Recursive Dynamic Simulator), which can be downloaded free from http://www.redysim.co.nr. ReDySim can be easily used by the students and researchers of multibody dynamics. Note here that the DeNOC-based algorithm was also used by the researchers from other domain, e.g., Patriciu et al. (2004) have adopted the concept for the analysis of conformational dependence of mass-metric tensor determinants in serial polymers with constraints.

The main motivation behind this paper is to bring forth the developments of the DeNOC-based dynamic modelling for multibody systems, which have taken place over more than one and half decades. The paper explains the fundamental principles of the DeNOC-based formulation, their benefits

and applications. Rest of the paper is organized as follows: Sect. 2 presents the DeNOC-based dynamic modelling for serial-chain systems, which forms the basis for the dynamic modelling of other type of systems, e.g., tree-type systems explained in Sect. 3. Application to closed-loop systems is explained in Sect. 4, whereas two software, namely, RoboAnalyzer and ReDySim, developed for the use by the students and researchers of multibody dynamics are explained in Sect. 5. The computational aspects are provided in Sect. 6. Finally, conclusions are given in Sect. 7.

## 2 DeNOC-based dynamic modelling for serial-chain systems

The Natural Orthogonal Complement (NOC) matrix proposed by Angeles and Lee (1988) relates the angular and linear velocities of the rigid bodies in a mechanical system to its associated joint-rates. It is used to develop a set of independent equations of motion from the unconstrained or uncoupled Newton-Euler (NE) equations using free-body diagrams. These independent set of equations was referred by the authors as the Euler-Lagrange equations of motion. Unlike the NOC, its decoupled form, i.e., the DeNOC, proposed by Saha (1995, 1997), allows one to write the expressions of each element of the matrices and vectors associated with the dynamic equations of motion in analytical recursive form.

### 2.1 Preliminaries and notation

An open-loop serial-chain system, e.g., a robotic manipulator shown in Fig. 1, has a fixed-base, denoted by #0, and $n$ moving rigid bodies or links, indicated with #1, ..., #$n$, coupled by $n$ single degree-of-freedom (DOF) kinematic pairs or joints numbered as 1, ..., $n$. The joints are generally revolute or prismatic. In presence of higher-DOF joints, they are modelled as combinations of single-DOF joints. For example, a spherical joint can be modelled as three intersecting revolute joints, whereas a cylindrical joint is modelled as a combination of revolute and prismatic joints. Few terms are defined below which will be used throughout the paper for the derivation of the dynamic models.

The 6-dimensional vectors, twist ($t_i$) of the $i$-th rigid link undergoing motion in the 3-dimensional Cartesian space and wrench ($w_i$), acting on the $i$-th link are defined by:

$$t_i \equiv \begin{bmatrix} \omega_i \\ v_i \end{bmatrix} \text{ and } w_i \equiv \begin{bmatrix} n_i \\ f_i \end{bmatrix} \qquad (1)$$

where $\omega_i$ is the 3-dimensional vector of angular velocity, and $v_i$ is the 3-dimensional vector of linear velocity of the mass center ($C_i$) of the $i$-th link, whereas $n_i$ and $f_i$ are the 3-dimensional vectors of the moment and force applied about and at $C_i$, respectively. The $6 \times 6$ matrices of mass $\mathbf{M}_i$, and angular velocity $\mathbf{W}_i$, of the $i$-th body are represented by:

$$\mathbf{M}_i \equiv \begin{bmatrix} \mathbf{I}_i & \mathbf{O} \\ \mathbf{O} & m_i \mathbf{1} \end{bmatrix} \text{ and } \mathbf{W}_i \equiv \begin{bmatrix} \omega_i \times \mathbf{1} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} \end{bmatrix} \qquad (2)$$
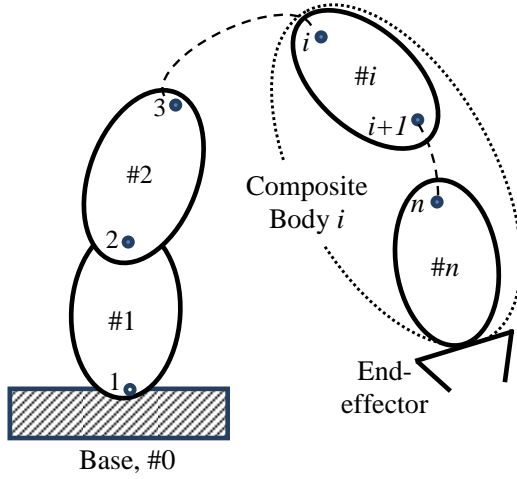
Figure 1. A robot manipulator.



Figure 2. A coupled link system.

where $\omega_i \times \mathbf{1}$ is the $3 \times 3$ cross-product tensor associated with the angular velocity vector $\omega_i$ which when operates on any 3-dimensional Cartesian vector $\boldsymbol{x}$ leads to the cross-product vector between $\omega_i$ and $\boldsymbol{x}$, i.e., $(\omega_i \times \mathbf{1})\boldsymbol{x} \equiv \omega_i \times \boldsymbol{x}$. Also, $\mathbf{1}$ and $\mathbf{O}$ are the $3 \times 3$ identity and zero matrices, respectively, whereas $\mathbf{I}_i$ and $m_i$ are the $3 \times 3$ inertia tensor about $C_i$, and the mass of the $i$-th link, respectively. For the serial-chain mechanical system shown in Fig. l, the method to obtain the dynamic equations of motion using the DeNOC matrices is as follows:

– Derive the DeNOC matrices.

– Obtain the unconstrained NE equations of motion from the free-body diagrams of each link, and

– Couple the DeNOC matrices with the unconstrained NE equations to obtain a set of constrained independent equations of motion which are same as the system's EL equations of motion.

The above steps are explained next in the following subsections.

## 2.2 Kinematic constraints

The kinematic constraints in terms of the velocities of two neighbouring links, say, #$i$ and #$j$, coupled by a revolute joint, as shown in Fig. 2, are given by

$$\omega_i = \omega_j + \dot{\theta}_i e_i \tag{3a}$$

$$\boldsymbol{v}_i = \boldsymbol{v}_j + \omega_j \times \boldsymbol{r}_j + \omega_i \times \boldsymbol{d}_i \tag{3b}$$

where $\omega_j$ and $\boldsymbol{v}_j$ are the angular velocity and velocity of the mass of link $j$, i.e., $C_j$, respectively. Similarly, $\omega_i$ and $\boldsymbol{v}_i$ are defined for the neighbouring link $i$, whereas $\dot{\theta}_i$ is the joint-rate of the $i$-th joint. The above six scalar equations can be
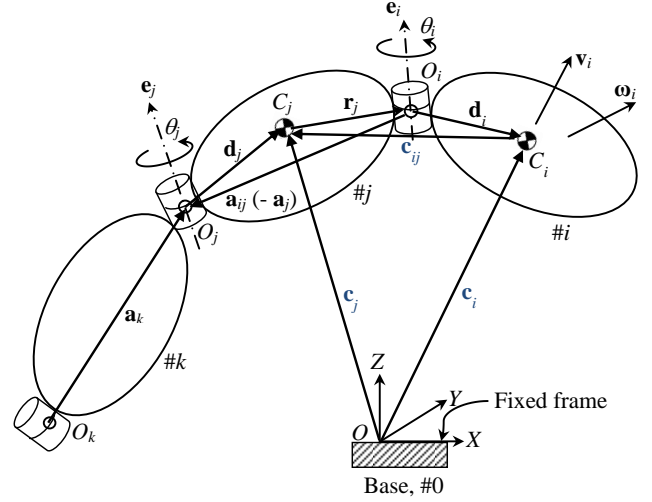
written in a compact form as

$$\boldsymbol{t}_i = \mathbf{B}_{ij}\boldsymbol{t}_j + \boldsymbol{p}_i\dot{\theta}_i \tag{4}$$

where $\mathbf{B}_{ij}$ is the $6 \times 6$ matrix and $\boldsymbol{p}_i$ is the 6-dimensional vector which are given by

$$\mathbf{B}_{ij} \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ c_{ij} \times \mathbf{1} & \mathbf{1} \end{bmatrix} \text{ and } \boldsymbol{p}_i \equiv \begin{bmatrix} e_i \\ e_i \times \boldsymbol{d}_i \end{bmatrix} \tag{5}$$

Here, $\boldsymbol{c}_{ij}$ is the 3-dimensional position vector from $C_i$ to $C_j$ given by $\boldsymbol{c}_{ij} \equiv -\boldsymbol{d}_i - \boldsymbol{r}_j$, and $\boldsymbol{c}_{ij} \times \mathbf{1}$ is the cross-product tensor associated with vector $\boldsymbol{c}_{ij}$. It is defined similar to $\omega_i \times \mathbf{1}$ of Eq. (2). Moreover, $\boldsymbol{e}_i$ is the unit vector parallel to the axis of rotation of the $i$-th revolute joint. Interestingly, matrix $\mathbf{B}_{ij}$ and vector $\boldsymbol{p}_i$ have the following interpretations:

– If links #$i$ and #$j$ are rigidly attached, $\mathbf{B}_{ij}$ propagates twist or velocities of #$j$ to #$i$. Hence, $\mathbf{B}_{ij}$ is termed in Saha (1999a) as the *twist-propagation* matrix, which satisfies

$$\mathbf{B}_{ij}\mathbf{B}_{jk} = \mathbf{B}_{ik} \text{ and } \mathbf{B}_{ii} = \mathbf{1} \tag{6}$$

– On the other hand the vector $\boldsymbol{p}_i$ takes into account the motion of the $i$-th joint. Hence, vector $\boldsymbol{p}_i$ is termed as the *joint-rate-propagation* vector. The vector $\boldsymbol{p}_i$ in Eq. (5) is defined for a revolute joint. For a prismatic joint, it is given by

$$\boldsymbol{p}_i \equiv \begin{bmatrix} \mathbf{0} \\ e_i \end{bmatrix} \tag{7}$$

Equation (4) can be written for $i = 1, ..., n$, as

$$(\mathbf{1} - \mathbf{B})\boldsymbol{t} = \mathbf{N}_{\mathrm{d}}\dot{\boldsymbol{\theta}} \tag{8a}$$

where $\mathbf{1}$ is the $6n \times 6n$ identity matrix, and the $6n \times 6n$ matrix $\mathbf{B}$ has the following representation:

$$\mathbf{B} = \begin{bmatrix} \mathbf{O} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{B}_{21} & \mathbf{O} & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O} & \cdots & \mathbf{B}_{n,n-1} & \mathbf{O} \end{bmatrix} \tag{8b}$$

It is now simple matter to invert the $6n \times 6n$ matrix, $(\mathbf{1} - \mathbf{B})$, and hence, Eq. (8a) can be rewritten as

$$t = \mathbf{N}\dot{\theta}, \quad \text{where} \quad \mathbf{N} \equiv \mathbf{N}_\mathrm{l}\mathbf{N}_\mathrm{d} \tag{9a}$$

In Eq. (9a), the matrix $\mathbf{N}$ is the $6n \times n$ Natural Orthogonal Complement (NOC) matrix, as introduced by Angeles and Lee (1988), whereas $\mathbf{N}_\mathrm{l}$ and $\mathbf{N}_\mathrm{d}$ are the decoupled form of the NOC or the DeNOC matrices proposed first time in Saha (1995). The $6n \times 6n$ matrix $\mathbf{N}_\mathrm{l}$ and the $6n \times n$ matrix $\mathbf{N}_\mathrm{d}$ are given by

$$\mathbf{N}_\mathrm{l} = \begin{bmatrix} \mathbf{1} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{B}_{21} & \mathbf{1} & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}_{n1} & \mathbf{B}_{n2} & \cdots & \mathbf{1} \end{bmatrix} \quad \text{and}$$

$$\mathbf{N}_\mathrm{d} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{p}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{p}_n \end{bmatrix} \tag{9b}$$

Note that in Eq. (9b), $\mathbf{N}_\mathrm{l}$ is a lower block-triangular matrix, whereas $\mathbf{N}_\mathrm{d}$ is a block-diagonal matrix, as indicated through their subscripts "l" and "d", respectively. Moreover, $\mathbf{O}$ and $\mathbf{0}$ are the $6 \times 6$ matrix of zeros and the 6-dimensional vector of zeros, respectively. The $n$-dimensional vector $\dot{\theta}$ is defined as

$$\dot{\theta} \equiv [\dot{\theta}_1, \cdots, \dot{\theta}_n]^T \tag{10}$$

which contains the joint-rates of all the joints in the serial-chain system shown in Fig. 1.

