Recursive Dynamics Algorithms for Serial, Parallel, and Closed-chain Multibody Systems

by
Subir Kumar Saha
Department of Mechanical Engineering, IIT Delhi
Hauz Khas, New Delhi 110 016
saha@mech.iitd.ac.in

Abstract

In this paper, it is shown how to obtain recursive dynamics algorithms for multibody systems with serial, parallel, and closed-loop chains using the concept of Decoupled Natural Orthogonal Complement (DeNOC) matrices. The DeNOC is the product of two block matrices to yield the Natural Orthogonal Complement (NOC), which is a velocity transformation matrix orthogonal to the kinematic constraint matrix of the system at hand. Note that one of the two DeNOC matrices is a lower block triangular and the other one is a block diagonal. This representation allows one to compute the inverse and forward dynamics algorithms of the constrained multibody systems recursively, which was not possible with the original representation of the NOC. As a result, the computational complexities of the algorithms are reduced in many instances, particularly, in forward dynamics with large number of bodies in a system, e.g., space robots, parallel robots, vehicle systems, etc. Moreover, many physical interpretations are available, for example, articulated body inertia, etc., which can be exploited for debugging of a program and architecture design. Illustrations with several multibody systems, e.g., two six-degrees-of-freedom serial manipulators, a parallel manipulator, a carpet scrapping machine with general closed-chain, are presented.

1 Introduction

The conventional approach to obtain the dynamic model, i.e., equations of motion, of a multibody system, consisting of serial, tree, or closed kinematic chains with rigid and flexible bodies, is to use either Newton-Euler (NE) or Euler-Lagrange (EL) formulations. While the NE equations of motion are obtained from the free-body diagrams, the EL equations resulted from the kinetic and potential energies of the system at hand. The former is not suitable for motion simulation, as it finds the internal moments and forces that do not affect the motion of the system. Alternatively, EL equations give an independent set of equations of motion that is good for motion simulation, however, requires complex calculations for the partial derivatives. With the advent of digital computations, it was possible to introduce the computer-oriented methods to reduce the dimension of the unconstrained dynamical equations of motion, namely, Newton-Euler equations, by eliminating the constraint forces. Such methods are reported in Huston and Passerello (1974), Wehage and Haug (1982), Kamman and Huston (1984), Angeles and Lee (1988), Saha and Angeles (1991), Saha (1997; 1999a-b; 2003), and others.

Let us focus on the methodologies proposed by Angeles and Lee (1988) and Saha (1997), which are the basis for the development of recursive dynamics algorithms proposed in this paper. Whereas the Natural Orthogonal Complement (NOC) concept (Angeles and Lee, 1988) was used to reduce a set of NE unconstrained equations of motion to a reduced-dimension independent EL equations of motion, the Decoupled NOC (DeNOC) matrices allowed one to obtain the elements of the associated matrices and vectors in analytical form, thereby, leading to recursive order (n), i.e., $O(n)$---n being the number of bodies in the multibody system---dynamics algorithms. Note that the NOC is a velocity transformation matrix which relates the joint rates of the system at hand to the angular and translational velocities referred to as the *twists* of the constrained bodies. The matrix is also the *orthogonal complement* of the

velocity constraints of the multibody system under study, which is defined as the matrix whose columns span the null-space of the matrix of the velocity constraints. Hence, the pre-multiplication of the transpose of the NOC with the unconstrained dynamic equations of motion eliminates the joint constraint moments and forces referred to as the constraint *wrenches*. The said orthogonal complement is not unique. In some approaches, an orthogonal complement is found with numerical schemes which are computationally intensive, requiring, for example, singular-value decomposition or eigenvalue computations (Wehage and Haug, 1982; Kamman and Huston, 1984; Stejskal and Valasek, 1996). In the NOC approach, the velocity constraint matrix is written in a linear homogeneous form, and the complement is obtained naturally from the velocity constraint expressions without any complex computations. Therefore, the term *natural* is used. The NOC was later extended to express it as a product of two matrices, one of which is lower block triangular and the other one is block diagonal (Saha, 1997; 1999a-b; 2003; Saha and Schiehlen, 2001; Mohan and Saha, 2007; Chaudhary and Saha, 2007). Hence, the adjective *decoupled* is added to the new expression of the NOC, i.e., DeNOC. The dynamic modeling based on the DeNOC matrices has the following features:

1. Each element of the generalized inertia matrix, and other matrices and vectors, can be expressed in an invariant form (Saha, 1999a-b).
2. Due to the availability of the explicit expressions for the associated matrices and equations, immediate computer implementation of the algorithm, e.g., inverse or forward dynamics, is not required.
3. The invariant forms in the inverse and forward dynamics algorithms exhibit simplifying features, which contributes to a lower computational complexity (Saha, 1999b; 2003) and lower numerical errors (Mohan and Saha, 2007).
4. An inverse dynamics algorithm, where the motions of a constrained system are given to find the joint actuator torques or forces, is obtained recursively, similar to the recursive Newton-Euler inverse dynamics algorithm of a serial robotic system (Craig, 1986)
5. Since the expressions for the matrix elements are available, the inertia matrix can be decomposed in an invariant form. The decomposition is required in forward dynamics and simulation. While forward dynamics implies the algebraic solution of the joint accelerations from the dynamic equations of motion, simulation pertains to the numerical integration of the joint accelerations to obtain the joint velocities and angles. The proposed decomposition shows the recursive nature of the forward dynamics algorithm (Saha, 1997), which is not possible with the numerical LU or LL^T decompositions (Stewart, 1973). Hence, a uniform development of the recursive inverse and forward dynamics algorithms for the constrained multibody systems is possible (Saha, 1999a-b; Saha and Schiehlen, 2001).
6. The decomposition also allows one to explicitly invert the inertia matrix of a serial-chain rigid multibody system, e.g., a robot manipulator (Saha, 1999b), which not only provides the recursive forward dynamics algorithm as in item 5 above, but also facilitates a deeper understanding of the dynamics involved in the constrained multibody system.
7. Even though the uniform recursive algorithms based on the DeNOC matrices were originally proposed for serial-chain open-loop rigid-body systems, the concept has been applied to parallel (Saha and Schiehlen, 2001; Khan et al., 2005) and general closed-chain systems (Chaudhary and Saha, 2007) as well.
8. It was shown in Mohan and Saha (2007) that the DeNOC concept can even be applied to those multibody systems which have structurally flexible bodies.
9. The expressions for the elements of the generalized inertia matrix arising out of the dynamic equations of motion before or after the decomposition allow for many physical interpretations, for example, composite mass matrix, articulated body inertia, etc.

10. The physical interpretations, as mentioned in item 9 above, can be exploited for the architecture selection of a constrained system, e.g., in Bhangale et al. (2004), and Saha et al. (2006).

While almost all proposed inverse dynamics algorithms for serial-chain mechanical systems, as shown in Fig. 1, are recursive and $O(n)$, e.g., Hollerbach (1980), Luh, Walker and Paul (1980), Kane and Levinson (1983), Angeles et. al. (1989), Sciavicco and Siciliano (1996), most of the traditional forward dynamics algorithms (Walker and Orin, 1982; Kane and Levinson, 1983; Angeles and Ma, 1988) are of $O(n^3)$, except one of the four methods reported in Walker and Orin (1982) that has $O(n^2)$ complexity. Note here that, in inverse dynamics, the explicit derivation of the equations of motion in terms of the generalized coordinates is not required. For example, in Sciavicco and Siciliano (1996), the joint torques are calculated from the trajectory of the end-effector. Alternatively, explicit expressions for the matrices, particularly, the generalized inertia matrix associated with the equations of motion in terms of the generalized coordinates must be evaluated for forward dynamics, e.g., in Angeles and Ma (1988) and others, in which the joint accelerations are solved from the equations of motion. Conventionally, a numerical approach is taken to solve for the joint accelerations. In Angeles and Ma (1988), first, the elements of the inertia matrix are evaluated. Then, the numerical decomposition of the matrix, namely, Cholesky decomposition (Stewart, 1973), is performed, and finally, the joint accelerations are solved by backward and forward substitutions. Since the complexity of Cholesky decomposition is of $O(n^3)$ (Stewart, 1973), the forward dynamics algorithm also requires $O(n^3)$ computations. There is, however, an alternative approach to obtain a forward dynamics algorithm recursively whose computational complexity is of order n, i.e., $O(n)$, e.g., Armstrong (1979), Featherstone (1983), Rodriguez (1987), Schiehlen (1991), Rodriguez and Kreutz-Delgado (1992), and Saha (1997). There is also a parallel $O(\log N)$ algorithm proposed in Fijany et al. (1995) which requires a special computer hardware, namely, a parallel architecture. Because of the special hardware requirements, parallel algorithms will not be discussed any further. It is interesting to note that, compared to the $O(n^3)$ schemes, e.g., Angeles and Ma (1988), all the $O(n)$ schemes are computational efficient when $n \geq 10$, as evident from Table 2. The $O(n)$ algorithms, however, calculate the joint accelerations that are smoother (Ascher et al., 1997; Mohan and Saha, 2007). As a result, the numerical integration is faster and, hence, the total time in simulation may be less even for $n < 10$, as compared to the $O(n^3)$ schemes.

The paper is organized as follows: Section 2 presents the modeling of serial-chain multibody systems, which forms the basis for other recursive algorithms, mainly, for parallel-type and general closed-chain systems. This is followed by the modeling of parallel multibody systems in Section 3, whereas recursive modeling of closed-chain systems is presented in Section 4. Finally, the conclusions are drawn in Section 5.

2 Modeling of Serial-chain Systems

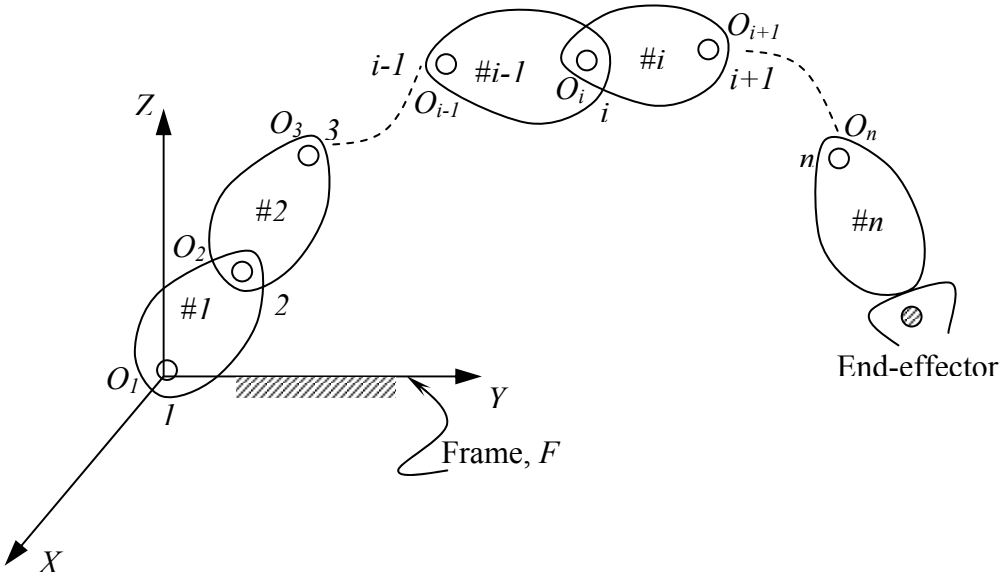
In this section, dynamic equations of motion of an n-link n-degree-of-freedom (DOF) serial-chain multibody system, e.g., a robot manipulator shown in Fig. 1(a), are derived. The methodology is based on the concept of the Decoupled Natural Orthogonal Compliment (DeNOC) matrices introduced in Saha (1997).

2.1 Uncoupled Newton-Euler equations

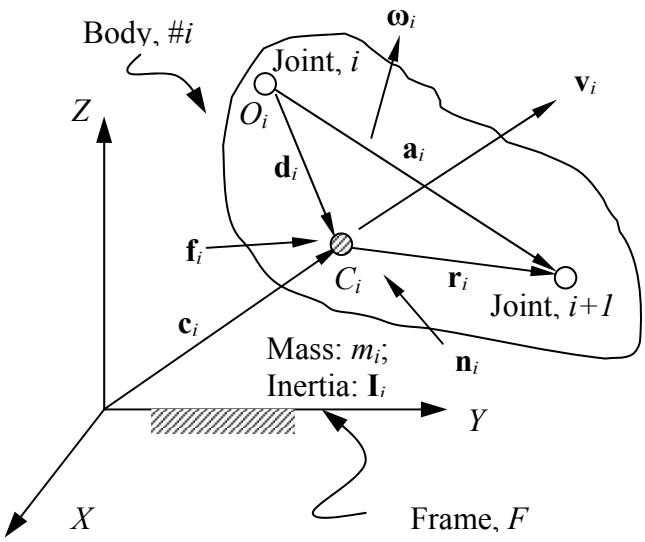
For a serial-chain system shown in Fig. 1(a), if m_i is the mass of the i^{th} link and $\mathbf{I}_i$ denotes its 3×3 inertia tensor about the mass center, C_i, as indicated in Fig. 1(b), the Newton-Euler equations of motion for the i^{th} link can be derived from its free-body diagram as

Euler's equation: $\mathbf{I}_i^c \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \mathbf{I}_i^c \boldsymbol{\omega}_i = \mathbf{n}_i^c$ (1a)

Newton's equation: $m_i \ddot{\mathbf{c}}_i = \mathbf{f}_i^c$ (1b)



(a) Serial chain



(b) The *i*th free-body

Figure 1 A serial-chain multibody system

where $\boldsymbol{\omega}_i$ and $\dot{\boldsymbol{\omega}}_i$ are the 3-dimensional vectors of angular velocity and angular acceleration of the *i*th body, respectively, whereas $\ddot{\mathbf{c}}_i$ is the 3-dimensional vector of linear acceleration or simply acceleration of the mass center, C_i. Moreover, $\mathbf{n}_i^c$ and $\mathbf{f}_i^c$ are the 3-dimensional vectors of the resultant moment about C_i and resultant force at C_i, respectively. Note that, one can also represent the equations of motion, eqs. (1a-b), with respect to point O_i where the *i*th body is coupled to its previous body, namely, (*i*-1)st

one, Fig. 1(a). To express the NE equations with respect to the point, O_i, the relations between its velocity, $\mathbf{v}_i$, and acceleration, $\dot{\mathbf{v}}_i$, with $\dot{\mathbf{c}}_i$ and $\ddot{\mathbf{c}}_i$, respectively, are first obtained as

$$\dot{\mathbf{c}}_i = \mathbf{v}_i + \boldsymbol{\omega}_i \times \mathbf{d}_i \tag{2a}$$

$$\ddot{\mathbf{c}}_i = \dot{\mathbf{v}}_i + \dot{\boldsymbol{\omega}}_i \times \mathbf{d}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{d}_i) \tag{2b}$$

Similarly, the relationships between the moment and force with respect to O_i, namely, $\mathbf{n}_i$ and $\mathbf{f}_i$, respectively, and $\mathbf{n}_i^c$ and $\mathbf{f}_i^c$ of eqs. (1a-b) are given by

$$\mathbf{n}_i = \mathbf{n}_i^c + \mathbf{d}_i \times \mathbf{f}_i^c \qquad \text{and} \qquad \mathbf{f}_i = \mathbf{f}_i^c \tag{3}$$

Upon substitution of eqs. (2-3) into eq. (1), yields

$$\mathbf{I}_i \dot{\boldsymbol{\omega}}_i + m_i \mathbf{d}_i \times \dot{\mathbf{v}}_i + \boldsymbol{\omega}_i \times \mathbf{I}_i \boldsymbol{\omega}_i = \mathbf{n}_i \tag{4a}$$

$$m_i[\dot{\mathbf{v}}_i + \dot{\boldsymbol{\omega}}_i \times \mathbf{d}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{d}_i)] = \mathbf{f}_i \tag{4b}$$

where from the parallel axis theorem, $\mathbf{I}_i = \mathbf{I}_i^c - m_i \mathbf{d}_i \times (\mathbf{d}_i \times \mathbf{1})$ —$\mathbf{I}_i$ being the 3×3 inertia tensor about O_i. Equations (4a-b) are now written in a compact form as (Saha and Schiehlen, 2001):

$$\mathbf{M}_i \dot{\mathbf{t}}_i + \mathbf{W}_i \mathbf{M}_i \mathbf{E}_i \mathbf{t}_i = \mathbf{w}_i \tag{5}$$

where the 6×6 matrices of the extended mass, $\mathbf{M}_i$, and of the extended angular velocity, $\mathbf{W}_i$, and the 6-dimensional vector of wrench of resultant external moments and forces, $\mathbf{w}_i$, are defined as

$$\mathbf{M}_i \equiv \begin{bmatrix} \mathbf{I}_i & m_i(\mathbf{d}_i \times \mathbf{1}) \\ -m_i(\mathbf{d}_i \times \mathbf{1}) & m_i \mathbf{1} \end{bmatrix}; \quad \mathbf{W}_i \equiv \begin{bmatrix} \boldsymbol{\omega}_i \times \mathbf{1} & \mathbf{O} \\ \mathbf{O} & \boldsymbol{\omega}_i \times \mathbf{1} \end{bmatrix}; \text{ and } \mathbf{w}_i \equiv \begin{bmatrix} \mathbf{n}_i \\ \mathbf{f}_i \end{bmatrix} \tag{6}$$

in which $\mathbf{d}_i \times \mathbf{1}$ and $\boldsymbol{\omega}_i \times \mathbf{1}$ are the 3×3 cross-product tensors associated with the vectors, $\mathbf{d}_i$ and $\boldsymbol{\omega}_i$,, respectively, such that $(\mathbf{d}_i \times \mathbf{1})\mathbf{x} \equiv \mathbf{d}_i \times \mathbf{x}$ and $(\boldsymbol{\omega}_i \times \mathbf{1})\mathbf{x} \equiv \boldsymbol{\omega}_i \times \mathbf{x}$, for any three-dimensional Cartesian vector, $\mathbf{x}$. Whereas vector $\mathbf{d}_i$ is shown in Fig. 1(b), $\boldsymbol{\omega}_i$ is the angular velocity of the ith body. Moreover, $\mathbf{1}$ and $\mathbf{O}$ are the 3×3 identity and zero matrices, respectively. Furthermore, the 6-dimensional vectors, namely, twist $\mathbf{t}_i$, and wrench $\mathbf{w}_i$, are given by

$$\mathbf{t}_i \equiv \begin{bmatrix} \boldsymbol{\omega}_i \\ \mathbf{v}_i \end{bmatrix} \quad \text{and} \quad \mathbf{w}_i \equiv \begin{bmatrix} \mathbf{n}_i \\ \mathbf{f}_i \end{bmatrix} \tag{7}$$

In eq. (5), vector $\dot{\mathbf{t}}_i$ is the time derivative of the twist vector, $\mathbf{t}_i$ of eq. (7). Also, $\mathbf{M}_i$ of eq. (6) is the mass matrix which compactly embodies the mass and inertia properties of the ith body about O_i. The 6×6 matrix, $\mathbf{E}_i$, is the coupling matrix and represented as

$$\mathbf{E}_i \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} \end{bmatrix}$$

Writing eq. (5), for the system of n moving bodies, i.e., for $i = 1, \ldots, n$, the $6n$ scalar equations of motion are then expressed as

$$\mathbf{M}\dot{\mathbf{t}} + \mathbf{WME}\mathbf{t} = \mathbf{w} \tag{8}$$

where the $6n \times 6n$ matrices, **M, W**, and **E**, are the generalized mass, velocity, and coupling matrices, respectively, namely,

$$\mathbf{M} \equiv diag[\mathbf{M}_1 \quad \cdots \quad \mathbf{M}_n]; \; \mathbf{W} \equiv diag[\mathbf{W}_1 \quad \cdots \quad \mathbf{W}_n]; \text{ and } \mathbf{E} \equiv diag[\mathbf{E}_1 \quad \cdots \quad \mathbf{E}_n] \tag{9}$$

Also, the $6n$-dimensional vectors of the generalized twist, twist-rate, and wrench, $\mathbf{t}$, $\dot{\mathbf{t}}$, and $\mathbf{w}$, respectively, are

$$\mathbf{t} \equiv [\mathbf{t}_1^T \quad \cdots \quad \mathbf{t}_n^T]^T, \dot{\mathbf{t}} \equiv [\dot{\mathbf{t}}_1^T \quad \cdots \quad \dot{\mathbf{t}}_n^T]^T \text{ and } \mathbf{w} \equiv [\mathbf{w}_1^T \quad \cdots \quad \mathbf{w}_n^T]^T \tag{10}$$

Equation (8) represents the $6n$ uncoupled scalar NE equations of motion for the n bodies in the serial-chain multibody system under study.

