Mechanism and Machine Theory 49 (2012) 234–255

Contents lists available at SciVerse ScienceDirect

Mechanism and Machine Theory

journal homepage: www.elsevier.com/locate/mechmt

Modular framework for dynamic modeling and analyses of legged robots

S.V. Shah, S.K. Saha *, J.K. Dutt

Department of Mechanical Engineering Indian Institute of Technology Delhi, New Delhi 110016, India

ARTICLE INFO

Article history:
Received 9 August 2010
Received in revised form 3 October 2011
Accepted 6 October 2011
Available online 26 November 2011

Keywords:
Legged robots
Kinematic modules
Modular framework
Recursive dynamics

ABSTRACT

In this paper, a legged robot is modeled as a floating-base tree-type system where the foot-ground interactions are represented as external forces and moments. Dynamic formulation thus obtained is independent of the configuration or state of the legged robot. Framework for dynamic modeling is proposed with the concept of kinematic modules, where each module is a set of serially connected links. Legged robots are then considered to have several kinematic modules, and kinematic constraints among these modules are obtained in a similar way as those between the links. The latter approach turns out to be a special case of the former where each module has only one link. A velocity transformation based approach is used to obtain the minimal-order equations of motion, and module-level analytical expressions for the vectors and matrices appearing in them. Recursive algorithms for inverse and forward dynamics are proposed by using inter- and intra-modular recursions for the first time. Analyses of a planar biped and spatial quadruped are presented using the proposed methodology. Effectiveness of the proposed algorithms to model-based control schemes is also provided.

Nomenclature

Notational rules followed in this paper are as follows:

- Matrices are in boldface upper-case letters.
- Column matrices, algebraic vectors and arrays are in boldface lower-case letters.
- Scalars are in light face letters.
- Entity corresponding to k^{th} link is denoted as $\mathbf{v}_k$, whereas an entity corresponding to i^{th} module is denoted as $\mathbf{v}^i$

Latin letters

$\mathbf{A}$	$6(n+1)\times 6(n+1)$ matrix
$\mathbf{A}_{k,k\text{-}1}$	6×6 link-twist propagation matrix from $(k\text{-}1)^{\text{th}}$ link to k^{th} link
$\mathbf{A}^{i,h}$	$6\eta^i\times 6\eta^h$ module-twist propagation matrix from h^{th} module to i^{th} module
$\mathbf{a}_{k,\,k\text{-}1}$	3-dimensional position vector form the origin of k^{th} link to the origin of $(k\text{-}1)^{\text{th}}$ link
$\tilde{\mathbf{a}}_{k,k-1}$	3×3 skew-symmetric matrix associated with vector $\mathbf{a}_{k,k\text{-}1}$
$\mathbf{C}$	$(n+n_0)\times(n+n_0)$ Matrix of Convective Inertia (MCI) terms
$\mathbf{D}$	$(n+n_0)\times(n+n_0)$ block diagonal matrix arising out of the $\mathbf{UDU}^T$ decomposition of the inertia matrix
$\mathbf{d}_k$	3-dimensonal vector from the origin of the k^{th} link to its center-of-mass
$\tilde{\mathbf{d}}_k$	3×3 skew-symmetric matrix associated with vector $\mathbf{d}_k$
$\mathbf{E}_k$, $\mathbf{E}^i$, $\mathbf{E}$	6×6 coupling matrix for k^{th} link, $6\eta^i\times 6\eta^i$ coupling matrix for i^{th} module, and $6(n+1)\times 6(n+1)$ generalized coupling matrix, respectively.
$\mathbf{e}_k$	3-dimensional unit vector parallel to the axis of rotation or translation
$\mathbf{f}_k$	3-dimensional vector of force applied at origin of the k^{th} link

* Corresponding author. Tel.: +91 11 26591135; fax: +91 11 26582053.
E-mail addresses: surilvshah@gmail.com (S.V. Shah), saha@mech.iitd.ac.in (S.K. Saha), jkdutt@mech.iitd.ac.in (J.K. Dutt).

0094-114X/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mechmachtheory.2011.10.006

$\mathbf{g}$	3-dimensional vector of gravitational acceleration
h_k, $\mathbf{h}^i$, $\mathbf{h}$	Scalar generalized forces due to Coriolis, centrifugal, gravity, external forces and moments, η^i-dimensional vector of h_k for the i^{th} module, and $(n+n_0)$-dimensional vector of $\mathbf{h}^i$ terms, respectively.
$\tilde{h}_k$, $\tilde{\mathbf{h}}^i$	Scalar, and η^i-dimensional vector, respectively.
$\mathbf{I}$, $\mathbf{I}^{ij}$	$(n+n_0)\times(n+n_0)$ Generalized Inertia Matrix (GIM), and $(i,j)^{\text{th}}$ block element of $\mathbf{I}$ with size $\eta^i\times\eta^j$
$\mathbf{I}_k$	3×3 inertia tensor of the k^{th} link
$\hat{\mathbf{I}}^i$	$\eta^i\times\eta^i$ inertia matrix of the i^{th} articulated module (block diagonal element of $\mathbf{D}$)
k_p, k_d and $\mathbf{K}_p$, $\mathbf{K}_d$	Scalar proportional and derivative gains and matrices of proportional and derivative gains.
k^i	k^{th} link of the i^{th} module
m_k	Mass of the k^{th} link
$\hat{m}_k$	Mass moment of inertia of articulated body k (scalar diagonal element of $\mathbf{D}$)
M_i	i^{th} module
$\mathbf{M}_k$, $\mathbf{M}^i$, $\mathbf{M}$	6×6 mass matrix for k^{th} link, $6\eta^i\times6\eta^i$ mass matrix for i^{th} module, and $6(n+1)\times6(n+1)$ generalized mass matrix, respectively
$\tilde{\mathbf{M}}_k$,$\hat{\mathbf{M}}_k$	6×6 mass matrices of k^{th} composite- and articulated-body, respectively
$\tilde{\mathbf{M}}^i$,$\hat{\mathbf{M}}^i$	$6\eta^i\times6\eta^i$ mass matrices of i^{th} composite- and articulated-module, respectively
$\mathbf{N}$	$6(n+1)\times(n+n_0)$ Natural Orthogonal Complement (NOC) matrix
$\mathbf{N}^i$	$6\eta^i\times\eta^j$ module-joint-rate propagation matrix associated with i^{th} module
$\mathbf{N}_I$, $\mathbf{N}_d$	$6(n+1)\times6(n+1)$ and $6(n+1)\times(n+n_0)$ Decoupled NOC (DeNOC) matrices
n	DOF of all the modules excluding the base module
n_0	DOF of the base module (i.e., floating base), $n_0=3$ for planar case and $n_0=6$ for spatial case
$\mathbf{n}_k$	3-dimensional vector of moments applied at origin of k^{th} link
O_k	Origin of the k^{th} link
$\dot{\mathbf{o}}_k$	3-dimensional linear velocity vector for the origin of the k^{th} link
$\mathbf{p}_k$	6-dimensional joint-motion propagation vector of the k^{th} joint
$\mathbf{q}$, $\dot{\mathbf{q}}$	$(n+n_0)$-dimensional vectors of independent coordinates, and their time rates, respectively
$\mathbf{q}_0$	n_0-dimensional independent coordinates of the floating-base
s	Total number of modules (excluding base module) in a system
$\mathbf{t}_k$, $\mathbf{t}^i$	6-dimesional vector of twist for k^{th} link, and $6\eta^i$-dimesional vector of module-twist for i^{th} module.
$\mathbf{U}$, $\mathbf{U}^{ij}$	$(n+n_0)\times(n+n_0)$ block upper-triangular matrix and $\eta^i\times\eta^j$ block element of $\mathbf{U}$, respectively.
$\mathbf{w}_k$, $\mathbf{w}^i$, $\mathbf{w}$	6-dimesional vector of wrench for the k^{th} link, $6\eta^i$-dimesional vector of wrench for the i^{th} module, and $6(n+1)$-dimensional vector of generalized wrench, respectively.
$\mathbf{w}^F$, $\mathbf{w}^C$, $\mathbf{w}^E$	$6(n+1)$-dimensional generalized wrenches due to external, constraint, and driving moments and forces, respectively
$\mathbf{0}$, $\mathbf{O}$	Vector and matrix of zeros, respectively, of compatible dimensions
$\mathbf{1}$	Identity matrix of compatible dimensions
Greek letters	
β_k, β^i	Parent of k^{th} link and parent of i^{th} module, respectively.
$\boldsymbol{\gamma}^i$	Array of all the modules upstream from and including the module i
$\boldsymbol{\eta}_k$, $\tilde{\boldsymbol{\eta}}_k$ and $\boldsymbol{\eta}^i$, $\tilde{\boldsymbol{\eta}}^i$	6-dimensional and $6\eta^i$-dimensional vectors, respectively
η^i	Total number of links in the i^{th} module
$\dot{\theta}_k$	k^{th} joint rate
$\boldsymbol{\kappa}_k$, $\tilde{\mathbf{K}}^i$	6-dimensional vector and $6\eta^i\times\eta^i$ matrix, respectively
$\boldsymbol{\mu}_k$, $\tilde{\boldsymbol{\mu}}_k$ and $\boldsymbol{\mu}^i$, $\tilde{\boldsymbol{\mu}}^i$	6-dimensional and $6\eta^i$-dimensional vectors, respectively
$\boldsymbol{\xi}^i$	Array of children of i^{th} module
τ_k, $\boldsymbol{\tau}^i$, $\boldsymbol{\tau}$	Torque or force at k^{th} joint, η^i-dimensional vector of joint torques and forces for i^{th} module, and $(n+n_0)$-dimensional vector of generalized forces and torques, respectively
φ_k, $\boldsymbol{\varphi}^i$, $\boldsymbol{\varphi}$	Scalar generalized non-inertial forces, η^i-dimensional vector of φ_k terms for the i^{th} module, and $(n+n_0)$-dimensional generalized vector of non-inertial forces, respectively
$\hat{\varphi}_i$, $\bar{\varphi}_i$ and $\hat{\boldsymbol{\varphi}}^i$, $\bar{\boldsymbol{\varphi}}^i$	Scalars, and η^i-dimensional vectors, respectively
$\boldsymbol{\psi}_k$,$\hat{\boldsymbol{\psi}}_k$, and $\boldsymbol{\psi}^i$,$\hat{\boldsymbol{\psi}}^i$	6-dimensional vectors, and $6\eta^i\times\eta^i$ matrices, respectively
$\boldsymbol{\omega}_k$	3-dimensional vector of angular velocity of the k^{th} link
$\tilde{\boldsymbol{\omega}}_k$	3×3 skew-symmetric matrix associated with $\boldsymbol{\omega}_k$
$\boldsymbol{\Omega}_k$, $\boldsymbol{\Omega}^i$, $\boldsymbol{\Omega}$	6×6 angular velocity matrix for the k^{th} link, $6\eta^i\times6\eta^i$ angular velocity matrix for the i^{th} module, and $6(n+1)\times6(n+1)$ generalized angular velocity matrix, respectively

1. Introduction

Research in the field of legged robots has made great stride in the last decade and is one of the active fields of robotics. Trajectory planning, dynamics and control form the basic framework for analysis of walking and running legged robots [1—12]. Fundamentally, legged robots are floating-base, variable-constraint, and tree-type systems moving with high joint acceleration. Thus, dynamics plays a vital role in their control. In recent times, there has been significant increase in the use of computational dynamics for simulation, analysis, design, and control of various robotic systems. This is mainly due to better comprehension of the dynamic formulation of increasingly complex systems. Efficient dynamic formulation is an essential part of any computational framework for analysis of legged robots. Proper modeling of foot-ground contact [13,14] is another important step in dynamic formulation. Contact may be defined using a hard constrained model, but doing so increases the number of constraint forces involved. This results into variable closed-open tree-type system for different combinations of foot-ground interactions. This is referred to as configuration-dependent approach. Another way to define contact is to use a compliant model. In this method, ground reactions are computed from feet positions and velocities, where the ground is represented as a system that can exert vertical conservative and dissipative forces (represented by a combination of spring and dashpot), and horizontal frictional forces. This is referred to as the configuration-independent approach. The latter method is generally preferred as it allows for a single set of dynamic equations of motion for the legged robots irrespective

of their phases, e.g., flight and single, double or multi supports in stance. Furthermore, by varying the parameters of the ground model, a variety of walking and running surfaces can be simulated. The dynamic formulation proposed in this paper is configuration independent and will be applicable to any type of legged robots.

Typically, Euler-Lagrange (EL) and Newton-Euler (NE) equations of motion are used for dynamic modeling of a mechanical system. With the advent of digital computers, various other recursive and non-recursive formulations [15—26] have emerged. Recursive formulations are attractive due to their simplicity and computational uniformity regardless of how complex the system is. In the field of robotics, recursive formulations have helped in achieving real-time computations and consequently different model-based control laws have emerged [27,28]. Saha [21] proposed one such recursive formulation for the serial-chain systems, which is based on the Decoupled Natural Orthogonal Complement (DeNOC) matrices and theories of linear algebra. The DeNOC matrices are nothing but the decoupled form of a velocity transformation matrix, used to eliminate constraint forces and moments from the NE equations of motion. Several other recursive formulations were also proposed in the literature [22—26]. For example, Featherstone [22] introduced the concept of articulated-body inertia from the NE equations of motion of a free-body interacting with its neighboring bodies, whereas Bae and Haug [23] used the variational principle to obtain recursive dynamics algorithm. In both the approaches, recursion was obtained from the velocity and acceleration relationships of the neighboring bodies. Rodriguez [24] on the other hand presented a complex algorithm based on Kalman filtering and smoothing theory. Later, Rodriguez et al. [25] introduced the concept of spatial operator algebra for recursive dynamics. Rosenthal [26] also presented a recursive formulation based on Kane's equations of motion. Amongst all the above recursive algorithms, the DeNOC-based methodology showed some unique advantages when applied to serial systems [29]. They are as follows:

- Analytical decomposition [21] of the Generalized Inertia Matrix (GIM) using the simple linear algebra approach of Gaussian elimination.
- Analytical expressions for the elements of the vectors and matrices appearing in the equations of motion [29].
- Uniform development of recursive algorithms [29] for inverse and forward dynamics.
- Explicit inversion of the GIM [30], which gives deeper insight into the associated dynamics.
- Recursive kinematics and dynamics of closed-chain systems [31].
- Dynamic model simplification and design of robotic manipulators [32].
- Efficient representation of a system with multiple Degrees-of-Freedom (DOF) joints [33].