## 2.3   Unconstrained Newton-Euler (NE) equations

The unconstrained or uncoupled Newton-Euler (NE) equations of motion for the $i$-th rigid-link (Saha, 1999a) can be written from its free-body diagram, Fig. 3, as

$$\mathbf{I}_i\dot{\omega}_i + \omega_i \times \mathbf{I}_i\omega_i = n_i \tag{11a}$$

$$m_i\dot{v}_i = f_i \tag{11b}$$

where $\dot{\omega}_i$ and $\dot{v}_i$ are the angular acceleration and acceleration of the mass center $C_i$, respectively. Moreover, $\mathbf{I}_i$ is the $3 \times 3$ inertia tensor of $i$-th link about its mass center $C_i$, and $m_i$ is its mass. Other variables were defined after Eq. (1). The above six scalar equations can be put in a compact form as
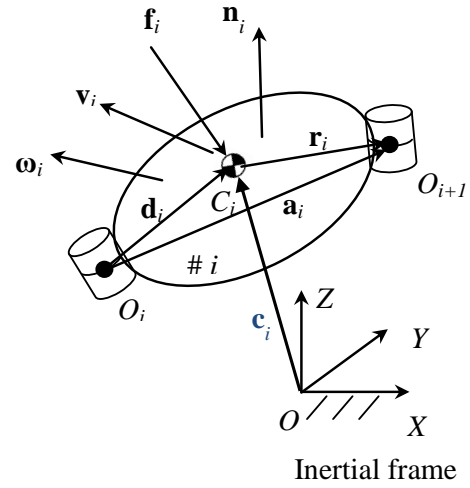


**Figure 3.** Free-body diagram of the $i$-th link.

$$\mathbf{M}_i\dot{t}_i + \mathbf{W}_i\mathbf{M}_it_i = w_i \tag{12}$$

where $t_i$, $w_i$ and $\mathbf{W}_i$, $\mathbf{M}_i$ are defined in Eqs. (1) and (2), respectively. Moreover, $\dot{t}_i$ is the time derivative of the twist $t_i$ of the $i$-th link. For the whole system of $n$ rigid links, the $6n$ scalar equations (for $i = 1, ..., n$, where $n$ is the number of moving rigid links in the serial chain system) can be written as

$$\mathbf{M}\dot{t} + \mathbf{W}\mathbf{M}t = w \tag{13}$$

In Eq. (13), $\dot{t}$ is the time derivative of the generalized twist, $t$. Moreover, $\mathbf{M}$ and $\mathbf{W}$ are the $6n \times 6n$ generalized mass matrix and generalized matrix of angular velocities, respectively, i.e.,

$$\mathbf{M} \equiv \mathrm{diag.}[\mathbf{M}_1, \cdots, \mathbf{M}_n] \quad \text{and} \quad \mathbf{W} \equiv \mathrm{diag.}[\mathbf{W}_1, \cdots, \mathbf{W}_n] \tag{14}$$

Moreover, $w$ and $t$ are the $6n$-dimensional vectors of generalized wrench and twist, respectively. They are defined as

$$w \equiv [w_1^T, \cdots, w_n^T]^T \quad \text{and} \quad t \equiv [t_1^T, \cdots, t_n^T]^T \tag{15}$$

## 2.4   Constrained equations using the DeNOC matrices

The kinematic constraints in velocities, i.e., Eq. (9a), then can be incorporated into the unconstrained NE equations of motion, Eq. (13). This is done by pre-multiplying $\mathbf{N}^T$ with the $6n$ unconstrained NE equations of motions of Eq. (13), i.e.,

$$\mathbf{N}^T \left( \mathbf{M}\dot{t} + \mathbf{W}\mathbf{M}t \right) = \mathbf{N}^T \left( w^\mathrm{E} + w^\mathrm{C} \right) \tag{16}$$

where $w$ is substituted as, $w \equiv w^\mathrm{E} + w^\mathrm{C}$, in which $w^\mathrm{E}$ and $w^\mathrm{C}$ are the $6n$-dimensional vectors of external and constraint wrenches, respectively. Since the constraint wrenches do

not do any work, $\mathbf{N}^T \boldsymbol{w}^{\mathrm{C}}$ vanishes (Angeles and Lee, 1988). Hence, $\mathbf{N}^T \boldsymbol{w}^{\mathrm{C}} = \mathbf{0}$. Substituting the expression of $\boldsymbol{t}$ from Eq. (9a) and its time derivative, $\dot{\boldsymbol{t}} = \mathbf{N}^T \ddot{\theta} + \dot{\mathbf{N}} \theta$ into Eq. (16), one can get the $n$ independent scalar dynamic equations of motion, namely,

$$\mathbf{I}\ddot{\theta} + \mathbf{C}\dot{\theta} = \tau \tag{17}$$

where, $\mathbf{I} \equiv \mathbf{N}^T \mathbf{M} \mathbf{N}$: the $n \times n$ generalized inertia matrix (GIM); $\mathbf{C} \equiv \mathbf{N}^T (\mathbf{M}\mathbf{N} + \mathbf{W}\mathbf{M}\mathbf{N})$: the $n \times n$ matrix of convective inertia terms (MCI); and $\tau \equiv \mathbf{N}^T \boldsymbol{w}^{\mathrm{E}}$: the $n$-dimensional vector of generalized forces of driving, and those resulting from gravity, dissipation, and other external forces like foot-ground interaction of a walking robot, etc., if any.

## 2.5   Analytical expression of the GIM

The analytical expression of the generalized inertia matrix (GIM) appearing in Eq. (17) plays an important role in simplifying, mainly, the forward dynamics agorithm (Saha, 1999a, 2003). In this section, the GIM $\mathbf{I}$ is derived using the expressions of the DeNOC matrices (Saha, 1995, 1997, 1999a, b, 2003). Substituting the expressions of the DeNOC matrices given by Eq. (9b) into the expression of the GIM appearing after Eq. (17), one gets

$$\mathbf{I} = \mathbf{N}_{\mathrm{d}}^T \tilde{\mathbf{M}} \mathbf{N}_{\mathrm{d}}, \quad \text{where} \quad \tilde{\mathbf{M}} \equiv \mathbf{N}_{\mathrm{l}}^T \mathbf{M} \mathbf{N}_{\mathrm{l}} \tag{18}$$

The $6n \times 6n$ symmetric matrix $\tilde{\mathbf{M}}$ can be written as

$$\tilde{\mathbf{M}} \equiv \begin{bmatrix} \tilde{\mathbf{M}}_1 & \mathbf{B}_{21}^T \tilde{\mathbf{M}}_2 & \cdots & \mathbf{B}_{n1}^T \tilde{\mathbf{M}}_n \\ \tilde{\mathbf{M}}_2 \mathbf{B}_{21} & \tilde{\mathbf{M}}_2 & \cdots & \mathbf{B}_{n2}^T \tilde{\mathbf{M}}_n \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{\mathbf{M}}_n \mathbf{B}_{n1} & \tilde{\mathbf{M}}_n \mathbf{B}_{n2} & \cdots & \tilde{\mathbf{M}}_n \end{bmatrix} \tag{19}$$

where the $6 \times 6$ matrix, $\tilde{\mathbf{M}}_i$, for $i = 1, \cdots, n$, can be obtained recursively, i.e.,

$$\tilde{\mathbf{M}}_i = \mathbf{M}_i + \mathbf{B}_{i+1,i}^T \tilde{\mathbf{M}}_{i+1} \mathbf{B}_{i+1,i} \tag{20}$$

in which $\tilde{\mathbf{M}}_{i+1} \equiv \mathbf{O}$, because there is no $(n+1)$st link in the serial-chain. Hence, $\tilde{\mathbf{M}}_n \equiv \mathbf{M}_n$. The matrix, $\tilde{\mathbf{M}}_i$, is interpreted as the mass matrix of the *Composite Body, i*, that consists of rigidly connected links #$i$, ..., #$n$, as indicated in Fig. 1. Finally, the $n \times n$ GIM $\mathbf{I}$ can be expressed as

$$\mathbf{I} \equiv \begin{bmatrix} i_{11} & & \mathrm{sym} \\ \vdots & \ddots & \\ i_{n1} & \cdots & i_{nn} \end{bmatrix}, \quad \text{where} \quad i_{ij} \equiv \boldsymbol{p}_i^T \tilde{\mathbf{M}}_i \mathbf{B}_{ij} \boldsymbol{p}_j \tag{21}$$

for $i = 1, ..., n$; $j = 1, ..., i$. The term $i_{ij}$ is a scalar and "sym" denotes symmetric elements of the GIM $\mathbf{I}$.

## 2.6   Recursive inverse dynamics algorithm

The inverse dynamics of a serial-chain system is defined as the process of determining the joint forces/torques when the joint motions of the system are known. The inverse dynamics algorithm calculates the joint torque, $\tau_i$, for $i = 1, ..., n$, in two recursive steps, namely, forward and backward recursions. They are given below.

### 2.6.1   Step 1: forward recursion

First, the 6-dimensional twist and twist-rate vectors of each link, i.e., $\boldsymbol{t}_i$ and $\dot{\boldsymbol{t}}_i$, respectively, are calculated, for $i = 1, ..., n$, using the following relations:

$$\boldsymbol{t}_i = \mathbf{B}_{i,i-1} \boldsymbol{t}_{i-1} + \boldsymbol{p}_i \dot{\theta}_i \tag{22}$$

$$\dot{\boldsymbol{t}}_i = \mathbf{B}_{i,i-1} \dot{\boldsymbol{t}}_{i-1} + \dot{\mathbf{B}}_{i,i-1} \boldsymbol{t}_{i-1} + \boldsymbol{p}_i \ddot{\theta}_i + \dot{\boldsymbol{p}}_i \dot{\theta}_i \tag{23}$$

$$\boldsymbol{w}_i = \mathbf{M}_i \dot{\boldsymbol{t}}_i + \mathbf{W}_i \mathbf{M}_i \boldsymbol{t}_i \tag{24}$$

In the above equations, $\boldsymbol{t}_0 = \mathbf{0}$ and $\dot{\boldsymbol{t}}_0 = \mathbf{0}$, as link #0 is fixed without any motion.

### 2.6.2   Step 2: backward recursion

The 6-dimensional vector, $\tilde{\boldsymbol{w}}_i$, and the scalar, $\tau_i$, for $i = n, ..., 1$, are calculated using the following relations:

$$\tilde{\boldsymbol{w}}_i = \mathbf{B}_{i+1,i}^T \tilde{\boldsymbol{w}}_{i+1}, \quad \text{and} \quad \tau_i = \boldsymbol{p}_i^T \tilde{\boldsymbol{w}}_i \tag{25}$$

where for $i = n$, $\tilde{\boldsymbol{w}}_{n+1} = 0$, as there is no $(n+1)$st link in the system. Hence, $\tilde{\boldsymbol{w}}_n = \boldsymbol{w}_n$. The effect of gravity can also be taken into account by providing negative acceleration due to gravity, $\boldsymbol{g}$, to the twist-rate of the first link as an additional term (Kane and Levinson, 1983), i.e.,

$$\dot{\boldsymbol{t}}_1 = \boldsymbol{p}_1 \ddot{\theta}_1 + \dot{\boldsymbol{p}}_1 \dot{\theta}_1 + \rho, \quad \text{where} \quad \rho \equiv \begin{bmatrix} \mathbf{0}^T, -\boldsymbol{g}^T \end{bmatrix} \tag{26}$$

Note that Eqs. (22)–(26) were reported in Saha (1999a) with different notations, which actually have the same interpretations as given above, i.e., twist ($\boldsymbol{t}_i$), twist-rate ($\dot{\boldsymbol{t}}_i$), wrench of composite body ($\tilde{\boldsymbol{w}}_i$), etc. Based on the above mentioned recursive inverse dynamics algorithm, a computer program was developed in C++ which was called RIDIM (Recursive Inverse Dynamic for Industrial Manipulators) (Saha, 1999a). Recently, a similar algorithm has been rewritten in Visual C# and implemented in the "IDyn" module of the newly developed software called RoboAnalyzer (Rajeevlochana and Saha, 2011; Rajeevlochana et al., 2012) which also has 3-dimensional visualisation of the system under study. It is explained in Sect. 6.1, and available free from http://www.roboanalyzer.com for the benefits of students and researchers of multibody dynamics community.

