Dynamics Algorithms for Multibody Systems

S. V. Shah[1], **S. K. Saha**[2] **and J. K. Dutt**[3]

[1]Robotics Research Centre, International Institute of Information Technology, Hyderabad 500 032, India
[2,3]Department of Mechanical Engineering, Indian Institute of Technology Delhi, New Delhi 110 016, India
[1]*surilvshah@gmail.com,* [2]*saha@mech.iitd.ac.in,* [3]*jkdutt@mech.iitd.ac.in*

Abstract

This paper presents dynamics algorithms for multibody systems using the concepts of kinematic modules and Decoupled Natural Orthogonal Complement (DeNOC) matrices. A multibody system is treated as open-loop serial- or tree-type consisting of several modules, where each module consists of one or more than one serially connected links. In fact, it is shown how the knowledge for a serial system can be extended to a module of a tree-type system. The resulting equations are used to obtain recursive inverse and forward dynamics algorithms at inter- and intra-modular levels The methodology is illustrated using a tree-type four Degrees-of-Freedom (DoF) robotic gripper and 100-DOF serial-type rope system. It is also shown how efficiency of the proposed algorithm benefits as the DOF of a system becomes very large.

Keywords: Kinematic modules, Multibody systems, and Recursive algorithms

1 Introduction

Studies on any multibody system is attempted to perform either force or motion analyses. Kinematic relationships between constituent links or bodies of a multibody system are foundation for aforesaid analyses. With increase in multibody applications, the systems to be studied have become more complex and contain large number of bodies. Moreover, many multibody systems comprise of smaller subsystems. Hence, simply breaking down a system to body-level does not provide macroscopic purview of several kinematic and dynamic properties, thereby, increasing the complexity of analyses in many instances.

In this work, dynamics is proposed using the concept of Decoupled Natural Orthogonal Complement (DeNOC) matrices, which when combined with the uncoupled Newton-Euler equations of motion give rise to the minimal-order constrained dynamic equations of motion of the system at hand by eliminating the constraint forces. This decoupled form in the DeNOC matrices brings the benefits like analytical recursive expressions for the elements of the vectors and matrices appearing in the dynamic equations of motion, $O(n)$ recursive dynamics, etc., as reported for the serial-type systems[1].

Several other approaches[2–5] were also proposed in the literature to obtain recursive dynamics algorithms. In comparison to those approaches, the DeNOC-based approach considered the system as a whole, i.e., NE equations of motion of all the bodies in a system are first grouped together. Next, simple matrix operations using the DeNOC matrices, derived from the velocity constraints of serially connected bodies or kinematic modules, are performed to yield minimal-order equations of motion. As a consequence, a global approach is possible in contrast to the body-centric approaches[2, 3]. Moreover, the use of the DeNOC matrices allows one to uniformly develop both recursive inverse and forward dynamics algorithms, and analytical inversion of the Generalized Inertia Matrix[1]. The analytical inversion was not possible using any of the other recursive $O(n)$ dynamic formulations.

It is worth noting that in contrast to[1], in the present work a multiple system is assumed to be an open-loop serial- or tree-type consisting of a set of many kinematic modules as shown in Fig. 1, where each module contains serially connected rigid links and emerges from the last link of the parent module. The links in the module M_i are denoted as $1^i, \ldots, k^i, \ldots, \eta^i$, where superscript i is the module number. Total number of modules, number of links in the module, and the total DoF of all the modules, are designated using letters, s, η^i, and n, respectively. It is worth noting that a tree-type system with no branch degenerates to a serial-type system. Moreover, a closed-loop multibody system can be represented as a tree-type system with additional constraint forces at the cut joints. Hence, the methodology presented in the paper mainly emphasis on a tree-type system. In this paper, an entity say, $\boldsymbol{v}$, corresponding to the kth link is denoted as $\boldsymbol{v}_k$, whereas an entity corresponding to the ith module is denoted as $\boldsymbol{v}^i$.

The paper is organized as follows: Section 2 presents dynamics modeling. Recursive algorithms are presented in Section 3. Computational efficiency is presented in Section 4. Finally, conclusions are given in Section 5.

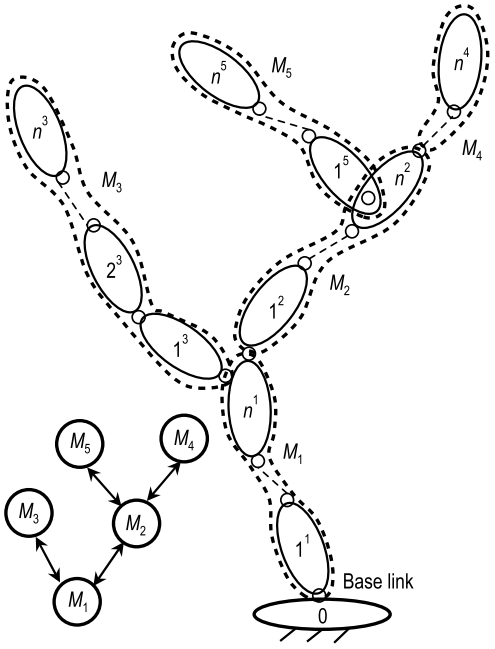

Fig. 1 A tree-type system and its modules

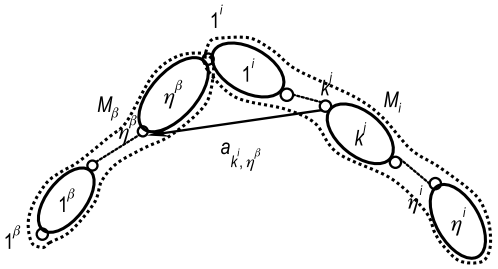

Fig. 2 Module M_i and its parent M_β

2 Dynamic Modeling

The DeNOC-based methodology for dynamic modeling begins with the Newton-Euler (NE) equations of motion. For the ith module, say M_i of Fig. 2, the NE equations of motion may be expressed in a compact form as

$$\boldsymbol{M}^i\dot{\boldsymbol{t}}^i + \boldsymbol{\Omega}^i\boldsymbol{M}^i\boldsymbol{E}^i\boldsymbol{t}^i = \boldsymbol{w}^i \tag{1}$$

where the matrices $\boldsymbol{M}^i$, $\boldsymbol{\Omega}^i$ and $\boldsymbol{E}^i$ are of sizes $6\eta^i \times 6\eta^i$ each, and $\boldsymbol{t}^i$ and $\boldsymbol{w}^i$ are the $6\eta^i$-dimensional vectors of module-twist and module-wrench associated with the ith module. Matrices, $\boldsymbol{M}^i$, $\boldsymbol{\Omega}^i$ and $\boldsymbol{E}^i$, and vector $\boldsymbol{t}^i$ and $\boldsymbol{w}^i$ are defined as

$$\begin{aligned} &\boldsymbol{M}^i \equiv \text{diag}\,[\boldsymbol{M}_1 \ldots \boldsymbol{M}_k \ldots \boldsymbol{M}_\eta]^i, \\ &\boldsymbol{\Omega}^i \equiv \text{diag}\,[\boldsymbol{\Omega}_1 \ldots \boldsymbol{\Omega}_k \ldots \boldsymbol{\Omega}_\eta]^i, \\ &\boldsymbol{E}^i \equiv \text{diag}\,[\boldsymbol{E}_1 \cdots \boldsymbol{E}_k \cdots \boldsymbol{E}_\eta]^i, \end{aligned} \quad \boldsymbol{w}^i \equiv \begin{bmatrix} \boldsymbol{w}_1 \\ \vdots \\ \boldsymbol{w}_k \\ \vdots \\ \boldsymbol{w}_\eta \end{bmatrix}^i \text{ and } \boldsymbol{t}^i \equiv \begin{bmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_k \\ \vdots \\ \boldsymbol{t}_\eta \end{bmatrix}^i \tag{2a}$$

