Robotica (2007) volume 25, pp. 739–750. © 2007 Cambridge University Press
doi:10.1017/S0263574707003852 Printed in the United Kingdom

$O(n)$ mass matrix inversion for serial manipulators and polypeptide chains using Lie derivatives

Kiju Lee†, Yunfeng Wang‡ and Gregory S. Chirikjian*†

†*Department of Mechanical Engineering, Johns Hopkins University, Baltimore, MD 21218, USA.*
‡*Department of Mechanical Engineering, The College of New Jersey, Ewing, NJ 08628, USA.*

(Received in Final Form: August 7, 2007. First published online: September 28, 2007)

SUMMARY
Over the past several decades, a number of $O(n)$ methods for forward and inverse dynamics computations have been developed in the multibody dynamics and robotics literature. A method was developed by Fixman in 1974 for $O(n)$ computation of the mass-matrix determinant for a serial polymer chain consisting of point masses. In other of our recent papers, we extended this method in order to compute the inverse of the mass matrix for serial chains consisting of point masses. In the present paper, we extend these ideas further and address the case of serial chains composed of rigid-bodies. This requires the use of relatively deep mathematics associated with the rotation group, $SO(3)$, and the special Euclidean group, $SE(3)$, and specifically, it requires that one differentiates real-valued functions of Lie-group-valued argument.

KEYWORDS: Serial manipulators; Mass-matrix inversion; Forward dynamics; Polymer chains; Polypeptides.

Nomenclature

$M = G_{11}$	the constrained mass matrix
G	the unconstrained mass matrix
$H = G^{-1}$	the inverted mass matrix of the unconstrained system
$\mathbf{q} = [q_1, \ldots, q_N]^T$	the vector of generalized coordinates
Soft variables	variables defining allowable motion of chains with kinematic constraints
Hard variables	constrained variables of the chain
$\mathbf{s} = [s_1, \ldots, s_f]^T$	the vector of soft variables
$\mathbf{h} = [h_1, \ldots, h_r]^T$	the vector of hard variables
m_i	the ith point mass or the mass of the ith rigid body
I_i	the moment of inertia of the ith rigid body in a frame attached at the center of mass
$T_i = (R_i, \mathbf{p}_i)$	the rigid-body transformation from the global reference frame to the Denavit-Hartenberg (D–H) frame i. $T_i \in SE(N)$, $R_i \in SO(N)$ and $\mathbf{p} \in I\!R^N$ for $N = 2$ or $N = 3$
$T_i^c = (R_i^c, \mathbf{p}_i^c)$	the transformation from the global reference frame to the ith center of mass
Q_i	the homogeneous transformation from T_i to T_i^c
$T_i^{i-1} = (R_i^{i-1}, p_i^{i-1})$	the relative transformation from the D–H frame $i-1$ to the D–H frame i
$A \oplus B = \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}$	for any square matrices A and B
screw$(\mathbf{e}_i, \theta, x)$	a transformation made by the rotation θ about the axis i and the translational displacement, x, along the same axis
$\mathcal{I} = I \oplus m I\!I_3 \in I\!R^{6\times 6}$	the inertia matrix in a frame at the center of mass of a rigid body
$J \in I\!R^{3\times 3}$	the Jacobian associated with parameterized rotations
$\mathcal{J} \in I\!R^{6\times 6}$	the Jacobian associated with parameterized rigid-body transformations
$I\!I_k$	$k \times k$ identity matrix
0_k	$k \times k$ zero matrix
$0_{l\times k}$	$l \times k$ zero matrix

* Corresponding author. E-mail: gregc@jhu.edu

1. Introduction

Serial chains consisting of n rigid bodies connected with rotational or prismatic joints have been studied for many years. The first $O(n)$ algorithm for dynamics computation was developed in the multibody systems literature in 1975.[1] In the robotics area, the Luh–Walker–Paul recursive Newton–Euler approach[2] has been a cornerstone of manipulator inverse dynamics for many years. Another $O(n)$ algorithm within a Lagrangian dynamics setting was presented in.[3] In addition, recursive techniques from linear filtering and smoothing theory for serial manipulators were introduced for both the forward and inverse dynamics problems.[5,6] In Ref. [6], two recursive factorization methods of the mass matrix were presented for fixed-base and mobile-base manipulators: Newton–Euler factorization and innovations factorization. As another approach, a decomposition method using analytical Gaussian Elimination (GE) of the inertia matrix[7] and a recursive forward dynamics algorithm for open-loop, serial-chain robots[8] were presented by Saha. His algorithm has $O(n)$ computational complexity and is also