## 2.7 Recursive forward dynamics algorithm

Forward dynamics of a serial-chain system is defined as the process of determining the joint accelerations when the joint-actuator torques/forces of the system are known. In order to compute the joint accelerations $\ddot{\boldsymbol{\theta}}$ recursively, the GIM, $\mathbf{I}$ of Eq. (17), is decomposed as $\mathbf{I} \equiv \mathbf{U}\mathbf{D}\mathbf{U}^T$ (Saha, 1995, 1997, 1999b) based on the Reverse Gaussian Elimination (RGE) method, where $\mathbf{U}$ and $\mathbf{D}$ are upper triangular and diagonal matrices, respectively. The $\mathbf{U}\mathbf{D}\mathbf{U}^T$ decomposition results in an efficient order $n$, i.e., $O(n)$, computational algorithm in contrast to $O(n^3)$ computations required by the Cholesky decomposition of the GIM (Strang, 1998).

For the development of recursive $O(n)$ forward dynamics algorithm, the constrained dynamics equations of motion, Eq. (17), are rewritten as

$$\mathbf{U}\mathbf{D}\mathbf{U}^T\ddot{\boldsymbol{\theta}} = \boldsymbol{\varphi} \tag{27}$$

where $\boldsymbol{\varphi} \equiv \boldsymbol{\tau} - \mathbf{C}\dot{\boldsymbol{\theta}}$. Then, three recursive steps are used to calculate the joint accelerations, which are given below.

### 2.7.1 Step 1

Solution for $\hat{\boldsymbol{\tau}}$, where $\hat{\boldsymbol{\tau}} \equiv \mathbf{D}\mathbf{U}^T\ddot{\boldsymbol{\theta}} \equiv \mathbf{U}^{-1}\boldsymbol{\varphi}$. It is found as follows: For $i = n - 1, ..., 1$, calculate

$$\hat{\tau}_i = \varphi_i - \boldsymbol{p}_i^T \boldsymbol{\eta}_{i,i+1} \tag{28}$$

where $\boldsymbol{\eta}_{i,i+1}$ is the 6-dimensional vector obtained recursively as

$$\boldsymbol{\eta}_{i,i+1} \equiv \mathbf{B}_{i+1,i}^T \boldsymbol{\eta}_{i+1} \quad \text{and} \quad \boldsymbol{\eta}_{i+1} \equiv \hat{\tau}_{i+1}\boldsymbol{\psi}_{i+1} + \boldsymbol{\eta}_{i+1,i+2} \tag{29}$$

in which $\boldsymbol{\eta}_{n,n+1} = \mathbf{0}$, and the 6-dimensional vector $\boldsymbol{\psi}_{i+1}$ is evaluated using the following relations:

$$\boldsymbol{\psi}_i = \frac{\hat{\boldsymbol{\psi}}_i}{\hat{m}_i}, \quad \text{where} \quad \hat{\boldsymbol{\psi}}_i \equiv \hat{\mathbf{M}}_i \boldsymbol{p}_i \quad \text{and} \quad \hat{m}_i \equiv \boldsymbol{p}_i^T \hat{\boldsymbol{\psi}}_i \tag{30}$$

In Eq. (30), the $6 \times 6$ matrix, $\hat{\mathbf{M}}_i$ is obtained recursively as

$$\hat{\mathbf{M}}_i = \mathbf{M}_i + \mathbf{B}_{i+1,i}^T \overline{\mathbf{M}}_{i+1} \mathbf{B}_{i+1,i},$$

where $\overline{\mathbf{M}}_{i+1} \equiv \hat{\mathbf{M}}_{i+1} - \hat{\boldsymbol{\psi}}_{i+1}\boldsymbol{\psi}_{i+1}^T$ and $\hat{\mathbf{M}}_n = \mathbf{M}_n \tag{31}$

The $6 \times 6$ symmetric matrix $\hat{\mathbf{M}}_i$ is the mass matrix of *Articulated Body, i*, defined as the links #i, ..., #n, coupled by the joints $i+1$, ..., $n$. This is in contrast to the definition of the *Composite Body, i*, given after Eq. (20), where the links are rigidly connected, i.e., the joints are locked. Note that the mass matrix of the *i*-th *Articulate Body* $\hat{\mathbf{M}}_i$ is nothing but the Articulated-Body-Inertia (ABI) of Featherstone (1987).

### 2.7.2 Step 2

Solution for $\tilde{\boldsymbol{\tau}}$, where, $\tilde{\boldsymbol{\tau}} \equiv \mathbf{U}^T\ddot{\boldsymbol{\theta}} \equiv \mathbf{D}^{-1}\hat{\boldsymbol{\tau}}$. It is found as follows: for $i = 1, ..., n$,
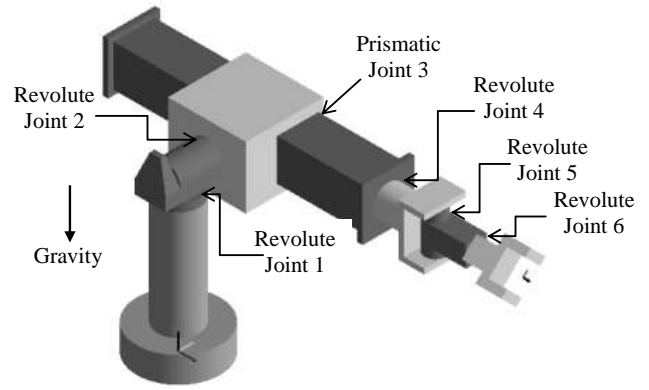
$$\tilde{\tau}_i = \frac{\hat{\tau}_i}{\hat{m}_i} \tag{32}$$



**Figure 4.** The Stanford arm.

### 2.7.3 Step 3

Solution for $\ddot{\boldsymbol{\theta}}$, where, $\ddot{\boldsymbol{\theta}} \equiv \mathbf{U}^{-T}\tilde{\boldsymbol{\tau}}$. It is found as follows: For $i = 2, ..., n$,

$$\ddot{\theta}_i = \tilde{\tau}_i - \boldsymbol{\psi}_i^T \boldsymbol{\mu}_{i,i-1} \tag{33}$$

where $\boldsymbol{\mu}_{i,i-1} \equiv \mathbf{B}_{i,i-1}\boldsymbol{\mu}_{i-1}$, $\boldsymbol{\mu}_{i-1} \equiv \boldsymbol{p}_{i-1}\ddot{\theta}_{i-1} + \boldsymbol{\mu}_{i-1,i-2}$, and for $i = 1$, $\boldsymbol{\mu}_{10} \equiv \mathbf{0}$.

Based on the above mentioned forward dynamics algorithm, another C++ program RFDSIM (Recursive Forward Dynamic and Simulation of Industrial Manipulators) was written which was reported in Saha (1999a). A similar algorithm was rewritten in Visual C# and implemented in the "FDyn" module of RoboAnalyzer software (Rajeevlochana et al., 2012; http://www.roboanalyzer.com) with which one can see animation of the systems under study. The numerical integrator used in RoboAnalyzer for the simulation purposes is based on the Runge-Kutta 4th order method (Bathe and Wilson, 1976).

## 2.8 Numerical example: a 6-DOF Stanford arm

The dynamic analyses of the 6-link 6-DOF serial-chain system with both revolute and prismatic joints, namely, the Stanford arm as shown in Fig. 4, were carried out using RoboAnalyzer. The Denavit and Hartenberg (DH) paramters, which were proposed by Denavit and Hartenberg (1955), and the mass and inertia propoerties are taken from Saha (1999a) as per the notations explained there and in Saha (2008). The numerical values are not reproduced here since the focus of this paper is to review the DeNOC-based formulations and their applicability. However, the joint torques (Joints 1–2, 4–6) and force (Joint 3) obtained from the "IDyn" module of RoboAnalyzer software for the following joint input motions are plotted in Fig. 5:

$$\theta_i = \theta_i(0) + \frac{\theta_i(T) - \theta_i(0)}{T}\left[t - \frac{T}{2\pi}\sin\left(\frac{2\pi}{T}t\right)\right]$$
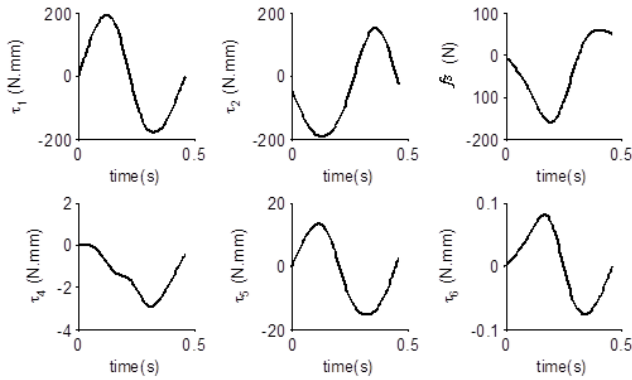
for $i = 1, 2, 4, 5, 6 \tag{34}$

**Figure 5.** Joint torques (1–2, 4–6) and force (3) for the Stanford arm.
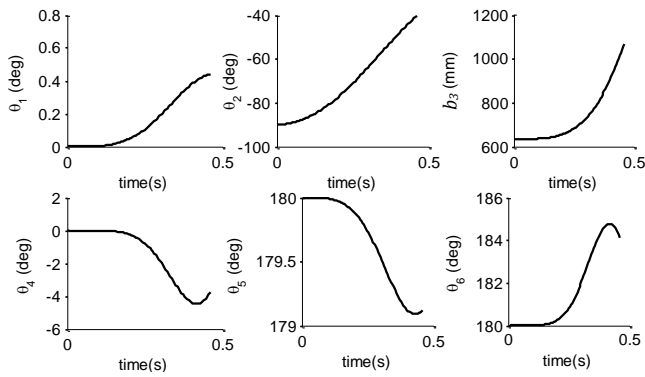


**Figure 6.** Simulated joint motions for the free-fall of the Stanford arm.

$$b_3 = b_3(0) + \frac{b_3(T) - b_3(0)}{T}\left[t - \frac{T}{2\pi}\sin\left(\frac{2\pi}{T}t\right)\right] \qquad (35)$$

where $\theta_i(0) = 0$, for $i = 1$–$2$, $4$–$6$ and $b_3(0) = 0$ are the variable DH parameters (Saha, 2008) or the joint variables at time $T = 0$, whereas the total time of motion is, $T = 10$ s. Gravity was acting in the negative $Z_1$-direction. The variable, $\tau_i$, for $i = 1$–$2$, $4$–$6$, and $f_3$ in Fig. 5 are the joint torques and force, respectively. The results were verified with those reported in Saha (2008).

The forward dynamics and simulation of the Stanford arm was also performed using "FDyn" module of RoboAnalyzer. The Stanford manipulator was assumed to fall freely under gravity without any external torques and force at the actuating joints. The initial positions were taken same as in the inverse dynamics analysis given after Eq. (35). The results are plotted in Fig. 6, where the variations of the joint motions with respect to time are shown. The results were also verified with those reported in Saha (2008).
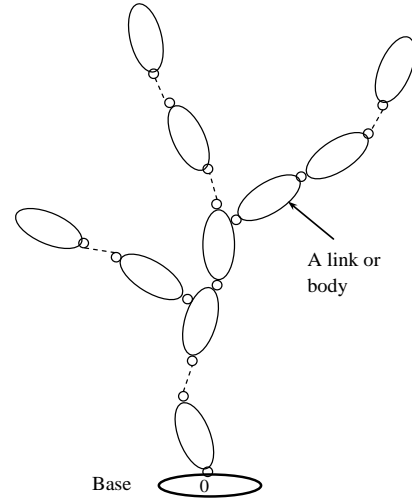


**Figure 7.** A tree-type system.

## 3   Tree-type systems

A tree-type system has a set of links connected by kinematic pairs, typically, a revolute or a prismatic joint, as shown in Fig. 7. Other type of joints, say, a universal or spherical, and a cylindrical, can be modelled as a combination of two or three intersecting revolute joints, and a pair of revolute-prismatic joints, respectively, as mentioned in the beginning of Sect. 2.1. Based on the modelling of serial-chain systems, Shah et al. (2011, 2013) extended the methodology to model a tree-type system. For this, the tree-type system was assumed to be a combination of several serial-chain systems called "kinematic modules". Consequently, multi-modular recursive algorithms for the tree-type systems were presented against "full-body-level" recursive dynamics algorithms of Featherstone (1987) and Rodriguez (1992). Each "module" of the tree-type architecture was defined as a set of serially connected links emerges from the last link of its parent module. For example, as indicated in Fig. 8, the parent module of $M_i$ is module $M_\beta$.