where $\boldsymbol{M}_k$, $\boldsymbol{\Omega}_k$ and $\boldsymbol{E}_k$ are the 6×6 mass, angular velocity and coupling matrices, respectively, $\boldsymbol{t}_k$ represents twist of the kth link, and $\boldsymbol{w}_k$ is the 6-dimensional vector of wrench for the kth link of module M_i. They are obtained as

$$\begin{aligned} \boldsymbol{M}_{k^i} &\equiv \begin{bmatrix} \boldsymbol{I}_k & m_k\tilde{\boldsymbol{d}}_k \\ -m_k\tilde{\boldsymbol{d}}_k & m_k\boldsymbol{1} \end{bmatrix}^i, \quad \boldsymbol{\Omega}_{k^i} \equiv \begin{bmatrix} \tilde{\boldsymbol{\omega}}_k & \boldsymbol{O} \\ \boldsymbol{O} & \tilde{\boldsymbol{\omega}}_k \end{bmatrix}^i, \\ \boldsymbol{E}_{k^i} &\equiv \begin{bmatrix} \boldsymbol{1} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix}^i, \quad \boldsymbol{t}_{k^i} \equiv \begin{bmatrix} \boldsymbol{\omega}_k \\ \dot{\boldsymbol{o}}_k \end{bmatrix}^i \text{ and } \boldsymbol{w}_{k^i} \equiv \begin{bmatrix} \boldsymbol{n}_k \\ \boldsymbol{f}_k \end{bmatrix}^i \end{aligned} \tag{2b}$$

where $\boldsymbol{d}_k, \boldsymbol{\omega}_k$, and $\dot{\boldsymbol{o}}_k$ represent the vector connecting origin of the kth link to its center-of-mass, the angular velocity vector of the kth link and the linear velocity vector of the origin of the kth link, respectively, whereas $\tilde{\boldsymbol{\omega}}_k$ and $\tilde{\boldsymbol{d}}_k$ are the 3×3 skew-symmetric matrices associated with $\boldsymbol{\omega}_k$ and $\boldsymbol{d}_k$. Moreover, $\boldsymbol{I}_k$ is a 3×3 inertia tensor about the origin, m_k is the mass of the kth link, $\boldsymbol{n}_k$ and $\boldsymbol{f}_k$ are the moment about and force applied at origin, and '$\boldsymbol{O}$' and '$\boldsymbol{1}$' denote null and identity matrices of compatible dimensions Combining (1) for $i = 1, \ldots, s$, the uncoupled NE equations of motion for the s-coupled modules are then put together as

$$\boldsymbol{M}\dot{\boldsymbol{t}} + \boldsymbol{\Omega}\boldsymbol{M}\boldsymbol{E}\boldsymbol{t} = \boldsymbol{w}, \text{ where } \boldsymbol{w} = \boldsymbol{w}^E + \boldsymbol{w}^C \tag{3}$$

In (3), $\boldsymbol{M}, \boldsymbol{\Omega}$, and $\boldsymbol{E}$ are the $6n \times 6n$ generalized matrices of mass, angular velocity, and coupling, respectively, and $\boldsymbol{t}$ and $\boldsymbol{w}$ are the $6n$-dimensional vectors of generalized twist and generalized wrench. They are given as follows:

$$\begin{aligned} &\boldsymbol{\Omega} \equiv \text{diag}\,[\boldsymbol{\Omega}^1 \ldots \boldsymbol{\Omega}^s], \\ &\boldsymbol{M} \equiv \text{diag}\,[\boldsymbol{M}^1 \ldots \boldsymbol{M}^s], \\ &\boldsymbol{E} \equiv \text{diag}\,[\boldsymbol{E}^1 \ldots \boldsymbol{E}^s] \end{aligned} \quad \boldsymbol{t} \equiv \begin{bmatrix} \boldsymbol{t}^1 \\ \vdots \\ \boldsymbol{t}^i \\ \vdots \\ \boldsymbol{t}^s \end{bmatrix} \text{ and } \boldsymbol{w} \equiv \begin{bmatrix} \boldsymbol{w}^1 \\ \vdots \\ \boldsymbol{w}^i \\ \vdots \\ \boldsymbol{w}^s \end{bmatrix} \tag{4}$$

2.1 Derivation of the DeNOC matrices

In order to obtain constrained equations of motion the module-level DeNOC matrices are derived in this subsection using the velocity constraints. As shown in Fig. 2, M_β is the parent of the module M_i. The symbol 'β' is used to signify a parent. For the ith module M_i, the $6\eta^i$-dimensional vector module-twist, $\boldsymbol{t}^i$, is obtained from the module-twist $\boldsymbol{t}^\beta$ of its parent module M_β, and the η^i-dimensional joint-rate, $\dot{\boldsymbol{q}}^i$, i.e.,

$$\boldsymbol{t}^i = \boldsymbol{A}^{i,\beta}\boldsymbol{t}^\beta + \boldsymbol{N}^i\dot{\boldsymbol{q}}^i \tag{5}$$

where $\boldsymbol{t}^i$ is defined in (2a), and $\dot{\boldsymbol{q}}^i$ is defined as follows:

$$\dot{\boldsymbol{q}}^i \equiv [\dot{\theta}_{1^i} \cdots \dot{\theta}_{k^i} \cdots \dot{\theta}_{\eta^i}]^T \tag{6}$$

where $\dot{\theta}_{k^i}$ is the kth joint rate of the ith module. The $6\eta^i \times 6\eta^\beta$ matrix $\boldsymbol{A}^{i,\beta}$ in (5) is associated with the twist-propagation of the parent module (βth) to the child module (ith) and hence referred to as module-twist propagation matrix. Moreover, the $6\eta^i \times \eta^i\boldsymbol{N}^i$ represents module-joint-rate propagation matrix.

Matrices $\boldsymbol{A}^{i,\beta}$ and $\boldsymbol{N}^i$ are given by

$$\boldsymbol{A}^{i,\beta} \equiv \begin{bmatrix} \boldsymbol{O} & \cdots & \boldsymbol{O} & \boldsymbol{A}_{1^i,\eta^\beta} \\ \vdots & & \vdots & \vdots \\ \boldsymbol{O} & \cdots & \boldsymbol{O} & \boldsymbol{A}_{k^i,\eta^\beta} \\ \vdots & & \vdots & \vdots \\ \boldsymbol{O} & \cdots & \boldsymbol{O} & \boldsymbol{A}_{\eta^i,\eta^\beta} \end{bmatrix} \text{ and } \boldsymbol{N}^i \equiv \begin{bmatrix} \boldsymbol{p}_1 & & \boldsymbol{0} & & \cdots & \boldsymbol{0} \\ \boldsymbol{A}_{2,1}\boldsymbol{p}_1 & \ddots & \boldsymbol{p}_{k-1} & & \ddots & \vdots \\ \vdots & \ddots & \boldsymbol{A}_{k,k-1}\boldsymbol{p}_{k-1} & & \ddots & \boldsymbol{0} \\ \boldsymbol{A}_{n,1}\boldsymbol{p}_1 & & \cdots & \ddots & \boldsymbol{A}_{\eta,\eta-1}\boldsymbol{p}_{\eta-1} & \boldsymbol{p}_\eta \end{bmatrix}^i \tag{7a}$$