2.2 Kinematic constraints

Referring to Figs. 1(a) and 2, it is assumed that the bodies of the serial-chain system are coupled by kinematic pairs or joints that are either revolute or prismatic. From the rigid body motion of the two bodies, namely, #(*i*-1) and #*i*, Fig. 2, the twist of the *i*th body defined in eq. (7) can be derived from the velocities of the (*i*-1)st body, and the joint motion of the *i*th joint. If the *i*th joint is revolute, then the angular velocity of the *i*th body and its velocity at point O_i, denoted with $\boldsymbol{\omega}_i$ and $\mathbf{v}_i$, respectively, are obtained as:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \mathbf{e}_i \dot{\theta}_i \tag{11a}$$

$$\mathbf{v}_i = \mathbf{v}_{i-1} + \boldsymbol{\omega}_{i-1} \times \mathbf{a}_{i-1,i} \tag{11b}$$

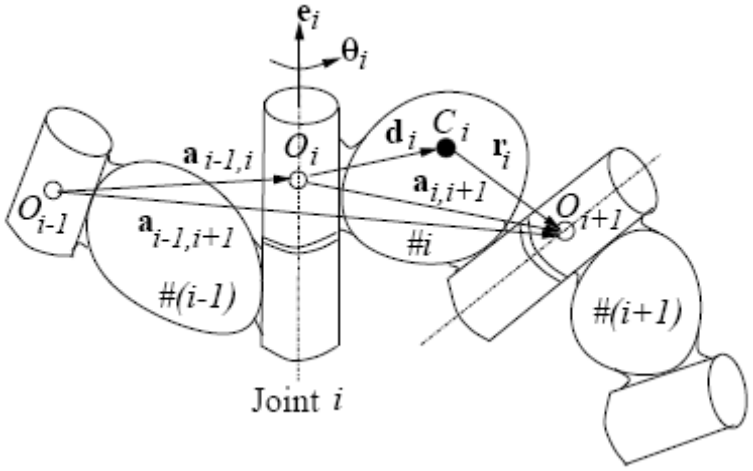


Figure 2 Three coupled bodies

where $\boldsymbol{\omega}_{i-1}$ and $\mathbf{v}_{i-1}$ are defined as the angular velocity and velocity of O_{i-1} of the (*i*-1)st body, respectively. Moreover, $\mathbf{e}_i$ is the 3-dimensional unit vector parallel to the axis of the revolute joint, as indicated in Fig. 2, and θ_i is the angular displacement of the *i*th revolute joint. Furthermore, the 3-dimensional vector, $\mathbf{a}_{i-1,i}$ denotes the position of O_i with respect to O_{i-1}, also shown in Fig. 2. The above six scalar velocity constraint equations can be written in compact form as

$$\mathbf{t}_i = \mathbf{A}_{i,i-1}\mathbf{t}_{i-1} + \mathbf{p}_i \dot{\theta}_i \tag{11c}$$

where the 6×6 matrix, $\mathbf{A}_{i,i-1}$, and the 6-dimensional vector, $\mathbf{p}_i$, are defined as

$$\mathbf{A}_{i,i-1} \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ \mathbf{a}_{i,i-1} \times \mathbf{1} & \mathbf{1} \end{bmatrix} \text{and } \mathbf{p}_i \equiv \begin{bmatrix} \mathbf{e}_i \\ \mathbf{0} \end{bmatrix} \tag{11d}$$

in which $\mathbf{a}_{i,i-1} \times \mathbf{1}$ is the cross-product tensor associated with vector $\mathbf{a}_{i,i-1}$ that is defined similar to $\boldsymbol{\omega}_i \times \mathbf{1}$ of eq. (6), i.e, $(\mathbf{a}_{i,i-1} \times \mathbf{1})\mathbf{x} = \mathbf{a}_{i,i-1} \times \mathbf{x}$, for any arbitrary 3-dimensional Cartesian vector $\mathbf{x}$. The vector, $\mathbf{a}_{i,i-1}$ is also given by, $\mathbf{a}_{i,i-1} = -\mathbf{a}_{i-1,i}$. It is interesting to note here that the matrix, $\mathbf{A}_{i,i-1}$, and the vector, $\mathbf{p}_i$, have the following interpretations:

- For two rigidly connected moving bodies, #(*i*-1) and #*i*, $\mathbf{A}_{i,i-1}$, propagates the twist of #(*i*-1) to #*i*. Hence, matrix $\mathbf{A}_{i,i-1}$ is termed as the twist propagation matrix (Saha, 1997; 1999a-b), which has the following properties:

$$\mathbf{A}_{i-1,i}\,\mathbf{A}_{i,i+1} = \mathbf{A}_{i-1,i+1}; \; \mathbf{A}_{ii} = \mathbf{1}; \text{ and } \mathbf{A}_{i-1,i}^{-1} = \mathbf{A}_{i,i-1} \text{ or } \mathbf{A}_{i,i+1}^{-1} = \mathbf{A}_{i+1,i} \tag{12a}$$

- Vector $\mathbf{p}_i$, on the other hand, takes into account the motion of the *i*th joint. Hence, $\mathbf{p}_i$ is termed as the joint-rate propagation vector, which is dependent on the type of joint. For example, the

expression of $\mathbf{p}_i$ in eq. (11d) is for a revolute joint shown in Fig. 2, whereas for a prismatic joint, vector $\mathbf{p}_i$ is given by

$$\mathbf{p}_i \equiv \begin{bmatrix} \mathbf{0} \\ \mathbf{e}_i \end{bmatrix} : \text{For a prismatic joint} \tag{12b}$$

where $\mathbf{e}_i$ is the unit vector parallel to the axis of linear motion. Correspondingly, $\dot{\theta}_i$ of eq. (11c) means the linear joint rate. Other joints are not treated here because any other joint, e.g., spherical or screw, can be treated as combination of three revolute, or revolute and prismatic pairs, respectively. For $i = 1, \ldots, n$, eq. (11c) is now put in a compact form for all the n joints as

$$\mathbf{t} = \mathbf{N}\,\dot{\boldsymbol{\theta}}, \quad \text{where } \mathbf{N} \equiv \mathbf{N}_l\,\mathbf{N}_d \tag{13a}$$

where $\mathbf{t}$ containing the twists of all the links is the $6n$-dimensional generalized twist, as defined in eq. (10). Using eq. (11c), for $i = 1, \ldots, n$, it is possible to express the generalized twist, $\mathbf{t}$, eq. (13a), as a linear transformation of the n-dimensional joint-rate vector, $\dot{\boldsymbol{\theta}}$ (Saha, 1997; 1999a-b), where the $6n \times 6n$ matrix, $\mathbf{N}_l$, the $6n \times n$ matrix, $\mathbf{N}_d$, and the n-dimensional vector, $\dot{\boldsymbol{\theta}}$, are defined as follows:

$$\mathbf{N}_l \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{A}_{21} & \mathbf{1} & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{n1} & \mathbf{A}_{n2} & \cdots & \mathbf{1} \end{bmatrix};\ \mathbf{N}_d \equiv \begin{bmatrix} \mathbf{p}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{p}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{p}_n \end{bmatrix};\ \text{and } \dot{\boldsymbol{\theta}} \equiv \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_n \end{bmatrix} \tag{13b}$$

The $6n \times n$ matrix, $\mathbf{N}$, in eq. (13a) is nothing but the original Natural Orthogonal Complement (NOC) matrix of the velocity constraints introduced by Angeles and Lee (1988), whereas its decoupled form, $\mathbf{N}_l$ and $\mathbf{N}_d$, are introduced in Saha (1997) and referred as the Decoupled NOC (DeNOC) matrices. The expressions of the DeNOC matrices allow one to develop the recursive inverse and forward dynamics algorithms (Saha, 1999a) required in control and simulation of serial-chain multibody systems, e.g., a robot manipulator, respectively. This will be shown Subsection 2.4.

2.3 Coupled equations of motion

The uncoupled NE equations of motion given by eq. (8) are re-written here as

$$\mathbf{M}\,\dot{\mathbf{t}} + \mathbf{WMEt} = \mathbf{w}^{\mathrm{E}} + \mathbf{w}^{\mathrm{C}} \tag{14}$$

where $\mathbf{w}$ of eq. (8) is substituted by, $\mathbf{w} \equiv \mathbf{w}^E + \mathbf{w}^C$ ---$\mathbf{w}^E$ and $\mathbf{w}^C$ being the external and constraint wrenches, respectively. The external wrench, $\mathbf{w}^E$, is contributed from the moments and forces due to the joint actuators, gravity, environmental effects, etc., whereas the constraint wrench, $\mathbf{w}^C$, is due to the presence of the joints that contains the reaction moments and forces at the joint interfaces. Since the constraint wrench, $\mathbf{w}^C$, does not do any useful work towards the motion of the robot links, the power consumed due to $\mathbf{w}^C$, i.e., $\boldsymbol{\Pi}^C \equiv \mathbf{t}^T\,\mathbf{w}^C$, vanishes. The sole purpose of $\mathbf{w}^C$ is to maintain the relative configuration of the bodies without getting separated. Using the expression for the generalized twist, $\mathbf{t}$, from eq. (13a), the vanishing power due to $\mathbf{w}^C$, is given by

$$\boldsymbol{\Pi}^C \equiv \mathbf{t}^T\,\mathbf{w}^C \equiv \dot{\boldsymbol{\theta}}^T\,\mathbf{N}^T\mathbf{w}^C \equiv \dot{\boldsymbol{\theta}}^{\mathrm{T}}\mathbf{N}_d^T\mathbf{N}_l^T\mathbf{w}^{\mathrm{C}} = 0 \tag{15a}$$

Note that for the n-body, n-DOF serial-chain system, the n-dimensional joint-rate vector, $\dot{\boldsymbol{\theta}}$, is independent. Hence, to satisfy eq. (15a), the following condition must hold good:

$$\mathbf{N}^T\mathbf{w}^C \equiv \mathbf{N}_d^T\mathbf{N}_l^T\mathbf{w}^{\mathrm{C}} = \mathbf{0} \tag{15b}$$

Upon multiplication of the transpose of the NOC, $\mathbf{N}^T$, to the uncoupled NE equations of motion, eq. (14), the following set of independent dynamic equations of motion results:

$$\mathbf{I}\,\ddot{\boldsymbol{\theta}} + \mathbf{h} = \boldsymbol{\tau}, \text{ where } \mathbf{h} \equiv \mathbf{C}\dot{\boldsymbol{\theta}} \tag{16a}$$

In deriving eq. (16a) the result of eq. (15b) and the time derivative of the generalized twist, $\mathbf{t}$, from eq. (13a), namely, $\dot{\mathbf{t}} = \mathbf{N}\ddot{\boldsymbol{\theta}} + \dot{\mathbf{N}}\dot{\boldsymbol{\theta}}$, are used. Moreover, the $n\times n$ matrices, $\mathbf{I}$ and $\mathbf{C}$, and the n-dimensional vectors, $\mathbf{h}$ and $\boldsymbol{\tau}$, are defined as

$\mathbf{I} \equiv \mathbf{N}^T\mathbf{M}\mathbf{N} \equiv \mathbf{N}_d^T\widetilde{\mathbf{M}}\mathbf{N}_d$: the $n\times n$ generalized inertia matrix (GIM), which is symmetric and positive definite;

$\mathbf{C} \equiv \mathbf{N}^T(\mathbf{M}\dot{\mathbf{N}} + \mathbf{W}\mathbf{M}\mathbf{E}\mathbf{N}) \equiv \mathbf{N}_d^T(\widetilde{\mathbf{M}}_{\mathrm{I}} + \widetilde{\mathbf{M}}_{\mathrm{w}} + \widetilde{\mathbf{M}}_{\mathrm{e}})\mathbf{N}_d$: the $n\times n$ matrix of convective inertia (MCI) terms;

$\mathbf{h} \equiv \mathbf{C}\dot{\boldsymbol{\theta}} = \mathbf{N}_d^T\widetilde{\mathbf{w}}'$: the n-dimensional vector of convective inertia (VCI) terms;

$\boldsymbol{\tau} \equiv \mathbf{N}^T\mathbf{w}^E \equiv \mathbf{N}_d^T\widetilde{\mathbf{w}}^{\mathrm{E}}$: the n-dimensional vector of generalized forces due to driving torques/forces, and those resulting from the gravity, environment and dissipation.

Also, the $6n\times 6n$ matrices, $\widetilde{\mathbf{M}}$, $\widetilde{\mathbf{M}}_l$, $\widetilde{\mathbf{M}}_w$, $\widetilde{\mathbf{M}}_e$, and the $6n$-dimensional vectors, $\widetilde{\mathbf{w}}^E$ and $\widetilde{\mathbf{w}}'$, are given below:

$$\widetilde{\mathbf{M}} \equiv \mathbf{N}_l^T\mathbf{M}\mathbf{N}_l;\ \widetilde{\mathbf{M}}_{\mathrm{I}} \equiv \mathbf{N}_l^T\mathbf{M}\dot{\mathbf{N}}_l,\ \widetilde{\mathbf{M}}_w \equiv \widetilde{\mathbf{M}}\mathbf{W},\ \widetilde{\mathbf{M}}_e \equiv \mathbf{N}_l^T\mathbf{W}\mathbf{M}\mathbf{E}\mathbf{N}_l; \tag{16b}$$

$$\widetilde{\mathbf{w}}' \equiv \mathbf{N}_l^T(\mathbf{M}\mathbf{t}' + \mathbf{W}\mathbf{M}\mathbf{E}\mathbf{t});\ \mathbf{t}' \equiv (\dot{\mathbf{N}}_l + \mathbf{N}_l\mathbf{W})\mathbf{N}_d\dot{\boldsymbol{\theta}};\ \text{and}\ \widetilde{\mathbf{w}}^{\mathrm{E}} \equiv \mathbf{N}_l^T\mathbf{w}^{\mathrm{E}} \tag{16c}$$

where $\dot{\mathbf{N}}_d = \mathbf{W}\mathbf{N}_{\mathrm{d}}$ is used, and the matrices, $\mathbf{N}_l$, $\mathbf{M}$, $\mathbf{W}$, $\mathbf{E}$, and the vector $\mathbf{w}^E$ are defined in the previous Subsection 2.1. Note above that $\mathbf{t}'$ is nothing but the twist-rate vector while $\ddot{\boldsymbol{\theta}} = \mathbf{0}$. It is pointed out here that eq. (16a) is nothing but the Euler-Lagrange equations of motion, which are obtained using the uncoupled Newton-Euler equations of motion and simple linear algebra concepts without performing any complex partial differentiations explicitly.

2.4 Recursive dynamics algorithms

In this section, recursive dynamics algorithms, namely, the inverse and forward dynamics of serial multibody systems are presented. Whereas inverse dynamics is defined as "given the system's geometrical and inertial parameters, along with its joint trajectories, find the joint torques and forces to follow the specified trajectory," forward dynamics referred to as the "evaluation of the joint accelerations, $\ddot{\boldsymbol{\theta}}$, from the dynamic equations of motion governed by eq. (16a) while its joint torques and forces, along with the robot's physical parameters, are known." The latter requires the algebraic solution of eq. (16a), which is linear in $\ddot{\boldsymbol{\theta}}$. Inverse and forward dynamics are generally required for the control and simulation of a system, respectively.