For the analyses purposes, the tree-type system was first kinematically modularized before its kinematic constraints were derived. The modules are denoted with $M_0$, $M_1$, $M_2$, etc., where a child module bears a number higher than its parent module. Moreover, the links inside any module, say, $M_i$, are denoted as $\#1^i$, ..., $\#k^i$, ..., $\#\eta^i$, where the superscript $i$ signifies the module number. Considering the tree-type system, there are $s$ number of modules in the system, and there are $\eta^i$ number of links in the $i$-th module. The total number of links in the whole system is then obtained by $n = \sum_i^s \eta^i$. The kinematic constraints were next derived at the intra-modular level, i.e., amongst the links inside a module, and inter-modular level, i.e., between the modules. The dynamic analyses were done using intra- (Inside the module)
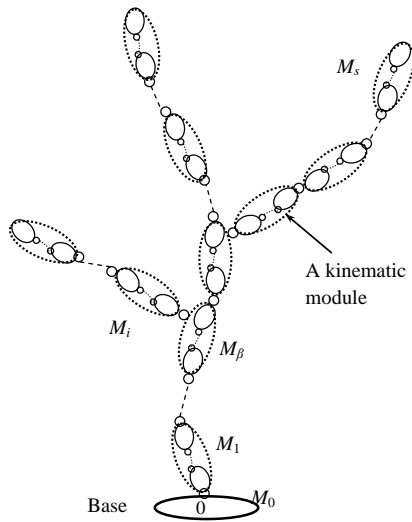
**Figure 8.** The multi-modular tree-type system.

and inter-modular (between the modules) recursions, as presented in Fig. 9.

### 3.1 Intra-modular kinematic constraints

Intra-modular kinematic constraints are effectively the velocity constraints between the links of a serial-chain system derived in Eqs. (3–10). Here, however, a little modification is proposed in the definition of each link's linear velocity $v_i$. In contrast to the definition of the velocity of the mass center of the $i$-th link, $C_i$, as $v_i$ of Eq. (1), it is defined in this section as the velocity of point $O_i$ where the $i$-th joint couples the $j$-th link with the $i$-th one, as indicated in Fig. 2. Such definition of $v_i$ in twist expression of Eq. (1) was necessitated mainly to take care of the branching issue of the serial-modules in the tree-type system, as shown in Fig. 8. The velocity of the $i$-th link defined here with respect to $O_i$ (sometimes referred to as the origin of the $i$-th link). It is actually the velocity of the previous link at its connection point, namely, the last link of the parent module where the first link of the child module is coupled. Hence, where branching occurs no additional computations are required for the calculation of the velocity of the first link belonging to the child module. This was not the case with the definition of the velocity of the $i$-th link with respect to its mass center $C_i$ in which additional computations would be required to calculate the velocity of the mass center $C_i$ from the origin $O_i$. Moreover, as the main objective of dynamic analyses is to calculate either joint torques or joint motions, selection of $O_i$ as a reference point, instead of the $C_i$, can lead to efficient recursive inverse and forward dynamics algorithms, as shown by Shah et al. (2011, 2013). In fact, for the serial-chain systems considered in Sect. 2, the same definition with respect to $O_i$ could have been adopted. This was done with the "IDyn" and "FDyn" modules of the RoboAnalyzer software. In Sect. 2, however, it was shown

how the simplest form of the NE equations of motion given by Eq. (1) can be used with the definition of the DeNOC matrices, as demonstrated in the original work of Saha (1995, 1997, 1999a, b, 2003).

Now, with the new definitions of $v_i$ with respect to $O_i$, Eq. (4) is rewritten as

$$t_i = \mathbf{A}_{ij} t_j + p_i \dot{\theta}_i \qquad (36\text{a})$$

where $t_i$ and $t_j$ are the 6-dimensional twist vectors defined in Eq. (1) but with respect to (w.r.t.) the new definition of $v_i$, i.e., w.r.t. point $O_i$. Accordingly, the $6 \times 6$ matrix $\mathbf{A}_{ij}$ is the new twist-propagation matrix. A different notation is used here to distinguish it from $\mathbf{B}_{ij}$ which was defined after Eq. (4) w.r.t. the definition of the velocity of $C_i$. The $6 \times 6$ matrix $\mathbf{A}_{ij}$, and the 6-dimensional joint-rate-propagation vector, $p_i$, are given by

$$\mathbf{A}_{ij} = \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ a_{ij} \times \mathbf{1} & \mathbf{1} \end{bmatrix}, \quad \text{and}$$

$$p_i = \begin{bmatrix} e_i \\ \mathbf{0} \end{bmatrix} \text{ for revolute}; \quad p_i = \begin{bmatrix} \mathbf{0} \\ e_i \end{bmatrix} \text{ for prismatic} \qquad (36\text{b})$$

where the 3-dimensional vector $a_{ij}$ is shown in Fig. 2. Notice the change in the expression of $p_i$ in Eq. (36b) in comparison to the same in Eq. (5) where $v_i$ was defined w.r.t. $C_i$. For serially connected rigid links in the $i$-th serial-chain module, one can write the expression for the generalized twist, $\bar{t}_i$, similar to Eq. (9a), as

$$\bar{t}_i = \overline{\mathbf{N}}_i \dot{\overline{\theta}}_i, \quad \text{where} \quad \overline{\mathbf{N}}_i \equiv [\overline{\mathbf{N}}_\mathrm{l} \overline{\mathbf{N}}_\mathrm{d}]^i \qquad (37)$$

In Eq. (37), the $6\eta^i$-dimensional generalized twist vector $\bar{t}_i$ and the $\eta^i$-dimensional generalized joint-rates vector $\dot{\overline{\theta}}_i$ are defined as follows:

$$\bar{t}_i \equiv \begin{bmatrix} t_1 \\ \vdots \\ t_k \\ \vdots \\ t_\eta \end{bmatrix}^i \quad \text{and} \quad \dot{\overline{\theta}}_i \equiv \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_k \\ \vdots \\ \dot{\theta}_\eta \end{bmatrix}^i \qquad (38)$$

where a bar ("–") over an entity in Eqs. (37) and (38) signifies that the quantity is related to a module and the superscript, $i$, outside the brackets identifies the module. As a consequence, the generic notation $t_k$ (or $t_{k^i}$) in Eq. (38) is the 6-dimensional twist vector for the $k^\text{th}$ link in the $i$-th module. The $6\eta^i \times 6\eta^i$ and $6\eta^i \times \eta^i$ DeNOC matrices for the serial-chain module, denoted as $\overline{\mathbf{N}}_{\mathrm{l}^i}$ and $\overline{\mathbf{N}}_{\mathrm{d}^i}$, respectively, are given by
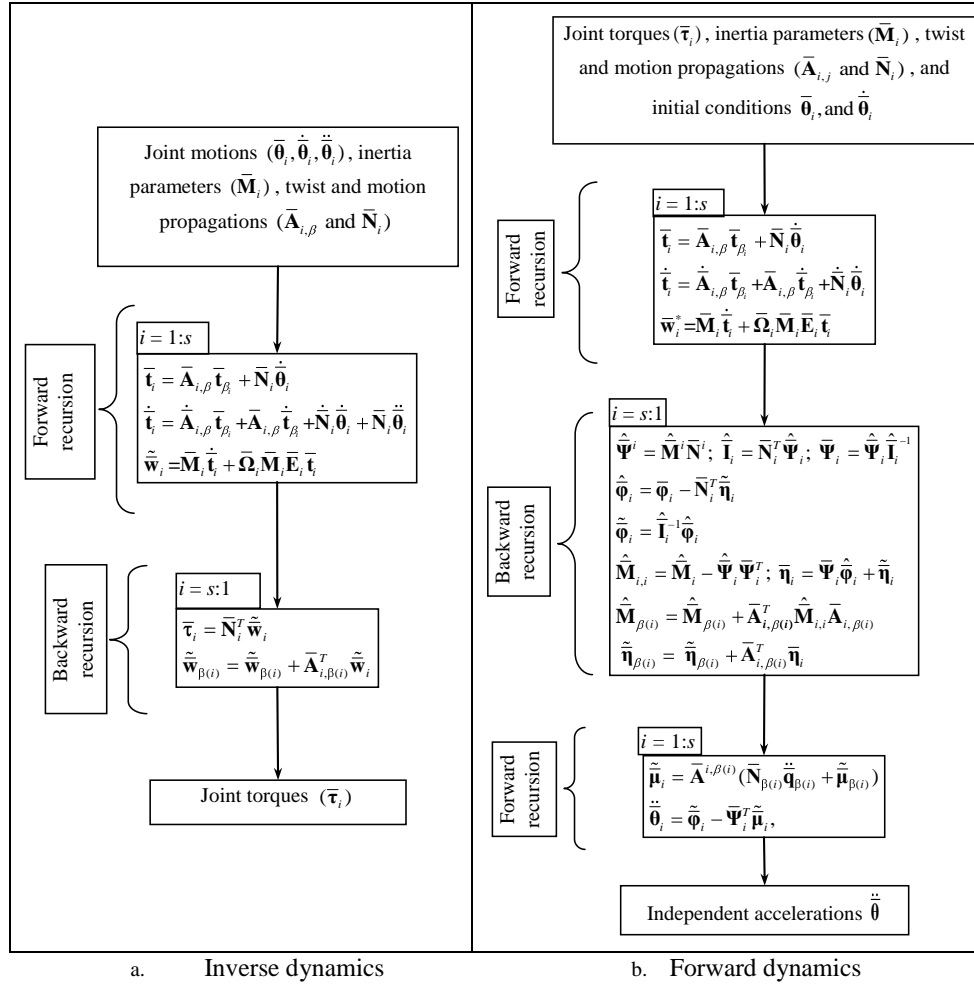
**Figure 9.** Recursive dynamics algorithms (Shah, 2011; Shah et al., 2013).

$$\overline{\mathbf{N}}_{1^i} \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{A}_{21} & \mathbf{1} & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{\eta 1} & \mathbf{A}_{\eta 2} & \cdots & \mathbf{1} \end{bmatrix}^i \quad \text{and}$$

$$\overline{\mathbf{N}}_{d^i} \equiv \begin{bmatrix} p_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & p_2 & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & p_\eta \end{bmatrix}^i \tag{39}$$

## 3.2  Inter-modular kinematic constraints

Having obtained the intra-modular kinematic constraints in the velocity-level, it is now possible to derive the inter-modular kinematic (velocity) constraints, i.e., between two neighbouring serial-chain modules. In a way, each module has been treated similar to a link in a serial-chain module presented in Sect. 2 or Sect. 3.1. For this, module $M_\beta$ is con-

sidered as the parent of module $M_i$, as shown in Fig. 8. This is similar to link $j$ of Fig. 2 which is the parent of link $i$. The $6\eta^i$-dimensional generalized twist $\overline{\mathbf{t}}_i$ is then obtained from the $6\eta^\beta$-dimensional generalized twist $\overline{\mathbf{t}}_\beta$ as

$$\overline{\mathbf{t}}_i = \overline{\mathbf{A}}_{i,\beta} \overline{\mathbf{t}}_\beta + \overline{\mathbf{N}}_i \dot{\overline{\boldsymbol{\theta}}}_i \tag{40}$$

where $\overline{\mathbf{A}}_{i,\beta}$ is the $6\eta^i \times 6\eta^\beta$ module-twist-propagation matrix which propagates the generalized twist of the parent module ($\beta$) to the child module ($i$) and $\overline{\mathbf{N}}_i$ is the $6\eta^i \times \eta^i$ module-joint-rate propagation matrix, which are given by

$$\overline{\mathbf{A}}_{i,\beta} \equiv \begin{bmatrix} \mathbf{O} & \cdots & \mathbf{O} & \mathbf{A}_{1^i,\eta^\beta} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{O} & \cdots & \mathbf{O} & \mathbf{A}_{\eta^i,\eta^\beta} \end{bmatrix} \tag{41a}$$

and

$$\overline{\mathbf{N}}_i = [\overline{\mathbf{N}}_l \overline{\mathbf{N}}_d]^i \equiv \begin{bmatrix} \boldsymbol{p}_1 & 0 & \cdots & 0 \\ \mathbf{A}_{21}\boldsymbol{p}_1 & \boldsymbol{p}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{\eta 1}\boldsymbol{p}_1 & \mathbf{A}_{\eta 2}\boldsymbol{p}_2 & \cdots & \boldsymbol{p}_\eta \end{bmatrix}^i \tag{41b}$$

The vectors $\overline{\boldsymbol{t}}_i$ and $\dot{\overline{\boldsymbol{\theta}}}_i$ are defined in Eq. (38). Next, the $6n$-dimensional generalized twist vector $\boldsymbol{t}$, and the $n$-dimensional generalized joint-rate vector $\dot{\boldsymbol{\theta}}$, for the whole tree-type system which comprises of $s$ modules and $n$ links are defined as

$$\boldsymbol{t} \equiv \begin{bmatrix} \overline{\boldsymbol{t}}_0^T & \overline{\boldsymbol{t}}_1^T & \cdots & \overline{\boldsymbol{t}}_i^T & \cdots & \overline{\boldsymbol{t}}_s^T \end{bmatrix}^T \quad \text{and}$$

$$\dot{\boldsymbol{\theta}} \equiv \begin{bmatrix} \dot{\overline{\boldsymbol{\theta}}}_0^T & \dot{\overline{\boldsymbol{\theta}}}_1^T & \cdots & \dot{\overline{\boldsymbol{\theta}}}_i^T & \cdots & \dot{\overline{\boldsymbol{\theta}}}_s^T \end{bmatrix}^T \tag{42}$$

where $\overline{\boldsymbol{t}}_0$ and $\dot{\overline{\boldsymbol{\theta}}}_0$ correspond to the base module $M_0$ which may not be fixed. For example, in the case of a spacecraft carrying a manipulator, the spacecraft floats with motion of 6-degrees-of-freedom (DOF). For the analysis purposes, its motion need to be specified for further motion analyses of other modules, e.g., the manipulator of the above system.