based on reverse GE applied to analytical expressions of the elements of the inertia matrix. It builds on the Natural Orthogonal Complement for the manipulator mass matrix developed by Angeles and Ma.[9] In a series of papers,[10–14] Anderson and his colleagues presented a numerical analysis and simulations of multi-rigid-body dynamic systems. An $O(n+m)$ algorithm for multibody systems with arbitrarily many closed loops, containing n generalized coordinates and m independent constraints, was presented in Ref. [15]. Featherstone showed a new efficient factorization of the joint space inertia matrix (JSIM) for branched kinematic trees.[16,17] A coordinate invariant algorithm for forward dynamics using Lie groups and Lie algebras was introduced in [18]. More recently, a recursive $O(n)$ forward dynamic computations was used to obtain a set of Hamiltonian equations for open-loop and closed-loop multibody systems.[19,20]

Interestingly, all of these approaches appear to be unaware of developments in the polymer physics literature in which Fixman[21] developed an $O(n)$ method for computing the mass-matrix determinant of a serial chain structure composed of rigid links and point masses. In a series of recent conference papers, we extended Fixman's method to yield a new method for $O(n)$ inversion of the mass matrix for planar serial manipulators and polymer chains consisting of point masses.[22,23] In Ref. [24], we examined chains of rigid bodies. The inverse of the constrained mass matrix (M^{-1}) is obtained by computing the inverse of the unconstrained mass matrix (H) composed of four block matrices which appear to be sparse and band-limited due to the special properties of the serial chain structure. Using these properties, M^{-1} is calculated by

$$M^{-1} = H_{11} - H_{12}(H_{22})^{-1}H_{21} \tag{1}$$

where H_{ij}'s are block matrices of H. This form is known as the Schur complement.[25] The main difference of our work and others that use the Schur compliment is that we adapt Fixman's method of partitioning generalized coordinates into soft and hard variables. This partitions H into four sparse and band-limited matrices instead of using the mathematical manipulations in Ref. [25]. In order to do this, the serial chain is viewed as a collection of rigid bodies, and the constraints between them that define the joints are written as functions on $SE(3)$. These functions are then differentiated using the appropriate concept from Lie theory. In contrast to other Lie-group-based approaches where dynamical phenomena on the groups of interest are considered, we perform operations on the space of differentiable functions on the group.

The rest of the paper is organized as follows. In Section 2, we briefly review Fixman's method and our extension to solve $M\mathbf{x} = \mathbf{b}$ for given M and $\mathbf{b}$ in $O(n)$ time for n-link serial chains consisting of n point masses. Further extensions to rigid-body applications are described in Section 3. In Section 4, we explain how to use the algorithm in detail, and include numerical examples for the PUMA 560 robot arm and a polypeptide chain. The Denavit–Hartenberg (D–H) parameterization is used to describe rigid-body motions for the examples.

2. Fixman's Theorem and Efficient Inversion of the Mass Matrix

2.1. Background

Given a set of n point masses $\{m_1, \ldots, m_n\}$ with a corresponding set of absolute positions $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, we define the $3n$-dimensional composite position vector as $\mathbf{x} = [\mathbf{x}_1^T, \ldots, \mathbf{x}_n^T]^T$. If N generalized coordinates, $q_1, \ldots, q_N$, are used to parameterize $\mathbf{x}$, then the partial derivatives of $\mathbf{x}$ with respect to the N generalized coordinates can be arranged in the $3n \times N$ Jacobian matrix:

$$\left[\frac{\partial \mathbf{x}}{\partial \mathbf{q}}\right] = \begin{pmatrix} \frac{\partial \mathbf{x}_1}{\partial q_1} & \cdots & \frac{\partial \mathbf{x}_1}{\partial q_k} & \cdots & \frac{\partial \mathbf{x}_1}{\partial q_N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{x}_j}{\partial q_1} & \cdots & \frac{\partial \mathbf{x}_j}{\partial q_k} & \cdots & \frac{\partial \mathbf{x}_j}{\partial q_N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{x}_n}{\partial q_1} & \cdots & \frac{\partial \mathbf{x}_n}{\partial q_k} & \cdots & \frac{\partial \mathbf{x}_n}{\partial q_N} \end{pmatrix} \in I\!R^{3n \times N}. \tag{2}$$