2.4.1 Inverse dynamics algorithm

The recursive inverse dynamics algorithm here has two recursions, namely, forward and backward. They are given below:

Forward Recursion

$$\mathbf{t}_1 = \mathbf{p}_1\dot{\theta}_1; \qquad \dot{\mathbf{t}}_1 = \mathbf{p}_1\ddot{\theta}_1 + \mathbf{W}_1\mathbf{p}_1\dot{\theta}_1$$
$$\mathbf{t}_2 = \mathbf{p}_2\dot{\theta}_2 + \mathbf{t}_1; \qquad \dot{\mathbf{t}}_2 = \mathbf{p}_2\ddot{\theta}_2 + \mathbf{W}_2\mathbf{p}_2\dot{\theta}_2 + \mathbf{A}_{21}\dot{\mathbf{t}}_1 + \dot{\mathbf{A}}_{21}\mathbf{t}_1$$
$$\vdots \qquad\qquad \vdots$$
$$\mathbf{t}_n = \mathbf{p}_n\dot{\theta}_n + \mathbf{t}_{n-1}; \qquad \dot{\mathbf{t}}_n = \mathbf{p}_n\ddot{\theta}_n + \mathbf{W}_n\mathbf{p}_n\dot{\theta}_n + \mathbf{A}_{n,n-1}\dot{\mathbf{t}}_{n-1} + \dot{\mathbf{A}}_{n,n-1}\mathbf{t}_{n-1}$$

Backward Recursion

$$\mathbf{w}_n^* = \mathbf{M}_n\dot{\mathbf{t}}_n + \mathbf{W}_n\mathbf{M}_n\mathbf{E}_n\mathbf{t}_n; \qquad \tau_n = \mathbf{p}_n^T\mathbf{w}_n^*$$
$$\mathbf{w}_{n-1}^* = \mathbf{M}_{n-1}\dot{\mathbf{t}}_{n-1} + \mathbf{W}_{n-1}\mathbf{M}_{n-1}\mathbf{E}_{n-1}\mathbf{t}_{n-1} + \mathbf{A}_{n,n-1}^T\mathbf{w}_n^*; \qquad \tau_{n-1} = \mathbf{p}_{n-1}^T\mathbf{w}_{n-1}^*$$
$$\vdots \qquad\qquad \vdots$$
$$\mathbf{w}_1^* = \mathbf{M}_1\dot{\mathbf{t}}_1 + \mathbf{W}_1\mathbf{M}_1\mathbf{E}_1\mathbf{t}_1 + \mathbf{A}_{21}^T\mathbf{w}_2^*; \qquad \tau_1 = \mathbf{p}_1^T\mathbf{w}_1^*$$

where $\mathbf{t}_i$, $\dot{\mathbf{t}}_i$ and $\mathbf{w}_i^*$ are the 6-dimensional vectors of twist, twist rate, and the inertia wrench. If gravity is present, it is taken into account by providing negative acceleration due to the gravity, denoted by **g**, to the acceleration of the first body, #1 (Kane and Levinson, 1983), i.e.,

$$\dot{\mathbf{t}}_1 = \mathbf{p}_1\ddot{\theta}_1 + \mathbf{W}_1\mathbf{p}_1\dot{\theta}_1 + \boldsymbol{\rho}, \text{ where } \boldsymbol{\rho} \equiv [\mathbf{0}, -\mathbf{g}^T]^T \tag{17}$$

Table 1 Computational complexities of recursive inverse dynamics algorithms (Saha, 1999a)

Algorithm	*Multiplications/ Divisions (M)*	*Addition/ Subtraction (A)*	*n = 6*	
Hollerbach (1980)	412n-277	320n-201	2195M	1719A
Luh et al. (1980)	150*n*-48	131*n*+48	852M	834A
Walker and Orin (1982)	137*n*-22	101*n*-11	800M	595A
RIDIM (Saha, 1999)	**120*n*-44**	**97*n*-55**	**676M**	**527A**
Khalil et al. (1986)	105*n*-92	94*n*-86	538M	478A
Angeles et al. (1989)	105*n*-109	90*n*-105	521M	435A
Balafoutis et al. (1988)	93*n*-69	81*n*-65	489M	421M

Note that an algorithm very similar to the above one was also reported in Saha (1999a), where the twist of each body is defined with respect to the mass-center, C_i, instead of point O_i of Fig. 1(b). The computational complexity of the inverse dynamics algorithm reported in Saha (199a) is given in Table 1, where the same is compared with others available in the literature. From Table 1, the benefit of the inverse dynamics algorithm based on the DeNOC concept is not so substantial. However, it will be shown next how the DeNOC concept is useful for the forward dynamics algorithm. The inverse dynamics algorithm of Saha (1999a) was implemented in a C++ program called RIDIM (Recursive Inverse Dynamics for Industrial Manipulator) that has a MS-Windows interface, as shown in Fig. 3. To run RIDIM, one needs the following inputs: For $i = 1, \ldots, n$,

1. Constant Denavit-Hartenberg (DH) parameters (Denavit and Hartenberg, 1955) of the robot under study. They are b_i, a_i, α_i, for a revolute joint, and θ_i, a_i, α_i, for a prismatic joint. The parameters, b_i, θ_i, a_i, and α_i, are referred as joint offset, joint angle, link length, and twist angle, respectively. The exact definitions and notations for the DH parameters used in RIDIM are explained in Saha (1999a).
2. Time history of the variable DH parameter, θ_i, for a revolute pair, and b_i, for a prismatic joint, and their first and second time derivatives, i.e., $\dot{\theta}_i$, $\dot{b}_i$, and $\ddot{\theta}_i$, $\ddot{b}_i$, respectively.
3. Mass of each body, m_i.

4. Vector denoting the distance of the $(i + 1)$st joint from the ith mass center, C_i, in $(i + 1)$st frame, i.e., $[\mathbf{r}_i]_{i+1}$.
5. Inertia tensor of the ith body about its mass center, C_i, in the $(i + 1)$st frame, $[\mathbf{I}_i]_{i+1}$.

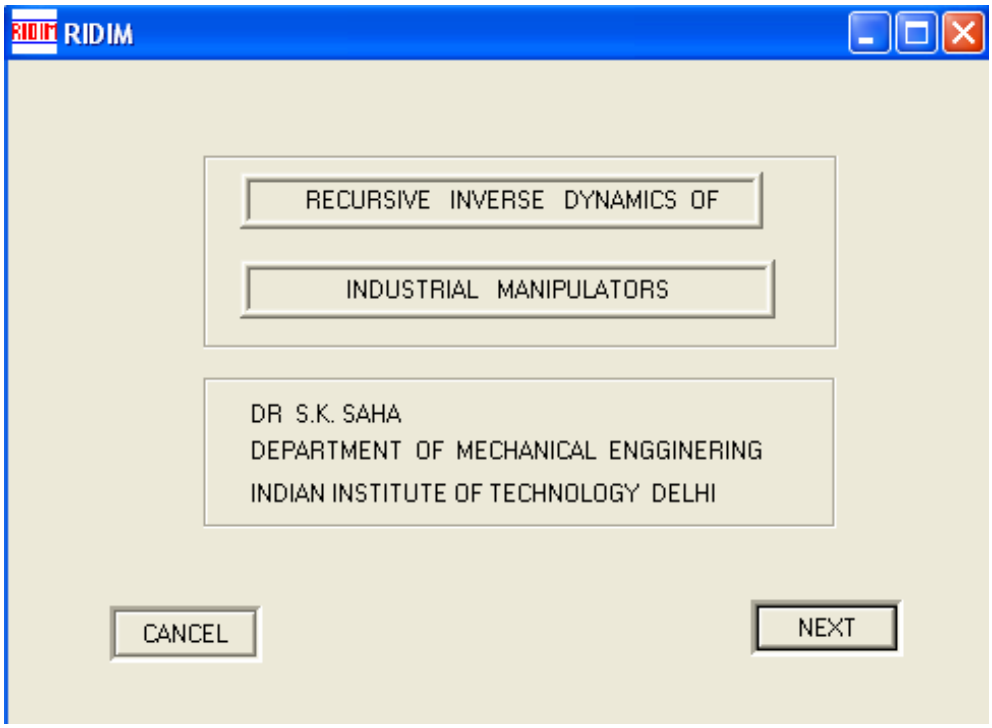


Figure 3 MS-Windows interface of RIDM

2.4.2 Forward dynamics algorithm

Conventionally, joint accelerations are solved from eq. (16a) using Cholesky decomposition of the GIM, **I**, as done by Walker and Orin (1982), Angeles (2003), and others, or in the MATLAB command of "**ϕ\I**" (Pratap, 2002), where **ϕ** represents the vector of generalized forces due to external moments and forces, gravity and Coriolis terms, etc., namely, $\boldsymbol{\varphi} \equiv \boldsymbol{\tau} - \mathbf{h}$. The above approach requires $O(n^3)$---n being the number of links in the robot---computations, and produces non-smooth joint accelerations (Ascher et al., 1997). On the contrary, the dynamic formulation based on the Decoupled Natural Orthogonal Complement (DeNOC) matrices presented here allows one to solve $\ddot{\boldsymbol{\theta}}$ from eq. (16a) recursively with $O(n)$ computations (Saha, 1999a-b; 2003; Mohan and Saha, 2007). Such recursive algorithms are known to provide smooth profiles for $\ddot{\boldsymbol{\theta}}$, as reported in Ascher et al. (1997), Mohan and Saha (2007), and others.

The algorithm is based on the $\mathbf{UDU}^T$ decomposition of the generalized inertia matrix, **I** of eq. (16a) (Saha, 1997), i.e., $\mathbf{I} = \mathbf{UDU}^T$, where **U** and **D** are the upper and diagonal matrices, respectively. Moreover, substituting for **I** and, $\boldsymbol{\varphi} \equiv \boldsymbol{\tau} - \mathbf{C}\dot{\boldsymbol{\theta}}$, in eq. (16a), one obtains

$$\mathbf{UDU}^T\ddot{\boldsymbol{\theta}} \equiv \boldsymbol{\varphi} \tag{18}$$

A three step recursive scheme is then used to calculate the joint accelerations from eq. (18), i.e.,

1. Solution for $\hat{\boldsymbol{\varphi}}$: The solution, $\hat{\boldsymbol{\varphi}} = \mathbf{U}^{-1}\boldsymbol{\varphi}$, is evaluated in terms of the scalar terms as

$$\hat{\phi}_i = \phi_i - \mathbf{p}_i^T \boldsymbol{\eta}_{i,i+1}, \text{ for } i = n, \ldots, 1 \tag{19a}$$

Note $\hat{\phi}_n \equiv \phi_n$, and the 6-dimensional vector, $\boldsymbol{\eta}_{i,i+1}$, is obtained as

$$\boldsymbol{\eta}_{i,i+1} \equiv \mathbf{A}_{i+1,i}^T \boldsymbol{\eta}_{i+1} \text{ and } \boldsymbol{\eta}_{i,i+1} \equiv \boldsymbol{\psi}_{i+1}\hat{\phi}_{i+1} + \boldsymbol{\eta}_{i+1,i+2} \tag{19b}$$

in which $\boldsymbol{\eta}_{n,n+1} = \mathbf{0}$. In order to calculate the new variable, $\boldsymbol{\psi}_{i+1}$ of eq. (19b), is the 6-dimensional vector, $\boldsymbol{\psi}_i$, is evaluated as

$$\boldsymbol{\psi}_i \equiv \frac{\mathbf{A}_{i+1,i}^T \hat{\boldsymbol{\psi}}_i}{\hat{m}_i}, \text{ where } \hat{\boldsymbol{\psi}}_i \equiv \hat{\mathbf{M}}_i \mathbf{p}_i \text{ and } \hat{m}_i \equiv \mathbf{p}_i^T \hat{\boldsymbol{\psi}}_i \tag{19c}$$

In eq. (19c), the 6×6 matrix, $\hat{\mathbf{M}}_i$, is called the "articulated body inertia" (Saha, 1997;1999a) that can be obtained recursively as

$$\hat{\mathbf{M}}_i \equiv \mathbf{M}_i + \mathbf{A}_{i+1,i}^T \overline{\mathbf{M}}_{i+1} \mathbf{A}_{i+1,i}, \text{ where } \overline{\mathbf{M}}_i \equiv \hat{\mathbf{M}}_i - \hat{\boldsymbol{\psi}}_i \boldsymbol{\psi}_i^T \tag{19d}$$

for $i = n$-1, …, *1*, and $\hat{\mathbf{M}}_n = \mathbf{M}_n$.

2. Solution for $\tilde{\boldsymbol{\varphi}}$: The solution, $\mathbf{D}\tilde{\boldsymbol{\varphi}} = \hat{\boldsymbol{\varphi}}$, involves the inverse of the diagonal matrix, **D** of eq. (18), which is simple, namely, $\mathbf{D}^{-1}$ has only nonzero diagonal elements that are the reciprocal of the corresponding diagonal elements of **D**. The scalar elements of vector $\tilde{\boldsymbol{\varphi}}$ is obtained as follows: For i = 1, …, n,

$$\tilde{\phi}_i = \hat{\varphi}_i / \hat{m}_i \tag{20}$$

where the scalar, $\hat{m}_i$, is obtained in eq.(19c).

3. Solution for $\ddot{\boldsymbol{\theta}}$: In this step, $\ddot{\boldsymbol{\theta}} = \mathbf{U}^{-T}\tilde{\boldsymbol{\varphi}}$, is calculated, for $i = 2$, …, n, as

$$\ddot{\theta}_i = \tilde{\phi}_i - \hat{\boldsymbol{\psi}}_i^T \tilde{\boldsymbol{\mu}}_{i,i-1} \tag{21a}$$

where $\ddot{\theta}_1 \equiv \tilde{\phi}_1$, and the 6-dimensional vector, $\tilde{\boldsymbol{\mu}}_{i,i-1}$, is obtained as

$$\tilde{\boldsymbol{\mu}}_{i,i-1} \equiv \mathbf{A}_{i,i-1}\tilde{\boldsymbol{\mu}}_{i-1} \text{ and } \tilde{\boldsymbol{\mu}}_{i-1} \equiv \mathbf{p}_{i-1}\ddot{\theta}_{i-1} + \tilde{\boldsymbol{\mu}}_{i-1,i-2} \tag{21b}$$

in which $\tilde{\boldsymbol{\mu}}_{10} \equiv \mathbf{0}$

The complexity of the above recursive forward dynamics algorithm is compared in Table 2 with some other algorithms. Note in Table 2 that if a rigid body in the kinematic chain is transformed to an equivalent body defined by its DH parameters the complexity of the above algorithm reduces further (Mohan and Saha, 2007). Also, compared to other existing recursive algorithms, the one proposed above based on the DeNOC concept is the best.

Table 2 Computational complexities of forward dynamics algorithms

Algorithm	Multiplication	Additions	n*	n^+
Mohan and Saha (2007)	**173*n*-128**	**150*n*-133**	10 (1602 M)	12 (1948M 1667A)
Saha (2003)	191n – 284	187n-325	10 (1626 M)	14 (2390M 2293A)
Featherstone (1983)	199n-198	174n-173	12 (2190 M)	14 (2588M 2263A)
Valasek [15]	226n-343	206n-345	13 (2595 M)	15 (3047M 2745A)
Brandle et.al [15]	250n-222	220n-198	14 (3278 M)	16 (3778M 3322A)
Walker and Orin (as implemented by Featherstone)	$\frac{1}{6}n^3+\frac{23}{2}n^2$ $+\frac{115}{3}n-47$	$\frac{1}{6}n^3+7n^2$ $+\frac{233}{6}n-46$	10 (1653 M)	12 (2357M 1716A)

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

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

For the numerical simulation, the above forward dynamics algorithm requires the following input: For i =1, …, n,

1. Input items 1 and 3-5 for the inverse dynamics algorithm presented in Subsection 3.1.
2. Initial values for the variable DH parameters, i.e., θ_i, for a revolute joint, and b_i, for a prismatic joint, and their first time derivatives, i.e., $\dot{\theta}_i$ and $\dot{b}_i$.
3. Time history of the input joint torques and forces, i.e., τ_i.
4. The vector, $\boldsymbol{\varphi} \equiv \boldsymbol{\tau} - \mathbf{C}\dot{\boldsymbol{\theta}}$, that can be obtained recursively from eq. (16a) using the inverse dynamics algorithm presented in Subsection 3.1 while $\ddot{\boldsymbol{\theta}} = \mathbf{0}$.

2.5 Numerical examples

In this section, two serial six-degree-of-freedom manipulators with PUMA and Standford arm architectures shown in Figs. 4 and 5, respectively, are considered.

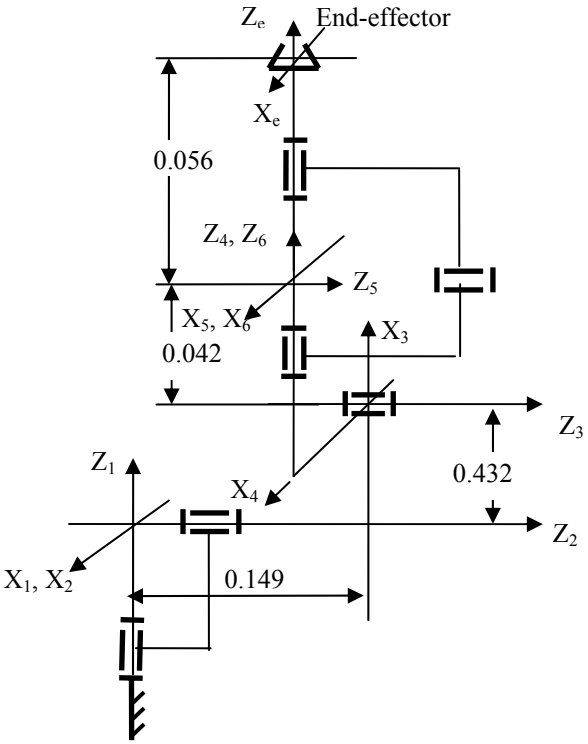


Figure 4 PUMA architecture

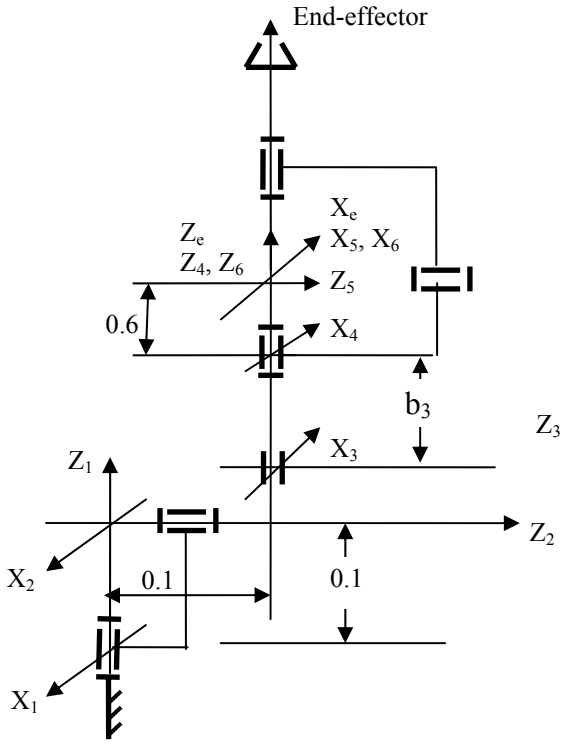


Figure 5 Stanford arm

2.5.1 PUMA

For the PUMA architecture with all joints revolute, as shown in Fig. 4, the inverse dynamics results were obtained for the DH and inertial parameters shown in Table 3. The joint angle for each joint is taken as (Angeles, 2003)

$$\theta_i = \theta_i(0) + \frac{\theta_i(T) - \theta_i(0)}{T}\left[t - \frac{T}{2\pi}\sin\left(\frac{2\pi}{T}t\right)\right] \tag{22a}$$

$$\dot{\theta}_i = \frac{\theta_i(T) - \theta_i(0)}{T}\left[1 - \cos\left(\frac{2\pi}{T}t\right)\right];\ \ddot{\theta}_i = \frac{\theta_i(T) - \theta_i(0)}{T}\left[\frac{2\pi}{T}\sin\left(\frac{2\pi}{T}t\right)\right] \tag{22b}$$

where T = 10sec, and initial joint values are $\theta_i(0) = \dot{\theta}_i(0) = \ddot{\theta}_i(0) = 0$, whereas the final joint values are $\theta_i(T) = \pi$, $\dot{\theta}_i(0) = \ddot{\theta}_i(0) = 0$, for i = 1, …, 6. The joint torques can then be calculated using RIDIM. The results are shown in Figs. 6(a-f). In Fig. 7(a-f), free-fall simulation results are shown where the PUMA manipulator is allowed to fall under gravity from its initial configuration shown in Table 3(a). For the numerical integration, an in-house developed C++ program based on Runga-Kutta $5^{th}/6^{th}$ order numerical integration method (Press et al., 1997) was used.