Upon substitution of the expressions of $\overline{\mathbf{N}}_i$ from Eq. (41b), for $i = 1,..., s$, in Eq. (40), and manipulating the expressions like Eqs. (8)–(9), one obtains the expression of the $6n$-dimensioanl generalized twist $\boldsymbol{t}$ for the whole tree-type system as

$$\boldsymbol{t} = \mathbf{N}_l \mathbf{N}_d \dot{\boldsymbol{\theta}} \tag{43}$$

in which, $\mathbf{N}_l$ and $\mathbf{N}_d$ are the $6(n+1) \times 6(n+1)$ and $6(n+1) \times (n+n_0)$ matrices, respectively, as the tree-type system was assumed to have module $M_0$ with one-link with $n_0$ DOF. Matrices $\mathbf{N}_l$ and $\mathbf{N}_d$ for the tree-type system are given by

$$\mathbf{N}_l \equiv \begin{bmatrix} \overline{\mathbf{1}}_0 & & & & \\ \overline{\mathbf{A}}_{10} & \overline{\mathbf{1}}_1 & & \mathbf{O}'s & \\ \overline{\mathbf{A}}_{20} & \overline{\mathbf{A}}_{21} & \overline{\mathbf{1}}_2 & & \\ \vdots & \vdots & \vdots & \ddots & \\ \overline{\mathbf{A}}_{s0} & \overline{\mathbf{A}}_{s1} & \cdots & \cdots & \overline{\mathbf{1}}_s \end{bmatrix},$$

where $\overline{\mathbf{A}}_{j,i} \equiv \mathbf{O}, \text{ if } M_j \notin \gamma_i$ \hfill (44)

and

$$\mathbf{N}_d \equiv \begin{bmatrix} \overline{\mathbf{N}}_0 & & & \mathbf{O}'s \\ & \overline{\mathbf{N}}_1 & & \\ & & \ddots & \\ \mathbf{O}'s & & & \overline{\mathbf{N}}_s \end{bmatrix} \tag{45}$$

In Eq. (44), $\overline{\mathbf{1}}_i$ is the $6\eta^i \times 6\eta^i$ identity matrix, whereas $\gamma_i$ stands for the array of all modules including module $M_i$ and outward to it, as shown within dashed line of Fig. 10. The
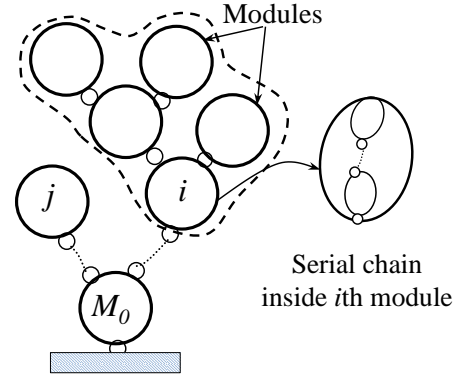


**Figure 10.** Definition of $\gamma_i$.

matrices $\mathbf{N}_l$ and $\mathbf{N}_d$ are the desired Decoupled Natural Orthogonal Compliment (DeNOC) matrices for the whole tree-type system at hand. Note here that the matrices, $\mathbf{N}_l$ and $\mathbf{N}_d$ of Eq. (9b), and $\mathbf{N}_{li}$ and $\mathbf{N}_{di}$ of Eq. (39), are the special cases of the DeNOC matrices derived in Eqs. (44) and (45), where each module has only one link without any branching.

### 3.3  Newton-Euler (NE) equations for tree-type systems

In contrast to the expressions for the Newton-Euler (NE) equations of the $i$-th link given by Eq. (11) or (12), a deviation in their expressions will be observed. This is due to the modified definition of the velocity of the $i$-th link, i.e., $\boldsymbol{v}_i$, with respect to point $O_i$. This was mentioned in Sect. 3.1. The NE equations of motion of the $k$-th link (as the letter "$i$" will be used to denote module) of the $i$-th module with respect to point $O_k$ can be expressed as (Shah et al., 2011, 2013)

$$\mathbf{I}_k \dot{\boldsymbol{\omega}}_k + m_k \boldsymbol{d}_k \times \dot{\boldsymbol{v}}_k + \boldsymbol{\omega}_k \times \mathbf{I}_k \boldsymbol{\omega}_k = \boldsymbol{n}_k \tag{46a}$$

$$m_k \boldsymbol{v}_k - m_k \boldsymbol{d}_k \times \dot{\boldsymbol{\omega}}_k - \boldsymbol{\omega}_k \times (m_k \boldsymbol{d}_k \times \boldsymbol{\omega}_k) = \boldsymbol{f}_k \tag{46b}$$

Combining Eqs. (46a)–(46b), one can obtain an expression equivalent to Eq. (12) as

$$\mathbf{M}_k \dot{\boldsymbol{t}}_k + \Omega_k \mathbf{M}_k \mathbf{E}_k \boldsymbol{t}_k = \boldsymbol{w}_k \tag{47a}$$

where the $6 \times 6$ matrices $\mathbf{M}_k$, $\Omega_k$, and $\mathbf{E}_k$ are defined as

$$\mathbf{M}_k \equiv \begin{bmatrix} \mathbf{I}_k & m_k \boldsymbol{d}_k \times \mathbf{1} \\ -m_k \boldsymbol{d}_k \times \mathbf{1} & m_k \mathbf{1} \end{bmatrix}, \quad \Omega_k \equiv \begin{bmatrix} \boldsymbol{\omega}_k \times \mathbf{1} & \mathbf{O} \\ \mathbf{O} & \boldsymbol{\omega}_k \times \mathbf{1} \end{bmatrix},$$

and $\mathbf{E}_k \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} \end{bmatrix}$ \hfill (47b)

Note in Eq. (47b), that $\mathbf{I}_k$ is the $3 \times 3$ mass moment of inertia tensor of the $k$-th link about $O_k$. Combining Eq. (47a) for all $\eta^i$ links of the $i$-th module and for all $s$ modules, one can write a compact expression equivalent to Eq. (13) as (Shah et al., 2011, 2013)

$$\mathbf{M}\dot{\boldsymbol{t}} + \Omega \mathbf{M} \mathbf{E} \boldsymbol{t} = \boldsymbol{w} \tag{48}$$

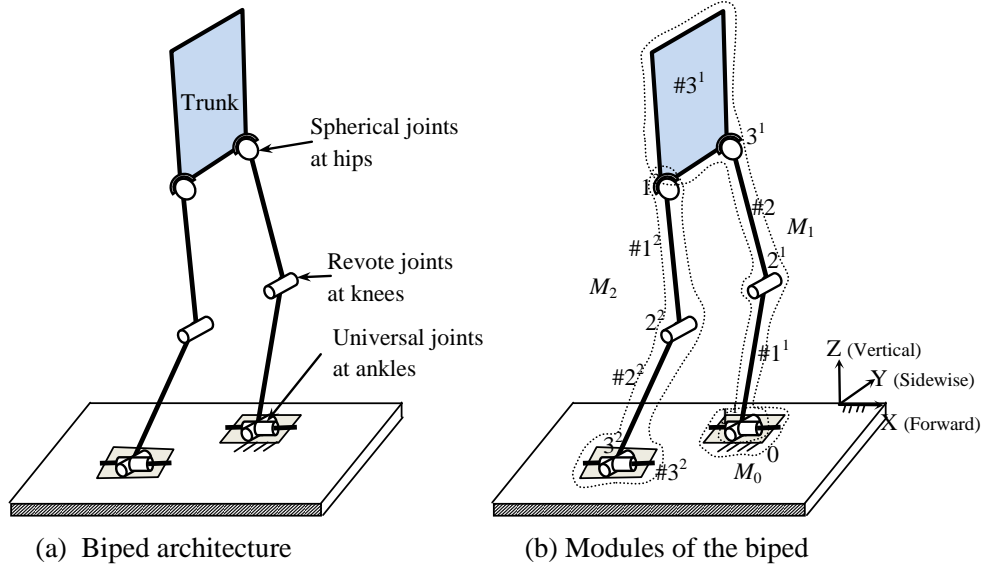(a) Biped architecture                    (b) Modules of the biped

**Figure 11.** A 7-link spatial biped.

where matrices $\mathbf{M}$, $\boldsymbol{\Omega}$, and $\mathbf{E}$ are the $6(n+1) \times 6(n+1)$ block-diagonal matrices defined similar to Eq. (14). For details, readers are referred to the Ph.D. thesis of Shah (2011) or the book by Shah et al. (2013).

### 3.4 Constrained equations for tree-type systems using the DeNOC matrices

The constrained equations of motion for the tree-type systems are derived in this subsection in a similar manner to that of the serial-chain system of Sect. 2, i.e., pre-multiply $\mathbf{N}_d^T \mathbf{N}_l^T$ of Eq. (43) to the unconstrained NE equations given by Eq. (48) to obtain a set of constrained independent equations of motion by eliminating the constraint wrenches. These constrained equations are also referred to as the Euler-Lagrange equations of motion of the tree-type system at hand. They are given by

$$\mathbf{I}\ddot{\boldsymbol{\theta}} + \mathbf{C}\dot{\boldsymbol{\theta}} = \boldsymbol{\tau} \tag{49}$$

where $\mathbf{I}$ is generalized inertia matrix (GIM), $\mathbf{C}$ is the matrix of convective inertia terms (MCI), and $\boldsymbol{\tau}$ is the vector of generalized driving forces, and due to gravity, dissipation, external forces, etc., which have expressions similar to those after Eq. (17).

Note that the expression of Eq. (49) is same as Eq. (17) but the sizes of the corresponding matrices and vectors are different because they represent two different architectures of the multibody systems. Based on Eq. (49), recursive inverse and forward dynamics algorithms for tree-type systems were developed by Shah (2011) and implemented in a software called ReDySim (Recursieve Dynamics Simulator) (Shah et al., 2012c). ReDySim was written in MATLAB environment and available free from http://www.redysim.co.nr.
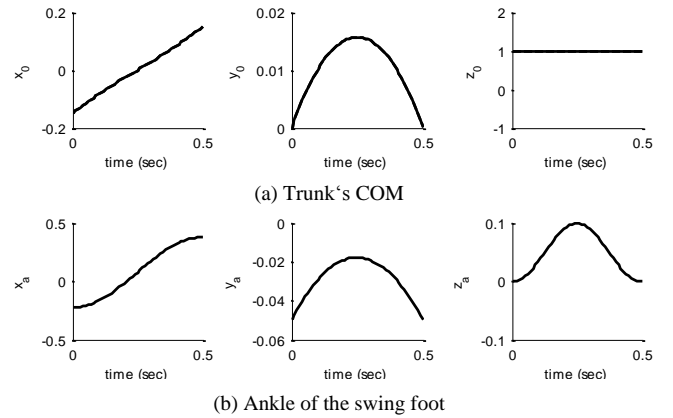


(a) Trunk's COM

(b) Ankle of the swing foot

**Figure 12.** Designed trajectories of the trunk's center of mass (COM) and ankle (Shah, 2011; Shah et al., 2013).

### 3.5 Numerical example: a spatial biped

In order to illustrate the recursive dynamics algorithms presented in this section, ReDySim was used to analyze a spatial biped shown in Fig. 11. The model parameters were taken from Shah (2011) which will appear in the book by Shah et al. (2013) also. They are not reproduced here due to the reasons cited in Sect. 2.8. However, the designed input motions of the trunk's centre-of-mass (COM) and ankle for stable walking (Shah, 2011) are shown in Figs. 12 and 13, respectively. Based on the inputs of Figs. 12 and 13, the inverse dynamics results were obtained which are shown in Fig. 14.