If we define, for $\mathbf{m} = (m_1, \ldots, m_n)$,

$$[\mathrm{diag}(\mathbf{m})] = \begin{pmatrix} m_1 I\!I_3 & 0 & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & m_i I\!I_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & m_n I\!I_3 \end{pmatrix} \in I\!R^{3n \times 3n},$$

then the generalized mass matrix is given by

$$G = \left[\frac{\partial \mathbf{x}}{\partial \mathbf{q}}\right]^T [\mathrm{diag}(\mathbf{m})] \left[\frac{\partial \mathbf{x}}{\partial \mathbf{q}}\right] \in I\!R^{N \times N}. \tag{3}$$

In the case when no constraints are imposed, $N = 3n$, and all of the matrices in Eq. (3) are square and invertible almost everywhere.[1] Therefore, it follows that, for $\tilde{\mathbf{m}} = (1/m_1, \ldots, 1/m_n)$,

$$H = \left[\frac{\partial \mathbf{q}}{\partial \mathbf{x}}\right] [\mathrm{diag}(\tilde{\mathbf{m}})] \left[\frac{\partial \mathbf{q}}{\partial \mathbf{x}}\right]^T. \tag{4}$$

Recall that the derivative of a scalar-valued function of vector-valued argument, $f(\mathbf{z})$ with $\mathbf{z} \in I\!R^n$, with respect to its argument is a row vector, $\frac{\partial f}{\partial \mathbf{z}} = \left[\frac{\partial f}{\partial z_1}, \ldots, \frac{\partial f}{\partial z_n}\right]$. This means that

$$\left[\frac{\partial \mathbf{q}}{\partial \mathbf{x}}\right] = \begin{pmatrix} \frac{\partial q_1}{\partial \mathbf{x}_1} & \cdots & \frac{\partial q_1}{\partial \mathbf{x}_k} & \cdots & \frac{\partial q_1}{\partial \mathbf{x}_n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial q_j}{\partial \mathbf{x}_1} & \cdots & \frac{\partial q_j}{\partial \mathbf{x}_k} & \cdots & \frac{\partial q_j}{\partial \mathbf{x}_n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial q_N}{\partial \mathbf{x}_1} & \cdots & \frac{\partial q_N}{\partial \mathbf{x}_k} & \cdots & \frac{\partial q_N}{\partial \mathbf{x}_n} \end{pmatrix}$$

where each entry in the above matrix is a 3-dimensional row vector.

[1] For general invertible matrices A and B, $(AB)^{-1} = B^{-1}A^{-1}$ and $(A^T)^{-1} = (A^{-1})^T$.

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

2.2. Fixman's method and extension to solve $M\mathbf{x}=\mathbf{b}$

We begin by introducing *Fixman's Theorem* to the robotics community and showing how extensions of Fixman's results can be used to efficiently invert the expression $M\mathbf{x}=\mathbf{b}$, where M is the mass matrix for a serial chain composed of point masses and $\mathbf{b}$ is any given vector with matching dimension.

Given an n-link serial chain with point masses, the vector of generalized coordinates is partitioned into the vector of soft variables and the vector of hard variables, such that $\mathbf{q}=[\mathbf{s}^T,\mathbf{h}^T]^T$ where $\mathbf{s}=[s_1,\ldots,s_f]^T$ and $\mathbf{h}=[h_1,\ldots,h_r]^T$ when $f+r=3n$.[21] Then, Eqs. (3) and (4) are also represented as partitioned matrices, such that

$$G=\begin{pmatrix} G_{11} & G_{12} \\ G_{12}^T & G_{22}\end{pmatrix};\quad H=\begin{pmatrix} H_{11} & H_{12} \\ H_{12}^T & H_{22}\end{pmatrix}.$$

Now we consider the fast inversion of G_{11} of the equation

$$\mathbf{x}=G_{11}^{-1}\mathbf{b} \tag{5}$$

where $G_{11}(=M)$ is the mass matrix for a serial chain with constraints. In general, G_{11} is a full matrix, and thus, the direct numerical inversion of G_{11} requires $O(n^3)$ computations. By using the fact that

$$G\cdot H=\begin{pmatrix} G_{11} & G_{12} \\ G_{12}^T & G_{22}\end{pmatrix}\cdot\begin{pmatrix} H_{11} & H_{12} \\ H_{12}^T & H_{22}\end{pmatrix}=\begin{pmatrix} \mathit{II} & 0 \\ 0 & \mathit{II}\end{pmatrix},$$