In (7a) '$\boldsymbol{O}$' and '$\boldsymbol{0}$' denote null matrix and null vector of compatible dimensions, whereas the block elements $\boldsymbol{A}_{k,k-1}$ and $\boldsymbol{p}_k$ in $\boldsymbol{N}^i$ are the 6×6 twist-propagation matrix and the 6-dimensional joint-motion propagation vector given by

$$\boldsymbol{A}_{k,k-1} \equiv \begin{bmatrix} 1 & \boldsymbol{O} \\ \tilde{\boldsymbol{a}}_{k,k-1} & \boldsymbol{1} \end{bmatrix}, \text{ and } \boldsymbol{p}_k \equiv \begin{bmatrix} \boldsymbol{e}_k \\ \boldsymbol{0} \end{bmatrix} \text{ (for revolute) } \boldsymbol{p}_k \equiv \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{e}_k \end{bmatrix} \text{ (for prismatic)} \tag{7b}$$

where $\tilde{\boldsymbol{a}}_{k,k-1}$ is the 3×3 skew-symmetric matrix associated with vector $\boldsymbol{a}_{k,k-1}$ connecting origins of the kth and $(k-1)$th links. Moreover, $\boldsymbol{e}_k$ is the unit vector along the axis of rotation or translation of the kth joint. Next, the definitions of generalized twist and generalized joint rate vectors of the multibody system at hand are introduced as

$$\boldsymbol{t} \equiv \begin{bmatrix} \boldsymbol{t}^1 \\ \vdots \\ \boldsymbol{t}^i \\ \vdots \\ \boldsymbol{t}^s \end{bmatrix}, \text{ and } \dot{\boldsymbol{q}} \equiv \begin{bmatrix} \dot{\boldsymbol{q}}^1 \\ \vdots \\ \dot{\boldsymbol{q}}^i \\ \vdots \\ \dot{\boldsymbol{q}}^s \end{bmatrix} \tag{8}$$

Now, substitution of (5) into (8) yields

$$\boldsymbol{t} = \boldsymbol{N}_l \boldsymbol{N}_d \dot{\boldsymbol{q}} \tag{9}$$

where the $6n\times 6n$ block lower triangular matrix $\boldsymbol{N}_l$ and the $6n\times n$ block diagonal matrix $\boldsymbol{N}_d$ are given by

$$\boldsymbol{N}_l \equiv \begin{bmatrix} \boldsymbol{1}^1 & & & \boldsymbol{O}'s \\ \boldsymbol{A}^{2,1} & \boldsymbol{1}^2 & & \\ \vdots & \ddots & \ddots & \\ \boldsymbol{A}^{s,1} & \cdots & \boldsymbol{A}^{s,s-1} & \boldsymbol{1}^s \end{bmatrix}; \quad \boldsymbol{A}^{j,i} \equiv \boldsymbol{O}, \text{ if } M_j \notin \boldsymbol{\gamma}^i$$
$$\boldsymbol{N}_d \equiv \text{diag} \left[\boldsymbol{N}^1 \cdots \boldsymbol{N}^i \cdots \boldsymbol{N}^s \right] \tag{10}$$

where $\boldsymbol{1}^i$ is the $6\eta^i \times 6\eta^i$ identity matrix and γ^i stands for the set of kinematic modules originating from module M_i including M_i as shown within the dashed boundary of Fig. 3. The matrices $\boldsymbol{N}_l$ and $\boldsymbol{N}_d$ are the desired DeNOC matrices for the system under study written in terms of the module information rather than in terms of the link information They are generalization over those proposed in[1].

2.2 Equations of motion

As vector of constraint forces and moments is orthogonal to the columns of velocity transformation matrix[6], one may show that the pre-multiplication of (3) by $\boldsymbol{N}_d^T \boldsymbol{N}_l^T$ yields the minimal set of equations of motion, where the constraint wrenches are eliminated, i.e.,

$$\boldsymbol{N}_d^T \boldsymbol{N}_l^T (\boldsymbol{M}\dot{\boldsymbol{t}} + \boldsymbol{\Omega M E t}) = \boldsymbol{N}_d^T \boldsymbol{N}_l^T \boldsymbol{w}^E \tag{11}$$

where $\boldsymbol{w}^E$ is the generalized wrench due to external moments and forces. Substituting (9) and its time derivative into (11), and rearranging them, the following equations of motion is obtained

$$\boldsymbol{I}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} = \boldsymbol{\tau} + \boldsymbol{\tau}^F \tag{12}$$

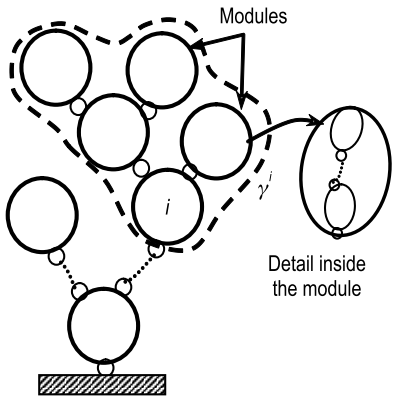

Fig. 3 Definition of γ_i

where the expressions of the $n \times n$ Generalized Inertia Matrix (GIM) $\boldsymbol{I}$, the $n \times n$ Matrix of Convective Inertia (MCI) $\boldsymbol{C}$, and the n-dimensional vector of the generalized external forces $\boldsymbol{\tau}$ are given as

$$\boldsymbol{I} \equiv \boldsymbol{N}^T \boldsymbol{M N},\ \boldsymbol{C} \equiv \boldsymbol{N}^T (\boldsymbol{M}\dot{\boldsymbol{N}} + \boldsymbol{\Omega M E N}), \boldsymbol{\tau} \equiv \boldsymbol{N}^T \boldsymbol{w}^E \tag{13}$$

where $\boldsymbol{N} = \boldsymbol{N}_l \boldsymbol{N}_d$. In order to analyze a closed-loop system, it is cut at appropriate joints to form several open-loop serial or tree-type sub-systems. The cut joints are substituted with Lagrange multipliers, which are then treated as external forces for the resulting open-loop sub-systems, serial- or tree-type. The concepts of "determinate" and "indeterminate" subsystems[7] are then used to decide whether the unknown Lagrange multipliers and the driving torques can be solved from the kinematic results of a subsystem or not. This resulted in elegant and efficient subsystem level recursions as reported in[7].

3 Recursive Dynamics

Based on the dynamic modeling presented in Section 2, recursive inverse and forward dynamics algorithms can be developed. In order to comprehend the steps better, the algorithms are developed for a four-DOF tree-type gripper shown in Fig. 4. However, a general algorithm applicable to any serial or tree type open-loop system can be written. The gripper in Fig. 4 can be used for holding an object to be manipulated, say, by a robot hand. It has three kinematic modules, namely, M_1, M_2, and M_3. Module M_1 has one link, module M_2 has two links and module M_3 has one link. Hence, the 24×24 and 24×4 matrices $\boldsymbol{N}_l$ and $\boldsymbol{N}_d$ are obtained from (10) as

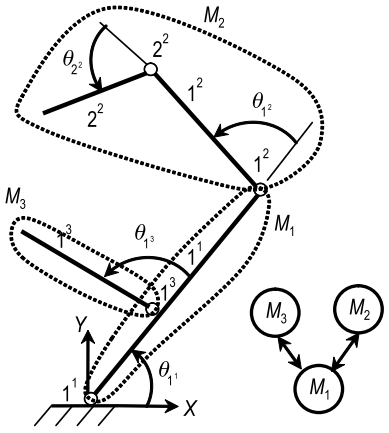

Fig. 4 Joint torques for robotic gripper

$$\boldsymbol{N}_l \equiv \begin{bmatrix} \boldsymbol{1}^1 & & \boldsymbol{O}'s \\ \boldsymbol{A}^{2,1} & \boldsymbol{1}^2 & \\ \boldsymbol{A}^{3,1} & \boldsymbol{O} & \boldsymbol{1}^3 \end{bmatrix}, \text{ and } \boldsymbol{N}_d \equiv \begin{bmatrix} \boldsymbol{N}^1 & & \boldsymbol{O}'s \\ & \boldsymbol{N}^2 & \\ \boldsymbol{O}'s & & \boldsymbol{N}^3 \end{bmatrix} \tag{14}$$