Forced simulation was performed next, as reported in Shah (2011), where the motion of the biped was studied under the application of joint torques calculated above, i.e.,
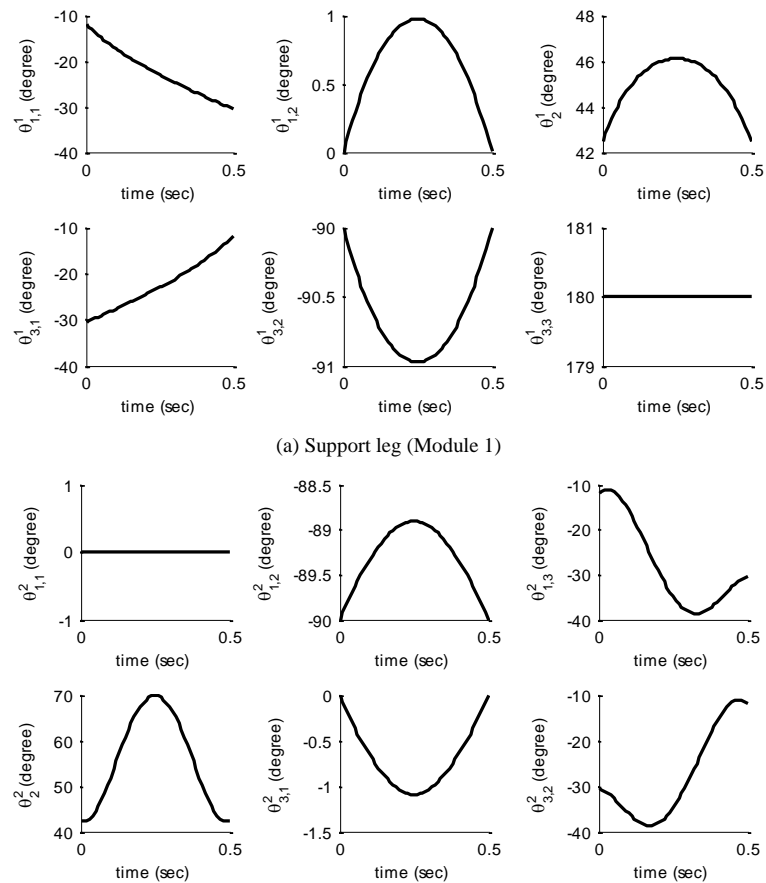
(a) Support leg (Module 1)



**Figure 13.** Joint trajectories of the biped obtained from the trajectories of trunk and ankle (Shah, 2011; Shah et al., 2013).

those shown in Fig. 14. The joint motions were calculated using the forward dynamics module of ReDySim. The plots for the simulated joint angles are shown in Fig. 15, along with the desired one. It can be seen that the simulated joint angles match with the desired joint angles up to 0.1 s, i.e., until 0.1 s movement of the biped. After this, the system behaves unexpectedly as evident from the divergent plots of the simulated angles in Fig. 15a. The deviation in the simulated angles is mainly attributed to what is known as zero eigen-value effect (Saha and Schiehlen, 2001). The physical system may also not behave as expected due to disturbances caused by unmodelled parameters like friction, backlash, etc., and non-exact geometrical and inertia parameters. Hence, a control scheme must be considered, as this forms a part and parcel of achieving proper walking. These aspects were explained in detail in Shah (2011) and Shah et al. (2013), and not elaborated further due to space limitation of the paper.

Note several advantages of the concept of the kinematic modules in the dynamics modelling of tree-type systems consisting of serially connected links (Shah, 2011; Shah et al., 2013), which are as follows:

– Extension of the body-to-body velocity transformation relationship to module-to-module velocity transformation relationship.

– Compact representation of the system's kinematic and dynamic models.

– Uniform development of the inverse and forward dynamics algorithms with inter- and intra-modular recursions.

– Module-level analytical expressions of the matrices and vectors appearing in the equations of motion.

– Ease of investigation of any inconsistency in the results of modules without the need to investigate the whole system.

– Possibility of hybrid recursive-parallel algorithms, where each module can be analyzed using recursive relations in parallel.
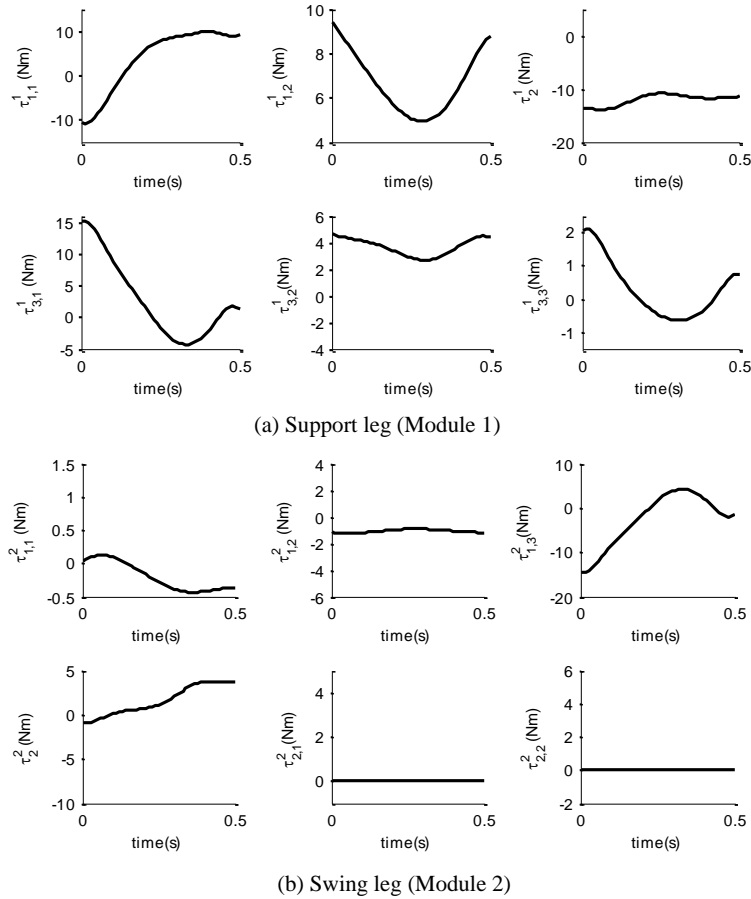
(a) Support leg (Module 1)



(b) Swing leg (Module 2)

**Figure 14.** Torques at different joints of the biped (Shah, 2011; Shah et al., 2013).

## 4  Closed-loop systems

The DeNOC-based dynamic modelling of serial-chain and tree-type open-loop systems presented in Sects. 2 and 3, respectively, can be extended to closed-loop systems provided one cuts the closed-loops of a system at suitable locations to make it open. Note that, one needs to use suitable constraint forces at the cut-joints to represent the actual presence of the joints. Such constraint forces are known in the literature as Lagrange multipliers (Chaudhary and Saha, 2007, 2009, and others). The multipliers need to be evaluated from the loop-closure constraints before they can be used as external forces to the resulting open-loop systems. In this section, a planar 4-bar mechanism shown in Fig. 16 is considered to illustrate the concept. However, the methodology is applicable to any general multi closed-loop systems, as shown in Chaudhary and Saha (2007), Shah (2011), and Shah et al. (2013).

Note that to model an open-loop system resulting from a closed-loop system, one needs to re-write Eqs. (13) or (48) as

$$\mathbf{N}^T(\mathbf{M}\dot{\boldsymbol{t}} + \boldsymbol{\Omega}\mathbf{M}\mathbf{E}\boldsymbol{t}) = \mathbf{N}^T(\boldsymbol{w}^E + \boldsymbol{w}^\lambda + \boldsymbol{w}^C) \qquad (50)$$

where $\boldsymbol{w}^\lambda$ is the $6n$-dimensional vector of generalized wrench due to Lagrange multipliers acting at the cut joints. For the 4-bar mechanism shown in Fig. 16a, the two cut-open serial-chain subsystems are shown in Fig. 16b. The resulting open-loop tree-type subsystems have one and two links, respectively, connected by one and two one-DOF revolute joints. Other terms have same meaning as in Sects. 2 and 3. In Eq. (50), $\mathbf{N}^T\boldsymbol{w}^C = 0$ for the reason given after Eq. (16), but $\mathbf{N}^T\boldsymbol{w}^\lambda \neq 0$. These terms are now the new unknowns to the inverse and forward dynamics problems that need to be evaluated with the help of loop-closure constraints.

For the closed-loop 1-2-3-4 of the 4-bar mechanism shown in Fig. 16a, one can write

$$\boldsymbol{a}_0 + \boldsymbol{a}_1 = \boldsymbol{a}_2 + \boldsymbol{a}_3 \qquad (51)$$

where 2-dimensional vectors of the planar system, $\boldsymbol{a}_i$ for $i = 0, 1, 2, 3$, represent the relative position vectors of the joints in the 4-bar mechanism, Fig. 16a. Differentiating Eq. (51) with respect to time, one obtains

$$\mathbf{J}\dot{\boldsymbol{\theta}} = 0, \quad \text{where } \dot{\boldsymbol{\theta}} \equiv \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dot{\theta}_3 \end{bmatrix}^T \qquad (52a)$$

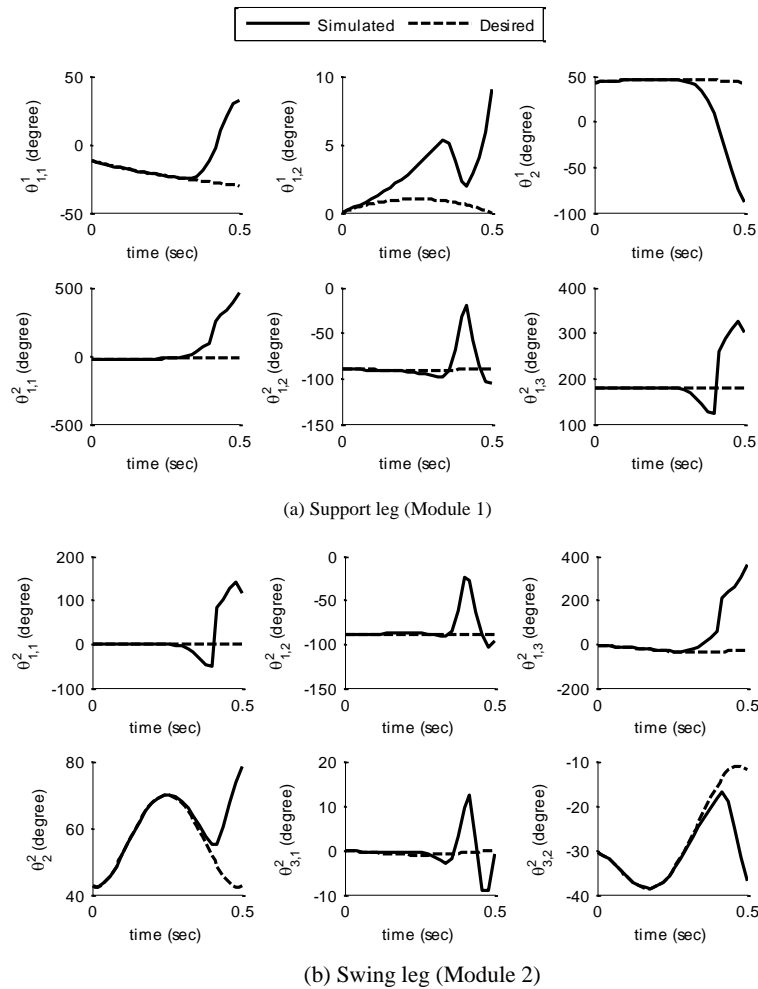(a) Support leg (Module 1)



(b) Swing leg (Module 2)

**Figure 15.** Simulated joint angles for the biped (Shah, 2011; Shah et al., 2013).

and the $2 \times 3$ Jacobian matrix for the 4-bar mechanism at hand can be given by

$$\mathbf{J} \equiv \begin{bmatrix} -a_1 s_1 - a_2 s_{12} + a_3 s_{123} & -a_2 s_{12} + a_3 s_{123} & -a_3 s_{123} \\ a_1 c_1 + a_2 c_{12} - a_3 c_{123} & a_2 c_{12} + a_3 c_{123} & a_3 c_{123} \end{bmatrix} \tag{52b}$$

where $s_{12} \equiv \sin(\theta_1 + \theta_2)$, $s_{123} \equiv \sin(\theta_1 + \theta_2 + \theta_3)$, and similarly $c_{12}$ and $c_{123}$ etc. Equation (52b) was also derived in Chaudhary et al. (2007, 2009) as

$$\mathbf{J} \equiv \begin{bmatrix} \mathbf{J}_{\mathrm{I}} \\ \mathbf{J}_{\mathrm{II}} \end{bmatrix}, \quad \text{where } \mathbf{J}_{\mathrm{I}} \equiv \mathbf{A}_{\mathrm{en}}^{\mathrm{I}} \mathbf{N}^{\mathrm{I}} \text{ and } \mathbf{J}_{\mathrm{II}} \equiv \mathbf{A}_{\mathrm{en}}^{\mathrm{II}} \mathbf{N}^{\mathrm{II}} \tag{53}$$