G_{11}^{-1} is now computed using blocks of H as

$$G_{11}^{-1}=H_{11}-H_{12}H_{22}^{-1}H_{12}^T. \tag{6}$$

This is known as a form of the Schur complement. Instead of Eq. (5), our approach will be to solve

$$\mathbf{x}=\left(H_{11}-H_{12}H_{22}^{-1}H_{12}^T\right)\mathbf{b}, \tag{7}$$

where each block matrix of H is calculated as follows:

$$\begin{aligned} H_{11}&=\left[\frac{\partial\mathbf{s}}{\partial\mathbf{x}}\right][\mathrm{diag}(\tilde{\mathbf{m}})]\left[\frac{\partial\mathbf{s}}{\partial\mathbf{x}}\right]^T; \\ H_{12}&=\left[\frac{\partial\mathbf{s}}{\partial\mathbf{x}}\right][\mathrm{diag}(\tilde{\mathbf{m}})]\left[\frac{\partial\mathbf{h}}{\partial\mathbf{x}}\right]^T=H_{21}^T; \\ H_{22}&=\left[\frac{\partial\mathbf{h}}{\partial\mathbf{x}}\right][\mathrm{diag}(\tilde{\mathbf{m}})]\left[\frac{\partial\mathbf{h}}{\partial\mathbf{x}}\right]^T. \end{aligned} \tag{8}$$

For serial manipulators, the matrices in Eq. (8) can be computed efficiently due to their structural properties. In the case of the planar n-link serial chain with constrained link lengths,[22] the vector of generalized coordinates is partitioned as $\mathbf{q}=[\theta_1,\ldots,\theta_n,L_1,\ldots,L_n]^T$, where θ_i's are the joint angles (soft variables) and L_i's are the link lengths (hard variables). The analytical expressions for soft and hard variables are written as some functions of related position vectors, such that $\theta_i=f(\mathbf{x}_i,\mathbf{x}_{i-1},\mathbf{x}_{i-2})$ and $L_i=g(\mathbf{x}_i,\mathbf{x}_{i-1})$. Since $\partial\theta_i/\partial\mathbf{x}_j=0$ except of when $j=i,i-1,i-2$ and $\partial L_i/\partial\mathbf{x}_j=0$ except of when $j=i,i-1$, there are $(3n-2)$ nonzero elements in $[\frac{\partial\mathbf{s}}{\partial\mathbf{x}}]\in\mathbb{R}^{n\times 2n}$ and $(2n-1)$ nonzero entries in $[\frac{\partial\mathbf{h}}{\partial\mathbf{x}}]\in\mathbb{R}^{n\times 2n}$, noting that each element is a 1×2 row vector.

If matrices have $O(n)$ nonzero entries, matrix multiplication can be made in $O(n)$ computations by extracting zero elements. Once all the blocks of H (which are also sparse and band-limited) are computed, Eq. (7) can be calculated in $O(n)$ as well. There are $O(n)$ algorithms to solve $H_{22}^{-1}\mathbf{c}$ for some vector $\mathbf{c}$ with a matching dimension.[26] The computational steps for an n-link spatial manipulator composed of n point masses are summarized as follows:

I. Define a partitioning of the generalized coordinates as $\mathbf{q}=[\mathbf{s}^T,\mathbf{h}^T]^T$ where $\mathbf{s}=[s_1,\ldots,s_f]^T$ is the vector of soft variables and $\mathbf{h}=[h_1,\ldots,h_r]^T$ is the vector of hard variables, such that $f+r=3n$.[2]
II. Obtain analytical expressions for each variable in terms of position vectors, such that $s_i=f(\mathbf{x})$ and $h_j=g(\mathbf{x})$, where s_i and h_j denote the ith soft variable and the jth hard variable, respectively, and f and g are some functions of position vectors.[3]
III. Compute all nonzero entries of $[\frac{\partial\mathbf{s}}{\partial\mathbf{x}}]\in\mathbb{R}^{f\times 3n}$ and $[\frac{\partial\mathbf{h}}{\partial\mathbf{x}}]\in\mathbb{R}^{r\times 3n}$. There are $O(n)$ nonzero elements for each matrix. The elements of these matrices are:

$$\left[\frac{\partial\mathbf{s}}{\partial\mathbf{x}}\right]_{ij}=\frac{\partial s_i}{\partial\mathbf{x}_j}\in\mathbb{R}^{1\times 3};\qquad\left[\frac{\partial\mathbf{h}}{\partial\mathbf{x}}\right]_{ij}=\frac{\partial h_i}{\partial\mathbf{x}_j}\in\mathbb{R}^{1\times 3}.$$