Table 1 Recursive inverse dynamics of gripper

Step 1. Forward recursion: Compute $\boldsymbol{t}, \dot{\boldsymbol{t}}$ and $\boldsymbol{w}$			
Module M_i	Inter-modular	Link k^i	Intra-modular
$i=1$	$\boldsymbol{t}^1 = \boldsymbol{N}^1\dot{\boldsymbol{q}}^1; \dot{\boldsymbol{t}}^1 = \dot{\boldsymbol{N}}^1\dot{\boldsymbol{q}}^1 + \boldsymbol{N}^1\ddot{\boldsymbol{q}}^1$ $\boldsymbol{w}^1 \equiv \boldsymbol{M}^1\dot{\boldsymbol{t}}^1 + \boldsymbol{\Omega}^1\boldsymbol{M}^1\boldsymbol{E}^1\boldsymbol{t}^1$	$k=1^1$	$\boldsymbol{t}_{1^1} = \boldsymbol{p}_{1^1}\dot{\theta}_{1^1}; \dot{\boldsymbol{t}}_{1^1} = \boldsymbol{\Omega}_{1^1}\boldsymbol{p}_{1^1}\dot{\theta}_{1^1} + \boldsymbol{p}_{1^1}\ddot{\theta}_{1^1}$ $w_{1^1} \equiv \boldsymbol{M}_{1^1}\dot{\boldsymbol{t}}_{1^1} + \boldsymbol{\Omega}_{1^1}\boldsymbol{M}_{1^1}\boldsymbol{E}_{1^1}\boldsymbol{t}_{1^1}$
$i=2$	$\boldsymbol{t}^2 = \boldsymbol{A}^{2,1}\boldsymbol{t}^1 + \boldsymbol{N}^2\dot{\boldsymbol{q}}^2$ $\dot{\boldsymbol{t}}^2 = \dot{\boldsymbol{A}}^{2,1}\boldsymbol{t}_1 + \boldsymbol{A}^{2,1}\dot{\boldsymbol{t}}^1 + \dot{\boldsymbol{N}}^2\dot{\boldsymbol{q}}^2 + \boldsymbol{N}^2\ddot{\boldsymbol{q}}^2$ $\boldsymbol{w}^2 \equiv \boldsymbol{M}^2\dot{\boldsymbol{t}}^2 + \boldsymbol{\Omega}^2\boldsymbol{M}^2\boldsymbol{E}^2\boldsymbol{t}^2$	$k=1^2$	$\boldsymbol{t}_{1^2} = \boldsymbol{A}_{1^2,1^1}\boldsymbol{t}_{1^1} + \boldsymbol{p}_{1^2}\dot{\theta}_{1^2};$ $\dot{\boldsymbol{t}}_{1^2} = \boldsymbol{A}_{1^2,1^1}\dot{\boldsymbol{t}}_{1^1} + \dot{\boldsymbol{A}}_{1^2,1^1}\boldsymbol{t}_{1^1} + \boldsymbol{\Omega}_{1^2}\boldsymbol{p}_{1^2}\dot{\theta}_{1^2} + \boldsymbol{p}_{1^2}\ddot{\theta}_{1^2}$ $\boldsymbol{w}_{1^2} \equiv \boldsymbol{M}_{1^2}\dot{\boldsymbol{t}}_{1^2} + \boldsymbol{\Omega}_{1^2}\boldsymbol{M}_{1^2}\boldsymbol{E}_{1^2}\boldsymbol{t}_{1^2}$
		$k=2^2$	$\boldsymbol{t}_{2^2} = \boldsymbol{A}_{2^2,1^2}\boldsymbol{t}_{1^2} + \boldsymbol{p}_{2^2}\dot{\theta}_{2^2};$ $\dot{\boldsymbol{t}}_{2^2} = \boldsymbol{A}_{2^2,1^2}\dot{\boldsymbol{t}}_{1^2} + \dot{\boldsymbol{A}}_{2^2,1^2}\boldsymbol{t}_{1^2} + \boldsymbol{\Omega}_{2^2}\boldsymbol{p}_{2^2}\dot{\theta}_{2^2} + \boldsymbol{p}_{2^2}\ddot{\theta}_{2^2}$ $\boldsymbol{w}_{2^2} \equiv \boldsymbol{M}_{2^2}\dot{\boldsymbol{t}}_{2^2} + \boldsymbol{\Omega}_{2^2}\boldsymbol{M}_{2^2}\boldsymbol{E}_{2^2}\boldsymbol{t}_{2^2}$
$i=3$	$\boldsymbol{t}^3 = \boldsymbol{A}^{3,1}\boldsymbol{t}^1 + \boldsymbol{N}^3\dot{\boldsymbol{q}}^3$ $\dot{\boldsymbol{t}}^3 = \dot{\boldsymbol{A}}^{3,1}\boldsymbol{t}_1 + \boldsymbol{A}^{3,1}\dot{\boldsymbol{t}}^1 + \dot{\boldsymbol{N}}^3\dot{\boldsymbol{q}}^3 + \boldsymbol{N}^3\ddot{\boldsymbol{q}}^3$ $\boldsymbol{w}^3 \equiv \boldsymbol{M}^3\dot{\boldsymbol{t}}^3 + \boldsymbol{\Omega}^3\boldsymbol{M}^3\boldsymbol{E}^3\boldsymbol{t}^3$	$k=1^3$	$\boldsymbol{t}_{1^3} = \boldsymbol{A}_{1^3,1^1}\boldsymbol{t}_{1^1} + \boldsymbol{p}_{1^3}\dot{\theta}_{1^3}$ $\dot{\boldsymbol{t}}_{1^3} = \boldsymbol{A}_{1^3,1^1}\dot{\boldsymbol{t}}_{1^1} + \dot{\boldsymbol{A}}_{1^3,1^1}\boldsymbol{t}_{1^1} + \boldsymbol{\Omega}_{1^3}\boldsymbol{p}_{1^3}\dot{\theta}_{1^3} + \boldsymbol{p}_{1^3}\ddot{\theta}_{1^3}$ $\boldsymbol{w}_{1^3} \equiv \boldsymbol{M}_{1^3}\dot{\boldsymbol{t}}_{1^3} + \boldsymbol{\Omega}_{1^3}\boldsymbol{M}_{1^3}\boldsymbol{E}_{1^3}\boldsymbol{t}_{1^3}$
Step 2. Backward recursion: : Compute $\boldsymbol{\tau} = \boldsymbol{N}_d^T\boldsymbol{N}_l^T\boldsymbol{w}$			
$i=3$	$\tilde{\boldsymbol{w}}^3 = \boldsymbol{w}^3; \boldsymbol{\tau}^3 = \boldsymbol{N}^{3^T}\tilde{\boldsymbol{w}}^3$	$k=1^3$	$\tilde{\boldsymbol{w}}_{1^3} = \boldsymbol{w}_{1^3}; \tau_{1^3} = \boldsymbol{p}_{1^3}^T\tilde{\boldsymbol{w}}_{1^3}$
$i=2$	$\tilde{\boldsymbol{w}}^2 = \boldsymbol{w}^2; \boldsymbol{\tau}^2 = \boldsymbol{N}^{2^T}\tilde{\boldsymbol{w}}^2$	$k=2^2$	$\tilde{\boldsymbol{w}}_{2^2} = \boldsymbol{w}_{2^2}; \tau_{2^2} = \boldsymbol{p}_{2^2}^T\tilde{\boldsymbol{w}}_{2^2}$
		$k=1^2$	$\tilde{\boldsymbol{w}}_{1^2} = \boldsymbol{w}_{1^2} + \boldsymbol{A}_{2^2,1^2}^T\tilde{\boldsymbol{w}}_{2^2}; \tau_{1^2} = \boldsymbol{p}_{1^2}^T\tilde{\boldsymbol{w}}_{1^2}$
$i=1$	$\tilde{\boldsymbol{w}}^1 = \boldsymbol{w}^1 + (\boldsymbol{A}^{2,1})^T\tilde{\boldsymbol{w}}^2 + (\boldsymbol{A}^{3,1})^T\tilde{\boldsymbol{w}}^3;$	$k=1^1$	$\tilde{\boldsymbol{w}}_{1^1} = \boldsymbol{w}_{1^1} + \boldsymbol{A}_{1^2,1^1}^T\tilde{\boldsymbol{w}}_{1^2} + \boldsymbol{A}_{1^3,1^1}^T\tilde{\boldsymbol{w}}_{1^3};$
	$\boldsymbol{\tau}^1 = \boldsymbol{N}^{1^T}\tilde{\boldsymbol{w}}^1;$		$\tau_{1^1} = \boldsymbol{p}_{1^1}^T\tilde{\boldsymbol{w}}_{1^1}$

Note: Gravity can be taken into account by adding gravitational acceleration to the accelerations of links whose parent is base link.