Table 3 The DH and inertia parameters of PUMA

(a) DH parameters

Link	Joint	a_i (m)	b_i (m)	α_i (rad)	θ_i (rad)
1	r	0	0	-π/2	JV [0]
2	r	0.432	0.149	0	JV [0]
3	r	0.02	0	-π/2	JV [0]
4	r	0	0.432	-π/2	JV [0]
5	r	0	0	-π/2	JV [0]
6	r	0	0.05	0	JV [0]

JV: Joint variable with initial values inside [and]; r: Revolute joint

(b) Mass and inertia parameters

Link	m_i	$r_{i,x}$	$r_{i,y}$	$r_{i,z}$	$I_{i,xx}$	$I_{i,xy}$	$I_{i,xz}$	$I_{i,yy}$	$I_{i,yz}$	$I_{i,zz}$
	(kg)	(m)			(kg-m^2)					
1	10.521	0	0	0.054	1.612	0	0	1.612	0	0.5091
2	15.761	0.292	0	0	0.4898	0	0	8.0783	0	8.2672
3	8.767	0.02	0	-0.197	3.3768	0	0	3.3768	0	0.3009
4	1.052	0	-0.057	0	0.181	0	0	0.1273	0	0.181
5	1.052	0	0	-0.007	0.0735	0	0	0.1273	0	0.0735
6	0.351	0	0	0.019	0.0071	0	0	0.0071	0	0.0141

2.5.2 Stanford arm

For the Stanford arm shown in Fig. 6, the DH and other parameters are shown in Table 4. Note that it differentiates from the PUMA architecture in a way that it has a prismatic pair in joint location 3. The joint trajectory for the revolute joints were taken same as PUMA, i.e., eqs. (22a-b), with the following data: T = 10sec, $\theta_i(0) = 0$, for $i \neq 2,3$, $\theta_2(0) = \pi/2$; and $\theta_i(T) = \pi/3$, for $i \neq 3$. For the third prismatic joint, the joint variable denoted by b_3 is taken same as

$$b_3 = b_3(0) + \frac{b_3(T) - b_3(0)}{T}\left[t - \frac{T}{2\pi}\sin\left(\frac{2\pi}{T}t\right)\right] \tag{23}$$

where $b_3(0) = 0$, and $b_3(T) = 0.1$m. The joint torques and force are then evaluated using RIDIM shown in Fig. 8(a-f), whereas the free-fall simulation is shown in Fig. 9(a-f). Note that the joint torques and force obtained shown in Fig. 8 match exactly with those reported in Cyril (1988) for the same arm.

Table 4 The DH and inertia parameters of Stanford arm

(a) DH parameters

Link	Joint	a_i (m)	b_i (m)	α_i (rad)	θ_i (rad)
1	r	0	0.1	$-\pi/2$	JV [0]
2	r	0	0.1	$-\pi/2$	JV [0]
3	p	0	JV [0]	0	0
4	r	0	0.6	$\pi/2$	JV [0]
5	r	0	0	$-\pi/2$	JV [0]
6	r	0	0.0	0	JV [0]

JV: Joint variable with initial values within [and]; r: Revolute joint; p: Prismatic joint

(b) Mass and inertia parameters

Link	m_i	$r_{i,x}$	$r_{i,y}$	$r_{i,z}$	$I_{i,xx}$	$I_{i,xy}$	$I_{i,xz}$	$I_{i,yy}$	$I_{i,yz}$	$I_{i,zz}$
	(kg)	(m)			(kg-m^2)					
1	9	0	-0.1	0	0.01	0	0	0.02	0	0.01
2	6	0	0	0	0.05	0	0	0.06	0	0.01
3	4	0	0	0	0.4	0	0	0.4	0	0.01
4	1	0	-0.1	0	0.001	0	0	0.001	0	0.0005
5	0.6	0	0	-0.06	0.0005	0	0	0.0005	0	0.0002
6	0.5	0	0	0.2	0.003	0	0	0.001	0	0.002

3 Modeling of Parallel Systems

Realizing the benefits of the recursive and minimum order formulations, an attempt is made here to obtain a recursive minimum order algorithm for parallel-type systems as well. Such systems are Stewart platform, hexapod and hexaslide machine tools, and like. Whereas recursive inverse and forward dynamics algorithms for serial-chain open-loop systems are available in the literature, as indicated in Tables 1 and 2, the same closed-chain systems is difficult. In this section, symmetric closed-chain multibody systems, namely, the parallel types are considered, which will be followed by general closed-chain systems in Section 4.

3.1. Existing formulations

Before proceeding to the development of unified recursive algorithms for the parallel systems, a look into the existing formulations is given in this section. Available literature on the dynamics of closed-chain systems suggest that it can be broadly classified as either recursive DAE (differential algebraic equations) or non-recursive minimum order dynamic equations. For example, the algorithms by Bae and Haug (1987b), Schiehlen (1990), Stejskal and Valasek (1996), Bae et al. (1999), and others write the equations of motion of a closed-chain system in the following manner:

- Cut the closed chains of the system to open into a serial or tree structure;
- Introduce Lagrange multipliers to substitute for the cut joints;
- Use a recursive scheme for the open chain system, e.g., (Featherstone, 1983; Bae and Haug, 1987a), etc., to obtain a recursive algorithm.

The above formulation does provide a recursive inverse and forward dynamics algorithms but form a DAE system which is known for its difficulty in numerical integration due to numerical instability and

constraint violation problems. On the contrary, the formulations by Brauchli and Weber (1991), Blajer et al. (1993), and Ghorbel (1994) follow the steps given below:

- Open the closed chains to make it an open chain or tree structure, as above;
- Write the equations of motion of the new system recursively in terms of a larger number of generalized coordinates (Featherstone, 1983; Bae and Haug, 1987a);

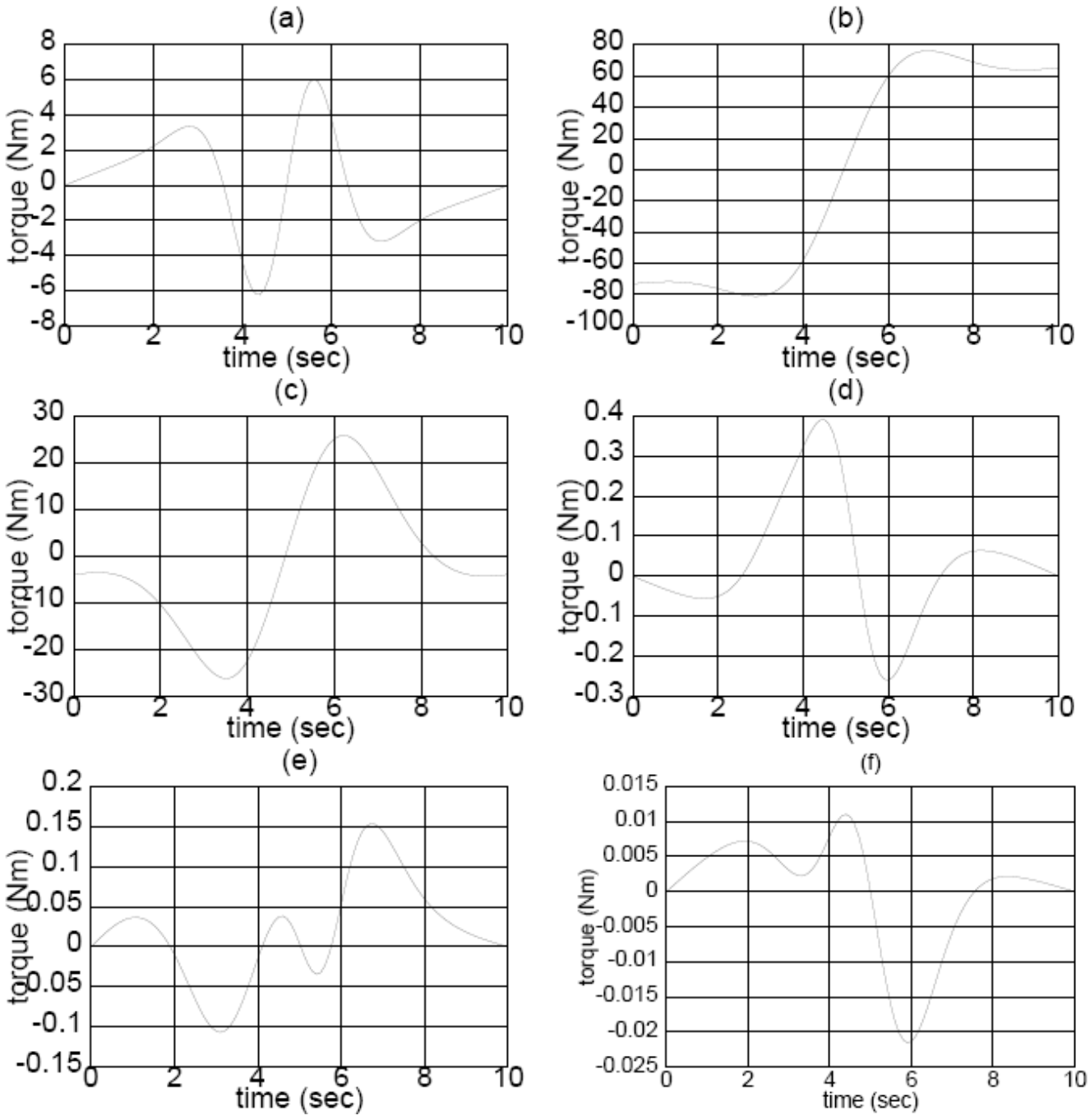


Figure 6 Torques for PUMA architecture: (a) Joint 1; (b) Joint 2; (c) Joint 3; (d) Joint 4; (e) Joint 5; (f) Joint 6

- Since the above generalized coordinates are not independent for the original closed-chain system, the loop closure conditions are introduced. This enables one to write the larger set of generalized coordinates in terms of an independent set. This step is, normally, done using a numerical method, e.g., using LU decomposition (Stewart, 1973). As a result the algorithm remains no longer recursive;
- Transform the equations of motion of the open/tree system to the closed system, which is also not recurisve.

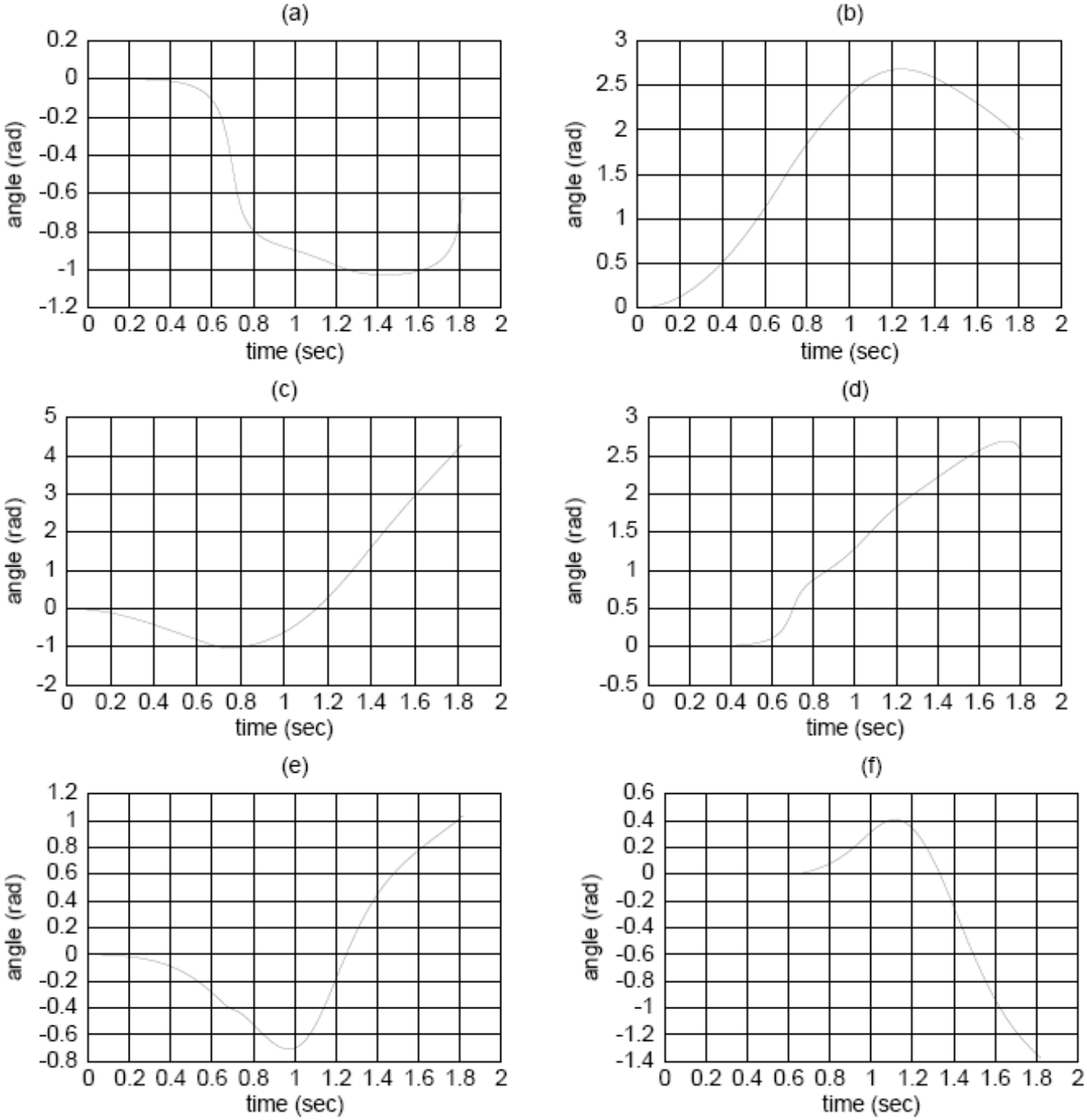


Figure 7 Simulated joint angles for PUMA: (a) Joint 1; (b) Joint 2; (c) Joint 3; (d) Joint 4; (e) Joint 5; (f) Joint 6

Due to the last two steps above, the advantages of the recursive algorithm in the second step are lost. As a remedy, a formulation was proposed in Saha and Schiehlen (2001) where it was shown how to solve for the dependent joint rates recursively form the independent or actuated joint rates leading to recursive ODE (order or ordinary differential equation) formulation. Note that an ODE formulation is generally preferred over the DAE formulation due to its well-known advantages (Shabana 1994). This is shown next.