The above advantages motivated the present authors to use the DeNOC-based approach for the tree-type legged robots. However, in contrast to previous work the proposed methodology uses the concept of kinematic modules, which encapsulates the work proposed in [21—26]. In this paper, legged robots are considered as a floating-base tree-type system consisting of a set of kinematic modules, where each module contains serially connected rigid links. Kinematic relationships are then obtained between the adjoining modules. This helps in introducing the concepts of module-twist, module-twist-propagation matrix, module-DeNOC matrices, etc. A modular framework for dynamics is then obtained using the module-DeNOC matrices. Empowered with the modular framework, algorithms have been developed to obtain inverse and forward dynamics results using inter- and intra-modular recursions. Study of forward dynamics enables one to predict the future state of the system from its present state, whereas the inverse dynamics studies are used to obtain model-based control laws. In order to show the effectiveness of the proposed concepts, dynamic analyses of a planar biped and spatial quadruped are presented in this paper.

The paper is organized as follows: Section 2 presents the concept of kinematic modules and the derivations for inter- and intra-modular kinematic constraints. Dynamic formulation is presented in Section 3, whereas its applications are given in Section 4. Numerical examples are given in Section 5, followed by the conclusions in Section 6.

2. Kinematic modules

Conventionally, a serial or a tree-type system is considered to have a set of links or bodies connected by kinematic pairs. However, in this work a generic approach is proposed where serial or tree-type architecture is considered to have a set of kinematic modules, where a module consists of either one or more than one serially connected links. A multi-modular tree-type system is shown in Fig. 1, where the modules are depicted by using dotted boundaries. The existing link-level approaches [21—24] turn out to be special cases of the proposed one for each module having only one link.

As a legged robot is generally mobile system, so, one of its links is considered to be a floating-base. It is referred to as module M_0 in Fig. 1. Once the floating-base is established, the system is modularized outward from it (See Fig. 1) such that each module 1) contains only serially connected links; and 2) emerges from the last link of its parent module. The modules are denoted with M_0, M_1, M_2, etc., where a child module bears a number higher than its parent module. Moreover, the links inside any module, say, M_i, are denoted as $\#1^i$, …, $\#k^i$, …, $\#\eta^i$, where the superscript i signifies the module number. The total number of modules excluding floating-base is denoted as s. The number of links in the i^{th} module and the total number of links in s modules are designated as η^i and n, respectively. Next, the kinematic constraints are derived at the intra-modular level, i.e., amongst the links inside a module.

2.1. Intra-modular level

As mentioned above, module i, for $i = 1, \ldots, s$, in a tree-type system contains only serially connected links, hence intra-modular constraints are derived first. The k^{th} link in the i^{th} module, denoted as $\#k^i$ (or $\#k$) in Fig. 2, is connected to link $\#(k\text{-}1)^i$ by a 1-DOF joint k^i. Note that the superscripts denoting the modules are omitted in this subsection to avoid clumsiness of the expression. The

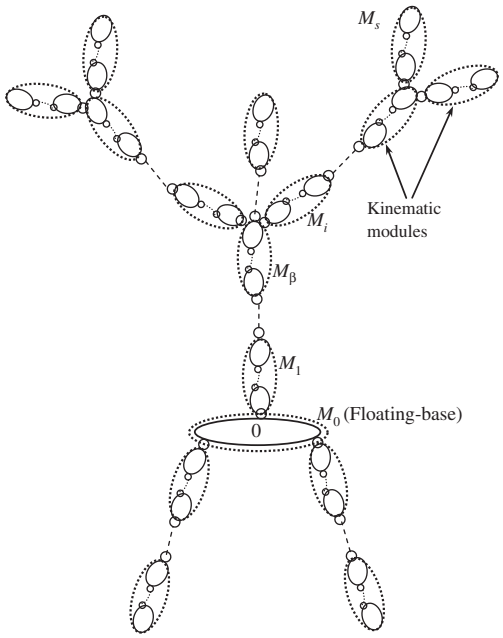


Fig. 1. A multi-modular tree-type system.

velocity constraints are then written in terms of the twist of the links. The 6-dimensional vector of twist associated with the angular velocity ($\boldsymbol{\omega}_k$) and linear velocity ($\dot{\mathbf{o}}_k$) of #k is defined as $\mathbf{t}_k \equiv [\boldsymbol{\omega}_k^T \; \dot{\mathbf{o}}_k^T]^T$. The twist is written in terms of the twist of (k-1), $\mathbf{t}_{k\text{-}1}$, as

$$\mathbf{t}_k = \mathbf{A}_{k,k-1}\mathbf{t}_{k-1} + \mathbf{p}_k\dot{\theta}_k \tag{1}$$

where $\mathbf{A}_{k,k\text{-}1}$, $\mathbf{p}_k$ and $\dot{\theta}_k$ are the 6×6 twist-propagation matrix, the 6-dimensional motion-propagation vector, and the k^{th} joint rate, respectively. The matrix $\mathbf{A}_{k,k\text{-}1}$ and vector $\mathbf{p}_k$ are given by

$$\mathbf{A}_{k,k-1} \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ \tilde{\mathbf{a}}_{k,k-1} & \mathbf{1} \end{bmatrix}, \text{and } \mathbf{p}_k \equiv \begin{bmatrix} \mathbf{e}_k \\ \mathbf{0} \end{bmatrix} (\text{for revolute pair}) \text{ or } \begin{bmatrix} \mathbf{0} \\ \mathbf{e}_k \end{bmatrix} (\text{for prismatic pair}). \tag{2}$$

In Eq. (2), $\tilde{\mathbf{a}}_{k,k-1} (= -\tilde{\mathbf{a}}_{k-1,k})$ is the 3×3 skew-symmetric matrix associated with vector $\mathbf{a}_{k\text{-}1,k}$, $\mathbf{e}_k$ is the unit vector along the axis of rotation of the k^{th} joint, '**O**' and '**1**' are the null and identity matrices of compatible sizes, and '**0**' is the null vector of

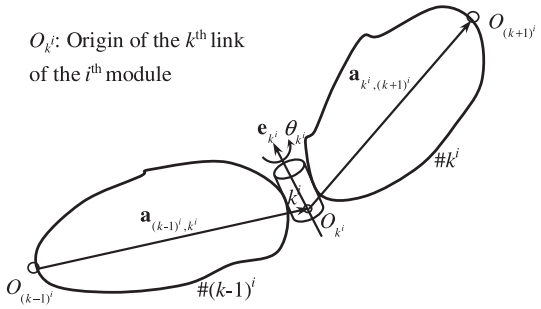


Fig. 2. The #k^{th} and #(k-1)$^{\text{th}}$ links coupled by joint k.

compatible dimension. Next for the module M_i, the $6\eta^i$-dimensional vector of module-twist ($\mathbf{t}^i$) and the η^i-dimensional vector of module-joint-rate ($\dot{\mathbf{q}}^i$) are defined as

$$\mathbf{t}^i \equiv \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_k \\ \vdots \\ \mathbf{t}_\eta \end{bmatrix}^i \text{ and } \dot{\mathbf{q}}^i \equiv \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_k \\ \vdots \\ \dot{\theta}_\eta \end{bmatrix}^i . \tag{3}$$

Substituting Eq. (1) into Eq. (3), for $k=1, \ldots, \eta^i$, the expression for the module twist is obtained as

$$\mathbf{t}^i = \mathbf{N}^i \dot{\mathbf{q}}^i \tag{4}$$

where the $6\eta^i \times \eta^i$ matrix $\mathbf{N}^i$ is given by

$$\mathbf{N}^i \equiv \begin{bmatrix} \mathbf{p}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}_{2,1}\mathbf{p}_1 & \mathbf{p}_{k-1} & \ddots & \vdots \\ \vdots & \mathbf{A}_{k,k-1}\mathbf{p}_{k-1} & \ddots & \mathbf{0} \\ \mathbf{A}_{\eta,1}\mathbf{p}_1 & \cdots & \mathbf{A}_{\eta,\eta-1}\mathbf{p}_{\eta-1} & \mathbf{p}_\eta \end{bmatrix}^i . \tag{5}$$

In Eq. (4), $\mathbf{t}^i$ actually represents the relative twist of the i^{th} module with respect to its parent module. Interestingly, the matrix $\mathbf{N}^i$ is nothing but the velocity transformation matrix or Natural Orthogonal Complement (NOC) of a serial chain [29]. It is worth noting that for module M_0, the floating-base (#0), expressions of $\mathbf{N}^0$ and $\dot{\mathbf{q}}^0$ in Eq. (4) simplify as

$$\mathbf{N}^0 = \mathbf{P}_0 \text{ and } \dot{\mathbf{q}}^0 = \dot{\mathbf{q}}_0 \tag{6}$$

where the $n_0 \times n_0$ matrix $\mathbf{P}_0$ and the n_0-dimensional vector $\dot{\mathbf{q}}_0$ ($n_0=3$ for planar case, and $n_0=6$ for spatial case) have the following representations:

$$\mathbf{P}_0 \equiv \begin{bmatrix} \mathbf{L}_0 & \mathbf{O} \\ \mathbf{O} & 1 \end{bmatrix}, \text{ and } \dot{\mathbf{q}}_0 \equiv \begin{bmatrix} \dot{\boldsymbol{\theta}}_0 \\ \mathbf{v}_0 \end{bmatrix}. \tag{7}$$

In Eq. (7), $\dot{\boldsymbol{\theta}}_0$, $\mathbf{L}_0$ and $\mathbf{v}_0$ are the rates of the independent rotation co-ordinates, the corresponding transformation matrix, and the linear velocity vector, respectively. Vector $\dot{\boldsymbol{\theta}}_0$ contains the time-rates of Euler angles or Euler-angle-joint motions [33] if the Euler angles are used to define the rotation of floating-base. The dimensions of $\mathbf{L}_0$ and $\dot{\boldsymbol{\theta}}_0$ will change if the rotation representation is done with Euler parameters.

2.2. Inter-modular level

A recursive relationship between the twists of two neighboring modules, say, M_i and its parent M_β ('β' signifies a parent) as shown in Fig. 3, is obtained next. Using Eq. (4), the module-twist $\mathbf{t}^i$ for M_i can be written in terms of the module-twist of its parent M_β, i.e., $\mathbf{t}^\beta$, as

$$\mathbf{t}^i = \mathbf{A}^{i,\beta}\mathbf{t}^\beta + \mathbf{N}^i \dot{\mathbf{q}}^i \tag{8}$$

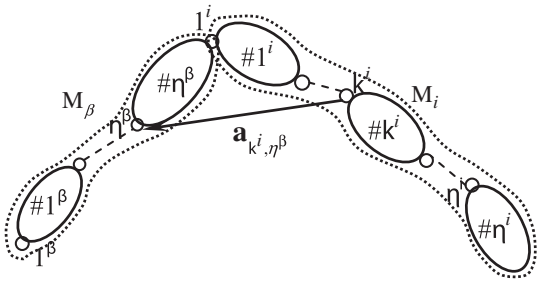


Fig. 3. Module M_i and its parent module M_β.

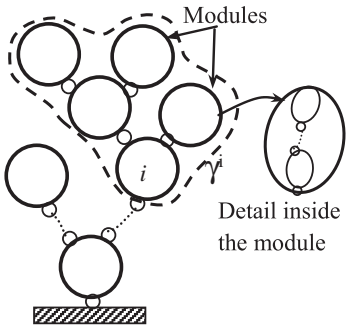


Fig. 4. Definition of $\boldsymbol{\gamma}^i$.

where the $6\eta^i \times 6\eta^\beta$ matrix $\mathbf{A}^{i,\beta}$ propagates the twist of the parent module (β^{th}) to the child module (i^{th}) and hence referred to as module-twist propagation matrix. The $6\eta^i \times \eta^i$ matrix $\mathbf{N}^i$ in Eq. (8) is nothing but the one obtained in Eq. (5) and termed here as the module-joint-rate propagation matrix. The matrix $\mathbf{A}^{i,\beta}$ is given by

$$\mathbf{A}^{i,\beta} \equiv \begin{bmatrix} \mathbf{O} & \cdots & \mathbf{O} & \mathbf{A}_{1^i,\eta^\beta} \\ \vdots & & \vdots & \vdots \\ \mathbf{O} & \cdots & \mathbf{O} & \mathbf{A}_{k^i,\eta^\beta} \\ \vdots & & \vdots & \vdots \\ \mathbf{O} & \cdots & \mathbf{O} & \mathbf{A}_{\eta^i,\eta^\beta} \end{bmatrix}. \tag{9}$$