3.1 Inverse dynamics

In inverse dynamics one attempts to find the joint torques and forces for a given set of joint motions, i.e., joint positions, velocity and acceleration. Table 1 shows the steps which comprise of intra- and inter-modular computations. In step 1, i.e., forward recursion, module-twist $\boldsymbol{t}^i$, module-twist-rate $\dot{\boldsymbol{t}}^i$, and module inertia wrench $\boldsymbol{w}^i$ are obtained using the information of the parent module, whereas in step 2, i.e., backward recursion, joint torques and forces ($\boldsymbol{\tau}^i$) are obtained based on (11–12). It may be noted here that the inter-modular steps are nothing but the compact representations of the intra-modular steps and hence, they are equivalent.

Numerical values for the joint torques for a given set of input motions are then obtained using the Recursive Dynamic Simulator (ReDySim)[9] developed in MATLAB[10] environment based on the proposed inverse dynamics algorithm. The trajectory for each joint is taken as per (15), whereas, the

initial and final joint angles are denoted as $\theta(0)$ and $\theta(T)$, respectively, are given in Table 2.

Table 2 Joint motions

Joint	$\theta(0)$	$\theta(T)$
1^1	0	60°
1^2	0	80°
2^2	0	80°
1^3	90°	120°

$$\theta = \theta(0) + \frac{\theta(T) - \theta(0)}{T}\left[t - \frac{T}{2\pi}\sin\left(\frac{2\pi}{T}t\right)\right] \tag{15}$$

The joint trajectory in (15) is so chosen that the initial and final joint rates and accelerations for all joints are zeros. This ensures smooth motion of the joints. The lengths and masses of the links are taken as $l_{1^1} = 0.1$ m, $l_{1^2} = l_{2^2} = l_{1^3} = 0.05$ m, $m_{1^1} = 0.4$ Kg, and $m_{1^2} = m_{2^2} = m_{1^3} = 0.2$ Kg.

Figure 5 shows the joint torques required over a time interval of $T = 1$ second to follow the prescribed trajectories given by (15). In order to validate the results a CAD model of the gripper mechanism was developed in ADAMS[8] software and used for computation of joint torques. The results are shown in Fig. 5, which show close match with those obtained using ReDySim. Hence the numerical results are validated. As far as the computational complexity is concerned, ReDySim took only 0.025 sec compared to 1.95 sec by ADAMS software when implemented in Intel T2300@1.66 GHz computing system.

3.2 Forward dynamics

The forward dynamics algorithm attempts to find the joint acceleration from the equations of motion of (12), while joint torques and forces are known. Numerical integration of these joint accelerations will then allow one to find the system's configuration at every time instance. The above forward dynamics computations followed by numerical integration are referred in literature as simulation. Unlike a recursive

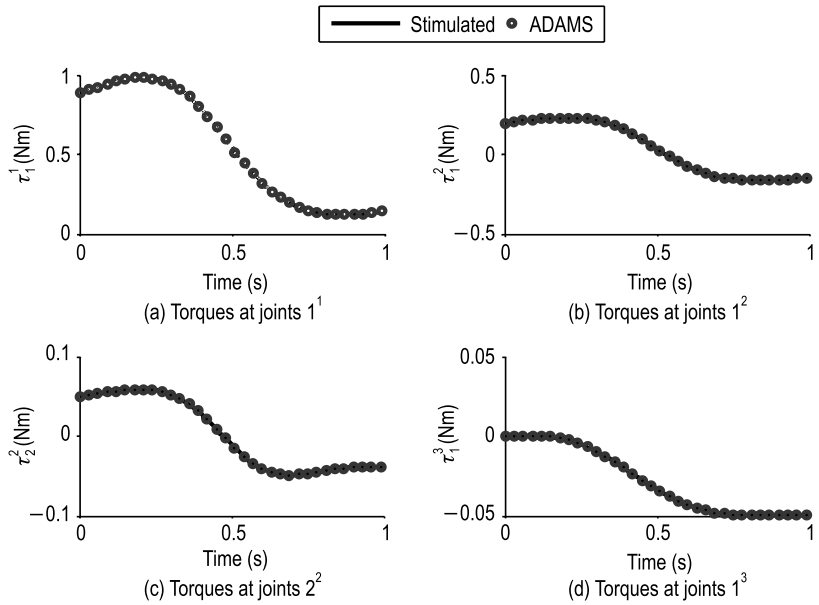

Fig. 5 Joint torques for robotic gripper

inverse dynamics algorithm presented above, a recursive forward dynamics algorithm is rather complex. Inverse dynamics calculations require only matrix-vector operations as shown in the previous subsection. In the case of forward dynamics, solution of the joint accelerations from (12) is required, necessitating, an order (n^3) numerical calculation. For large n, such algorithms should be avoided due to poor computational efficiency and problems associated with numerical instability[11]. Hence, recursive order (n) forward dynamics algorithm is preferred. Here one such algorithm is presented, which is illustrated with 4-DOF gripper shown in Fig. 4. Steps are given below:

1) The GIM of the 3-module gripper system is obtained using (13–14) as

$$\boldsymbol{I} \equiv \boldsymbol{N}_d^T(\boldsymbol{N}_l^T \boldsymbol{M} \boldsymbol{N}_l)\boldsymbol{N}_d \equiv \begin{bmatrix} \boldsymbol{N}^{1^T}\tilde{\boldsymbol{M}}^1\boldsymbol{N}^1 & & \text{sym} \\ \boldsymbol{N}^{2^T}\tilde{\boldsymbol{M}}^2\boldsymbol{A}^{2,1}\boldsymbol{N}^1 & \boldsymbol{N}^{2^T}\tilde{\boldsymbol{M}}^2\boldsymbol{N}^2 & \\ \boldsymbol{N}^{3^T}\tilde{\boldsymbol{M}}^3\boldsymbol{A}^{3,1}\boldsymbol{N}^1 & \boldsymbol{O} & \boldsymbol{N}^{3^T}\tilde{\boldsymbol{M}}^3\boldsymbol{N}^3 \end{bmatrix} \tag{16}$$

where $\boldsymbol{I}$ is the 4×4 matrix. In (16), the block elements $\boldsymbol{N}^i$ and $\boldsymbol{A}^{j,i}$ are obtained in (7a). Moreover, the mass matrix of the composite module $\tilde{\boldsymbol{M}}^i$, for $i = 3, 2, 1$, is obtained recursively as

$$\tilde{\boldsymbol{M}}^3 = \boldsymbol{M}^3, \tilde{\boldsymbol{M}}^2 = \boldsymbol{M}^2, \tilde{\boldsymbol{M}}^1 = \boldsymbol{M}^1 + \sum_{j=3,2} (\boldsymbol{A}^{j,1})^T \tilde{\boldsymbol{M}}^j \boldsymbol{A}^{j,1} \tag{17}$$