In Eq. (53), $\mathbf{N}^{\mathrm{I}}$ and $\mathbf{N}^{\mathrm{II}}$ are the $6 \times 1$ and $12 \times 2$ NOC matrices for the two open-chain subsystems I and II, respectively, shown in Fig. 16b, whereas $\mathbf{A}_{\mathrm{en}}^{\mathrm{I}}$ and $\mathbf{A}_{\mathrm{en}}^{\mathrm{II}}$ are the $2 \times 6$ and $2 \times 12$ twist-propagation matrices for the last link from the point of contact to its previous link to the point where the joint is cut, i.e., Joint 4 of Fig. 16b. Such Jacobians using the

DeNOC matrices were also derived in Saha (2008). The resulting constrained dynamic equations of motion for the two subsystems are then written as

$$\mathbf{I}^{\mathrm{I}} \ddot{\boldsymbol{\theta}}^{\mathrm{I}} + \mathbf{C}^{\mathrm{I}} \dot{\boldsymbol{\theta}}^{\mathrm{I}} = \boldsymbol{\tau}^{\mathrm{I}} + (\boldsymbol{\tau}^{\lambda})^{\mathrm{I}} \tag{54a}$$

$$\mathbf{I}^{\mathrm{II}} \ddot{\boldsymbol{\theta}}^{\mathrm{II}} + \mathbf{C}^{\mathrm{II}} \dot{\boldsymbol{\theta}}^{\mathrm{II}} = \boldsymbol{\tau}^{\mathrm{II}} + (\boldsymbol{\tau}^{\lambda})^{\mathrm{II}} \tag{54b}$$

Depending on the type of dynamics problem, i.e., inverse or forward, Eqs. (54a)–(54b) can be solved using recursive "subsystem" or "system" approach. For inverse dynamics, "subsystem recursion" provides a better efficiency, as pointed out by Chaudhary and Saha (2009).

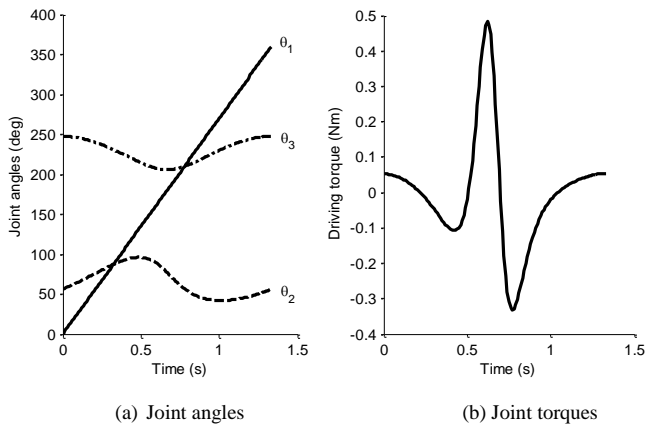## 4.1   Numerical example: a 4-bar mechanism

For the numerical results, ReDySim software mentioned in Sect. 3 was used using the lengths of crank (#1), output link (#2), coupler (#3) and fixed-base (#0) as 0.038 m, 0.1152 m, 0.1152 m and 0.0895 m, respectively. The masses
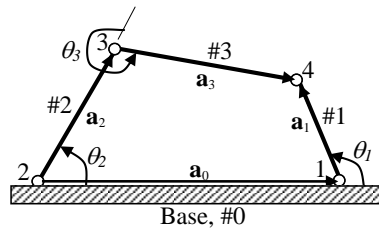
(a) Closed-loop system
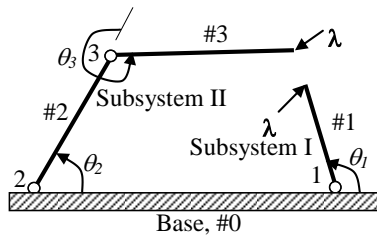


(b) Two equivalent open-loop systems

**Figure 16.** A 4-bar mechanism.



(a) Joint angles



(b) Joint torques

**Figure 17.** Inverse dynamics for a 4-bar mechanism.

of the crank, output link and coupler were taken as 1.5 kgs, 3 kgs and 5 kgs, respectively. The input joint angle and the joint torque at joint 1 are plotted in Fig. 17. The forward dynamics of the 4-bar mechanism was carried out using the same initial configuration, as specified in the inverse dynamics. The simulation was done for the free-fall of the mechanism under gravity without any external torque applied at joint 1. The joint angles and rates are plotted in Fig. 18. The results of inverse dynamics and forward dynamics were validated with MATLAB's SimMechanics model, as reported in Shah et al. (2012c).



(a) Joint angles



(b) Joint rates
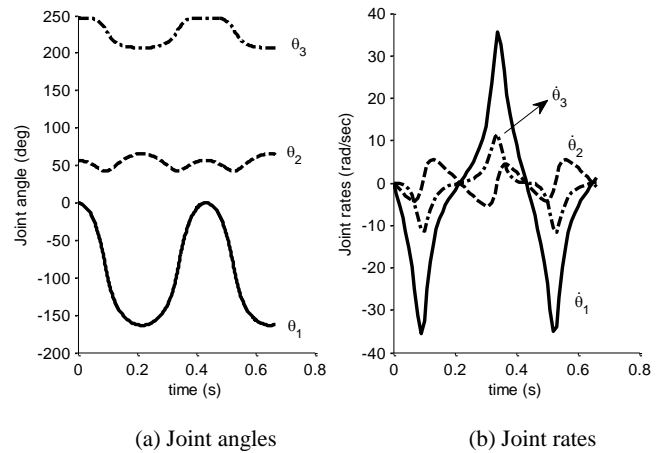
**Figure 18.** Forward dynamics and simulation results for the 4-bar mechanism.

## 5 Computational efficiency

In this section, computational efficiencies of the DeNOC-based algorithms are investigated. Figure 18 shows comparisons of computational complexities required by several inverse dynamics algorithms, when a system has 1-DOF (Fig. 19a), 2-DOF (Fig. 19b), 3-DOF (Fig. 19c) and equal numbers of 1-2- and 3-DOF joints (Fig. 19d). It may be seen that the recursive inverse dynamics algorithm given in Fig. 9 of Sect. 3 performs as fast as the fastest algorithm available in the literature when the system has only 1-DOF joints, as evident from Fig. 19a. However, when multiple-DOF joints are introduced in the system, the algorithms of Sect. 3 (Shah, 2011), which have been implemented in ReDySim, outperforms the other algorithms available in the literature. This is clear from Fig. 19b–d.

From Fig. 20, it is also clear that the forward dynamics algorithm of Fig. 9 explained in Sect. 3 (Shah, 2011) performs better than any other algorithm available in the literature. More the number of multiple-DOF joints, more the improvement in the computational efficiency, as shown in Fig. 20b–d. This is mainly due the implicit inversion of the GIM based on the Reverse Gaussian Elimination (Saha, 1995, 1997) of the GIM, and simplification of the expressions associated with multiple-DOF joints (Shah et al., 2012b). In the tree-type robotic systems, such as biped, quadruped, etc., where the DOF of the system is more than 30, and the system consists of many multiple-DOF joints, the DeNOC-based algorithms significantly improve the computational efficiency.

## 6 Software for students and researchers

In order to build up the interest in the areas of multibody dynamics and to provide efficient tools to perform the dynamic analyses, the following multibody simulation tools
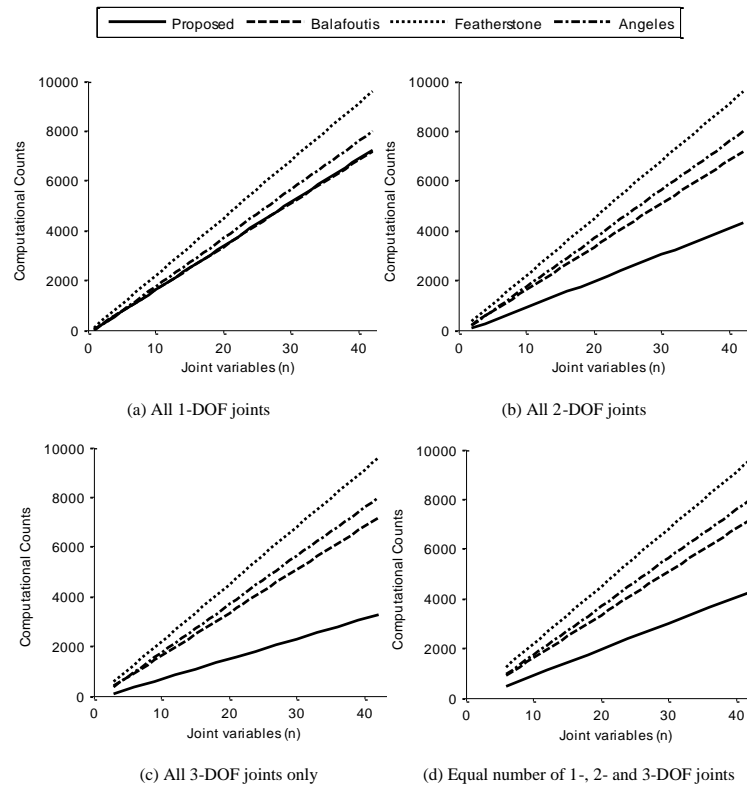
**Figure 19.** Performance of several inverse dynamics algorithms (Shah, 2011; Shah et al., 2013).

were developed for the students and researchers using the DeNOC-based formulations presented in Sects. 2–4:

– RoboAnalyzer: for serial-chain open-loop systems

– ReDySim (Recursive Dynamic Simulator): for general tree-type systems

These are explained briefly in the following subsections.

## 6.1  RoboAnalyzer

RoboAnalyzer (Rajeevlochana et al., 2012) is a 3-dimensional model-based software to solve kinematics and dynamics problems of an serial-chain open-loop system. It was developed using Visual C# and OpenGL that take the description of a serial-chain system using the DH parameters (Denavit and Hartenberg, 1955; Saha, 2008), and the mass and inertia properties of each link. In RoboAnalyzer, one can also see the animation of the analyzed systems. For the benefit of the users, CAD models of the standard systems like KUKA, PUMA robots, and others were made available for analysis. The software is freely downloadable from http://www.roboanalyzer.com.

## 6.2  ReDySim (Recursive Dynamics Simulator)

Recursive Dynamic Simulator (ReDySim) (Shah et al., 2012c) is a multibody dynamics simulation tool which was developed in MATLAB environment. It was developed based on the concept of the DeNOC matrices, and kinematic modules of a tree-type system, as explained in Sect. 3. ReDySim can be used to perform inverse dynamics and simulation of multibody systems. It has two modules, namely, fixed-base and floating-base modules. The latter was not presented in this paper due to limited page restriction of a paper. However, the interested readers can refer to Shah (2011) or Shah et al. (2013) for the dynamics analyses of biped, quadruped and six-legged walking robots using the floating-base concept. Note here that the architectural information of the tree-type systems were provided using the modified-DH parameters (MDH) parameters, as proposed by Khalil and Kleinfinger (1986), instead of the DH parameters defined in Saha (2008). This was done mainly to improve the computational efficiency of the tree-type systems.