Matrix $\mathbf{A}_{k^i,\eta^\beta}$ in the expression of $\mathbf{A}^{i,\beta}$ above denotes the twist-propagation matrix from the twist of link $\#\eta^\beta$ in module M_β to the twist of link $\#k^i$ in module M_i. It is obtained similar to $\mathbf{A}_{k,k\text{-}1}$ of Eq. (2) and is associated with the vector $\mathbf{a}_{k^i,\eta^\beta}$ as shown in Fig. 3. Eq. (8) is the generic representation of Eq. (1). Next, the generalized twist vector, $\mathbf{t}$, composed of all module-twists, and the generalized joint rate vector, $\dot{\mathbf{q}}$, composed of all module-joint-rates are defined as

$$\mathbf{t} \equiv \begin{bmatrix} \mathbf{t}^0 \\ \vdots \\ \mathbf{t}^i \\ \vdots \\ \mathbf{t}^s \end{bmatrix}, \text{ and } \dot{q} \equiv \begin{bmatrix} \dot{\mathbf{q}}^0 \\ \vdots \\ \dot{\mathbf{q}}^i \\ \vdots \\ \dot{\mathbf{q}}^s \end{bmatrix} \tag{10}$$

where each module in the tree-type system is treated similar to a link in a serial system. This analogy helps one to extrapolate many concepts of the serial systems [29] directly to the multi-modular systems. One such extraction is Eq. (8). Substituting Eq. (8) into Eq. (10), and rearranging the terms, the generalized twist vector, $\mathbf{t}$, can be expressed as

$$\mathbf{t} = \mathbf{N}_l \mathbf{N}_d \dot{\mathbf{q}} \tag{11}$$

in which $\mathbf{N}_l$ and $\mathbf{N}_d$ are the $6(n+1)\times 6(n+1)$ and $6(n+1)\times(n+n_0)$ matrices. They are given by

$$\mathbf{N}_l \equiv \begin{bmatrix} \mathbf{1}^0 & & & & \\ \mathbf{A}^{1,0} & \mathbf{1}^1 & & \mathbf{O}'s & \\ \mathbf{A}^{2,0} & \mathbf{A}^{2,1} & \mathbf{1}^2 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \mathbf{A}^{s,0} & \mathbf{A}^{s,1} & \cdots & \mathbf{A}^{s,s-1} & \mathbf{1}^s \end{bmatrix}, \quad \mathbf{N}_d \equiv \begin{bmatrix} \mathbf{N}^0 & & & \mathbf{O}'s \\ & \ddots & & \\ & & \mathbf{N}^i & \\ & & & \ddots \\ \mathbf{O}'s & & & \mathbf{N}^s \end{bmatrix} \text{ and } \mathbf{A}^{j,i} \equiv \mathbf{O}, \text{ if } M_j \notin \boldsymbol{\gamma}^i. \tag{12}$$

In Eq. (12), $\mathbf{1}^i$ is the $6\eta^i \times 6\eta^i$ identity matrix, and $\boldsymbol{\gamma}^i$ stands for a set of all modules including and outward from module M_i, as shown in Fig. 4. The matrices $\mathbf{N}_l$ and $\mathbf{N}_d$ are the module-DeNOC matrices for the floating-base tree-type system shown in Fig. 1. Under the special case of each module having only one link, matrices $\mathbf{N}^i$ and $\mathbf{A}^{i,j}$ simplify as $\mathbf{N}^i \equiv \mathbf{p}_i$ and $\mathbf{A}^{i,j} \equiv \mathbf{A}_{i,j}$ where the link index is assumed same as that of the module. Moreover, in the absence of any branching and the base being fixed, $\mathbf{N}_l$ and $\mathbf{N}_d$ degenerate to those obtained in [29], i.e.,

$$\mathbf{N}_l \equiv \begin{bmatrix} \mathbf{1} & & \mathbf{O} & \cdots & \mathbf{O} \\ & \ddots & & & \\ \mathbf{A}_{2,1} & & \mathbf{1} & & \vdots \\ & \ddots & & \ddots & \\ \vdots & & \mathbf{A}_{k,k-1} & & \mathbf{O} \\ & & & \ddots & \\ \mathbf{A}_{\eta,1} & & \cdots & \mathbf{A}_{\eta,\eta-1} & \mathbf{1} \end{bmatrix}, \text{ and } \mathbf{N}_d \equiv \begin{bmatrix} \mathbf{p}_1 & & & \mathbf{0}'s \\ & \ddots & & \\ & & \mathbf{p}_k & \\ & & & \ddots \\ \mathbf{0}'s & & & \mathbf{p}_n \end{bmatrix}. \tag{13}$$

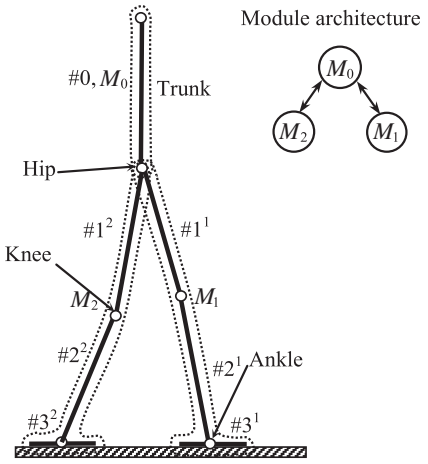


Fig. 5. A 7-link planar biped.

It is pointed out here that depending on the definition of a kinematic module the structure of the matrices $\mathbf{N}_l$ and $\mathbf{N}_d$ of Eq. (12) will vary as their representation is not unique. Moreover, under the above stated special case the expressions simplify to the DeNOC matrices of serial-chain [29]. Note that one can also write the DeNOC matrices simply with body-level recursion without defining any kinematic modules, as derived in Subsection 2.3. However, the resultant $\mathbf{N}_l$ and $\mathbf{N}_d$ matrices are not suitable for two layer algorithms (inter- and intra-levels) that makes the analysis much more elegant and comprehensive. This is one of the fundamental contributions of this paper.

2.3. An example: DeNOC matrices for a planar biped

To illustrate the concept of the module-DeNOC matrices, they are derived here for a 7-link planar biped. Trunk of the biped is assumed to be the floating-base M_0, and two legs form modules M_1 and M_2 as shown in Fig. 5. The biped has revolute joints at the hip, knee, and ankle. It has nine degrees-of-freedom that includes 2 each at hips, knees and ankles, and 3-DOF associated with floating-base, i.e., trunk. The DeNOC matrices are then obtained using Eq. (12) as

$$\mathbf{N}_l \equiv \begin{bmatrix} \mathbf{1}^0 & & \mathbf{0}'s \\ \mathbf{A}^{1,0} & \mathbf{1}^1 & \\ \mathbf{A}^{2,0} & \mathbf{0} & \mathbf{1}^2 \end{bmatrix}, \text{and } \mathbf{N}_d \equiv \begin{bmatrix} \mathbf{N}^0 & & \mathbf{0}'s \\ & \mathbf{N}^1 & \\ \mathbf{0}'s & & \mathbf{N}^2 \end{bmatrix} \tag{14}$$

where $\mathbf{N}_l$ and $\mathbf{N}_d$ are the 39×39 and 39×9 matrices, respectively. Moreover, $\mathbf{1}^0$, $\mathbf{1}^1$ and $\mathbf{1}^2$ are the 3×3, 18×18 and 18×18, identity matrices. The block elements $\mathbf{A}^{1,0}$, $\mathbf{A}^{2,0}$, $\mathbf{N}^1$, $\mathbf{N}^2$, and $\mathbf{N}^0$ are obtained using Eqs. (5), (6), (9) as

$$\mathbf{A}^{i,0} = \begin{bmatrix} \mathbf{A}_{1^i,0} \\ \mathbf{A}_{2^i,0} \\ \mathbf{A}_{3^i,0} \end{bmatrix}, \mathbf{N}^i = \begin{bmatrix} \mathbf{p}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_{2,1}\mathbf{p}_1 & \mathbf{p}_2 & \mathbf{0} \\ \mathbf{A}_{3,1}\mathbf{p}_1 & \mathbf{A}_{3,2}\mathbf{p}_2 & \mathbf{p}_3 \end{bmatrix}^i (for\ i = 1,2), \text{and } \mathbf{N}^0 = \mathbf{P}_0. \tag{15}$$

Matrices $\mathbf{A}^{1,0}$, $\mathbf{A}^{2,0}$, $\mathbf{N}^1$, $\mathbf{N}^2$, and $\mathbf{N}^0$ are of sizes 18×6, 18×6, 18×3, 18×3, and 3×3, respectively. Size of each block element of $\mathbf{N}_l$ and $\mathbf{N}_d$ depends on the total number of links and joint variables in the modules. It is interesting to note from Eqs. (14), (15) that the diagonal block elements $\mathbf{N}^i$ (i.e., $\mathbf{N}^1$ and $\mathbf{N}^2$) in the expression of $\mathbf{N}_d$ actually represent the module joint motion propagation matrix, which is nothing but the combined form of the DeNOC matrices or simply NOC matrix of the serial chain module, whereas the matrices $\mathbf{A}^{1,0}$ and $\mathbf{A}^{2,0}$ relate motion propagations between the zeroth and first, and zeroth and second modules, respectively. Thus, one can systematically correlate kinematic relationships between the modules using Eqs. (14), (15). On the contrary, the use of body-level representation for the biped using the scheme of [29] or [35] leads to the following DeNOC matrices:

$$\mathbf{N}_l \equiv \begin{bmatrix} \mathbf{1} & & & & & & \\ \mathbf{A}_{1^1,0} & \mathbf{1} & & & & & \\ \mathbf{A}_{2^1,0} & \mathbf{A}_{2^1,1^1} & \mathbf{1} & & & & \\ \mathbf{A}_{3^1,0} & \mathbf{A}_{3^1,1^1} & \mathbf{A}_{3^1,2^1} & \mathbf{1} & & & \\ \mathbf{A}_{1^2,0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & & \\ \mathbf{A}_{2^2,0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{2^2,1^2} & \mathbf{1} & \\ \mathbf{A}_{3^2,0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{3^2,1^2} & \mathbf{A}_{3^2,2^2} & \mathbf{1} \end{bmatrix}, \mathbf{N}_d \equiv \begin{bmatrix} \mathbf{P}_0 & & & & & & \mathbf{0}'s \\ & \mathbf{p}_{1^1} & & & & & \\ & & \mathbf{p}_{2^1} & & & & \\ & & & \mathbf{p}_{3^1} & & & \\ & & & & \mathbf{p}_{1^2} & & \\ & & & & & \mathbf{p}_{2^2} & \\ \mathbf{0}'s & & & & & & \mathbf{p}_{3^2} \end{bmatrix} \tag{16}$$

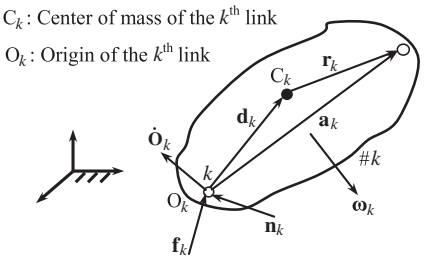


Fig. 6. Motion of the k^{th} link of the i^{th} module.

Comparing Eq. (14) with Eq. (16), it is clear that Eq. (16) does not give any information about the motion propagation amongst the modules. Every twist propagation matrix represented by $\mathbf{A}_{k^i,l^j}$ between two successive links must have information of the parent link. With representation in Eq. (14), only the parent module or in other words parent of the first link has to be known. Hence, the algorithm continues like a serial chain system reflected in the matrix $\mathbf{N}^i$ of Eq. (15). This allows the development of the algorithms in two levels, namely, inter- and intra-module levels, which is not possible with the body-level representation of the DeNOC matrices given by Eq. (16). Moreover, the vast knowledge available for serial-chain systems can be readily adopted for intra-modular recursion.

3. Dynamic formulation

For the i^{th} module, the Newton-Euler (NE) equations of motion of a module are written similar to a serial-chain system [34] as

$$\mathbf{M}^i\dot{\mathbf{t}}^i + \boldsymbol{\Omega}^i\mathbf{M}^i\mathbf{E}^i\mathbf{t}^i = \mathbf{w}^i, \textit{ where } \mathbf{w}^i \equiv \mathbf{w}^{i(E)} + \mathbf{w}^{i(C)} + \mathbf{w}^{i(F)}. \tag{17}$$

The right hand side of the equation gives the module-wrench vector, $\mathbf{w}^i$, which is composed of $\mathbf{w}^{i(\mathrm{E})}$, the wrench generated by external forces, $\mathbf{w}^{i(\mathrm{C})}$, the wrench due to constrained forces and moments, and $\mathbf{w}^{i(\mathrm{F})}$, the wrench due to link-ground interaction. The $6\eta^i \times 6\eta^i$ matrices $\mathbf{M}^i$, $\boldsymbol{\Omega}^i$ and $\mathbf{E}^i$, and the $6\eta^i$-dimensional module-wrench vector $\mathbf{w}^i$ are defined as

$$\begin{aligned} \mathbf{M}^i &\equiv diag\left[\mathbf{M}_1 \quad \cdots \mathbf{M}_k \cdots \quad \mathbf{M}_\eta\right]^i, \\ \boldsymbol{\Omega}^i &\equiv diag\left[\boldsymbol{\Omega}_1 \quad \cdots \boldsymbol{\Omega}_k \cdots \quad \boldsymbol{\Omega}_\eta\right]^i, \\ \mathbf{E}^i &\equiv diag\left[\mathbf{E}_1 \quad \cdots \mathbf{E}_k \cdots \quad \mathbf{E}_\eta\right]^i, \end{aligned} \quad and\ \mathbf{w}^i \equiv \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_k \\ \vdots \\ \mathbf{w}_\eta \end{bmatrix}^i \tag{18}$$

where $\mathbf{M}_k$, $\boldsymbol{\Omega}_k$ and $\mathbf{E}_k$ are the 6×6 matrices of angular velocity, mass and coupling, and $\mathbf{w}_k$ is the 6-dimensional wrench vector for the k^{th} link of module M_i. They are given by

$$\mathbf{M}_{k^i} \equiv \begin{bmatrix} \mathbf{I}_k & m_k\tilde{\mathbf{d}}_k \\ -m_k\tilde{\mathbf{d}}_k & m_k\mathbf{1} \end{bmatrix}^i, \boldsymbol{\Omega}_{k^i} \equiv \begin{bmatrix} \tilde{\boldsymbol{\omega}}_k & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\omega}}_k \end{bmatrix}^i, \mathbf{E}_{k^i} \equiv \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}^i, \textit{ and } \mathbf{w}_{k^i} \equiv \begin{bmatrix} \mathbf{n}_k \\ \mathbf{f}_k \end{bmatrix}^i. \tag{19}$$