2) Following the Block Reverse Gaussian elimination (BRGE), similar to the RGE[12] for serial-type system, the GIM in (16) is decomposed as $\boldsymbol{I} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$, where the 4×4 block upper triangular matrix $\boldsymbol{U}$ and block diagonal matrix $\boldsymbol{D}$ have the following representations:

$$\boldsymbol{U} \equiv \begin{bmatrix} \boldsymbol{1}^1 & \boldsymbol{N}^{1^T}(\boldsymbol{A}^{2.1})^T\boldsymbol{\Psi}^2 & \boldsymbol{N}^{1^T}(\boldsymbol{A}^{3.1})^T\boldsymbol{\Psi}^3 \\ & \boldsymbol{1}_2 & \boldsymbol{O} \\ \boldsymbol{O}'s & & \boldsymbol{1}_3 \end{bmatrix}, \text{ and } \boldsymbol{D} \equiv \begin{bmatrix} \boldsymbol{N}^{1^T}\hat{\boldsymbol{\Psi}}^1 & & \boldsymbol{O}'s \\ & \boldsymbol{N}^{2^T}\hat{\boldsymbol{\Psi}}^2 & \\ \boldsymbol{O}'s & & \boldsymbol{N}^{3^T}\hat{\boldsymbol{\Psi}}^3 \end{bmatrix} \tag{18}$$

block elements $\boldsymbol{\Psi}^i$ and $\hat{\boldsymbol{\Psi}}^i$, for $i = 1, 2, 3$, are expressed as

$$\begin{aligned} &\text{For } i = 1, 2, 3 \\ &\hat{\boldsymbol{\Psi}}^i = \hat{\boldsymbol{M}}^i\boldsymbol{N}^i \text{ and } \boldsymbol{\Psi}^i = \hat{\boldsymbol{\Psi}}^i\hat{\boldsymbol{I}}^{i^{-1}} \text{ where } \hat{\boldsymbol{I}}^i = \boldsymbol{N}^{i^T}\hat{\boldsymbol{\Psi}}^i \end{aligned} \tag{19}$$

In (19), the mass matrix of articulated module $\hat{\boldsymbol{M}}^i$, generalization over the articulated body inertia (Saha, 1999) of a serial-type system, for $i = 3, 2, 1$, is obtained recursively as

$$\hat{\boldsymbol{M}}^3 = \boldsymbol{M}^3, \hat{\boldsymbol{M}}^2 = \boldsymbol{M}^2, \hat{\boldsymbol{M}}^1 = \boldsymbol{M}^1 + \sum_{j=3,2} (\boldsymbol{A}^{j,1})^T \boldsymbol{\Phi}^j \hat{\boldsymbol{M}}^j \boldsymbol{A}^{j,1} \tag{20}$$

where the $6\eta^i \times 6\eta^i$ matrix $\boldsymbol{\Phi}^i$ is obtained as

$$\boldsymbol{\Phi}^j = \boldsymbol{1} - \boldsymbol{\Psi}^j\boldsymbol{N}^{j^T}, \text{ for } j = 2, 3 \tag{21}$$

It is worth noting that in contrast to[12], here, the block decomposition of GIM is followed, which allows one to obtain microscopic purview of the several dynamic properties.

3) Based on the $\boldsymbol{UDU}^T$ decomposition of the GIM, the constrained equations of motion, (12), are rewritten as follows:

$$\boldsymbol{UDU}^T\ddot{\boldsymbol{q}} = \boldsymbol{\phi}, \text{ where } \boldsymbol{\phi} = \boldsymbol{\tau} - \boldsymbol{C}\dot{\boldsymbol{\theta}} \tag{22}$$

The joint accelerations are then solved using three sets of linear algebraic equations, namely

$$\begin{aligned} &1) \quad \boldsymbol{U}\hat{\boldsymbol{\phi}} = \boldsymbol{\phi}, \text{ where } \hat{\boldsymbol{\phi}} = \boldsymbol{DU}^T\ddot{\boldsymbol{\theta}} \\ &2) \quad \boldsymbol{D}\tilde{\boldsymbol{\phi}} = \hat{\boldsymbol{\phi}}, \text{ where } \tilde{\boldsymbol{\phi}} = \boldsymbol{U}^T\ddot{\boldsymbol{\theta}} \\ &3) \quad \boldsymbol{U}^T\ddot{\boldsymbol{q}} = \tilde{\boldsymbol{\phi}} \end{aligned} \tag{23}$$

The Recursive forward dynamics algorithm to obtain the joint acceleration using (23) is presented in Table 3. Numerical results are obtained for the fall of the gripper under gravity. The accelerations were numerically integrated twice using fixed step ordinary differential equation solver based on Runge-Kutta formula, for the following initial joint angles and rates $\theta_{1^1} = -60^\circ$, $\theta_{2^1} = \theta_{2^2} = \theta_{1^3} = 0$, and $\dot{\theta}_{1^1} = \dot{\theta}_{1^2} = \dot{\theta}_{2^2} = \dot{\theta}_{1^3} = 0$. The step size was taken as 0.001 sec. Figure 6 shows the plots for the simulated joint angles.