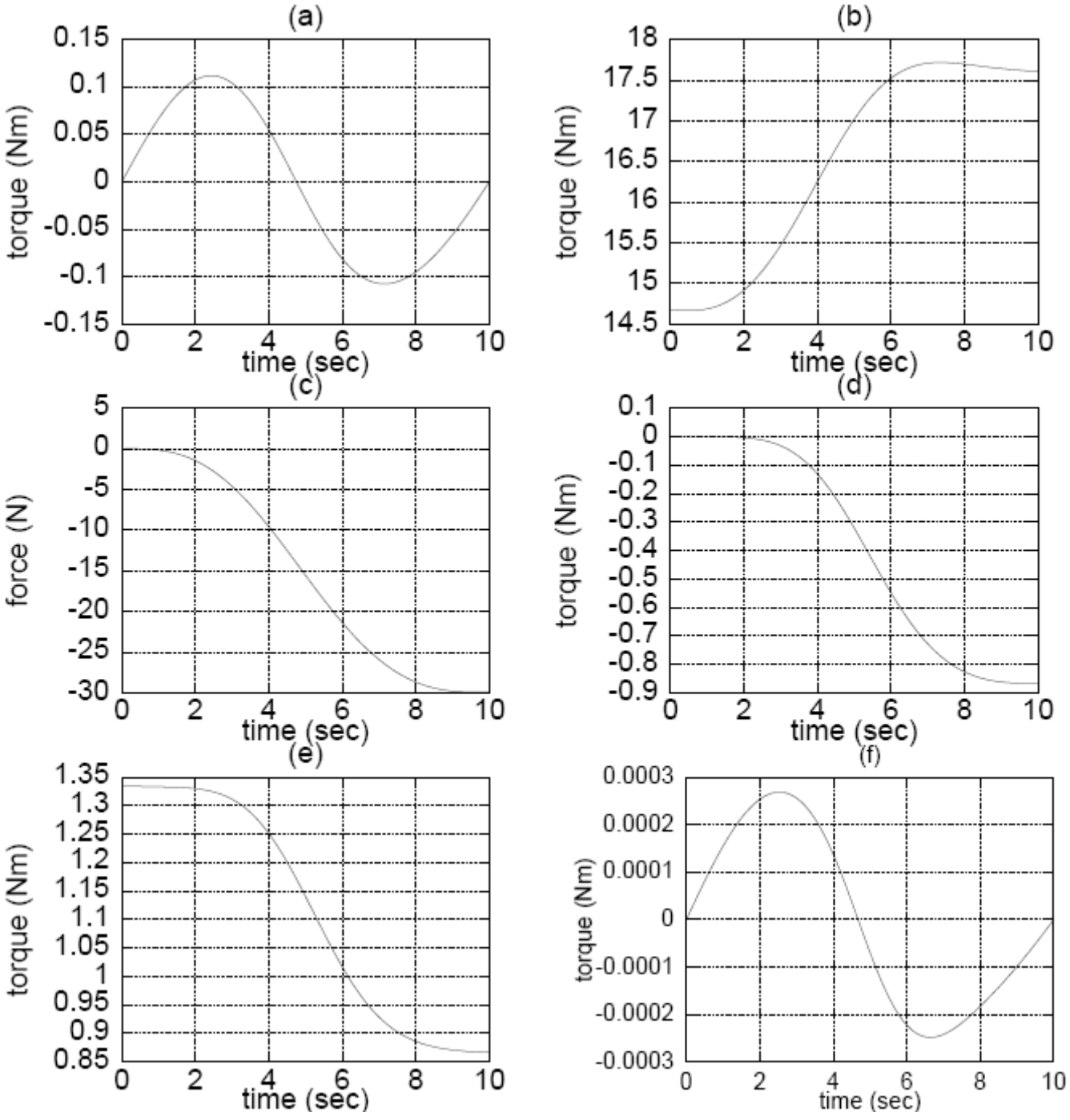


Figure 8 Torques and force for Stanford arm: (a) Joint 1; (b) Joint 2; (c) Joint 3; (d) Joint 4; (e) Joint 5; (f) Joint 6

3.2 Recursive formulation for unactuated joints

In this Subsection, the focus is closed-chain parallel type systems shown in Fig. 10(a). In order to derive the recursive kinematic relations, the following assumptions are made:

- Each leg has only one actuator at the base joint denoted as 1;

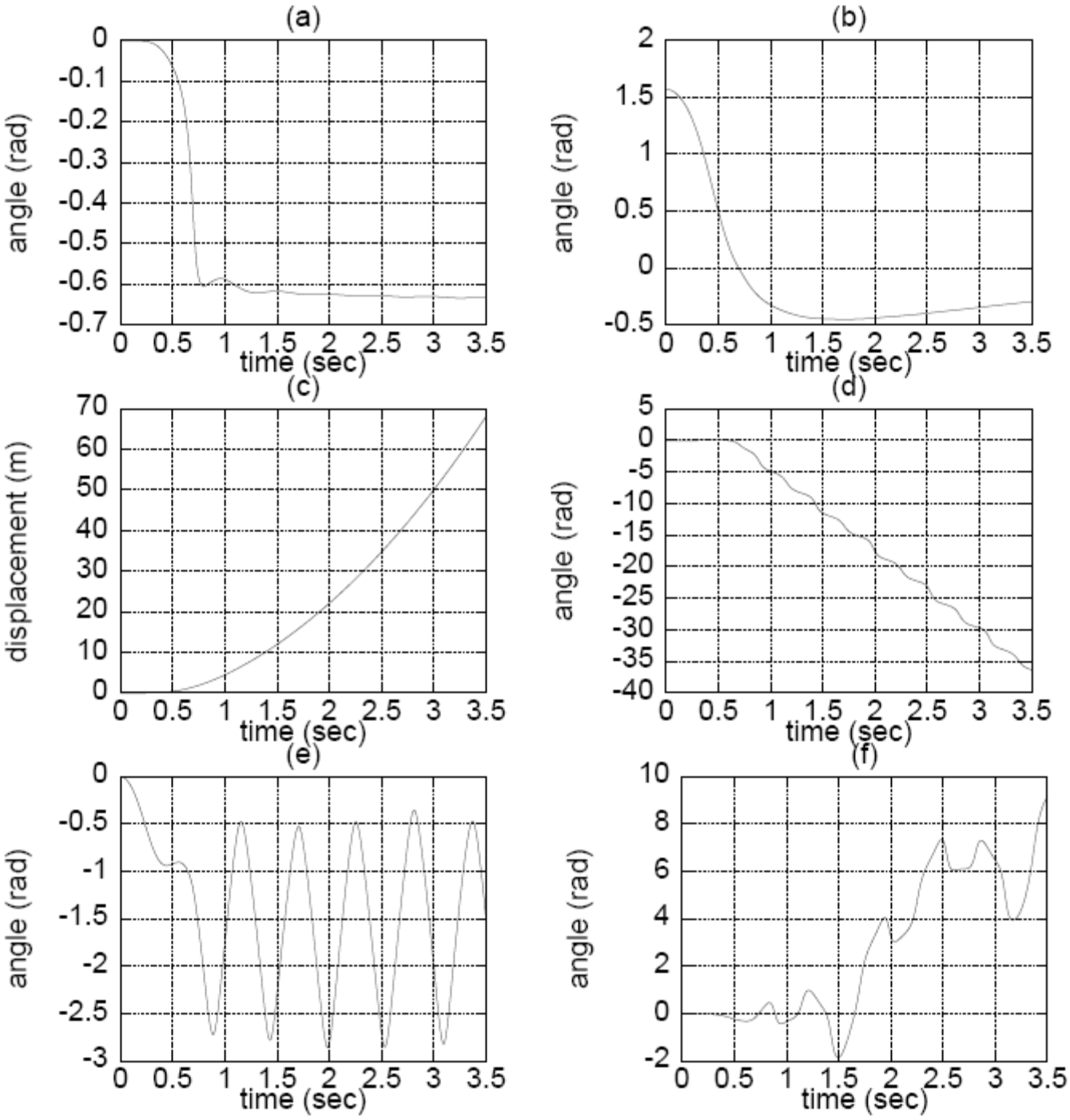
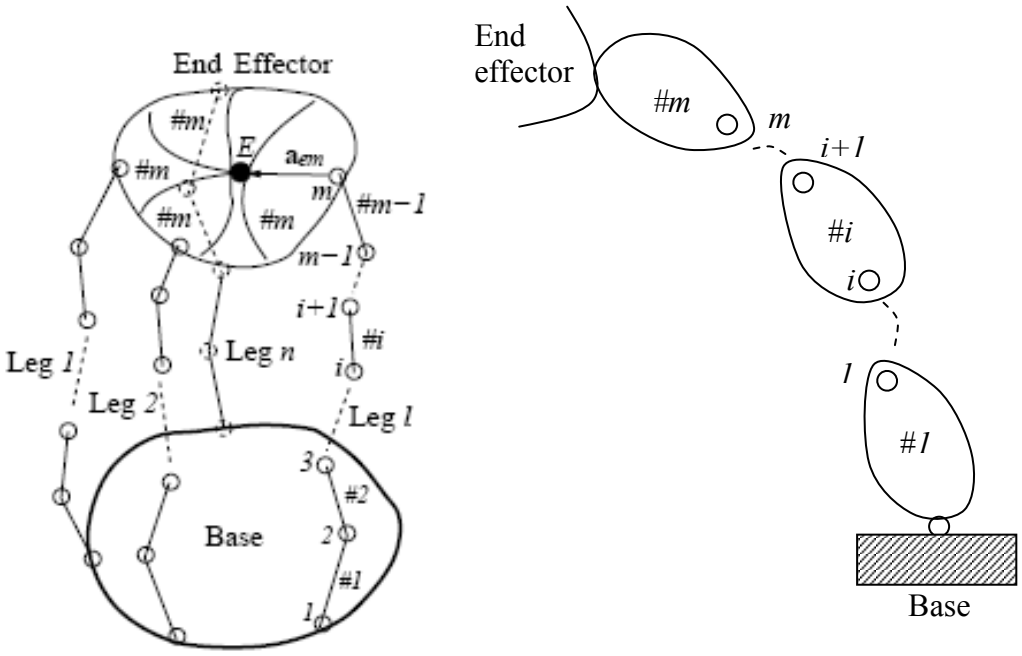


Figure 9 Simulated joint displacements for Stanford arm: (a) Joint 1; (b) Joint 2; (c) Joint 3; (d) Joint 4; (e) Joint 5; (f) Joint 6



(a) Architecture (b) The *l*th leg

Figure 10 Parallel system

- The system is non-redundant, i.e., the degrees of freedom (DOF) is equal to the number of driving actuators;
- Number of legs is equal to the DOF;
- Joint positions are known either from the sensor data, or from a suitable algorithm, e.g., using the one proposed in Hiller (1995);
- No friction or damping;
- Gravity is taken into account by adding negative acceleration due to gravity to the base body of the system given in eq. (17) for serial-chain systems.

Referring to each leg of the parallel system, Fig. 10(b) and eq. (11c), the twist of the end-effector can be expressed in terms of the twist of the *m*th body, namely,

$$\mathbf{t}_e = \mathbf{A}_{em}\mathbf{t}_m \tag{24}$$

The unactuated joint rates, for $i = m, \ldots, 2$, are then solved recursively. For that, a useful relation is introduced next. Let $\mathbf{a}=\mathbf{b}+\mathbf{c}x$, where $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ are vectors of same dimension, while x is a scalar. One may determine x as

$$x = \frac{1}{\delta_c}\mathbf{c}^T(\mathbf{a}-\mathbf{b}), \text{ where } \qquad \delta_c \equiv \mathbf{c}^T\mathbf{c} \tag{25a}$$

Substituting the value for x from eq. (25a) into the expression, $\mathbf{a}=\mathbf{b}+\mathbf{c}x$**,** and rearranging the terms, one may eliminate x from eq. (25a) to obtain a relationship between $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ as

$$\mathbf{\Phi}_c\mathbf{a} = \mathbf{\Phi}_c\mathbf{b}, \text{ where } \qquad \mathbf{\Phi}_c \equiv \mathbf{1} - \frac{1}{\delta_c}\mathbf{c}\mathbf{c}^T \tag{25b}$$

In eq. (25b), $\mathbf{1}$ is the identity matrix of compatible dimension, whereas the symmetric matrix, $\mathbf{\Phi}_c$, projects vectors $\mathbf{a}$ and $\mathbf{b}$ along the axis orthogonal to $\mathbf{c}$. The matrix, $\mathbf{\Phi}_c$, is singular and has the following properties:

$$\mathbf{\Phi}_c^T\mathbf{\Phi}_c = \mathbf{\Phi}_c\mathbf{\Phi}_c^T = \mathbf{\Phi}_c \tag{25c}$$

To eliminate the unactuated joint velocities, eq. (25b) is employed here. For example, to eliminate the unactuated *m*th joint rate of each leg, $\dot{\theta}_m$, eq. (24) is first written in the form of $\mathbf{a}=\mathbf{b}+\mathbf{c}x$, i.e.,

$$\mathbf{t}_e = \mathbf{A}_{em}(\mathbf{A}_{m,m-1}\mathbf{t}_{m-1} + \mathbf{p}_m\dot{\theta}_m) = \widetilde{\mathbf{t}}_{m-1} + \widetilde{\mathbf{p}}_m\dot{\theta}_m \text{, where } \widetilde{\mathbf{t}}_{m-1} \equiv \mathbf{A}_{e,m-1}\mathbf{t}_{m-1} \text{, and } \widetilde{\mathbf{p}}_m \equiv \mathbf{A}_{em}\mathbf{p}_m \quad (26)$$

where the relation of the twist of the *m*th body, $\mathbf{t}_m$, in terms of the twist of the (*m*-1)st body, i.e., $\mathbf{t}_{m-1}$, in the form of eq. (11c) is used. Applying the rules of eqs. (25a-b) to eq. (26), one readily obtains the following:

$$\dot{\theta}_m = \frac{1}{\delta_m}\widetilde{\mathbf{p}}_m^T(\mathbf{t}_e - \widetilde{\mathbf{t}}_{m-1}) \text{, where } \delta_m \equiv \widetilde{\mathbf{p}}_m^T\widetilde{\mathbf{p}}_m \quad (27a)$$

and

$$\mathbf{\Phi}_m\mathbf{t}_e = \mathbf{\Phi}_m\widetilde{\mathbf{t}}_{m-1} \text{, where} \qquad \mathbf{\Phi}_m \equiv \mathbf{1} - \frac{1}{\delta_m}\widetilde{\mathbf{p}}_m\widetilde{\mathbf{p}}_m^T \quad (27b)$$

The process is repeated until the first actuated joint rate, i.e., joint 1, is reached, i.e.,

$$\mathbf{\Phi}_2\mathbf{t}_e = \mathbf{\Phi}_2\widetilde{\mathbf{t}}_1 \text{ or } \mathbf{\Phi}_2\mathbf{t}_e = \mathbf{\Phi}_2\mathbf{A}_{e1}\mathbf{p}_1\dot{\theta}_1 \quad (28)$$

Since the matrix, $\mathbf{\Phi}_2$, is singular the twist of the end-effector, $\mathbf{t}_e$, cannot be expressed in terms of the actuated joint rate, $\dot{\theta}_1$. However, if all the legs are considered, eq. (28) can be combined as

$$\mathbf{\Phi}\mathbf{t}_e = \sum_{l=1}^{n}[\mathbf{\Phi}_2\mathbf{A}_{e1}\mathbf{p}_1\dot{\theta}_1]^l \text{, where } \mathbf{\Phi} \equiv \sum_{l=1}^{n}\mathbf{\Phi}_2^l \quad (29a)$$

where superscript '*l*' stands for leg. Equations (29a) now allows one to obtain the twist of the end-effector in terms of all the actuated joint rates, namely, $\dot{\mathbf{\theta}} \equiv \lfloor \dot{\theta}_1^1 \quad \cdots \quad \dot{\theta}_1^n \rfloor$, as

$$\mathbf{t}_e = \mathbf{J}\dot{\mathbf{\theta}} \text{, where } \mathbf{J} \equiv \mathbf{\Phi}^{-1}[(\mathbf{\Phi}_2\mathbf{A}_{e1}\mathbf{p}_1)^1, \cdots, (\mathbf{\Phi}_2\mathbf{A}_{e1}\mathbf{p}_1)^n] \quad (29b)$$

The 6×*n* matrix, **J,** is the well-known Jacobian matrix, which generally exists the system's configuration is such that the 6×6 matrix, $\mathbf{\Phi}$, is singular.

3.3 The DeNOC matrices for parallel systems

Solving for theunactuated joint rates for each leg in Subsection 3.2, one can now express the twists of all bodies in a leg in terms of its actuated joint rates, namely,

$$\mathbf{t}_1 = \mathbf{p}_1\dot{\theta}_1 \quad (30a)$$

$$\mathbf{t}_2 = \mathbf{A}_{21}\mathbf{t}_1 + \mathbf{p}_2\dot{\theta}_2 = \mathbf{\Psi}_2\mathbf{A}_{21}\mathbf{t}_1 + \frac{1}{\delta_2}\mathbf{p}_2\widetilde{\mathbf{p}}_2^T\mathbf{t}_e \quad (30b)$$

$$\vdots$$

$$\mathbf{t}_m = \mathbf{A}_{m,m-1}\mathbf{t}_{m-1} + \mathbf{p}_m\dot{\theta}_m = \mathbf{\Psi}_m\mathbf{A}_{m,m-1}\mathbf{t}_{m-1} + \frac{1}{\delta_m}\mathbf{p}_m\widetilde{\mathbf{p}}_m^T\mathbf{t}_e \quad (30c)$$

where $\mathbf{\Psi}_i$, for $i = 2, \ldots, m$, is the 6×6 matrix, which can be obtained as

$$\mathbf{\Psi}_i \equiv \mathbf{1} - \frac{1}{\delta_i}\mathbf{p}_i\widetilde{\mathbf{p}}_i^T\mathbf{A}_{ei} \quad (31)$$

Combining eqs. (30a-c) for all legs, and performing some algebraic manipulations, one can then write the following (Saha and Schiehlen, 2001):

$$\mathbf{t} = \mathbf{N}_l\mathbf{N}_c\mathbf{N}_d\dot{\mathbf{\theta}} \quad (32a)$$

where the 6*mn*-dimensional twist and *n*-dimensional joint rate vector are as follows:

$$\mathbf{t} \equiv \begin{bmatrix} \mathbf{t}_1^{(1n)} \\ \vdots \\ \mathbf{t}_n^{(1n)} \end{bmatrix} \text{, where } \mathbf{t}_i \equiv \begin{bmatrix} \mathbf{t}_i^1 \\ \vdots \\ \mathbf{t}_i^n \end{bmatrix} \text{ and } \dot{\mathbf{\theta}} \equiv \begin{bmatrix} \dot{\theta}_1^1 \\ \vdots \\ \dot{\theta}_1^n \end{bmatrix} \quad (32b)$$

Moreover, the $6mn \times 6mn$ lower block triangular matrix, $\mathbf{N}_l$, the $6mn \times 6n$ matrix, $\mathbf{N}_c$, and the $6n \times 6n$, block diagonal matrix, $\mathbf{N}_d$, are given by

$$\mathbf{N}_l \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} & \cdots & \mathbf{O} \\ \widetilde{\mathbf{A}}_{21} & \mathbf{1} & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \widetilde{\mathbf{A}}_{m1} & \widetilde{\mathbf{A}}_{m2} & \cdots & \mathbf{1} \end{bmatrix}^{(1n)}, \quad \mathbf{N}_c \equiv \begin{bmatrix} \mathbf{1} \\ \mathbf{C}\overline{\mathbf{A}}_{e1} \\ \vdots \\ \mathbf{C}\overline{\mathbf{A}}_{e1} \end{bmatrix}^{(1n)}, \text{ and } \mathbf{N}_d \equiv \begin{bmatrix} \mathbf{p}_1^1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{p}_1^2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{p}_1^n \end{bmatrix} \tag{32c}$$

where the superscript (1n) of the matrix notation imply that the non-zero element matrices are to be read with the superscript (1n), e.g., $\widetilde{\mathbf{A}}_{ij}^{(1n)} \equiv diag[\widetilde{\mathbf{A}}_{ij}^1 \cdots \widetilde{\mathbf{A}}_{ij}^n]$, etc. Moreover, the 6×6 matrices, $\widetilde{\mathbf{A}}_{ij}^l$ and $\overline{\mathbf{A}}_{e1}^l$, and the $6n\times6n$ constant matrix, $\mathbf{C}$, are as follows:

$$\widetilde{\mathbf{A}}_{ij}^l \equiv (\mathbf{\Psi}_i \mathbf{A}_{i,i-1} \mathbf{\Psi}_{i-1} \mathbf{A}_{i,i-2} \cdots \mathbf{\Psi}_{j+1} \mathbf{A}_{i,j-1})^l \tag{32d}$$

and

$$\mathbf{C} \equiv \begin{bmatrix} \mathbf{1} & \cdots & \mathbf{1} \\ \vdots & \ddots & \vdots \\ \mathbf{1} & \cdots & \mathbf{1} \end{bmatrix} \tag{32e}$$

where $\mathbf{1}$ represents the 6×6 identity matrix. The matrices, $\mathbf{N}_l$, $\mathbf{N}_c$, and $\mathbf{N}_d$, are the DeNOC matrices for the parallel system. Compared to its serial counterpart, the differences are

- For serial-chain systems, the DeNOC matrices are two, namely, $\mathbf{N}_l$ and $\mathbf{N}_d$, which are lower block triangular and block diagonal, respectively. Matrix $\mathbf{N}_l$ is a function of $\mathbf{A}_{ij}$ only.
- In case of parallel systems, the DeNOC matrices are three. Here, matrix $\mathbf{N}_l$ is not only the function of $\mathbf{A}_{ij}$, but also of $\mathbf{\Psi}_I$ that takes care of the effect of the unactuated joint rates on the twist of rigid bodies in a leg. Furthermore, the matrix, $\mathbf{N}_c$, accounts for the interaction between all leg motions through the twist of the end-effector, $\mathbf{t}_e$. Finally, the matrix, $\mathbf{N}_d$, has the same structure as that of a serial-chain system (Saha, 1997; 1999a-b).