In Eq. (19), $\tilde{\boldsymbol{\omega}}_k$ and $\tilde{\mathbf{d}}_k$ are the 3×3 skew-symmetric matrices associated with the 3-dimensional vectors $\boldsymbol{\omega}_k$ and $\mathbf{d}_k$, respectively. Moreover, $\mathbf{I}_k$ and m_k are the 3×3 inertia tensor about O_k and mass of the k^{th} link, respectively, and $\mathbf{n}_k$ and $\mathbf{f}_k$ are the moment about and force acting at O_k for the k^{th} link, respectively. A typical k^{th} link of the i^{th} module is shown in Fig. 6. The NE equations of motion for the $(s+1)$ modules may then be put together as

$$\mathbf{M}\dot{\mathbf{t}} + \boldsymbol{\Omega}\mathbf{M}\mathbf{E}\mathbf{t} = \mathbf{w}^E + \mathbf{w}^C + \mathbf{w}^F \tag{20}$$

where the generalized mass matrix $\mathbf{M}$, the angular velocity matrix $\boldsymbol{\Omega}$, and the coupling matrix $\mathbf{E}$, are each of size $6(n+1)\times 6(n+1)$. They are given by

$$\mathbf{M} \equiv diag\left[\mathbf{M}^0 \quad \ldots \quad \mathbf{M}^s\right], \boldsymbol{\Omega} \equiv diag\left[\boldsymbol{\Omega}^0 \quad \cdots \quad \boldsymbol{\Omega}^s\right], \textit{ and } \mathbf{E} \equiv diag\left[\mathbf{E}^0 \quad \cdots \quad \mathbf{E}^s\right]. \tag{21}$$

Pre-multiplication of Eq. (20) by $\mathbf{N}_d^T\mathbf{N}_l^T$ eliminates the constraint wrenches [19], i.e., $\mathbf{N}_d^T\mathbf{N}_l^T\mathbf{w}^C = \mathbf{0}$. As a result a minimal set of equations of motion for floating-base multi-modular tree-type system is obtained as

$$\mathbf{N}_d^T\mathbf{N}_l^T\left(\mathbf{M}\dot{\mathbf{t}} + \boldsymbol{\Omega}\mathbf{M}\mathbf{E}\mathbf{t}\right) = \mathbf{N}_d^T\mathbf{N}_l^T\left(\mathbf{w}^E + \mathbf{w}^F\right). \tag{22}$$

Substituting Eq. (11) and its time derivative into Eq. (22) leads to the following:

$$\mathbf{I}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} = \boldsymbol{\tau} + \boldsymbol{\tau}^F \tag{23}$$

where the expressions for the Generalized Inertia Matrix (GIM) $\mathbf{I}$, Matrix of Convective Inertia (MCI) term $\mathbf{C}$, vector of the generalized external force $\boldsymbol{\tau}$, and the vector of forces due to foot-ground interaction $\boldsymbol{\tau}^F$ are given by

$$\begin{aligned}&\mathbf{I}\equiv\mathbf{N}^T\mathbf{M}\mathbf{N},\ \mathbf{C}\equiv\mathbf{N}^T\left(\mathbf{M}\dot{\mathbf{N}}+\mathbf{M}\mathbf{E}\mathbf{N}\right),\ \boldsymbol{\tau}\equiv\mathbf{N}^T\mathbf{w}^E \text{ and } \boldsymbol{\tau}^F\equiv\mathbf{N}^T\mathbf{w}^F\\ &\text{where } \mathbf{N}\equiv\mathbf{N}_l\mathbf{N}_d.\end{aligned} \tag{24}$$

Eq. (23) is nothing but the Euler-Lagrange equations of motion in minimum number of co-ordinate derived from the Newton-Euler equations and associated DeNOC matrices.

The proposed formulation based on the concept of kinematic modules brings several advantages. For example, 1) Generalization of the link-to-link velocity transformation relationships to module-to-module velocity transformation relationships where the former is a special case of the latter; 2) Compact representation of the system's kinematic and dynamic model; 3) Provision of module-level analytical expressions for the matrices and vectors appearing in the equations of the motion; one such analytical expression for the GIM is provided in Eq. (A.2); 4) Ease of adopting already developed algorithms for the serial-chain systems [29] to multi-modular tree-type systems; 5) Possibility of repeating the computations of a module to other modules if the system has same type of modules, i.e., equal number of links and types of joints; 6) Ease of investigating any inconsistency in the results of a tree-type system with same type of modules; in this case, only one module has to be investigated; and (7) Hybrid recursive-parallel algorithms can be obtained where modules are solved parallelly, whereas links inside the modules are solved in recursive manner.

4. Applications

The dynamic formulation presented above facilitates to develop inter- and intra-modular recursive inverse and forward dynamics algorithms for multi-modular tree-type systems in a unified manner. They are presented next.

4.1. Inverse dynamics

The objective of inverse dynamics is to find the joint torques and forces of a legged robot for a given set of joint motions. It is worth mentioning that the legged robots under study are underactuated. For example, the planar biped shown in Fig. 5 has 9-DOF, out of which, only 6-DOF associated with hips, knees and ankles having revolute joints are actuated. The other 3-DOFs, two-translational and one-rotational coordinate, of the trunk moving in a plane are not actuated.

In order to obtain the actuated joint torques for such underactuated robots, the equations of motion in Eq. (23) are rewritten by separating the floating-base acceleration, represented by n_0-dimensional vector $\ddot{\mathbf{q}}_0$, from the joint accelerations, represented by n_0-dimnesional vector $\ddot{\mathbf{q}}_\theta$, as

$$\begin{bmatrix}\mathbf{I}_0 & \mathbf{I}_{\theta 0}^T\\ \mathbf{I}_{\theta 0} & \mathbf{I}_\theta\end{bmatrix}\begin{bmatrix}\ddot{\mathbf{q}}_0\\ \ddot{\mathbf{q}}_\theta\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\ \boldsymbol{\tau}_\theta\end{bmatrix}-\begin{bmatrix}\mathbf{h}_0\\ \mathbf{h}_\theta\end{bmatrix} \tag{25}$$

where $\mathbf{I}_0$ ($\equiv\mathbf{I}^{0,0}$) is the $n_0\times n_0$ matrix associated with the floating-base, $\mathbf{I}_\theta$ is the $n\times n$ matrix representing the remainder of the tree-type legged robot, and

$$\mathbf{I}_{\theta 0}\equiv\begin{bmatrix}\mathbf{I}^{1,0}\\ \vdots\\ \mathbf{I}^{s,0}\end{bmatrix},\ \ddot{\mathbf{q}}_0=\ddot{\mathbf{q}}^0,\ \ddot{\mathbf{q}}_\theta\equiv\begin{bmatrix}\ddot{\mathbf{q}}^1\\ \vdots\\ \ddot{\mathbf{q}}^s\end{bmatrix},\boldsymbol{\tau}\equiv\begin{bmatrix}\mathbf{0}\\ \boldsymbol{\tau}_\theta\end{bmatrix},\boldsymbol{\tau}_\theta\equiv\begin{bmatrix}\boldsymbol{\tau}^1\\ \vdots\\ \boldsymbol{\tau}^s\end{bmatrix},\ \begin{bmatrix}\mathbf{h}_0\\ \mathbf{h}_\theta\end{bmatrix}\equiv\mathbf{h}\equiv\mathbf{C}\,\dot{\mathbf{q}}-\boldsymbol{\tau}^F,\mathbf{h}_0\equiv\mathbf{h}^0 \text{ and } \mathbf{h}_\theta\equiv\begin{bmatrix}\mathbf{h}^1\\ \vdots\\ \mathbf{h}^s\end{bmatrix}.$$

Next, the inverse dynamics problem for the legged robot is formulated using Eq. (25) as

$$\ddot{\mathbf{q}}_0=-\mathbf{I}_0^{-1}\tilde{\mathbf{h}}_0,\ \text{where } \tilde{\mathbf{h}}_0\equiv\mathbf{I}_{\theta 0}^T\ddot{\mathbf{q}}_\theta+\mathbf{h}_0 \tag{26}$$

$$\boldsymbol{\tau}_\theta=\mathbf{I}_{\theta 0}\ddot{\mathbf{q}}_0+\tilde{\mathbf{h}}_\theta,\ \text{where } \tilde{\mathbf{h}}_\theta\equiv\mathbf{I}_\theta\mathbf{q}_\theta+\mathbf{h}_\theta. \tag{27}$$

Therefore, inverse dynamics of legged robots involves calculation of the acceleration of floating-base and joint torques. The steps to compute $\ddot{\mathbf{q}}_0$ and $\boldsymbol{\tau}_\theta$ are shown below:

Step 1: Find $\mathbf{t}, \mathbf{t}'$, *and* $\mathbf{w}=\left(\mathbf{M}\mathbf{t}'+\boldsymbol{\Omega}\mathbf{M}\mathbf{E}\mathbf{t}\right)-\mathbf{w}^F$

In this step, $\mathbf{t}^i, \mathbf{t}^{i\prime}$, *and* $\mathbf{w}^i$ (for $i=0, \ldots, s$) are obtained recursively for each module starting from M_0. It is worth noting that $\mathbf{t}'$ is the generalized twist-rate vector $\mathbf{t}$ while $\ddot{\mathbf{q}}_0=\mathbf{0}$.

Step 2: Find $\tilde{\mathbf{h}}=\left[\tilde{\mathbf{h}}_0^T\ \tilde{\mathbf{h}}_\theta^T\right]^T=\mathbf{N}_d^T\mathbf{N}_l^T\mathbf{w}, \mathbf{I}_0$ and $\mathbf{I}_{\theta 0}$

Having the results from Step 1, this step computes $\tilde{\mathbf{h}}^i$ and $\mathbf{I}^{i,0}$ (for $i=s, \ldots, 0$) using backward recursion. It may be noted that $\mathbf{I}^{i,0}$ is the $(i,0)^{\text{th}}$ block element of GIM, analytical expression of which is obtained in Eq. (A.2).

Step 3: Find $\ddot{\mathbf{q}}_0=-\mathbf{I}_0^{-1}\tilde{\mathbf{h}}_0$ and $\boldsymbol{\tau}_\theta=\mathbf{I}_{\theta 0}\mathbf{q}_0+\tilde{\mathbf{h}}_\theta$

Finally, acceleration of the floating-base, $\ddot{\mathbf{q}}^0$, and the joint torques, $\boldsymbol{\tau}^i$, for $i=1, \ldots, s$, are obtained. The details of the inter-modular and intra- modular recursions for the above three steps are shown in Fig. A.1 of Appendix A.2.

4.2. Forward dynamics

The main objective of forward dynamics is to find the accelerations associated with DOF of the legged robot for a given set of initial positions and velocities. These accelerations can then be integrated twice to obtain the velocities and positions at the next time step. Forward dynamics problem is formulated using Eq. (23) as

$$\mathbf{I}\mathbf{q}=\boldsymbol{\tau}-\mathbf{h},\ \text{where}\ \mathbf{h}\equiv\mathbf{C}\dot{\mathbf{q}}-\boldsymbol{\tau}^F=\mathbf{N}_d^T\mathbf{N}_l^T\mathbf{w} \tag{28}$$

In the proposed forward dynamics algorithm, the accelerations associated with each module are solved recursively without actually performing any explicit inversion of the GIM of Eq. (24). The algorithm is based on the block $\mathbf{UDU}^T$ decomposition of the GIM, obtained in Appendix A.1, similar to the $\mathbf{UDU}^T$ decomposition of GIM for serial-chain systems [21]. Here it is generalized to the multi-modular tree-type systems for the first time. Based on this decomposition, the equations of motion, Eq. (28), are re-written as

$$\mathbf{U}\mathbf{D}\mathbf{U}^T\ddot{\mathbf{q}}=\boldsymbol{\tau}-\mathbf{h} \tag{29}$$

where $\mathbf{U}$ and $\mathbf{D}$ are the $(n+n_0)\times(n+n_0)$ block upper-triangular and diagonal matrices, respectively, the expressions of which are derived in Eqs. (A.5)–(A.6). Solution of Eq. (29) for recursive computation of the acceleration $\ddot{\mathbf{q}}$ requires the following steps:

Step 1: Computation of $\mathbf{t}, \mathbf{t}', \textit{and}\ \mathbf{w}=\left(\mathbf{M}\mathbf{t}'+\boldsymbol{\Omega}\mathbf{M}\mathbf{E}\mathbf{t}\right)-\mathbf{w}^F$

This step is similar to step 1 of the inverse dynamics algorithm given in Subsection 4.1. However, here $\mathbf{t}'$ corresponds to the generalized twist-rate when $\ddot{\mathbf{q}}=\mathbf{0}$.

Step 2: Computation of $\ddot{\mathbf{q}}$

Next, $\ddot{\mathbf{q}}$ is solved recursively using three sets of linear algebraic equations, i.e.,

$$\begin{aligned}&\text{i)}\ \mathbf{U}\hat{\boldsymbol{\varphi}}=\boldsymbol{\tau}-\mathbf{N}_d^T\mathbf{N}_l^T\mathbf{w},\ \text{where}\ \hat{\boldsymbol{\varphi}}=\mathbf{D}\mathbf{U}^T\ddot{\mathbf{q}}\\ &\text{ii)}\ \mathbf{D}\tilde{\boldsymbol{\varphi}}=\hat{\boldsymbol{\varphi}},\ \text{where}\ \tilde{\boldsymbol{\varphi}}=\mathbf{U}^T\ddot{\mathbf{q}}\\ &\text{iii)}\ \mathbf{U}^T\ddot{\mathbf{q}}=\tilde{\boldsymbol{\varphi}}\end{aligned} \tag{30}$$

The inter- and intra-modular steps for recursive calculation of independent acceleration ($\ddot{\mathbf{q}}$) are shown in Fig. A.2 of Appendix A.3. It is worth noting that the intra-modular recursion is carried out using the methodology for serial chain system [29].

To study the effectiveness of the proposed recursive inverse and forward dynamics algorithms, their computational counts are provided in Table 1(a). It may be noted that the counts for the tree-type system does not depend on the number of modules,

Table 1
Comparison of the computational counts.