ReDySim was also used to solve flexible systems like ropes, etc. which were modelled as hyper-degrees-of-freedom rigid-link systems (Shah, 2011). In fact, the simulation of long chains with the aid of ReDySim showed considerable improvement over commercial software like Recur-Dyn in terms of the computational time and correctness of
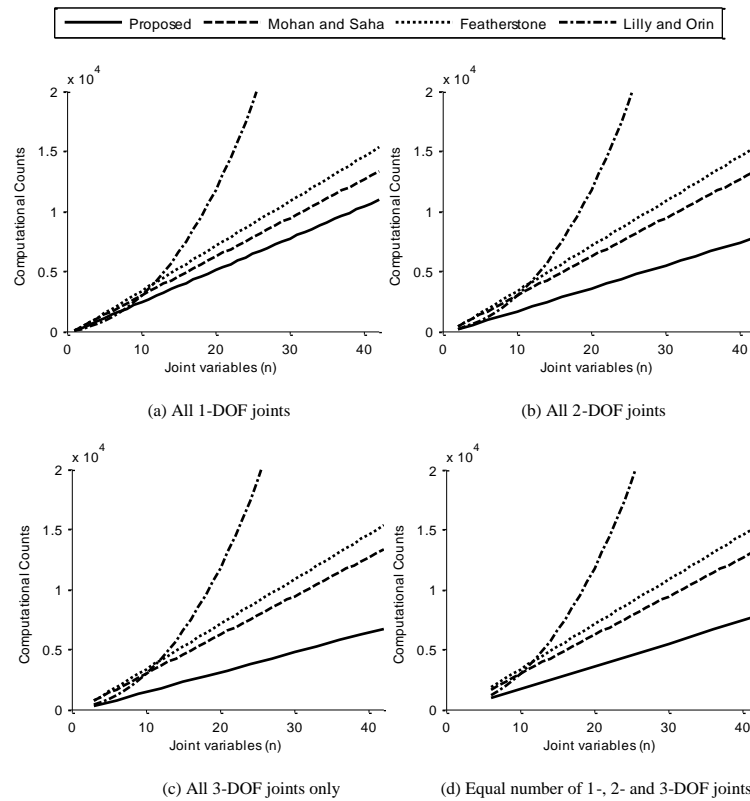
**Figure 20.** Performance of forward dynamics algorithms (Shah, 2011; Shah et al., 2013).

the results. These results were separately communicated to a journal for publication (Agarwal et al., 2012). ReDySim can be downloaded free from http://www.redysim.co.nr, where the user's manual and some demos are also made available for the benefits of the users.

## 7   Conclusions

Recursive algorithms are popular due to their efficiency, computational uniformity, and numerical stability. This paper gave insight to the evolution of the Decoupled Natural Orthogonal Complement (DeNOC) matrices for more than one and half decades. It was shown that the use of DeNOC matrices in the dynamic analyses of multibody systems led to the development of several recursive dynamics algorithms for serial-chain, tree-type, and closed-loop systems. Several numerical examples, e.g., the Stanford arm (serial-chain), spatial biped (tree-type), and 4-bar mechanism (closed-loop), were given to explain the concepts presented in this paper. Advantages and computational efficiencies of the DeNOC-based methodologies suggest that it should be used when the DOF of a system is large and the application is real-time. Finally, two types of software meant for serial-chain open-loop systems, and general tree-type systems with abilities to solve closed-loop systems, i.e., RoboAnalyzer and ReDySim, re-

spectively, were explained. It is expected that the algorithms, and more importantly, the software will benefit immensely the students and researchers of multibody dynamics community.

## References

Agarwal, A., Shah, S. V., Bandyopadhyay, S., and Saha, S. K.: Dynamics of serial chains with large degrees-of-freedom, J. Multibody System Dynamics, under review, 2012.

Angeles, J. and Lee, S.: The formulation of dynamical equations of holonomic mechanical systems using a natural orthogonal complement, ASME J. Appl. Mech., 55, 243–244, 1988.

Angeles, J. and Ma, O.: Dynamic simulation of n-axis serial robotic manipulators using a natural orthogonal complement, Int. J. Robot. Res., 7, 32–47, 1988.

Angeles, J., Ma, O., and Rojas, A.: An algorithm for the inverse dynamics of n-axis general manipulator using Kane's formulation of dynamical equations, Computers and Mathematics with Applications, 17, 1545–1561, 1989.

Bathe, K. J. and Wilson, E. L.: Numerical Methods in Finite Element Analysis, 1st Edn., Prentice-Hall, New Jersy, USA, 1976.

Blajer, W., Bestle, D., and Schiehlen, W.: An orthogonal complement matrix formulation for constrained multibody systems, ASME J. Mech. Design, 116, 423–428, 1994.

Cameron, J. M. and Book, W. J.: Modeling mechanisms with nonholonomic joints using the Boltzmann-Hammel equations, Int. J. Robot. Res., 16, 47–59, 1997.

Chaudhary, H. and Saha, S. K.: Constraint wrench formulation for closed-loop systems using two-level recursions, ASME J. Mech. Design, 129, 1234–1242, 2007.

Chaudhary, H. and Saha, S. K.: Dynamics and Balancing of Multibody Systems, Springer, 2009.

Cyril, X.: Dynamics of Flexible Link Manipulators, Ph.D. thesis, Dept. of Mech. Eng., McGill University, Canada, 1988.

Denavit, J. and Hartenberg, R. S.: A kinematic notation for lower-pair mechanisms based on matrices, ASME J. Appl. Mech., 77, 215–221, 1955.

Dimitrov, D.: Dynamics and Control of Space Manipulators During a Satellite Capturing Operation, Ph.D. thesis, Graduate School of Engineering, Tohoku University, Japan, 2005.

Eberhard, P. and Schiehlen, W.: Computational dynamics of multibody systems: History, formalisms, and applications, ASME J. Comput. Nonlin. Dyn., 1, 3–12, 2006.

Featherstone, R.: Robot Dynamics Algorithms, Kluwer Academic Publishers, 1987.

Garcia de Jalon, J., Alvarez, E., de Ribera, F. A., Rodriguez, I., and Funes, F. J.: A fast and simple semi-recursive formulation for multi-rigid-body systems, in: Advances in Computational Multibody Systems, edited by: Ambrosio, J. A. C., Springer, 1–23, 2005.

Greenwood, D. T.: Principles of Dynamics, Prentice-Hall of India, New Delhi, 1988.

Hanzaki, A. R., Saha, S. K., and Rao, P. V. M.: An improved dynamic modeling of a multibody system with spherical joints, Multibody Syst. Dyn., 21, 325–345, 2009.

Hemami, H. and Weimer, F. C.: Modeling of nonholonomic dynamic systems with applications, ASME J. Appl. Mech., 48, 177–182, 1981.

Huston, R. L. and Passerello, C. E.: On constraint equations – A new approach, ASME J. Appl. Mech., 41, 1130–1131, 1974.

Kamman, J. W. and Huston, R. L.: Constrained multibody system dynamics: An automated approach, Comput. Struct., 18, 999–1003, 1984.

Kane, T. R. and Levinson, D. A.: The use of Kane's dynamical equations for robotics. Int. J. Robot. Res., 2, 3–21, 1983.

Khan, W. A., Krovi, V. N., Saha, S. K., and Angeles, J.: Recursive kinematics and inverse dynamics for a planar 3R parallel manipulator, J. Dyn. Syst.-T. ASME, 127, 529–536, 2005.

Khalil, W. and Kleinfinger, J.: A new geometric notation for open and closed-loop robots, Proc. of the IEEE Int. Conf. on Robotics and Automation, 3, 1174–1179, 1986.

Khatib, O.: Unified approach for motion and force control of robot manipulators: The operational space formulation, IEEE J. Robotics and Automation, RA-3, 43–53, 1987.

Kim, S. S. and Vanderploeg, M. J.: A general and efficient method for dynamic analysis of mechanical systems using velocity transformations, J. Mech. Transm.-T. ASME, 108, 176–182, 1986.

Mani, N. K., Haug, E. J., and Atkinson, K. E.: Application of singular value decomposition for analysis of mechanical system dynamics, J. Mech. Transm.-T. ASME, 107, 82–87, 1985.

McPhee, J. J.: On the use of linear graph theory in multibody system dynamics, Nonlinear Dynam., 9, 73–90, 1996.

Mohan, A. and Saha, S. K.: A recursive, numerically stable, and efficient algorithm for serial robots, Multibody Syst. Dyn., 17, 291–319, 2007.

Park, F. C., Bobrow, J. E., and Ploen, S. R.: A Lie group formulation of robot dynamics, Int. J. Robot. Res., 14, 606–618, 1995.

Patriciu, A., Chirikjian, S. G., and Pappub, R. V.: Analysis of the conformational dependence of mass-metric tensor determinants in serial polymers with constraints, J. Chem. Phys., 121, 12708, doi:10.1063/1.1821492, 2004.

Rajeevlochana, C. G. and Saha, S. K.: RoboAnalyzer: 3D model based robotic learning software, Proc. of the Int. Conf. on Multibody Dynamics, Vijayawada, India, 24–26 February, 3–13, 2011.

Rajeevlochana, C. G., Jain, A., Shah, S. V., and Saha, S. K.: Recursive robot dynamics in RoboAnalyzer, in: Machines and Mechanisms (Proc. of the 15th Nat. Conf. on Machines and Mechanisms), edited by: Bandopadhyay, S., Gurunathan, S. K., and Ramu, P., Narosa Publishing House, New Delhi, 482–490, ISBN: 978-81-8487-192-0, 2012.

Roberson, R. E. and Schwertassek R.: Dynamics of Multibody Systems, Springer, Berlin, 1988.

Rodriguez, G., Jain, A., and Kreutz-Delgado, K.: Spatial operator algebra for multibody system dynamics, J. Astronaut. Sci., 40, 27–50, 1992.

Saha, S. K.: The $UDU^T$ decomposition of manipulator inertia matrix, Proc. of the IEEE Int. Conf. on Robotics and Automation, Nagoya, Japan, 21–27 May, 3, 2829–2834, 1995.

Saha, S. K.: A decomposition of the manipulator inertia matrix, IEEE Trans. on Robotics and Automation, 13, 301–304, 1997.

Saha, S. K.: Dynamics of serial multibody systems using the decoupled natural orthogonal complement matrices, ASME J. Appl. Mech., 66, 986–996, 1999a.

Saha, S. K.: Analytical expression for the inverted inertia matrix of serial robots, Int. J. Robot. Res., 18, 116–124, 1999b.

Saha, S. K.: Simulation of industrial manipulators based on the $UDU^T$ decomposition of inertia matrix, Multibody Syst. Dyn., 9, 63–85, 2003.

Saha, S. K.: Introduction to Robotics, Tata McGraw-Hill, New Delhi, 2008.

Saha, S. K. and Angeles, J.: Dynamics of nonholonomic mechanical systems using a natural orthogonal complement, ASME J. Appl. Mech., 58, 238–243, 1991.

Saha, S. K. and Schiehlen, W. O.: Recursive kinematics and dynamics for closed loop multibody systems, Int. J. Mech. Structures Machines, 29, 143–175, 2001.

Saha, S. K., Shirinzadeh, B., and Alici, G.: Dynamic model simplification of serial manipulators, Proc. of the Int. Symp. on Robotics and Automation, San Miguel Regla Hotel, Hgo, Mexico, 25–28 August, 14–19, 2006.

Schiehlen, W.: Multibody Systems Handbook, Springer-Verlag, Berlin, 1990.

Schiehlen, W.: Multibody system dynamics: Roots and perspectives, Multibody Syst. Dyn., 1, 49–188, 1997.

Shabana, A. A.: Computational Dynamics, Wiley, New York, 2001.

Shah, S. V.: Modular Framework for Dynamic Modeling and Analyses of Tree-type Robotics Systems, Ph.D. thesis, Dept. of Mech. Eng., IIT Delhi, 2011.

Shah, S. V., Saha, S. K., and Dutt, J. K.: Modular framework for dynamics of tree-type legged robots, Mech. Mach. Theory, 49, 234–255, 2012a.

Shah, S. V., Saha, S. K., and Dutt, J. K.: Denavit-Hartenberg (DH) parametrization of Euler-angles. ASME J. Nonlinear and Computational Dynamics, 7, 021006, doi:10.1115/1.4005467, 2012b.

Shah, S. V., Nandihal, P. V., and Saha, S. K.: Recursive Dynamics Simulator (ReDySim): A multibody dynamics solver, Theor. Appl., 2, 063011, doi:10.1063/2.1206311, 2012c.

Shah, S. V., Saha, S. K., and Dutt, J. K.: Dynamics of Tree-type Robotics Systems, Springer, 2013.

Stokes, A. and Brockett, R.: Dynamics of kinematic chains, Int. J. Robot. Res., 15, 393–405, 1996.

Strang, G.: Linear Algebra and its Applications, Harcourt, Brace, Jovanovich, Publisher, Florida, 1998.

Sundarraman, P., Saha, S. K., Vasa, N. J., Baskaran, R., Sunilkumar, V., and Raghavendra, K.: Modeling and analysis of a fuel-injection pump used in diesel engines, Int. J. Automot. Techn., 13, 193–203, 2012.

Wehage, R. A. and Haug, E. J.: Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems, ASME J. Mech. Design, 104, 247–255, 1982.

Wittenburg J.: Dynamics of Multibody systems, Springer, Berlin, 2008.