3.4 Dynamic model

For the dynamic model, the $6mn$ uncoupled NE equations of motion are again expressed in the form of eq. (8), where the $6mn\times6mn$ matrices, $\mathbf{M}$, $\mathbf{W}$, and $\mathbf{E}$ are as follows:

$$\mathbf{M} \equiv \begin{bmatrix} \mathbf{M}_1^{(1n)} & \cdots & \mathbf{O} \\ \vdots & \ddots & \vdots \\ \mathbf{O} & \cdots & \mathbf{M}_m^{(1n)} \end{bmatrix}; \ \mathbf{W} \equiv \begin{bmatrix} \mathbf{W}_1^{(1n)} & \cdots & \mathbf{O} \\ \vdots & \ddots & \vdots \\ \mathbf{O} & \cdots & \mathbf{W}_m^{(1n)} \end{bmatrix}; \text{ and } \mathbf{E} \equiv \begin{bmatrix} \mathbf{E}_1^{(1n)} & \cdots & \mathbf{O} \\ \vdots & \ddots & \vdots \\ \mathbf{O} & \cdots & \mathbf{E}_m^{(1n)} \end{bmatrix} \tag{33a}$$

where the $6n\times6n$ block matrices, $\mathbf{M}_i^{(1n)}$, $\mathbf{W}_i^{(1n)}$, and $\mathbf{E}_i^{(1n)}$ are given by

$$\mathbf{M}_i^{(1n)} \equiv \begin{bmatrix} \mathbf{M}_i^1 & \cdots & \mathbf{O} \\ \vdots & \ddots & \vdots \\ \mathbf{O} & \cdots & \mathbf{M}_i^n \end{bmatrix}; \mathbf{W}_i^{(1n)} \equiv \begin{bmatrix} \mathbf{W}_i^1 & \cdots & \mathbf{O} \\ \vdots & \ddots & \vdots \\ \mathbf{O} & \cdots & \mathbf{W}_i^n \end{bmatrix}; \text{ and } \mathbf{E}_i^{(1n)} \equiv \begin{bmatrix} \mathbf{E}_i^1 & \cdots & \mathbf{O} \\ \vdots & \ddots & \vdots \\ \mathbf{O} & \cdots & \mathbf{E}_i^n \end{bmatrix} \tag{33b}$$

Rest of the methodology remains same as presented for the serial-chain system in Subsection 2.3.

3.5 Numerical example: Five-bar parallel manipulator

A five-bar planar parallel manipulator, also known as planar Delta robot (Brauchli and Weber, 1991), is shown in Fig. 11, in which the actuated joints are the first joints of both the legs that are indicated with pointing arcs. The parameters associated with the manipulator are given by

$$a_1^1 = 0.6m; a_2^1 = 1.1m; a_1^2 = 0.7m; a_2^2 = 1.2m \tag{34a}$$

$$m_1^1 = 3.0kg; m_2^1 = 1.6kg; m_1^2 = 2.0kg; m_2^2 = 1.1kg; g = 9.81m/s^2 \tag{34b}$$

where a_i^l, for *i,l* = 1,2, and a_0 are the link lengths, as shown in Fig. 11, and m_i^l, for *i,l* = 1,2, are the masses of the link *I* in the *l*th leg. Other values associated with the trajectories of the actuated joint, i.e., the joints 1 of legs 1 and 2 in the form of eq. (22a) are as follows:

$$\theta_1^1(0) = 30^o; \theta_1^1(T) = 90^o; \theta_1^2(0) = 30^o; \theta_1^2(T) = 70.3^o; \text{and } T = 20\,\text{sec} \tag{35}$$

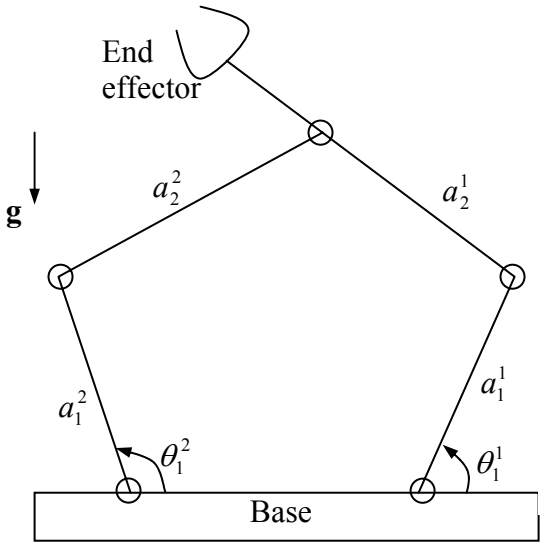


Figure 11 Five-bar manipulator

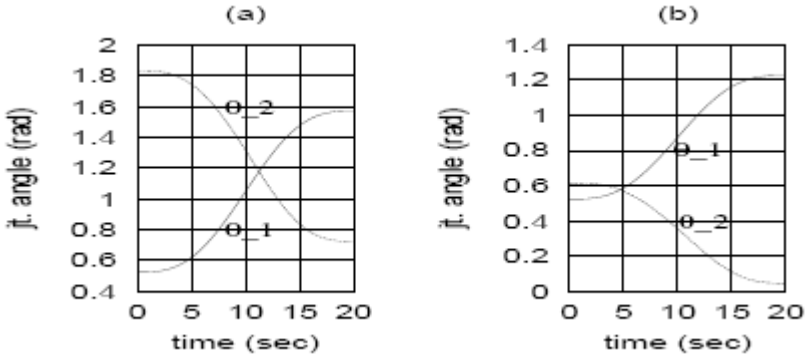


Figure 12 Joint angles (a) Leg 1; (b) Leg 2

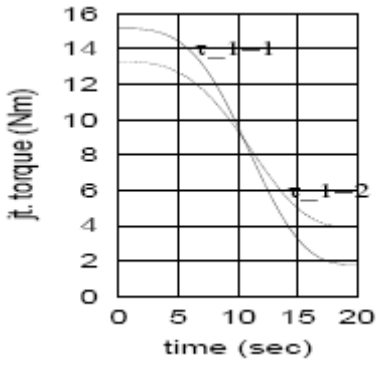


Figure 13 Actuated joint torques

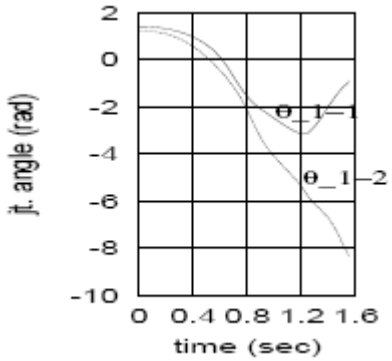


Figure 14 Simulation results

The required actuated joint torques of the vertical planar parallel manipulator following the trajectory given by eqs. (22a) and (35) are shown in Fig. 13, where 'τ_1-1' and 'τ_1-2' represent the torques τ_1^1 and τ_1^2, respectively. Free-fall simulation with the initial conditions given below:

$$\theta_1^1(0) = 80^o(1.4rad); \theta_1^2(0) = 70^o(1.22rad); \dot{\theta}_1^1(0) = \dot{\theta}_1^2(0) = 0rad/\text{sec} \tag{36}$$

is carried out next whose results are shown in Fig. 14, where 'θ_1-1' and 'θ_1-2' mean θ_1^1 and θ_1^2, respectively. Note that after about 1.4 sec of simulation time the program stopped executing because the manipulator comes to an extreme position when it is singular.

4 Modeling of Closed-chain Systems

As pointed out in Section 3, semi-recursive formulations for general closed-chain multibody systems were developed with the DAE and ODE approaches. Here, a subsystem-level recursive method is presented for the inverse dynamics of a general closed-chain multibody system, which is based on the methodology proposed in Chaudhary and Saha (2007). To achieve such recursion, the closed-chains are first cut at appropriate joints to make a spanning tree. The spanning tree is an open system, that can have several serial or tree-type subsystems which are classified as 'determinate' or 'indeterminate,' as explained in Subsection 4.1. Next, the constrained equations of motion for the determinate subsystems are derived and solved for the Lagrange multipliers representing the constraint wrenches at the cut joints, and the driving torques/forces, if any. Interestingly, the determination of the Lagrange multipliers makes some of the indeterminate subsystems determinate, and the process is repeated. The above two steps are referred here as the subsystem-level recursion. The methodology also allows one to compute the joint moments and forces at the uncut unactuated joints using the body-level recursion (Chaudhary and Saha, 2007).

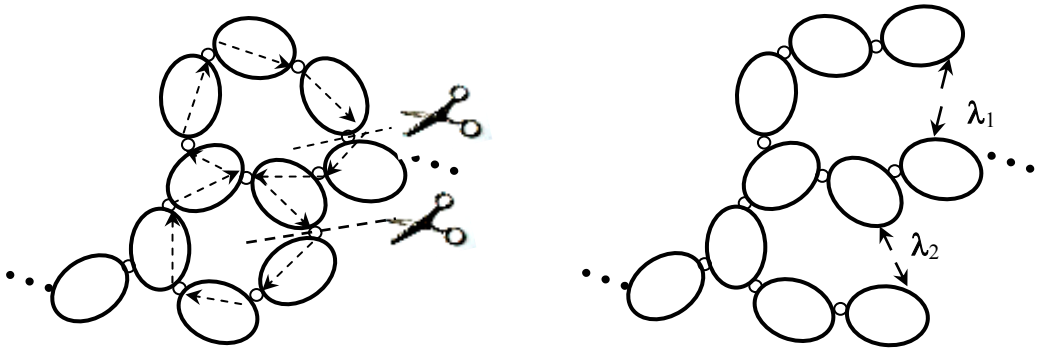
4.1 Subsystems and their classifications

Assume that there are one or more closed kinematic loops in a closed-chain multibody system under study, as shown in Fig. 15(a). In order to convert such multiloop system into an equivalent open-loop system, the closed kinematic loops are cut at some appropriate joints, as indicated in Fig. 15(a). Its open-loops are then shown in Fig. 15(b), where λ_1 and λ_2 represent the Lagrange multipliers to keep the complete system in equilibrium. For a complex multiloop system, the joints to be cut can be identified using the graph theory approach (McPhee, 1996). The resulting cut system is called the spanning tree of the original closed-loop system, as illustrated in Fig. 16. The distinct branches of the spanning tree which originate from the base body, #0, are referred as "subsystems," which could be either serial- or tree-type. For the purpose of defining the architecture of the spanning tree, base body is generally chosen as the fixed body of the system under study. Any other body whose position, velocity and acceleration are known can also be selected as the base body. Next, for the serial subsystems, the bodies are numbered from #1 that is connected to the base body, as indicated in Fig. 16. For a tree-type subsystem, the longest chain from the base body, #0, is called the main chain and assumed to have n^0 bodies, whereas all other serial branches are called subchains. The subchains are assumed to be connected to the main chain, as shown in Fig. 16 for subchains *k* and ℓ. Any subchain is identified by its base body, e.g., in subsystem I, the subchain *k* stems from the *k*th body of the main chain, i.e., $\#k^0$, and has n^k bodies. If more than one subchains emerge from a body of the main chain, the each subchain can be identified by double subscripts. For example, *k*1 and *k*2, can emerge from the *k*th body of the main chain having n^{k1} and n^{k2} bodies. Similarly the subchain, ℓ, is connected to the ℓth body of the main chain, $\#\ell^0$, and has n^ℓ bodies, as shown in Fig. 16. In addition, the Roman numerals are prefixed before the number of a body to recognize a subsystem to which the body belongs. For example, as indicated in Fig. 2, $\#\mathrm{I}-1^k$ denotes the 1st body of the *k*th subchain in subsystem I, whereas $\mathrm{I}-1^k$ denotes the first joint of the *k*th subchain in subsystem I. Note that the symbol, "#", is used to distinguish labeling of bodies from that of joints. The index of subchain is dropped in the serial

subsystems, as in subsystems II and III, because the main chain has no subchains. Assuming that all joints are of one degree-of-freedom, and the total number of moving bodies is n, then the degree of freedom (DOF) of the spanning tree is given by

$$\text{DOF} = 6 + \sum_{j=I}^{s} n_j \tag{37}$$

where n_j is the number bodies in the jth subsystem with the base body having six DOF. Moreover, s denotes the number of subsystems. For the spanning tree shown in Fig. 16, $n_I = n^0 + n^k + n^\ell$, $n_{II} = r$, $n_{III} = 2$, and hence, $n = n_I + n_{II} + n_{III}$.



(a) The closed loops (b) Its open-loops

Figure 15 Closed chains in a multiloop multibody system

On a free body in the spanning tree of Fig. 2 there may be as many as four categories of moments and forces or wrenches acting on it, namely, 1) external wrench from the environment that are external to the system, and those provided by the actuators to the system; 2) inertia wrench due to the motion of the body; 3) Lagrange multipliers representing the constraint or reaction wrench at the cut joints; and 4) the constraint wrench at the uncut joints. For a non-redundant spanning tree resulting from the closed-loop system, the total number of unknowns, namely, the Lagrange multipliers and the driving torques/forces, is equal to the DOF of the spanning tree. Such a spanning tree is referred here as a determinate system. For example, the spanning tree of the four-bar mechanism, as shown in Fig. 17, is determinate, as it has three unknowns, namely, λ_1, λ_2, and τ_D, and three-DOF. Following the above definition, any subsystem originating from the base body of the spanning tree can also be categorized as *determinate*. The determinate subsystem is the one in which the number of unknowns, i.e., the Lagrange multipliers and the driving torques/forces associated with the subsystem are equal to its DOF. If the condition for the determinate is not satisfied then the subsystem is called *indeterminate*. It can be shown that once the unknowns for the determinate subsystems are solved, one or more of the remaining indeterminate subsystem(s) converted into determinate one. For example, there are two spanning trees for the four-bar mechanism depending on which joint is cut. In Fig. 17(a), subsystems I and II are determinate and indeterminate, respectively. Subsystem I has two unknowns, λ_1 and λ_2, with two DOF whereas subsystem II has three unknowns, λ_1, λ_2, and τ_D, with one-DOF. Alternatively, In Fig. 17(b) both subsystems are indeterminate.

4.2 Equations of motion

In this Subsection, a methodology for the development of recursive dynamics algorithm, mainly, for inverse dynamics, of a closed-loop system is presented. First, the closed-loop system is converted into a spanning tree by cutting the appropriate joints of the closed-loops, presented in Subsection 4.1. The loop-closure constraints are then incorporated into the equations of motion as Lagrange multipliers. The

equations of motion for the subsystems and the spanning tree are then systematically derived, as explained next.