	Proposed O(n)	Literature O(n) and O(n^3)
(a) Floating-base		
Inverse dynamics	$(164n+60)$M $(156n+66)$A	Not available
Forward dynamics	$(209n+60)$M $(201n+66)$A	O(n): $(224n+158)$M $(205n+131)$A (McMillan and Orin,1998)
		$O(n^3): \left(\frac{1}{6}n^3+17\frac{1}{2}n^2+175\frac{1}{3}n+137\right)M\left(\frac{1}{6}n^3+13n^2+160\frac{5}{6}n+91\right)A$ (*McMillan and Orin, 1998*)
(b) Fixed-base		
Inverse dynamics	$(94n\text{-}81)$M $(82n\text{-}75)$A	O(n) : $(130n\text{-}68)$M $(101n\text{-}56)$A (Featherstone, 2008)
Forward dynamics	$(209n-198)$M $(201n-185)$A	O(n) : $(300n\text{-}267)$M $(279n\text{-}259)$A (Featherstone, 2008)

M: Multiplications/Divisions; A: Addition/Subtraction.

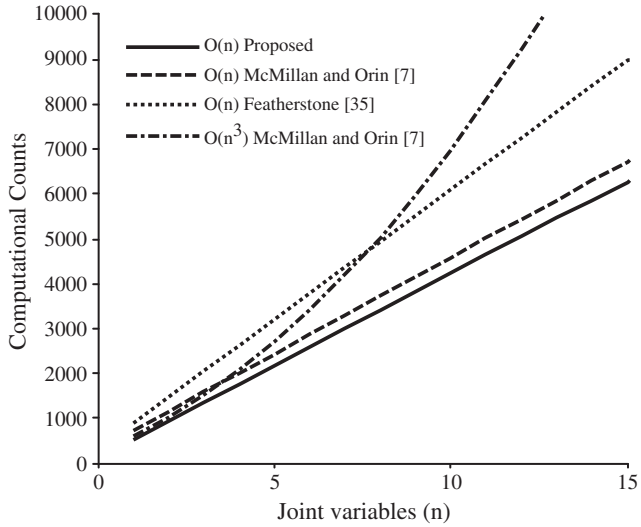


Fig. 7. Comparison of recursive forward dynamics algorithms for the floating-base system.

instead depend only on number of links, n, which is expected. For the forward dynamics of legged robots the proposed algorithm is slightly better than the recursive algorithm of [7], whereas the complexity for the inverse dynamics algorithm could not be compared due to non-availability of similar data in the literature. Nevertheless, Featherstone [35] gave the computational complexities for a fixed-base system which are given in Table 1(b) along with the implementation of the proposed algorithm to a fixed-base system.

Fig. 7 shows comparison of the proposed forward dynamics algorithm with those available in literature. The algorithm is compared with O(n) and O(n^3) algorithms proposed by McMillan and Orin [7]. The comparison is also done with Featherstone's algorithm [35] when implemented for floating-base system in a straightforward manner. The comparison in Fig. 7 clearly indicates that the proposed algorithm performed best amongst all recursive algorithms.

4.3. Simulation of controlled legged robots

A controller is an essential part of a robotic system in achieving desired motions. A Proportional Integral Derivative (PID) controller is widely used in industries for control of processes or machines. In a robotic system, a typical PID controller independently controls the motion of each joint ignoring the effects of the system-dynamics. However, to accomplish a complex motion, the PID controller does not always result in the best performance, as shown in [28]. On the other hand, the use of model-based controllers [27,28] has become popular in order to improve the performance of the conventional PID controllers. The model-based controllers work based on the information of the dynamic model of a system. Hence, the recursive algorithms for inverse and forward dynamics proposed in Subsections 4.1 and 4.2 are valuable in control of robots due to their efficiency, computational uniformity and less numerical errors in the dynamic computations as compared to non-recursive algorithms.

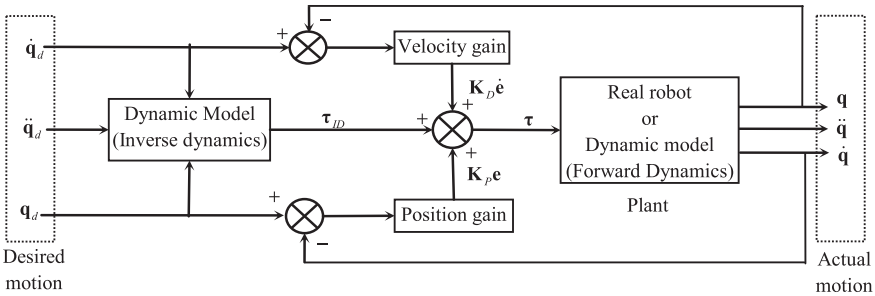


Fig. 8. Feed-forward control scheme.

While the inverse dynamics algorithm helps in calculating the controlling torques of the actuators located at different joints, the forward dynamics algorithm predicts the behavior of the robot. The latter also allows real time simulation, which helps in predicting the future state of a system for corrective control measures. In this section, it will be shown how both the algorithms can be used to study the behavior of legged robots under feed-forward control scheme. In order to validate the results thus obtained, model-based computed-torque control that is very popular in robotic application [27,28] is also employed.

4.3.1. Feed-forward control

Feed-forward control scheme consists of linear servo feedback and feed-forward of the nonlinear dynamic model of robot. It is the simplest form of the non-linear controller which can be employed for motion control of a robot. The control law for this scheme applied for the legged robots at hand, as shown in Fig. 8, is given by

$$\boldsymbol{\tau} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau}_\theta \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau}_{ID} + \mathbf{K}_p\left(\mathbf{q}_{\theta d} - \mathbf{q}_\theta\right) + \mathbf{K}_d\left(\dot{\mathbf{q}}_{\theta d} - \dot{\mathbf{q}}_\theta\right) \end{bmatrix} \tag{31}$$

where $\mathbf{K}_p \equiv diag\begin{bmatrix} k_{p_1} & \cdots & k_{p_n} \end{bmatrix}$ *and* $\mathbf{K}_d \equiv diag\begin{bmatrix} k_{d_1} & \cdots & k_{d_n} \end{bmatrix}$ are the matrices of proportional and derivative gains, and $\boldsymbol{\tau}_{ID}$ is the vector of joint torques obtained from the recursive inverse dynamics algorithm presented in Subsection 4.1. It is worth noting that the feed-forward control can achieve performance as good as computed-torque control with the proper tuning of the control gains. The gains can be calculated using, for example, a design procedure proposed by Kelly et al. [36]. In the absence of a real legged robot, its dynamic model given by Eq. (29) was used. To predict the real robot's behavior, acceleration $\ddot{\mathbf{q}}$ was obtained first before it was integrated twice by using a variable order stiff solver based on the Numerical Differentiation Formulas (NDFs).

4.3.2. Computed-torque control

Computed-torque control works on the principal of feedback linearization of nonlinear systems. For the floating-base legged robots under study, the control law is given by

$$\boldsymbol{\tau} = \mathbf{I}\boldsymbol{\alpha} + \mathbf{h}. \tag{32}$$

Substituting Eq. (32) into Eq. (28) the resulting equations of motion of the closed-loop control system are represented as $\ddot{\mathbf{q}} = \boldsymbol{\alpha}$. For $\boldsymbol{\alpha}$ being the linear function of the state variables, the closed-loop equations are also linear functions of the state variables. As a result, the problem of control is simply to solve a system of linear differential equations. For the floating-base legged robots, the components of $\boldsymbol{\tau}$ associated with the generalized forces of the floating-base are zeros. Hence, Eq. (32) is rewritten as

$$\boldsymbol{\tau} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau}_\theta \end{bmatrix} = \begin{bmatrix} \mathbf{I}_0 & \mathbf{I}_{\theta 0}^T \\ \mathbf{I}_{\theta 0} & \mathbf{I}_\theta \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_0 \\ \boldsymbol{\alpha}_\theta \end{bmatrix} + \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_\theta \end{bmatrix} \tag{33}$$

which is further simplified as

$$\boldsymbol{\alpha}_0 = -\mathbf{I}_0^{-1}\left(\mathbf{I}_{\theta 0}^T \boldsymbol{\alpha}_\theta + \mathbf{h}_0\right) \tag{34}$$

$$\boldsymbol{\tau}_\theta = \mathbf{I}_{\theta 0}\boldsymbol{\alpha}_0 + (\mathbf{I}_\theta \boldsymbol{\alpha}_\theta + \mathbf{h}_\theta). \tag{35}$$

Substituting Eq. (34) into Eq. (35), the computed-torque control law for the actuated joints of the legged robots is obtained as

$$\boldsymbol{\tau} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau}_\theta \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \left(\mathbf{I}_\theta - \mathbf{I}_{\theta 0}\mathbf{I}_0^{-1}\mathbf{I}_{\theta 0}^T\right)\boldsymbol{\alpha}_\theta + \left(\mathbf{h}_\theta - \mathbf{I}_{\theta 0}\mathbf{I}_0^{-1}\mathbf{h}_0\right) \end{bmatrix}. \tag{36}$$

In. Eq. (36), $\boldsymbol{\tau}_\theta$ is the vector of required actuator torques and forces, whereas $\boldsymbol{\alpha}_\theta$ representing the servo part is given by

$$\boldsymbol{\alpha}_\theta = \ddot{\mathbf{q}}_{\theta d} + \mathbf{K}_p\left(\mathbf{q}_{\theta d} - \mathbf{q}_\theta\right) + \mathbf{K}_d\left(\dot{\mathbf{q}}_{\theta d} - \dot{\mathbf{q}}_\theta\right) \tag{37}$$

where $\mathbf{K}_p = diag\begin{bmatrix} k_{p_1} & \cdots & k_{p_n} \end{bmatrix}$ *and* $\mathbf{K}_d = diag\begin{bmatrix} k_{d_1} & \cdots & k_{d_n} \end{bmatrix}$. The scheme for computed-torque control is shown in Fig. 9, whereas the gains k_{p_i} *and* k_{d_i} are obtained by assuming critically damped response, i.e., $k_{d_i} = 2\sqrt{k_{p_i}}$. It is worth noting that the direct use of Eq. (36) to compute the driving torque, $\boldsymbol{\tau}_\theta$, is computationally inefficient as the equations cannot be solved recursively due to the presence of the terms $\mathbf{I}_{\theta 0}\,\mathbf{I}_0^{-1}\,\mathbf{I}_{\theta 0}^T$ and $\mathbf{I}_{\theta 0}\mathbf{I}_0^{-1}\mathbf{h}_0$. On the contrary, Eqs. (34)–(35) have representations similar to

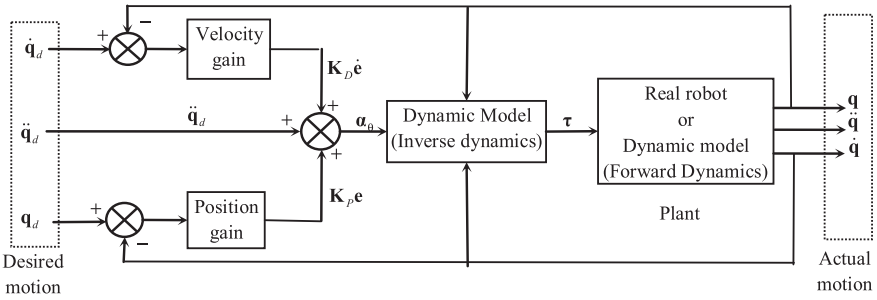


Fig. 9. Computed-torque control scheme.

Eqs. (26), (27). As a result, they can be obtained recursively using the proposed recursive inverse dynamics algorithm by substituting $\ddot{\mathbf{q}}_\theta = \boldsymbol{\alpha}_\theta$ and $\ddot{\mathbf{q}}_0 = \boldsymbol{\alpha}_0$ in Eq. (26), (27).

5. Numerical examples

In this section, the results are reported for a 7-link planar biped and a 9-link spatial quadruped by using the proposed recursive algorithms.

5.1. Planar biped

A 7-link biped consisting of a trunk and two legs is shown in Fig. 10. Modularization of the biped has already been discussed in Subsection 2.3 and shown in Fig. 5. The trunk's center-of-mass (COM) (x_0, z_0) and angle φ_0, and the joint angles $\theta_{1^1}, \theta_{2^1}, \theta_{3^1}, \theta_{1^2}, \theta_{2^2}$, *and* θ_{3^2} are assumed to be the generalized independent coordinates of the biped as depicted in Fig. 10. The length and mass of the links are taken as $l_0 = l_{1^1} = l_{2^1} = l_{1^2} = l_{2^2} = 0.5$ *m*, $l_{3^1} = l_{3^2} = 0.15$ *m*, $m_0 = 5$ Kg, $m_{1^1} = m_{2^1} = m_{1^2} = m_{2^2} = 1$ Kg and $m_{3^1} = m_{3^2} = 0.2$ *Kg*.