Table 3 Recursive forward dynamics of gripper

Step 1. Backward recursion: Compute $\hat{\boldsymbol{\phi}}$ and $\tilde{\boldsymbol{\phi}}$ from $\boldsymbol{U}\hat{\boldsymbol{\phi}} = \boldsymbol{\phi}$ and $D\tilde{\boldsymbol{\phi}} = \hat{\boldsymbol{\phi}}$			
Module M_i	Inter-modular	Link k^i	Intra-modular
$i = 3$	$\hat{\boldsymbol{\varphi}}^3 = \boldsymbol{\varphi}^3; \tilde{\boldsymbol{\varphi}}^3 = \hat{\boldsymbol{I}}^{3^{-1}}\hat{\boldsymbol{\varphi}}^3; \boldsymbol{\eta}^3 = \boldsymbol{\Psi}^3\hat{\boldsymbol{\varphi}}^3$	$k = 1^3$	$\hat{\varphi}_{1^3} = \varphi_{1^3}; \tilde{\varphi}_{1^3} = \hat{\varphi}_{1^3}/\hat{m}_{1^3}; \boldsymbol{\eta}_{1^2} = \boldsymbol{\psi}_{2^2}\hat{\varphi}_{2^2}$
$i = 2$	$\hat{\boldsymbol{\varphi}}^2 = \boldsymbol{\varphi}^2; \tilde{\boldsymbol{\varphi}}^2 = \hat{\boldsymbol{I}}^{2^{-1}}\hat{\boldsymbol{\varphi}}^2; \boldsymbol{\eta}^2 = \boldsymbol{\Psi}^2\hat{\boldsymbol{\varphi}}^2$	$k = 2^2$	$\hat{\varphi}_{2^2} = \varphi_{2^2}; \tilde{\varphi}_{2^2} = \hat{\varphi}_{2^2}/\hat{m}_{2^2}\boldsymbol{\eta}_{2^2} = \boldsymbol{\psi}_{2^2}\hat{\varphi}_{2^2}$
		$k = 1^2$	$\tilde{\boldsymbol{\eta}}_{1^2} = \boldsymbol{A}^T_{2^2,1^2}\boldsymbol{\eta}_{2^2}$; $\hat{\varphi}_{1^2} = \varphi_{1^2} - \boldsymbol{p}^T_{1^2}\tilde{\boldsymbol{\eta}}_{1^2}; \tilde{\varphi}_{1^2} = \hat{\varphi}_{1^2}/\hat{m}_{1^2}$
$i = 1$	$\tilde{\boldsymbol{\eta}}^1 = (\boldsymbol{A}^{21})^T\boldsymbol{\eta}^2 + (\boldsymbol{A}^{31})^T\boldsymbol{\eta}^3$; $\hat{\boldsymbol{\varphi}}^1 = \boldsymbol{\varphi}^1 - \boldsymbol{N}^{1^T}\tilde{\boldsymbol{\eta}}^1; \tilde{\boldsymbol{\varphi}}^1 = \hat{\boldsymbol{I}}^{1^{-1}}\hat{\boldsymbol{\varphi}}^1$	$k = 1^1$	$\tilde{\boldsymbol{\eta}}_{1^1} = \boldsymbol{A}^T_{1^2,1^1}\boldsymbol{\eta}_{1^2} + \boldsymbol{A}^T_{1^3,1^1}\boldsymbol{\eta}_{1^3}$ $\hat{\varphi}_{1^1} = \varphi_{1^1} - \boldsymbol{p}^T_{1^1}\tilde{\boldsymbol{\eta}}_{1^1}; \tilde{\varphi}_{1^1} = \hat{\varphi}_{1^1}/\hat{m}_{1^1}$
Step 2. Forward recursion: Compute $\ddot{\boldsymbol{q}}$ from $\boldsymbol{U}^T\ddot{\boldsymbol{q}} = \tilde{\varphi}$			
$i = 1$	$\ddot{\boldsymbol{q}}^1 = \tilde{\boldsymbol{\varphi}}^1; 1$	$k = 1^1$	$\ddot{\theta}_{1^1} = \tilde{\varphi}_{1^1}; \boldsymbol{\mu}_{1^1} = \boldsymbol{p}_{1^1}\ddot{\theta}_{1^1}$
$i = 2$	$\tilde{\boldsymbol{\mu}}^2 = \boldsymbol{A}^{2,1}\boldsymbol{\mu}^1; \ddot{\boldsymbol{q}}^2 = \tilde{\boldsymbol{\varphi}}^2 - \boldsymbol{\Psi}^{2^T}\tilde{\boldsymbol{\mu}}^2$	$k = 1^2$	$\tilde{\boldsymbol{\mu}}_{1^2} = \boldsymbol{A}_{1^2,1^1}\boldsymbol{\mu}_{1^1}; \ddot{\theta}_{1^2} = \tilde{\varphi}_{1^2} - \boldsymbol{\psi}^T_{1^2}\tilde{\boldsymbol{\mu}}_{1^2}$; $\boldsymbol{\mu}_{1^2} = \boldsymbol{p}_{1^2}\ddot{\theta}_{1^2} + \tilde{\boldsymbol{\mu}}_{1^2}$
		$k = 2^2$	$\tilde{\boldsymbol{\mu}}_{2^2} = \boldsymbol{A}_{2^2,1^2}\boldsymbol{\mu}_{1^2}; \ddot{\theta}_{2^2} = \tilde{\varphi}_{2^2} - \boldsymbol{\psi}^T_{2^2}\tilde{\boldsymbol{\mu}}_{2^2}$
$i = 3$	$\tilde{\boldsymbol{\mu}}^3 = \boldsymbol{A}^{3,1}\boldsymbol{\mu}^1; \ddot{\boldsymbol{q}}^3 = \tilde{\boldsymbol{\varphi}}^3 - \boldsymbol{\Psi}^{3^T}\tilde{\boldsymbol{\mu}}^3$	$k = 1^3$	$\tilde{\boldsymbol{\mu}}_{1^3} = \boldsymbol{A}_{1^3,1^1}\boldsymbol{\mu}_{1^1}; \ddot{\theta}_{1^3} = \tilde{\varphi}_{1^3} - \boldsymbol{\psi}^T_{1^3}\tilde{\boldsymbol{\mu}}_{1^3}$

Note:
1) $\boldsymbol{\varphi}(\boldsymbol{\varphi}^i$ or $\varphi_{k^i})$ is the input to the algorithm and can be calculated using inverse dynamics by substituting $\ddot{\boldsymbol{q}} = \boldsymbol{0}$.
2) $\tilde{\boldsymbol{\eta}}^i = \boldsymbol{0}$ if ith module has no child, similarly $\tilde{\boldsymbol{\eta}}_{k^i} = \boldsymbol{0}$ if kth link of the ith module has no child.
3) $\tilde{\boldsymbol{\mu}}^i = \boldsymbol{0}$ if parent of the ith module is base module (M_0), similarly $\tilde{\boldsymbol{\mu}}_{k^i} = \boldsymbol{0}$ if parent of the kth link of ith module is base link (#0).

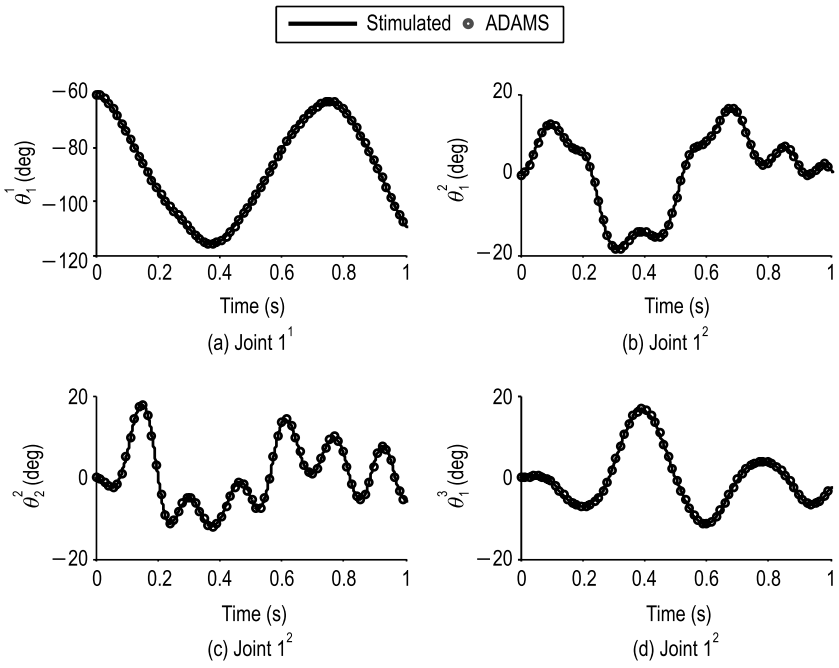

Fig. 6 Simulated joint angles for robotic gripper

The results were also compared with those obtained from ADAMS software for the same step size. As far as the computational complexity is concerned, it was found that the ReDySim based on forward dynamics algorithm took only 0.29 sec compared to 31 sec by ADAMS. Hence, the customized proposed algorithm performed better than a commercial algorithm, which was expected.

4 Computational Efficiency

Recursive inverse and forward dynamics algorithms presented in the previous section have complexity of order (n)[9] as shown in Table 4. It is interesting to note that the computational complexity does not depend on the number of modules but on the total number of DoF of a system, n. In the case of inverse dynamics, the proposed algorithm performed as good as the fastest algorithm available in the literature, whereas the forward dynamics algorithm outperforms the algorithms available in the literature. It is worth noting that work in[11] was limited to a serial chain, on the other hand, here, it is generalize to a multibody system consisting of several kinematics modules. Moreover, the algorithms have inter-modular recursion, which inside has intra-modular recursion. This leads to an elegant method for solving multibody systems. Furthermore, several spatial transformations have been optimized further in order to improve the computational complexity[9].