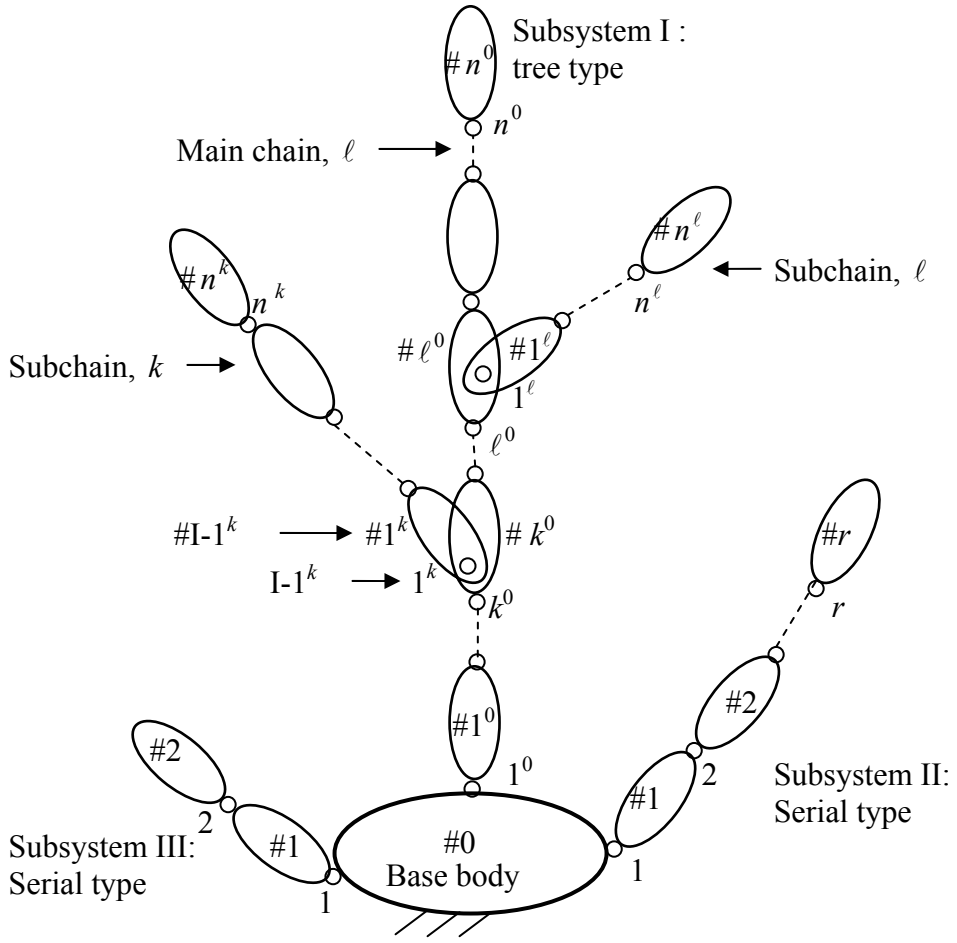


Figure 16 A spanning tree

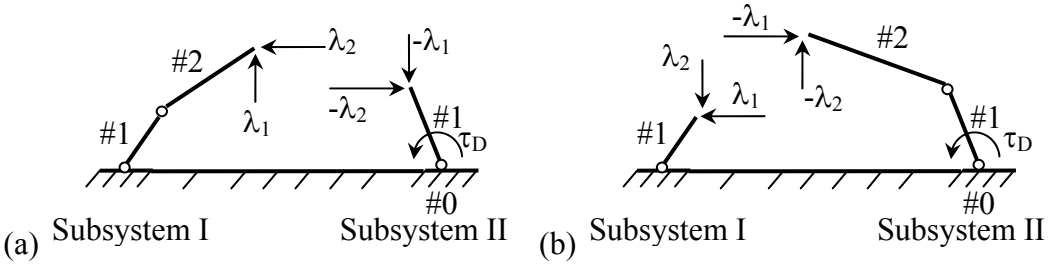


Fig. 17 Open systems for four-bar mechanism

4.2.1 Subsystem

The Newton-Euler (NE) equations of motion for the *i*th rigid body of the *j*th subsystem can be represented using eq. (5), whereas for the subsystem *j* having n_j moving bodies, the $6n_j$ scalar unconstrained equations of motion are expressed as

$$\mathbf{M}_j\dot{\mathbf{t}}_j + \mathbf{W}_j\mathbf{M}_j\mathbf{E}_j\mathbf{t}_j = \mathbf{w}_j \tag{38}$$

where the $6n_j$-vectors $\mathbf{t}_j$, $\dot{\mathbf{t}}_j$, and $\mathbf{w}_j$, respectively, are the generalized twist, twist-rate, and wrench, whereas the $6n_j \times 6n_j$ matrices, $\mathbf{M}_j$, $\mathbf{W}_j$, and $\mathbf{E}_j$, are the generalized mass, angular velocity, and coupling matrices, respectively. For a serial subsystem with the bodies numbered from 1 to n_j, the $6n_j$-dimensional vectors, $\mathbf{t}_j$, $\dot{\mathbf{t}}_j$, and $\mathbf{w}_j$, and the $6n_j \times 6n_j$ matrices, $\mathbf{M}_j$, $\mathbf{W}_j$, and $\mathbf{E}_j$, are defined as shown in eq. (9). For the complete tree-type system, i.e., Subsystem I of Fig. 16, the generalized vectors, $\mathbf{t}$, $\dot{\mathbf{t}}$, $\mathbf{w}$, and the generalized matrices, $\mathbf{M}$, $\mathbf{W}$ and $\mathbf{E}$, are then defined as

$$\mathbf{t}_j \equiv \begin{bmatrix} \mathbf{t}^0 \\ \mathbf{t}^k \\ \mathbf{t}^\ell \end{bmatrix};\ \dot{\mathbf{t}}_j \equiv \begin{bmatrix} \dot{\mathbf{t}}^0 \\ \dot{\mathbf{t}}^k \\ \dot{\mathbf{t}}^\ell \end{bmatrix};\ \mathbf{w}_j \equiv \begin{bmatrix} \mathbf{w}^0 \\ \mathbf{w}^k \\ \mathbf{w}^\ell \end{bmatrix} \tag{39a}$$

and

$$\mathbf{M}_j \equiv diag[\mathbf{M}^0 \quad \mathbf{M}^k \quad \mathbf{M}^\ell];\ \mathbf{W}_j \equiv diag[\mathbf{W}^0 \quad \mathbf{W}^k \quad \mathbf{W}^\ell];\ \mathbf{E}_j \equiv diag[\mathbf{E}^0 \quad \mathbf{E}^k \quad \mathbf{E}^\ell] \tag{39b}$$

where the components of vectors and matrices are of sizes according to the serial subchains in the tree-type Subsystem I. For example, $\mathbf{t}^0$ and $\mathbf{M}^0$ are $6n^0$-dimensional vector and $6n^0 \times 6n^0$ matrix, respectively. Note that, in contrast to serial-chain systems where the wrench, $\mathbf{w}$ of eq. (14), is composed of the wrenches, $\mathbf{w}^E$, due to externally applied moments and forces on it including those provided by the driving actuators, and $\mathbf{w}^C$, due to the nonworking constraint moments and forces at the uncut joints only. However, for the open-chain systems resulting from closed loops, an additional term, namely, $\mathbf{w}^\lambda$, representing the constraint moments and forces at the cut joints must be present, i.e., $\mathbf{w} \equiv \mathbf{w}^E + \mathbf{w}^C + \mathbf{w}^\lambda$. For a subsystem, eq. (38) is now rewritten as

$$\mathbf{M}_j \dot{\mathbf{t}}_j + \mathbf{M}_j \mathbf{W}_j \mathbf{t}_j = \mathbf{w}_j^E + \mathbf{w}_j^C + \mathbf{w}_j^\lambda \tag{40}$$

where $\mathbf{w}_j^E$, $\mathbf{w}_j^C$ and $\mathbf{w}_j^\lambda$ denote the $6n_j$-dimensional vectors of corresponding wrenches associated with the jth subsystem. It can be shown that the pre-multiplication of the transpose of the natural orthogonal complement (NOC) matrix, $\mathbf{N}_j$, associated with the velocity constraint of the subsystem, with the unconstrained NE equations of motion, eq. (40), leads to a set of n_j constrained equations of motion free from the constraint wrenches at the uncut joints, as done in Subsection 2.3, i.e.,

$$\mathbf{N}_j^T(\mathbf{M}_j \dot{\mathbf{t}}_j + \mathbf{W}_j \mathbf{M}_j \mathbf{E}_j \mathbf{t}_j) = \mathbf{N}_j^T(\mathbf{w}_j^e + \mathbf{w}_j^\lambda) \tag{9}$$

where the term, $\mathbf{N}_j^T \mathbf{w}_j^C$, vanishes, as pointed out in Subsection 2.3. Note that the size of the NOC matrix, $\mathbf{N}_j$, is $6n_j \times n_j$ if all n_j moving bodies of the jth subsystem are coupled with one-DOF joints. Now, introducing the notation for the inertia wrench of the jth subsystem as $\mathbf{w}_j^*$, i.e., $\mathbf{M}_j \dot{\mathbf{t}}_j + \mathbf{W}_j \mathbf{M}_j \mathbf{E}_j \mathbf{t}_j \equiv \mathbf{w}_j^*$, eq. (41) is rewritten as

$$\mathbf{N}_j^T \mathbf{w}_j^* = \tau_j^E + \tau_j^\lambda \tag{10}$$

where

$\tau_j^E \equiv \mathbf{N}_j^T \mathbf{w}_j^E$: the n_j-dimensional vector of generalized forces due to the external moments and forces, and those resulting from the actuators, gravity, and dissipation;

$\boldsymbol{\tau}_j^\lambda \equiv \mathbf{N}_j^T \mathbf{w}_j^\lambda$: the n_j-dimensional vector of generalized forces due to the constraint moments and forces at the cut joints, i.e., the Lagrange multipliers.

Equation (42) contains the n_j scalar equations that are linear in Lagrange multipliers, and the driving torques/forces associated with the jth subsystem. Note that, for a determinate subsystem, n_j is equal to the number of unknown Lagrange multipliers and the driving torques/forces, if any, that are

associated with it and can be solve uniquely. With the unknowns solved for the determinate subsystems, some or all of all the indeterminate subsystems become determinate, and the process can be repeated. This is referred here as the subsystem-level recursion.

4.2.2 Spanning tree

In case all the subsystems are indeterminate one needs to consider the whole spanning tree, whose constrained equations are obtained using eq. (42) as

$$\mathbf{N}_j^T \mathbf{w}_j^* = \boldsymbol{\tau}_j^E + \boldsymbol{\tau}_j^\lambda \text{ for } j=\text{I}, \ \ldots, s \tag{43}$$

where s is the number of subsystems, which can be either serial or tree-type originating from the base body of the spanning tree. Equation (43) is written in a compact form as:

$$\mathbf{N}^T \mathbf{w}^* = \boldsymbol{\tau}^e + \boldsymbol{\tau}^\lambda \tag{44}$$

where

$$\mathbf{w}^* \equiv \begin{bmatrix} \mathbf{w}_I^* \\ \vdots \\ \mathbf{w}_s^* \end{bmatrix}; \boldsymbol{\tau}^E \equiv \begin{bmatrix} \boldsymbol{\tau}_I^E \\ \vdots \\ \boldsymbol{\tau}_s^E \end{bmatrix}; \text{ and } \boldsymbol{\tau}^\lambda \equiv \begin{bmatrix} \boldsymbol{\tau}_I^\lambda \\ \vdots \\ \boldsymbol{\tau}_s^\lambda \end{bmatrix} \tag{45}$$

and $\mathbf{N}$ is the $6n \times n$ NOC matrix for the spanning tree, which is

$$\mathbf{N} = diag(\mathbf{N}_I \quad \cdots \quad \mathbf{N}_s) \tag{46}$$

in which $\mathbf{N}_j$ is the NOC matrix for the jth subsystem be it a serial or tree-type, derived next in Subsection 4.3. Moreover, $\mathbf{w}^*$ is the $6n$-dimensional vector, and τ^E and $\boldsymbol{\tau}^\lambda$ are the n-dimensional vectors, where $n \equiv n_I + \cdots + n_S$ is the total number bodies in the spanning tree. For a non-redundant spanning tree resulting from the closed-loop system, the total number of unknowns, the driving torques/forces, and the Lagrange multipliers, is equal to the total number of bodies, n, and the DOF of the system. Hence, the scalar equations of Eq. (14) can be solved using any standard method such as LU decomposition (Stewart, 1973). This approach is termed here as the system approach, where eqs. (42) and (44) are used. Similar methodology is also reported in Nikravesh and Gim (1993), Rodriguez et al. (1992), Anderson and Critchley, (2003), Shabana (1994)

4.3 The DeNOC matrices for spanning tree

The natural orthogonal complement (NOC) matrix is used to derive the constrained equations of motion, eqs. (40) and (42), for a subsystem and the spanning tree, respectively. For the serial-type subsystem, the NOC matrix can be obtained as the multiplication of two block matrices, as done in eq. (13a), which are called the decoupled natural orthogonal complement (DeNOC) matrices. For a tree-type subsystem, they were derived next. Referring to the tree-type system of Fig. 16, i.e., Subsystem I, the generalized twist, for the main chain, denoted as $\mathbf{t}^0$, is obtained similar to eq. (13a) as

$$\mathbf{t}^0 = \mathbf{N}_l^0 \mathbf{N}_d^0 \dot{\boldsymbol{\theta}}^0 \tag{47}$$

where superscript '0' indicates the main chain mentioned in Subsection 4.1. Moreover, for the kth subchain of subsystem I, the $6n^k$-dimensional vector of generalized twist, denoted as $\mathbf{t}^k$, is given by

$$\mathbf{t}^k = -\mathbf{N}_l^k \mathbf{A}_0^k \mathbf{t}_0^k + \mathbf{N}_l^k \mathbf{N}_d^k \dot{\boldsymbol{\theta}}^k \tag{48}$$

where the $6n^k \times 6n^k$ matrix, $\mathbf{N}_l^k$, and the $6n^k \times n^k$ matrix $\mathbf{N}_d^k$ are the DeNOC matrices of the kth subchain, and $\mathbf{t}_0^k \equiv \mathbf{t}_k^0$ is nothing but the twist of the kth body in the main chain, 0. The twist, $\mathbf{t}_k^0$, can be obtained from eq. (47) as

$$\mathbf{t}_k^0 = \mathbf{N}_{lk}^0 \mathbf{N}_d^0 \dot{\boldsymbol{\theta}}^0 \tag{49a}$$

where the $6\times 6n^0$ matrix, $\mathbf{N}_{lk}^0$, is given by

$$\mathbf{N}_{lk}^0 \equiv \begin{bmatrix} \mathbf{A}_{k,1}^0 & \cdots & \mathbf{A}_{k,k-1}^0 & \mathbf{1} & \mathbf{O} & \cdots & \mathbf{O} \end{bmatrix} \tag{49b}$$

Substituting eq. (49a) into eq. (49b) yields

$$\mathbf{t}^k = \mathbf{N}_l^{k0} \mathbf{N}_d^0 \dot{\boldsymbol{\theta}}^0 + \mathbf{N}_l^k \mathbf{N}_d^k \dot{\boldsymbol{\theta}}^k, \text{ where } \mathbf{N}_l^{k0} \equiv -\mathbf{N}_l^k \mathbf{A}_0^k \mathbf{N}_{lk}^0 \tag{50}$$

For other subchains one can similarly obtain eq. (50). Now, the generalized twist for the tree-type subsystem I, Fig. 16, i.e., the 6*n*-vector, **t**, can be expressed as

$$\mathbf{t} = \mathbf{N}\dot{\boldsymbol{\theta}}, \text{ where } \mathbf{N} \equiv \mathbf{N}_l \mathbf{N}_d \tag{51}$$

where $n \equiv n^0 + n^k + n^\ell$ being the total number of moving bodies in the tree-type system and the 6*n*-vector, **t,** *n*-dimensional vector, $\dot{\boldsymbol{\theta}}$, the $6n\times 6n$ matrix, $\mathbf{N}_l$, and the $6n\times n$ matrix, $\mathbf{N}_d$, are defined by

$$\mathbf{t} \equiv \begin{bmatrix} \mathbf{t}^0 \\ \mathbf{t}^k \\ \mathbf{t}^\ell \end{bmatrix};\ \dot{\boldsymbol{\theta}} \equiv \begin{bmatrix} \dot{\boldsymbol{\theta}}^0 \\ \dot{\boldsymbol{\theta}}^{\mathrm{k}} \\ \dot{\boldsymbol{\theta}}^\ell \end{bmatrix};\ \mathbf{N}_l \equiv \begin{bmatrix} \mathbf{N}_l^0 & \mathbf{O} & \mathbf{O} \\ \mathbf{N}_l^{k0} & \mathbf{N}_l^k & \mathbf{O} \\ \mathbf{N}_l^{\ell 0} & \mathbf{O} & \mathbf{N}_l^\ell \end{bmatrix}; \text{ and } \mathbf{N}_d \equiv \begin{bmatrix} \mathbf{N}_d^0 & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{N}_d^k & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{N}_d^\ell \end{bmatrix} \tag{52}$$

For additional subchains, one can modify the expressions of **t**, $\dot{\boldsymbol{\theta}}$, $\mathbf{N}_l$, and $\mathbf{N}_d$, as given in eq. (52). Equations (51-52) together provide the DeNOC matrices for the tree-type system at hand, which are used to reduce the dimension of the system's NE equations of motion, as mentioned in Subsection 4.2.