5.1.1. Synthesis of joint trajectories

For the simulation of the 7-link 9-DOF biped, the desired joint trajectories to be applied at the six actuated joints are synthesized first. As evident from Eq. (26), the motions of the trunk and the joints are related. Therefore, the actuated joint motions must be synthesized such that they ensure stable motion of the trunk as well. For this, the concept of Inverted Pendulum Method (IPM) [8,9] was used, as explained in Appendix B. The desired actuated joint trajectories, $\theta_{1^1}, \theta_{2^1}, \theta_{3^1}, \theta_{1^2}, \theta_{2^2}$ *and* θ_{3^2}, are obtained from the motions of trunk and foot using the inverse kinematics relationships of the biped, and these are similar to those of a serial

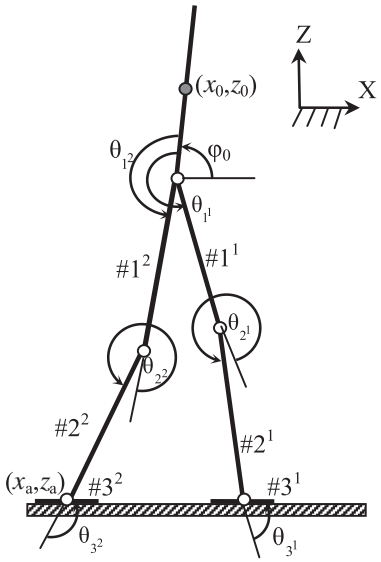
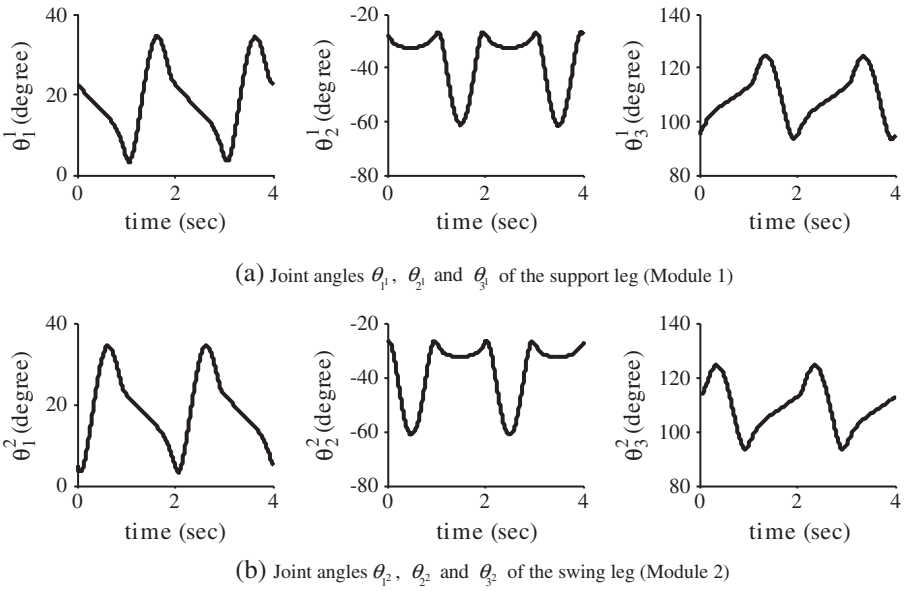


Fig. 10. A 7-link planar biped.

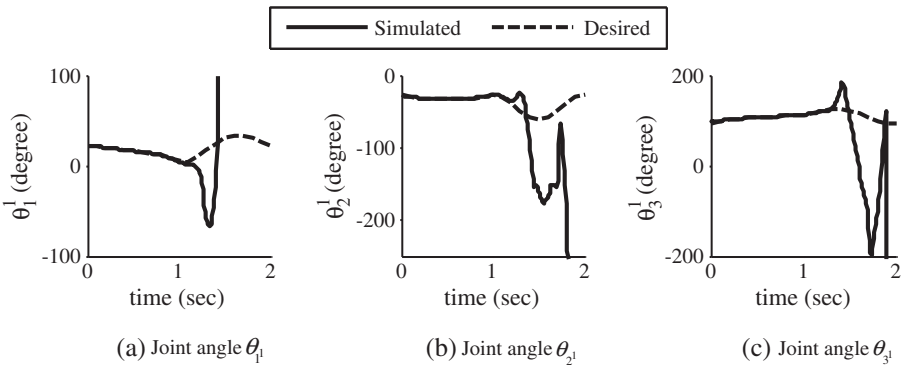


(a) Joint angles θ_{1^1}, θ_{2^1} and θ_{3^1} of the support leg (Module 1)

(b) Joint angles θ_{1^2}, θ_{2^2} and θ_{3^2} of the swing leg (Module 2)

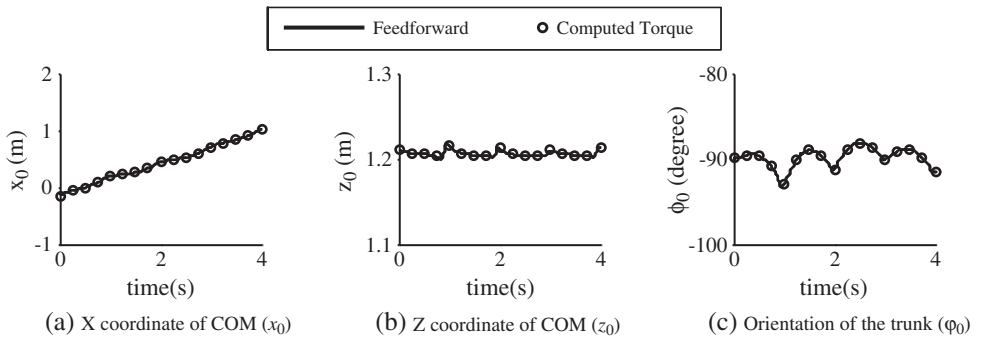
Fig. 11. Joint trajectories of the planar biped obtained from trunk and ankle trajectories by using inverse kinematics relationships.

manipulator [37]. The joint trajectories are depicted in Fig. 11. It is worth mentioning here that it is only the actuated joint trajectories, which were input to the dynamics algorithms, not the trajectories of the trunk and foot. During dynamic computation the accelerations of the trunk were solved from Eq. (26) first, followed by numerical integration to obtain its velocity and position. These motions of the trunk were then used for the computation of dynamic quantities necessary in the algorithms.



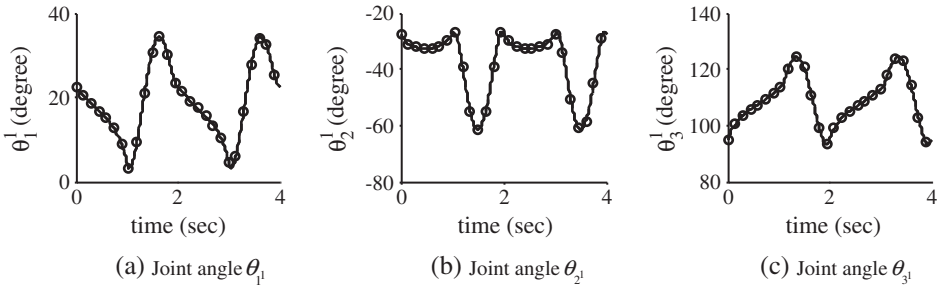
(a) Joint angle θ_{1^1} (b) Joint angle θ_{2^1} (c) Joint angle θ_{3^1}

Fig. 12. Simulated joint angles for the support leg (Module M_1) of the biped without control.



(a) X coordinate of COM (x_0) (b) Z coordinate of COM (z_0) (c) Orientation of the trunk (φ_0)

Fig. 13. Simulated behavior for the trunk of the biped.



(a) Joint angle θ_{1^1} (b) Joint angle θ_{2^1} (c) Joint angle θ_{3^1}

Fig. 14. Simulated joint angles for the support leg (Module M_1) of the biped with control.

5.1.2. Simulation without control

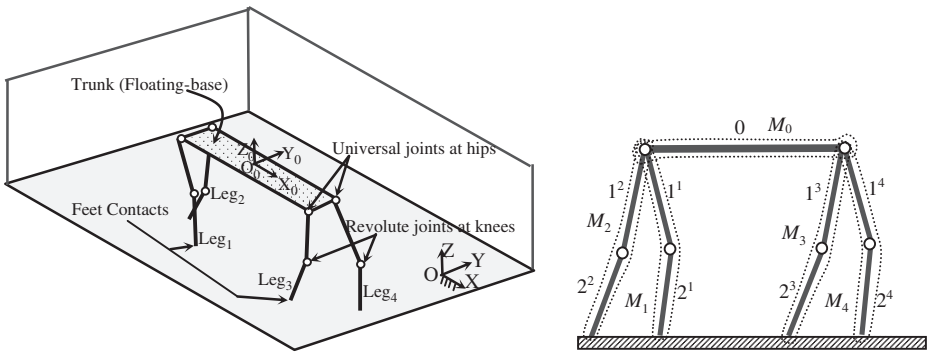
Forward dynamics was then performed simply with the values of the joint torques obtained using the inverse dynamics without any control. Theoretically, the torque calculated using inverse dynamics (Subsection 4.1) should produce the desired trajectories when simulation (Subsection 4.2) was performed using the same dynamic model. However, due to numerical errors and the well-known zero eigen value effect [31], there will be deviations in the simulated trajectories from the desired ones, as seen in Fig. 12, where the motion of the biped is uncontrollable after 1 s. In practice, there will be deviations in the trajectories followed by a real robot due to modeling inaccuracy and other unmodeled phenomena like backlash in the gears and similar elements in the actuators, joint friction, etc. Hence, a suitable strategy is required for control of a robot, both in simulation and in actual practice. This justifies the control algorithms proposed in Subsection 4.3 and corresponding results given in Subsection 5.1.3 for the stable walking of the biped.

5.1.3. Simulation with control

Dynamic simulation presented in the previous subsection showed that the biped was able to follow the trajectory for only 1 s. Hence, feed-forward control scheme is applied to simulate the controlled motion of the planar biped. The proposed recursive inverse dynamics algorithm (Subsection 4.1) was used for the implementation of feed-forward control law given by Eq. (31), while the recursive forward dynamics algorithm (Subsection 4.2) was used to simulate the controlled motion of the biped. In Eq. (31), gains were taken as $k_{p_i} = 49$ and $k_{d_i} = 14$. They were obtained empirically so that the controlled motions match the designed trajectories. For comparison, the computed-torque control scheme based on Eqs. (36), (37) was also used with the same control gains. Fig. 13 shows the variation of the trunk's COM and angle φ_0 for 4 s., even though the simulation was run for 12 s. It is evident from Fig. 13 (a) that the biped travels in the forward direction (i.e., X) with stable periodic motion. Moreover, there is a close match between the joint angles under feed-forward and computed-torque control schemes as shown in Fig. 14, thus, validating the results. This shows that the desired trajectories can be attained using either of the control laws.

5.2. Spatial quadruped

In order to study the effectiveness of the proposed algorithms in spatial motion, a 9-link, 18-DOF spatial quadruped shown in Fig. 15(a) was considered. It has universal joints at the hips, and revolute joints at the knees. The quadruped is divided into five kinematic modules, as shown in Fig. 15(b). As trot is the common gait [6] in four legged animals, it is chosen as the gait of the



(a) A 9-link quadruped (b) Kinematic modules

Fig. 15. A spatial quadruped and its modularization.

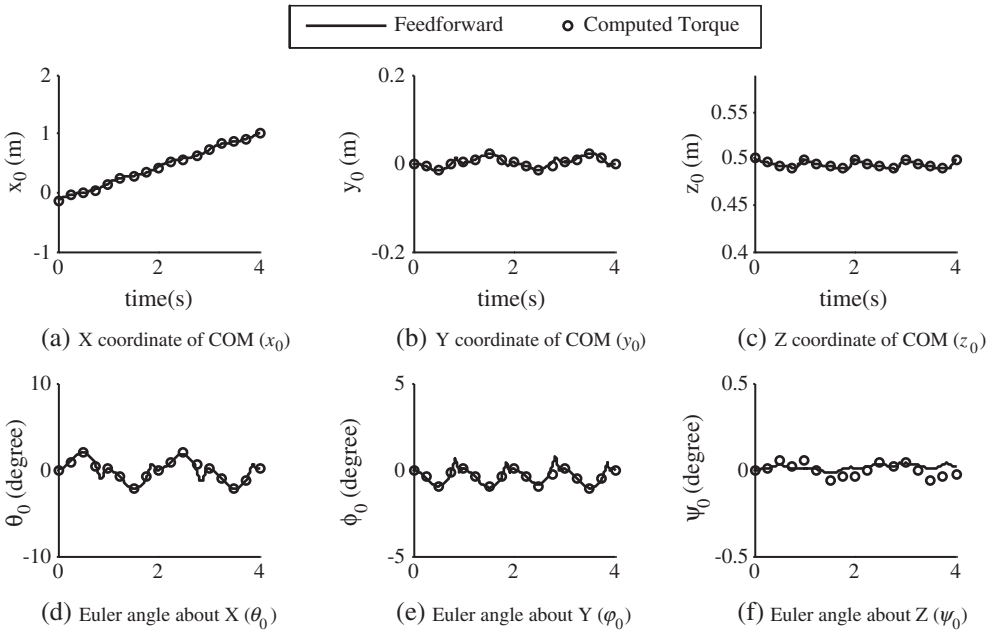


Fig. 16. Simulated behavior for the trunk of the quadruped.

quadruped under study. In quadruped trotting, the diagonally placed legs move in synchronization. Similar to planar biped, the inverted pendulum based approach was again used to obtain the trunk trajectories, which were then used to obtain the desired trajectories of the actuated joints, i.e., $\boldsymbol{\theta}_{1^1} = \begin{bmatrix} \theta_{1_1^1} & \theta_{1_2^1} \end{bmatrix}^T, \theta_{2^1}, \boldsymbol{\theta}_{1^2} = \begin{bmatrix} \theta_{1_1^2} & \theta_{1_2^2} \end{bmatrix}^T, \theta_{2^2}, \boldsymbol{\theta}_{1^3} = \begin{bmatrix} \theta_{1_1^3} & \theta_{1_2^3} \end{bmatrix}^T, \theta_{2^3}, \boldsymbol{\theta}_{1^4} = \begin{bmatrix} \theta_{1_1^4} & \theta_{1_2^4} \end{bmatrix}^T$, *and* θ_{2^4}. The length and mass of the links were taken as $l_0 = 1$ m, $l_{1^1} = l_{2^1} = l_{1^2} = l_{2^2} = l_{1^3} = l_{2^3} = l_{1^4} = l_{2^4} = 0.3$ m, $m_0 = 4$ Kg, and $m_{1^1} = m_{2^1} = m_{1^2} = m_{2^2} = m_{1^3} = m_{2^3} = m_{1^4} = m_{2^4} = 1$ Kg.

Simulations were performed for motion under the feed-forward and computed-torque control laws. Values for the gains were taken as $k_{p_i} = 49$ and $k_{d_i} = 14$ for both the feed-forward and the computed-torque control schemes. The variation of the COM and Euler angles of the trunk are shown in Fig. 16, which depicts that the quadruped travels in the forward direction (X) with a periodic motion. Fig. 17 shows the variation of joint angles for leg 4 (Module M_4). Like the 7-link biped, the simulation results show that the joint angles follow the desired trajectories under both the control laws, proving that the quadruped attains cyclic gait without toppling.

5.3. Computational efficiency

In order to compare the computational efficiencies of the proposed O(n) recursive algorithms with the conventionally used non-recursive O(n^3) algorithm, an in-house Order (n^3) algorithm was also developed. The CPU times required by both the O(n) and O(n^3) algorithms are shown in Table 2.