It may be seen from Table 4 that for large n significant saving can be obtained. For that, highly flexible rope or hose[15] was simulated in which the system was considered to have many small rigid links. The rope of 2.5 m length was considered to have 50 cylindrical elements. Each element is of 50 mm length,

Table 4 Comparison of the computational count

	Computational count
Inverse dynamics	$(94n-81)M(82n-75)A$ (Proposed) $(93n-69)M(81n-65)A$[13]
Forward dynamics	$(135n-116)M(131n-123)A$ (Proposed) $(173n-128)M(150n-133)A$[11] $(199n-198)M(174n-173)A$[2] $\left(\frac{1}{6}n^3+10\frac{3}{2}n^3+40\frac{1}{3}n-51\right)$ $M\left(\frac{1}{6}n^3+7n^2+50\frac{5}{6}n-51\right)A$[14]

M: Multiplications/Divisions; A: Additions/Subtractions

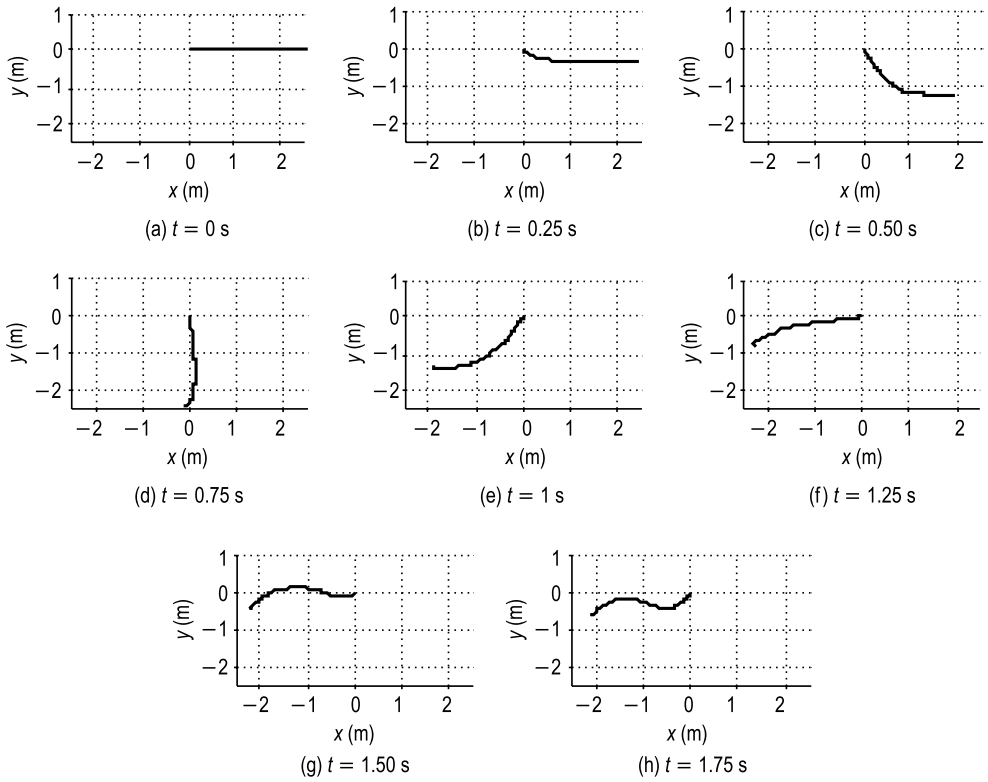

Fig. 7 Simulation of 2.5 m rope

5 mm radius, and 4 grams mass. Moreover, each link is connected with its parent link by a universal joint. As a result system has 100 DOF. Simulation under free-fall is performed for 1.75 seconds using ReDySim and took 3.44 sec. The configurations of the rope are shown in Fig. 7.

Comparison of the proposed algorithm with those available in the literature for the chain with the large DOF is also shown in Fig. 8. The comparison showed that the proposed algorithm performed much better than the algorithms available in the literature. Note that if the algorithm by Featherstone was used for the rope system CPU time of 4.54 sec would have been required in the same computer platform, i.e., E8400@3 GHz computing system.

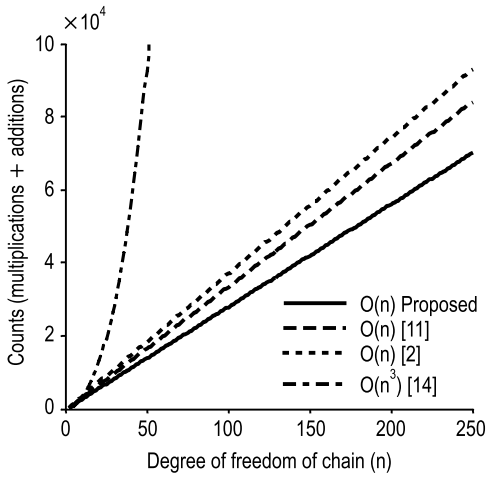

Fig. 3 Comparison of counts for high DOF chains

5 Conclusions

In this work recursive dynamics algorithms for open-loop multibody systems have been proposed using the concepts of kinematic modules and decoupled natural orthogonal complement (DeNOC) matrices. However, the methodology is easily adaptable to closed-loop systems where the loops are cut open and the cut joints are replaced with the appropriate constraint forces. The resulting system then becomes an open-loop system where the constraint forces need to be solved with other unknown. The proposed recursive algorithms were shown using the 4-DOF gripper and 100-DOF rope system. It is shown that the latter system, i.e., the system with large number of DOF, benefits more compared to any other similar algorithms available in the literature. Recursive Dynamics Simulator (ReDySim), developed based on the proposed algorithms, can be freely downloaded from www.redysim.co.nr where many demonstrative examples including those reported in this paper are made available.

Acknowledgment

The authors would like to thank Dr. Sandipan Bandyopadhyay, Assistant Professor in the Department of Engineering Design at IIT Madras, for introducing the problem of multibody system with large Degrees-of-Freedom like the rope system to the authors.

References

[1] Saha, S. K., Dynamics of serial multibody systems using the decoupled natural orthogonal complement matrices, *ASME J. of Applied Mechanics*, Vol. 66, pp. 986–996, 1999.

[2] Featherstone, R., The calculation of robotic dynamics using articulated body inertias, *International J. of Robotics Research*, Vol. 2, pp. 13–30, 1983.

[3] Bae, D. S. and Haug, E. J., A recursive formulation for constrained mechanical system dynamics: Part I, Open Loop Systems, *Mechanics of Structure and Machine*, Vol. 15, pp. 359–382, 1987.

[4] Rodriguez, G., Jain, A. and Kreutz-Delgado, K., A spatial operator algebra for manipulator modeling and control, *International J. of Robotics Research*, Vol. 10(4), pp. 371–381, 1991.

[5] Rosenthal, D. E., An order *n* formulation for Robotic systems, *Journal of Astronautical Sciences*, Vol. 38(4), pp. 511–529, 1990.

[6] Angeles, J. and Ma, O., Dynamic simulation of n-axis serial Robotic manipulators using a natural orthogonal complement, *International J. of Robotics Research*, Vol. 7(5), pp. 32–47, 1988.
[7] Chaudhary, H. and Saha, S. K., Dynamics and balancing of multibody systems, Springer, Berlin, 2009.
[8] Automated Dynamic Analysis of Mechanical System (ADAMS), 2004, Version 2005.0.0, MSC. Software.
[9] Shah S. V., Saha S. K. and Dutt J. K., Dynamics of Tree-type Robotic Systems, Intelligent Systems, Control and Automation: Science and Engineering Bookseries, Springer, Netherlands, 2013.
[10] Matlab, Version 7.4 Release 2007a, MathWorks, Inc, 2007.
[11] Mohan, A. and Saha, S. K., A recursive, numerically stable, and efficient algorithm for serial Robots, *Multibody System Dynamics*, Vol. 17(4), pp. 291–319, 2007.
[12] Saha, S. K., A decomposition of the manipulator inertia matrix, *IEEE Trans. on Robotics and Automation*, Vol. 13(2), pp. 301–304, 1997.
[13] Balafoutis, C. A. and Patel, R. V., Dynamic analysis of Robot manipulators: A cartesian tensor approach, Kluwer Academic Publishers, Boston, 1991.
[14] Lilly, K.W. and Orin, D. E., Alternate formulations for the manipulator inertia matrix, *Int. J. of Robotics Research*, Vol. 10(1), 64–74, 1991.
[15] Fritzkowski, P. and Kaminski, H., Dynamics of a rope modeled as a multi-body system with elastic joints, *Computational Mechanics*, Vol. 46, 901–909, 2010.