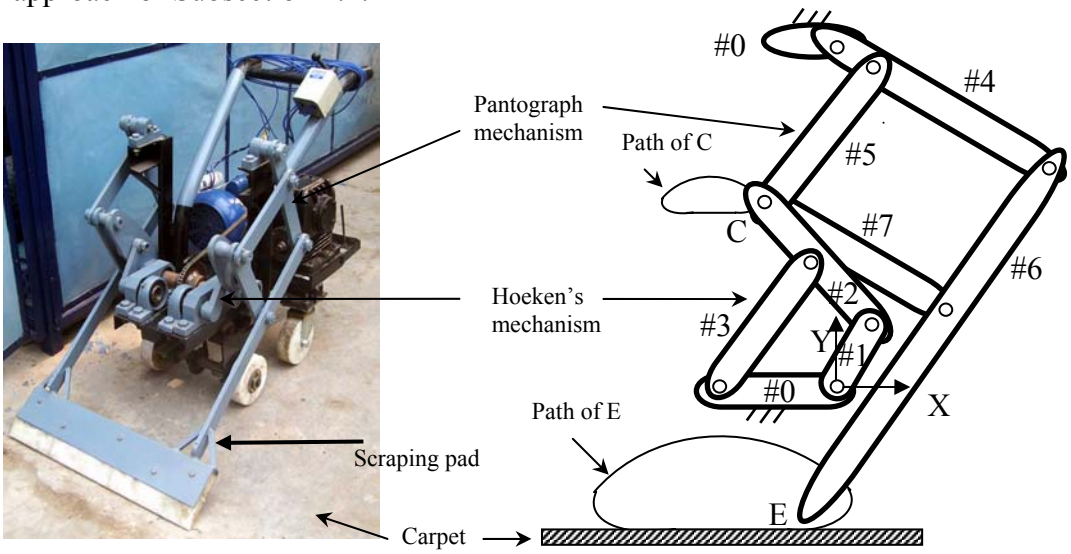
4.4 Numerical example: Carpet scrapping machine

In this Subsection example of an eight-body carpet scraping machine developed to clean a carpet after it is woven (Saha et al., 2003) is shown in Fig. 18. Two mechanisms, namely, the Hoeken's four-bar and the Pantograph, are used in this machine, as indicated in Fig. 18(b). The Hoeken's mechanism is a crank-rocker mechanism whose coupler generates a partially straight path. The straight line stroke generated by the Hoeken's mechanism is magnified by the Pantograph mechanism. The spanning tree of the mechanism is obtained by cutting the joints between links #1-#2, #2-#5, and #2-#7, of the closed-loops, #0-#1-#2-#3, #0-#1-#2-#5-#4, and #0-#1-#2-#7-#6-#4, respectively. The resulting spanning tree, Fig. 19, has three subsystems, I, II, and III. The links and joints of the subsystems are now numbered as per the scheme described in Subsection 4.1. For example, link #3 in Fig. 18 is indicated in Fig. 19 as #1 of subsystem I, i.e., link #I-1. Subsystem I has two moving links, #I-1 and #I-2, with 2-DOF. Subsystem II has one only moving link, #II-1, which is connected to its previous body, i.e., #0, at joint, II-1. Both the subsystems, I and II, are serial types, whereas subsystem III is tree-type with four moving links numbered as #III-1^0, #III-2^0, #III-3^0, and #III-1^1, which are coupled by four revolute joints denoted as, III-1^0, III-2^0, III-3^0, and III-1^1. Note that each subsystem originates from the base body, #0, which is fixed. Moreover, to avoid clumsiness in Fig. 19, the subsystems notations, I, II, and III are not used in the link and joint numbers. Furthermore, the joint angles θ_1 and θ_2 of subsystem II, θ_1 of subsystem I, and θ_1^0, θ_2^1, θ_3^0, θ_1^1 of subsystem III are treated as generalized coordinates. The input motion is provided to joint 1 of the subsystem, II, by applying torque τ_D, which needs to be calculated for the known motion of the mechanism. Additionally, three unknown vectors of Lagrange multipliers, $\boldsymbol{\lambda}_i$ for i=1, 2, 3, as indicated in Fig. 18, are,

$$\boldsymbol{\lambda}_1 = [\lambda_{1x} \quad \lambda_{1y}]^T,\ \boldsymbol{\lambda}_2 = [\lambda_{2x} \quad \lambda_{2y}]^T, \text{ and } \boldsymbol{\lambda}_3 = [\lambda_{3x} \quad \lambda_{3y}]^T.$$

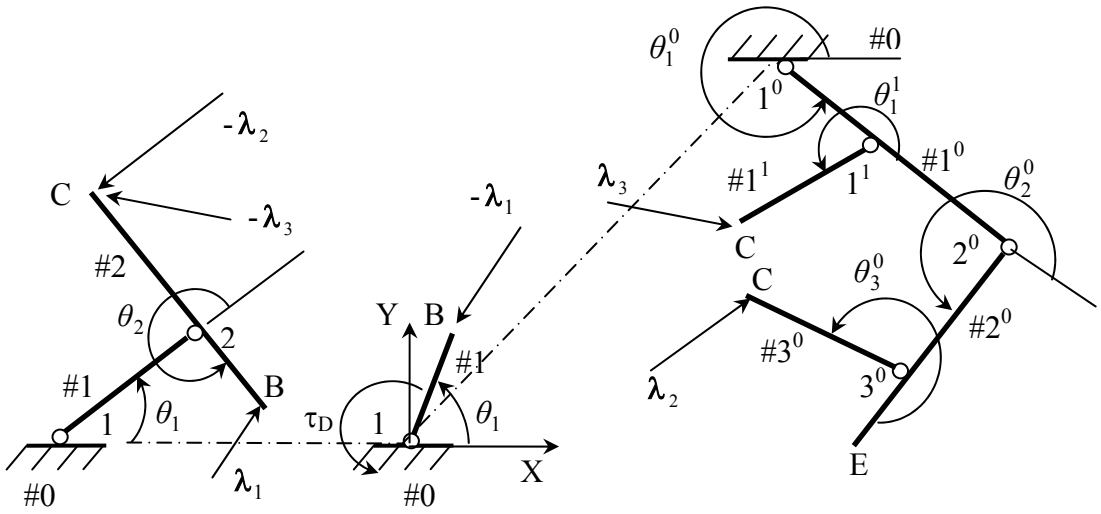
Being the motion of the mechanism planar, the two components for each Lagrange multiplier, represent the reaction forces at the cut revolute joints. Hence, the total number of scalar unknowns is 7, namely, λ_{1x}, λ_{1y}, λ_{2x}, λ_{2y}, λ_{3x}, λ_{3y}, and τ_D. Note that the DOF of the spanning tree (subsystems, I, II, and III, plus

the base body) is also seven. Hence, the spanning tree is determinate. Next, a determinate subsystem is sought. Note that the subsystem, III, has four unknowns, λ_{2x}, λ_{2y}, λ_{3x}, λ_{3y} and four DOF. Hence, it is determinate, and one can solve for the four unknowns using the four constrained equations of motion for the subsystem, III. The associated matrix size is 4×4. In the next step, evaluated λ_{2x}, λ_{2y}, λ_{3x}, λ_{3y} are taken as known external forces to the subsystem, I, that make it determinate, where λ_{1x}, λ_{1y} are the unknowns and the DOF is two. Hence, the two unknowns can be solved using the constrained equations of motion for subsystem I. Now, only τ_D remains as the unknown in subsystem II, which can be solved by one constrained equation of motion of the subsystem. The above three steps correspond to the subsystem approach of Subsection 4.2.



(a) Photograph (b) Multiloop mechanism

Figure 18 Carpet scraping machine



Subsystem I Subsystem II Subsystem III

Figure 19 Subsystems of spanning tree for the carpet scraping machine

For the multiloop scraping mechanism shown in Fig. 18(b), the physical parameters are shown in Table 5. They are used here to find the inverse dynamics result, namely, the driving torque of the mechanism.

The input motion provided to link #II-1 is a constant speed of 45 *rpm* (4.712 *rad/s*). The fixed frame, *XYZ*, is located at joint II-1, Fig. 19, where axis Z is perpendicular to the page. Joints I-1 and III-1^0 are located at (-0.089*m*, 0) and (0.038*m*, 0.410*m*), respectively. Joint between #I-1 and #II-2 is located at the mid of link #II-2. Joint III-1^1 is at 0.096*m* on link #III-1^0 from joint III-1^0. The result is shown in Fig. 20, which is compared with that obtained from the model developed using the commercial software, MSC.ADAMS 2005 (Automated Dynamic Analysis of Mechanical Systems). As per as the computational complexity of the present inverse dynamics algorithm is concerned, it is in the order of $O(4^3/3+3^3/3+1)$, compared to $O(21^3/3)$ using the traditional approach, i.e., the one based on the uncoupled NE equations of motion. Note here that no forward dynamics algorithm is presented here for the general closed-loop systems. However, it can be anticipated that the order of computational complexity for such algorithm for the carpet scrapping machine will also be $O(4^3/3+3^3/3+1)$, as the inversion of the associated inertia matrices would be involved. This implies that the order of computation is sum of the cubes of the number of bodies in all the subsystems, whereas, for a serial-chain system, it is proportional to the number of the bodies in the chain.

Table 5 Link parameters of the scraping machine

Subsystem	Link	Length (m)	Mass (kg)	Moment of inertia @ link origin (kg-m^2)
I	1	$a_{1,2}$ =0.115	3.0	0.0133
	2	$a_{2,B}$ =0.115	5.0	2.212×10^{-2}
II	1	$a_{II\text{-}1,II\text{-}2}$ =0.038	1.5	7.258×10^{-4}
III	1^0	a^0_{12} =0.335	4.2	0.157
	2^0	a^0_{23} =0.239	10.5	2.449
	3^0	$a^0_{3,c}$ =0.239	3.0	0.057
	1^1	$a^1_{1,B}$ =0.239	3.0	0.057

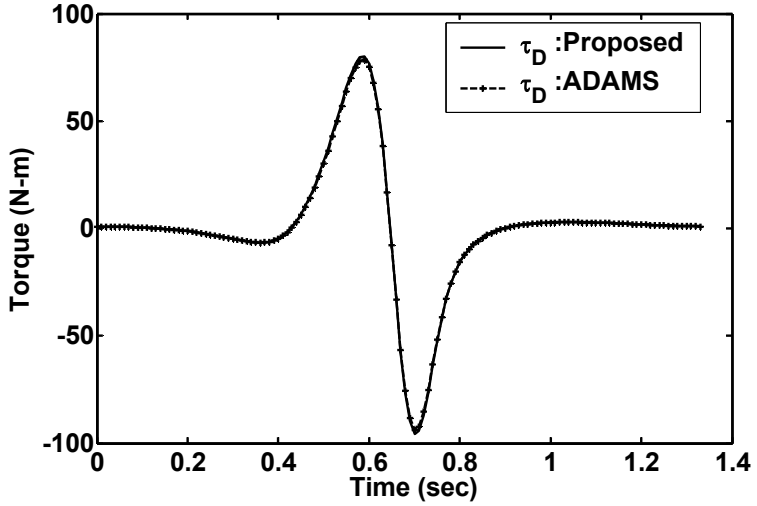


Figure 20 Driving torque for the scrapping machine

5 Conclusions

This paper presents several recursive algorithms for serial, parallel, and general closed-chain multibody systems. Whereas both inverse and forward dynamics results for the first two categories were presented

the inverse dynamics results for the last category was presented. From the computational complexity point of view, these algorithms are very efficient when the number of bodies in a multibody system is large. Furthermore, if the numerical stability aspect, particularly, for realistic rendering during computer simulation, is considered these algorithms also perform well.

Acknowledgements

The author acknowledges some of the figures, tables, results, and texts, mainly, in Section 4, by Dr. Himanshu Chaudhary, which were generated during his Ph. D thesis work under the supervision of the author.

References

Anderson, K.S., and Critchley, J. H., 2003, "A Generalized Recursive Coordinate Reduction Method for Multibody System Dynamics," Int. Journal for Computational Engineering, V. 1, N. 2&3, pp. 181-199.

Angeles, J., 2003, Fundamentals of Robotic Mechanical Systems, Second Edition, Springer-Verlag, New York.

Angeles, J. and Lee, S., 1988, "The Formulation of Dynamical Equations of Holonomic Mechanical Systems Using a Natural Orthogonal Complement," ASME Journal of Applied Mechanics, V. 55, N. 1, pp. 243-244.

Angeles, J., and Ma, O., 1988, "Dynamic simulation of n-axis serial robotic manipulators using a natural orthogonal complement," Int. J. Rob. Res., V. 7, N. 5, pp. 32-47.

Angeles, J., Ma, O., and Rojas, A., 1989, "An algorithm for the inverse dynamics of n-axis general manipulator using Kane's formulation of dynamical equations," Computers and Mathematics with Applications, V. 17, N. 12, pp. 1545-1561.

Armstrong, W.W., 1979, "Recursive solution to the equations of motion of an n-link manipulator," Proc. 5th World Cong. on Th. Mach. and Mech. (ASME), Montreal, Canada, V. 2, pp. 1343-1346.

Ascher, U.M., Pai, D.K., and Cloutier, B.P., 1997, "Forward dynamics, elimination methods, and formulation sti_ness in robot simulation," Int. J. Rob. Res., V. 16, N. 6, pp. 749-758.

Bae, B., and Haug, E.J., 1987a, "A recursive formulation for constrained mechanical system dynamics: Part I. Open loop systems," Mech. Struct. & Mach., V. 15, N. 3, pp. 359-382.

Bae, D., and Haug, E.J., 1987b, "A recursive formulation for constrained mechanical system dynamics: Part II. Closed loop systems," Mech. Struct. & Mach., V. 15, N. 4, pp.481-506.

Bae, D., Han, J.M., and Yoo, H.H., 1999, "A generalized recursive formulation for constrained mechanical system dynamics," Mech. Struct. & Mach., V. 27, N. 3, pp. 293-315.

Bhangale, P.P., Saha, S.K., and Agrawal, V.P., 2004, "A dynamic model based robot arm selection criterion," Int. J. of Multibody System Dynamics, V. 12, N. 2, pp. 95-115.

Blajer, W., Schiehlen, W., and Schirm, W., 1993, “Dynamic analysis of constrained multibody systems using inverse kinematics," Mech. and Mach. Th., V. 28, N. 3, pp. 397-405.

Brauchli, H., and Weber, R., 1991, “Dynamical equations in natural coordinates," Computer Meth. in Appl. Mech. and Eng., V. 91, pp. 1403-1414.

Chaudhary, H., and Saha, S.K., 2007, “Constraint wrench formulation for closed-loop systems using two-level recursions,” ASME J. of Mechanical Design, V. 129, Dec., pp. 1234-1242.

Craig, J.J., 1986, Introduction to Robotics: Mechanics and Control, Addison-Wesley. Singapore.

Cyril, X., 1988, Dynamics of Flexible-Link Manipulators, Ph. D thesis, McGill University, Canada.

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

Featherstone, R., 1983, “The calculation of robot dynamics using articulated-body Inertias," Int. J. Rob. Res., V. 2, N. 1, pp. 13-30.

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

Fijany, A., Sharf, I., and D'Eleuterio, M.T.D., 1995, “Parallel O(logN) algorithms for computation of manipulator forward dynamics, IEEE Trans. on R&A, V. 11., N. 3.

Ghorbel, F., Chetelat, O., and Longchamp, R., 1994, “A reduced model for constrained rigid bodies with application to parallel robots," IFAC Robot Control (Proc. 4th Symp. On Robot Control, Capri, Italy), pp. 45-50.

Hiller, M., 1995, “Multiloop kinematic chains," Kinematics and Dynamics of Multi-body Systems (Ch. 4), in J. Angeles and A. Kecskemethy (editors), Springer-Verlag, New York.

Hollerbach, J.M., 1980, “A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity," IEEE Trans. on Sys., Man, and Cybernatics, V. SMC-10, pp. 730-736.

Huston, H., and Passerello, C.E., 1974, “On constraint equations--A new approach," ASME J. Appl. Mech., V. 41, pp. 1130-1131.

Kamman, J.W., and Huston, R.L., 1984, “Dynamics of constrained multibody systems," ASME J. Appl. Mech., V. 51, Dec., pp. 899-903.

Kane, T.R., and Levinson, D.A., 1983, “The use of Kane's dynamical equations for robotics," Int. J. Rob. Res., V. 2, N. 3, pp. 3-21.

Khan, W.A., Krovi, V.N., Saha, S.K., and Angeles, J., 2005, “Recursive kinematics and inverse dynamics for a planar 3R parallel manipulator,” ASME Journal of Dynamic Systems, Measurement and Control, V. 27, N. 4, pp. 529-536.

Luh, J.Y.S., Walker, M.W., and Paul, R.P.C., 1980, “On-line computational scheme for mechanical manipulators," ASME J. Dyn. Sys., Meas., and Cont., V. 102, pp.69-76.

McPhee, J. J., 1996, “On the use of linear graph theory in multibody system dynamics,” Nonlinear Dynamics, V. 9, pp. 73-90.

Mohan, A., and Saha, S.K., 2007, “A recursive, numerically stable, and efficient simulation algorithm for serial robots,” Int. J. of Multibody System Dynamics, V. 17, N. 4, May, pp. 291-319.

Nikravesh, P. E. and Gim, G., 1993, “Systematic construction of the equations of motion for multibody systems containing closed kinematic loops,” Journal of Mechanical Design, V. 115, pp. 143-149.

Pratap, R., 2002, MATLAB 6: A Quick Introduction for Scientists and Engineers, Oxford University Press, New York.

Press, W.H., Teukolsky, S.A., Vellerling, W.T., and Flannery, B.P., 1997, Numerical Recipes in C, Cambridge University Press, 2nd Ed., New Delhi.

Rodriguez, G., 1987, “Kalman filtering, smoothing, and recursive robot arm forward and inverse dynamics," IEEE Trans. on R&A, V. RA-3, N. 6, pp. 624-639.

Rodriguez, G., and Kreutz-Delgado, K., 1992, “Spatial operator factorization and Inversion of the manipulator mass matrix," IEEE Trans. on R&A, V. 8, N. 1, pp. 65-76.

Saha, S.K., 1997, “A decomposition of the manipulator inertia matrix,” IEEE Trans. on R&A, V. 13, N. 2, Apr., pp. 301-304.

Saha, S.K., 1999a, “Dynamics of Serial Multibody Systems Using the Decoupled Natural Orthogonal Complement Matrices,” ASME Journal of Applied Mechanics, V. 66, pp. 986-996.

Saha, S.K., 1999b, “Analytical Expression for the inverted inertia matrix of serial robots," Int. J. of Rob. Res., V. 18, N. 1, Jan., pp.116-124.

Saha, S.K., and Angeles, J., 1991, “Dynamics of nonholonomic mechanical systems using a natural orthogonal complement," ASME J. Appl. Mech., V. 58, Mar., pp. 238-243.

Saha, S.K., and Schiehlen, W.O., 2001, “Recursive kinematics and dynamics for closed loop multibody systems,” Int. J. of Mechanics of Structures and Machines, V. 29, N. 2, pp.143-175.

Saha, S. K., Prasad, R., and Mandal, A.K., 2003, “Use of Hoeken’s and Pantograph mechanisms for Carpet Scraping Operations,” Proc. of 11th Nat. Conf. On Machines and Mechanisms, Dec. 18-19, IIT Delhi, pp. 732-738.

Saha, S.K., Shirinzadeh, B., and Alici, G., 2006, “Dynamic model simplification of serial manipulators,” CD-Proc. of the Int. Symp. on Robotics and Automation, San Miguel Regla Hotel, Hgo, Mexico, Aug. 25-28, pp. 14-19.

Schiehlen, W., 1990, “Multibody systems and robot dynamics," RoManSy 8 (Proc. 8th CISM-IFToMM Symp. on Theory and Practice of Robots and Manipulators), Ed: A. Morecki, et. al., Warsaw Univ. of Tech. Publ., Warsaw, pp. 14-21.

Schielen, W., 1991, “Computational aspects in multibody system dynamics," Computer Methods in Applied Mechanics and Engineering, V. 90, N. 1-3, pp. 569-582.

Sciavicco, L, and Siciliano, B., 1996, Modeling and Control of Robot Manipulators, McGraw-Hill, New York.

Shabana, A. A., 1994, Computational Dynamics, John Wiley & Sons, Inc., New York

Stejskal, V., and Valasek, M., 1996, Kinematics and Dynamics of Machinery, Marcel Dekker, Inc., New York.

Stewart, G.E., 1973, Introduction to Matrix Computations, Academic Press, Inc., New York.

Walker, M.W., and Orin, D.E., 1982, “Efficient dynamic computer simulation of robotic mechanisms," ASME J. Dyn. Sys., Meas., and Cont., V. 104, Sept., pp. 205-211.

Wehage, R.A., and Haug, E.J., 1982, “Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems," ASME J. Mech. Des., V. 104, pp. 247-255.