Comparing the results the following observations are made:

- The proposed O(n) forward dynamics algorithm performed better than the O(n^3) algorithm under both computed-torque and feed-forward control laws.

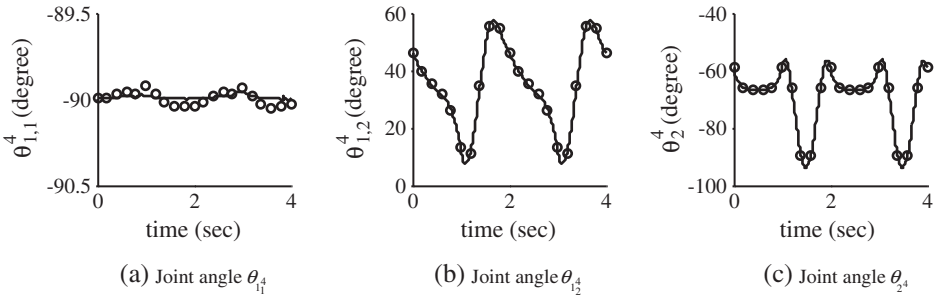


Fig. 17. Simulated joint angles for the fore leg (Leg 4, Module M_4) of the quadruped with control.

Table 2
Time taken* for computation of acceleration ($\ddot{\mathbf{q}}$) for one step, and simulation for 1 s.

Legged robots	Control	Algorithm	Time for $\ddot{\mathbf{q}}$ in ms (% improvement)	Simulation time in ms (% improvement)
Biped	Feed-forward	$O(n^3)$	2.88	5034
		$O(n)$	2.40 (16.67%)	4312 (14.31%)
	Computed-torque	$O(n^3)$	2.89	2920
		$O(n)$	2.41 (16.67%)	2241(23.29%)
Quadruped	Feed-forward	$O(n^3)$	4.48	16170
		$O(n)$	3.49 (22.10%)	10443 (35.44%)
	Computed-torque	$O(n^3)$	4.49	9675
		$O(n)$	3.50 (22.10%)	7031 (27.30%)

*Results are obtained using Intel E8400@3 GHz computing system.

- The enhanced efficiency of the proposed $O(n)$ forward dynamics algorithm is noticeable in the case of spatial quadruped as the system is more complex with higher number of links and joints compared to the planar biped.
- Simulation involving computation of $\ddot{\mathbf{q}}$ followed by numerical integration shows better percentage improvement than the computation of $\ddot{\mathbf{q}}$ only. This implies that the numerical integrations are much faster in the case of an $O(n)$ algorithm compared to the $O(n^3)$ algorithm.

6. Conclusions

This paper presents a general framework for the dynamic modeling and analyses of the legged robots using the concept of kinematic modules where each module is a set of serially connected links only. Legged robots having general module architecture have been considered. Recursive kinematic relationships were obtained between any two adjoining modules and the concepts like module-twist propagation, module-level Decoupled Natural Orthogonal Complement (DeNOC) matrices, etc. were introduced. Such concepts helped to treat a system with large number of links as a system with smaller number of modules. This generalizes the concept of the DeNOC-based formulation originally proposed for a serial-chain system consisting of link to multi-modular tree-type systems, which is one of the basic contributions of this work. The constrained equations of motion were then obtained using the module-DeNOC matrices.

Empowered with the proposed modular framework, the inverse and forward dynamics algorithms with inter- and intra- modular recursions have been developed for the simulation of controlled legged robots, which were illustrated with a planar biped and a spatial quadruped. Inter-modular recursion has been conceived in this work for the first time. The algorithms in terms of the computational counts were found to be better than the algorithms available in literature (Fig. 7), whereas in terms of CPU time they were found to be 16% to 30% more efficient than an $O(n^3)$ algorithm (Table 2).

Appendix A. Recursive algorithms

Some derivations and details leading to recursive inverse and forward dynamics algorithms in Subsections 4.1 and 4.2 are shown in this appendix.

A.1. Decomposition of the GIM

The GIM of Eq. (24) can also be written as

$$\mathbf{I}=\mathbf{N}_d^T\tilde{\mathbf{M}}\mathbf{N}_d, \text{ where } \tilde{\mathbf{M}}=\mathbf{N}_l^T\mathbf{M}\mathbf{N}_l \tag{A.1}$$

where $\tilde{\mathbf{M}}$ is the generalized mass matrix of composite-module, and $\mathbf{N}_l$ and $\mathbf{N}_d$ are the module-DeNOC matrices. Substituting $\mathbf{N}_l$, $\mathbf{N}_d$, and $\mathbf{M}$ from Eqs. (12) and (21) into Eq. (A.1), the module level analytical expressions for the GIM are obtained as follows:

For $i=0, \ldots, s$, and $j=0, \ldots, i$

$$\begin{aligned}\mathbf{I}^{i,j}\equiv\mathbf{I}^{j,i} &= \mathbf{N}^{i^T}\tilde{\mathbf{M}}^i\mathbf{A}^{i,j}\mathbf{N}^j, \textit{ if } \mathrm{M}_i\in\boldsymbol{\gamma}^j\left(\textit{For } i=j, \mathbf{A}^{i,j}=1\right)\\ &= \mathbf{O} \textit{ otherwise}\end{aligned} \tag{A.2}$$

where $\mathbf{I}^{i,j}$ represents the $(i,j)^{\text{th}}$ block element of $\mathbf{I}$. Moreover, matrix $\tilde{\mathbf{M}}^i$ represents the mass and inertia properties of a system comprised of the rigidly connected links of modules which are upstream of the i^{th} module. This is generalization over the mass matrix of a composite body [17]. The matrix $\tilde{\mathbf{M}}^i$ is given by

$$\tilde{\mathbf{M}}^i = \mathbf{M}_i + \sum_{j \in \boldsymbol{\xi}^i} \left(\mathbf{A}^{j,i}\right)^T \tilde{\mathbf{M}}^j \mathbf{A}^{j,i} \tag{A.3}$$

where $\boldsymbol{\xi}^i$ denotes the children of module M_i. If $\boldsymbol{\xi}^i = \{\}$, i.e., no children, the composite-module mass matrix is simply equal to module-mass-matrix, namely, $\tilde{\mathbf{M}}^i \equiv \mathbf{M}^i$. In Eq. (A.3), $\mathbf{A}^{j,i}$ is the $6\eta^j \times 6\eta^i$ module twist propagation matrix, and is similar to $\mathbf{A}^{i,\beta}$ of Eq. (9). The analytical expression derived in Eq. (A.2) is the foundation for obtaining the decomposition of the GIM given by Eq. (A.1). The $\mathbf{UDU}^T$ decomposition of the GIM, $\mathbf{I}$, for a serial robot was originally proposed in [21]. The same concept is extended here for a floating-base tree-type system through module-level $\mathbf{UDU}^T$ decomposition of the GIM using the Block Reverse Gaussian Elimination (BRGE). The decomposed GIM can be expressed as

$$\mathbf{I} = \mathbf{U}\mathbf{D}\mathbf{U}^T. \tag{A.4}$$

The $(n+n_0)\times(n+n_0)$ block upper-triangular and diagonal matrices $\mathbf{U}$ and $\mathbf{D}$ are given by

$$\mathbf{U} \equiv \begin{bmatrix} \mathbf{1}^0 & \mathbf{U}^{0,1} & \cdots & \mathbf{U}^{0,s} \\ \mathbf{0} & \mathbf{1}^1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{U}^{s-1,s} \\ \mathbf{0}'s & \cdots & \mathbf{0} & \mathbf{1}^s \end{bmatrix}, \text{ and } \mathbf{D} \equiv \begin{bmatrix} \hat{\mathbf{I}}^0 & & \mathbf{0}'s \\ & \hat{\mathbf{I}}^1 & \\ \mathbf{0}'s & & \ddots\ \hat{\mathbf{I}}^s \end{bmatrix} \tag{A.5}$$

where the $\eta^i \times \eta^j$ matrix $\mathbf{U}^{i,j}$ and matrix $\hat{\mathbf{I}}^i$ are obtained as follows:

$$\begin{aligned} \mathbf{U}^{i,j} &\equiv \left(\mathbf{A}^{j,i}\right)^T \mathbf{N}^{i^T} \boldsymbol{\psi}^j, \quad \text{if } j \in \boldsymbol{\gamma}_i \\ &\equiv \mathbf{O}, \text{ otherwise} \\ \hat{\mathbf{I}}^i &\equiv \mathbf{N}^{i^T} \hat{\boldsymbol{\psi}}^i. \end{aligned} \tag{A.6}$$

The $6\eta^i \times \eta^i$ matrices $\boldsymbol{\Psi}^i$ and $\hat{\boldsymbol{\psi}}^i$ of Eq. (A.6) are given by

$$\boldsymbol{\Psi}^i = \hat{\boldsymbol{\psi}}^i \hat{\mathbf{I}}^{i^{-1}} \text{ and } \hat{\boldsymbol{\psi}}^i = \hat{\mathbf{M}}^i \mathbf{N}^i \tag{A.7}$$

where the $6\eta^i \times 6\eta^i$ matrix $\hat{\mathbf{M}}^i$ contains the mass and inertia properties of the articulated-module i. Thus the concept of articulated body [22] is generalized to articulated module. The matrix $\hat{\mathbf{M}}^i$ is obtained as

$$\hat{\mathbf{M}}^i = \mathbf{M}^i + \sum_{j \in \boldsymbol{\xi}^i} \left(\mathbf{A}^{j,i}\right)^T \hat{\mathbf{M}}^{j,j} \mathbf{A}^{j,i} \tag{A.8}$$

where if $\boldsymbol{\xi}^i = \{\}, \hat{\mathbf{M}}^i \equiv \mathbf{M}^i$. The $6\eta^i \times 6\eta^i$ matrix $\hat{\mathbf{M}}^{j,j}$ is defined as

$$\hat{\mathbf{M}}^{j,j} = \hat{\mathbf{M}}^j - \hat{\boldsymbol{\psi}}^j \boldsymbol{\psi}^{j^T}. \tag{A.9}$$

The proposed $\mathbf{UDU}^T$ decomposition helps in obtaining the recursive forward dynamics of Subsection 4.2 and provides better insight into the dynamics involved through inertia properties, which may help in deciding any inconsistency in the dynamic behavior of the legged robots. For example, $\hat{\mathbf{I}}^i$ in Eq. (A.6) can be interpreted as the inertia matrix of the i^{th} articulated module. One may study the stability of the independent module based on ill-conditioning of $\hat{\mathbf{I}}^i$ rather than having a global picture based on the ill-conditioning of the GIM. In other words one can predict instability of the modules and take corrective measures accordingly.

A.2. Recursive inverse dynamics

The inter- and intra-modular steps for the recursive inverse dynamics proposed in Subsection 4.1 are presented below in Figure A.1. It is worth noting that inter-modular steps are the compact representation of the intra-modular steps.

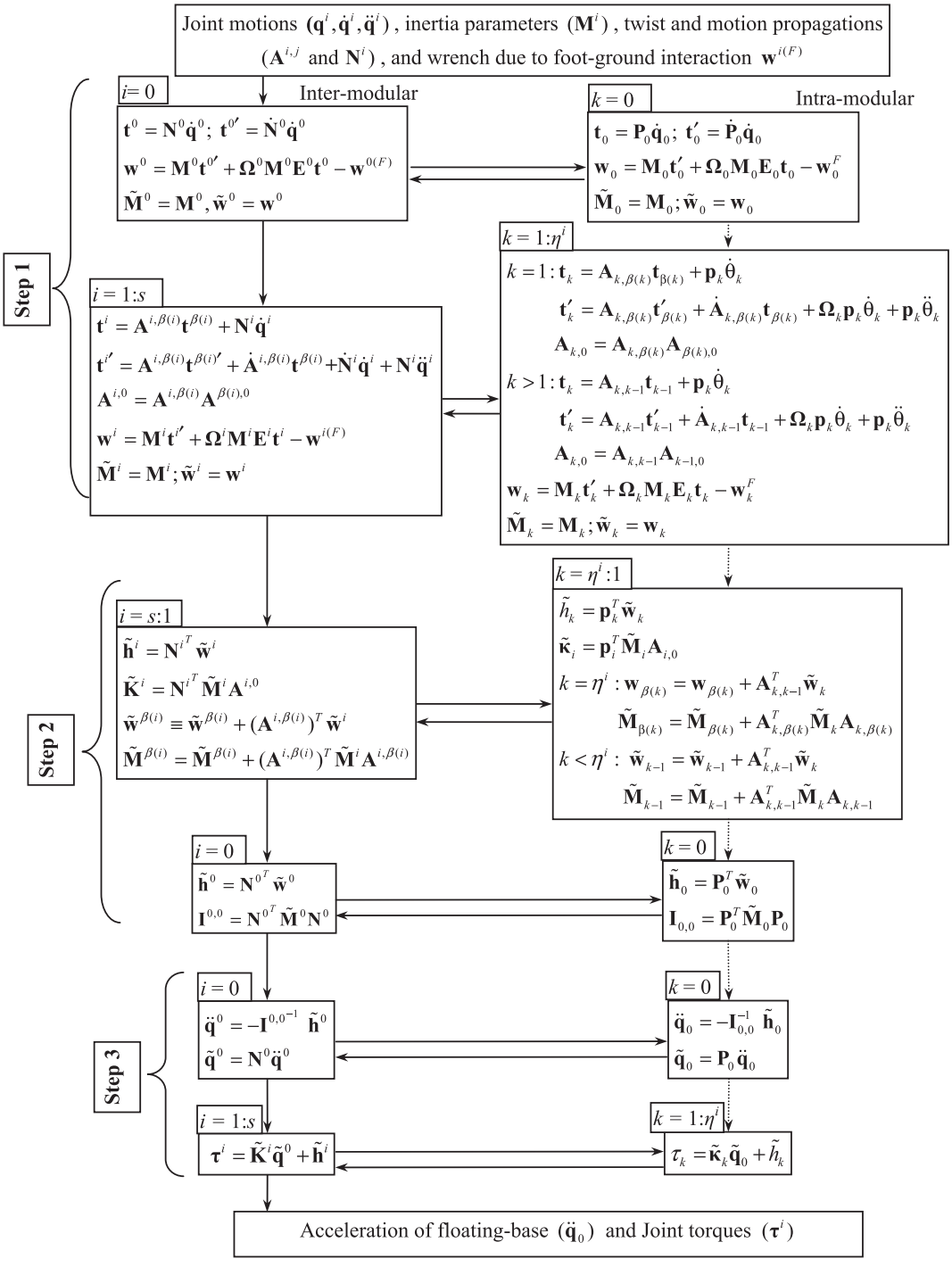


Fig. A.1. Inter- and intra-modular steps of recursive dynamics.

A.3. Recursive forward dynamics

The inter-modular and intra modular steps for recursive computation of the independent acceleration ($\ddot{\mathbf{q}}$) is presented in Fig. A.2 below based on recursive forward dynamics presented in Subsection 4.2.

Note that the derivation of the intra-modular steps from the inter-modular steps is not straightforward, as in the case of inverse dynamics algorithm of Fig. A.1, because these steps require analytical $\mathbf{U}_i\mathbf{D}_i\mathbf{U}_i^T$ decomposition of the matrix $\hat{\mathbf{I}}^i$. Intra-modular computations have been carried out in a way similar to a serial system [29].

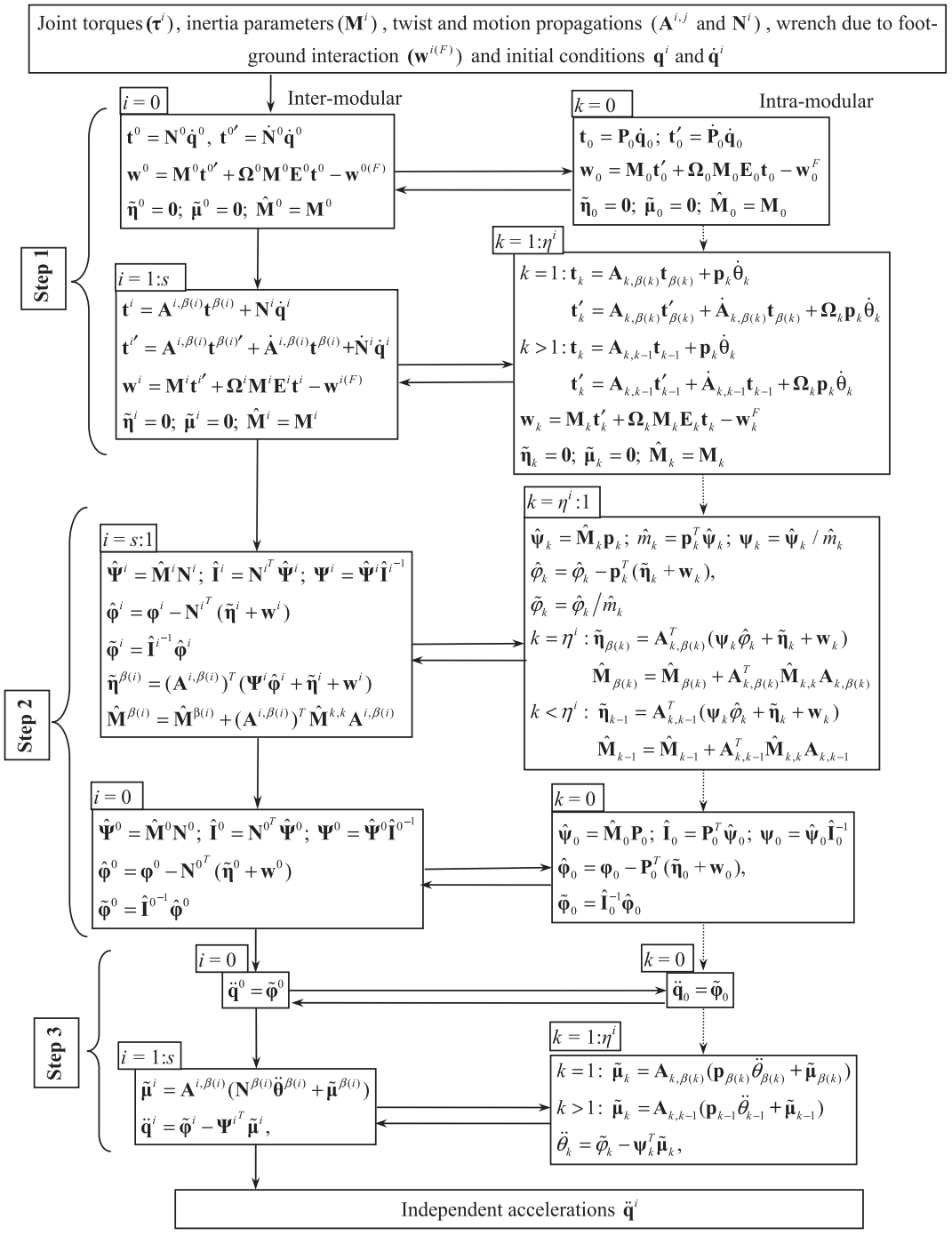


Fig. A.2. Inter- and intra-modular steps of recursive forward dynamics.

Appendix B. Trunk and swing foot trajectories

As discussed in Subsection 5.1.1, the joint trajectories are obtained from the motion of trunk and foot using inverse kinematics relationships. In order to obtain the joint trajectories that ensure stable walking pattern, the trunk's trajectories are synthesized using the Inverted Pendulum Method (IPM) [8,9]. It is worth noting that the motion of the trunk is not the input to the inverse dynamics algorithm; instead, it is obtained using Eq. (26) during the dynamics computation.

The IPM assumes that the mass of the biped is concentrated at the hip, and that the feet remain horizontal throughout the robot's motion cycle. The trunk is considered to move horizontally with the same height from the ground, i.e., $\varphi_0 = 90^\circ$ and $z_0 = z_c$. Thus, the Center-of-Mass (COM) of the trunk has same motion as that of the hip. For one cycle with a time period of T, the X and Z coordinates of the hip, denoted by x_h and z_h, are given by:

$$\begin{aligned} &For\ 0 \le t \le T \qquad x_h(t) = x_h(0)\cosh(\omega t) + \left(\frac{\dot{x}_h(0)}{\omega}\right)\sinh(\omega t) \\ &\qquad\qquad\qquad\quad z_h(t) = z_c - \left(\frac{l_0}{2}\right) \end{aligned} \tag{B.1}$$

where $x_h(0)$ and $\dot{x}_h(0)$ are the initial position and velocity of the hip, and $\omega = \sqrt{g/z_c}$, where g is acceleration due to gravity. In order to obtain the steady and repeatable walking pattern, the repeatability conditions of $x_h(0) = -x_h(T) and\ \dot{x}_h(0) = \dot{x}_h(T)$ as proposed in [8] were used. Trajectories for the ankle of the swing foot were synthesized [8] as

$$\begin{aligned} &For\ 0 \le t \le T \qquad x_{ankle}(t) = -l_s\cos\left(\frac{\pi}{T}t\right) \\ &\qquad\qquad\qquad\quad z_{ankle}(t) = \frac{h_f}{2}\left[1 - \cos\left(\frac{\pi}{T}t\right)\right] \end{aligned} \tag{B.2}$$

where l_s, and h_f are stride length and maximum foot height. The values of T, l_s, h_f and z_c are taken as 1 s, 0.3 m, 0.1 m and 0.96 m for the biped, and 1 s, 0.3 m, 0.08 m and 0.5 m for the quadruped.

References

[1] M.H. Raibert, Hopping in legged system— modeling and simulation for the two-dimensional one-legged case, IEEE Transactions on Systems, Man, and Cybernetics 14 (3) (1984) 451–463.
[2] M. Vukobratovic, B. Borovac, D. Surla, D. Stokic, Biped Locomotion: Dynamics, Stability, Control and Application, Springer, Berlin, 1989.
[3] C.L. Shih, W.A. Gruver, T.T. Lee, Inverse kinematics and inverse dynamics for control of a biped walking machine, Journal of Robotic Systems 10 (4) (1993) 531–555.
[4] K. Hirai, M. Hirose, Y. Haikawa, T. Takenaka, The development of Honda humanoid robot, IEEE Int. Conf. on Robotic and Automation, 1998, pp. 1321–1326.
[5] M. Buehler, R. Battaglia, A. Cocosco, G. Hawker, J. Sarkis, K. Yamazaki, SCOUT: a simple quadruped that walks, climbs and runs, IEEE Int. Conf. on Rob. and Auto., 1998, pp. 1707–1712.
[6] M.D. Berkemeier, Modeling the dynamics of quadrupedal running, International Journal of Robotics Research 17 (1998) 971–985.
[7] S. McMillan, D.E. Orin, Forward dynamics of multilegged vehicles, IEEE Int. Conf. on Robotics and Automation, 1998, pp. 464–470.
[8] J.H. Park, K.D. Kim, Biped robot walking using gravity-compensated inverted pendulum mode and computed torque control, Proc. IEEE Int. Conf. Rob. and Aut., 1998, pp. 3528–3533.
[9] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, H. Hirukawa, The 3D linear inverted pendulum mode: a simple modeling for a biped walking pattern generation, Proc. of IROS, 2001, pp. 239–246.
[10] U. Saranli, M. Buehler, D.E. Koditschek, RHex — a simple and highly mobile hexapod robot, International Journal of Robotics Research 20 (7) (2001) 616–631.
[11] M. Vukobratovic, V. Potkonjak, K. Babkovic, B. Borovac, Simulation model of general human and humanoid motion, Multibody System Dynamics 17 (1) (2007) 71–96.
[12] S. Stramigioli, V. Duindam, G. van Oort, A. Goswami, Compact analysis of 3D bipedal gait using geometric dynamics of simplified models, IEEE Int. Conf. on Rob. and Auto., 2009, pp. 1978–1984.
[13] D.E. Stewart, Rigid-body dynamics with friction and impact, SIAM Review 42 (1) (2000) 3–39.
[14] G. Gilardi, I. Sharf, Literature survey of contact dynamics modelling, Mechanism & Machine Theory 37 (10) (2002) 1213–1239.
[15] P. Eberhard, W. Schiehlen, Computational dynamics of multibody systems: history, formalisms, and applications, ASME Journal of Computational and Nonlinear Dynamics 1 (1) (2006) 3–12.
[16] J.M. Hollerbach, A recursive Lagrangian formulation of manipulator dynamics and comparative study of dynamics formulation complexity, IEEE Transactions on Systems, Man, and Cybernetics 11 (1980) 730–736.
[17] M.W. Walker, D.E. Orin, Efficient dynamic computer simulation of robotic mechanisms, Journal of Dynamic Systems, Measurement, and Control 104 (1982) 205–211.
[18] T.R. Kane, D.A. Levinson, The use of Kane's dynamical equations for robotics, International Journal of Robotics Research 2 (3) (1983) 3–21.
[19] J. Angeles, O. Ma, Dynamic simulation of n-axis serial robotic manipulators using a natural orthogonal complement, International Journal of Robotics Research 7 (5) (1988) 32–47.
[20] W. Blajer, D. Bestle, W. Schiehlen, An orthogonal complement matrix formulation for constrained multibody systems, ASME Journal of Mechanical Design 116 (1994) 423–428.
[21] S.K. Saha, A decomposition of the manipulator inertia matrix, IEEE Transactions on Robotics and Automation 13 (2) (1997) 301–304.
[22] R. Featherstone, The calculation of robotic dynamics using articulated body inertias, International Journal of Robotics Research 2 (1983) 13–30.
[23] D.S. Bae, E.J. Haug, A recursive formulation for constrained mechanical system dynamics: part I, open loop systems, Mechanics of Structure and Machine 15 (1987) 359–382.
[24] G. Rodriguez, Kalman filtering, smoothing, and recursive robot arm forward and inverse dynamics, IEEE Journal of Robotics and Automation 3 (6) (1987) 624–639.
[25] G. Rodriguez, A. Jain, K. Kreutz-Delgado, A spatial operator algebra for manipulator modeling and control, International Journal of Robotics Research 10 (4) (1991) 371–381.
[26] D.E. Rosenthal, An order n formulation for robotic systems, Journal of Astronautical Sciences 38 (4) (1990) 511–529.
[27] J.J. Craig, Introduction to Robotics, Mechanics and Control, Pearson Education, Delhi, 2006.
[28] R. Kelly, V. Santibanez, A. Loria, Control of Robot Manipulators in Joint Space, Springer-Verlag, London, 2005.

[29] S.K. Saha, Dynamics of serial multibody systems using the decoupled natural orthogonal complement matrices, ASME Journal of Applied Mechanics 66 (1999) 986–996.
[30] S.K. Saha, Analytical expression for the inverted inertia matrix of serial robots, International Journal of Robotic Research 18 (1) (1999) 116–124.
[31] S.K. Saha, W.O. Schiehlen, Recursive kinematics and dynamics for closed loop multibody systems, Mechanics and Structures Machines: An International Journal 29 (2) (2001) 143–175.
[32] S.K. Saha, B. Shirinzadeh, Gl. Alici, Dynamic Model Simplification of Serial Manipulators, ISRA, Mexico, 2006.
[33] S.V. Shah, S.K. Saha, J.K. Dutt, Denavit-Hartenberg parameters of Euler-angle-joints for order (n) recursive forward dynamics, ASME int. conf. on multibody sys., nonlinear dyn. and control, USA, 2009.
[34] H. Chaudhary, S.K. Saha, Dynamics and Balancing of Multibody Systems, Springer, Berlin, 2009.
[35] R. Featherstone, Rigid Body Dynamics Algorithms, Springer, New York, 2008.
[36] R. Kelly, R. Salgado, PD control with computed feedforward of robot manipulators: a design procedure, IEEE Trans. Robot. Automat. 10 (4) (1994) 566–571.
[37] S.K. Saha, Introduction to Robotics, Tata Mcgraw Hill, New Delhi, India, 2008.