IV. Compute H_{11}, $H_{12}(=H_{21}^T)$, and H_{22} using Eq. (8). Recall that matrix multiplications for band-limited matrices can be done in $O(n)$ by extracting zero elements from the matrices. (For example, the 'sparse(·)' command in Matlab stores all nonzero entries of a sparse matrix as an array.)
V. Compute Eq. (7) as follows:

$$\mathbf{c}=H_{12}^T\mathbf{b}\rightarrow\mathbf{d}=H_{22}^{-1}\mathbf{c}\rightarrow\mathbf{e}=H_{12}\mathbf{d}$$
$$\mathbf{f}=H_{11}\mathbf{b}$$

Finally,

$$(G_{11})^{-1}\mathbf{b}=\mathbf{f}-\mathbf{e}.$$

Multiplication of a sparse matrix with $O(n)$ nonzero entries with a vector with the corresponding size requires $O(n)$ computations. Note that H_{22} is also sparse and band-limited, and, therefore, $H_{22}^{-1}\mathbf{c}$ can be computed using an $O(n)$ algorithm, such as LU decomposition.[26]

3. Extension to Rigid Bodies

While Fixman's theorem represents a clever insight into how to directly exploit the serial nature of a chain consisting of point masses, the mathematics required is nothing more than multivariable calculus. This is because the positions of point masses are quantities that belong to $\mathbb{R}^3$, and

[2] For spatial chains with rigid-bodies, $f+r=6n$.
[3] In the rigid-body case, each generalized coordinate will be written as a function of homogeneous transformations, such that $s_i=f'(T_{i-1},T_i)$ and $h_j=g'(T_{j-1},T_j)$.

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

taking gradients in this space is a common mathematical operation. In contrast, it is not at all clear without invoking higher mathematics how to do the same for rigid bodies. In other words, whereas it makes sense to compute gradients of the form $\partial/\partial \mathbf{x}_i$ where $\mathbf{x}_i \in I\!R^3$, and the unconstrained Jacobian $[\partial \mathbf{x}/\partial \mathbf{q}]$ in Eq. (2) is square, when considering rigid bodies, would it mean anything to compute $\partial/\partial R_i$ where $R_i \in SO(3)$? Also, the dimensions of the associated Jacobians would certainly not be square given that rotation matrices have nine elements and only three free parameters. Hence, in this section we address how to compute derivatives in an appropriate way for functions of rotations and rigid-body motions in order to extend Fixman's approach.

3.1. Rotations and skew-symmetric matrices

To begin, recall that if R is a rotation matrix, then $R^T R = I\!I_3$, and

$$\frac{d}{dt}(R^T R) = \frac{d}{dt}(I\!I_3) = 0_3,$$

and so

$$R^T \dot{R} = -\dot{R}^T R = -(R^T \dot{R})^T.$$

Due to the skew-symmetry of this matrix, we can write $\boldsymbol{\omega} = (R^T \dot{R})^\vee$, where the operator '$\vee$' is defined by $(S)^\vee = \mathbf{s}$ where $\mathbf{s} = [s_1, s_2, s_3]^T$ and

$$S = \begin{pmatrix} 0 & -s_3 & s_2 \\ s_3 & 0 & -s_1 \\ -s_2 & s_1 & 0 \end{pmatrix} = -S^T.$$

Any 3×3 skew-symmetric matrix, S, can be written as $S = \sum_{i=1}^{3} s_i E_i$ where

$$E_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}; \quad E_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix};$$

$$E_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

These can be written in a vector form as $(E_i)^\vee = \mathbf{e}_i$, such that

$$\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}; \quad \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}; \quad \mathbf{e}_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Then it follows that $\mathbf{s} = \sum_{i=1}^{3} s_i \mathbf{e}_i$ since the $\vee$ operation is linear. We will use this fact later.

If the vector $\boldsymbol{\omega}$ is the angular velocity as seen in a body-fixed frame of reference, the kinetic energy of a rigid-body is then

$$K = \frac{1}{2} m \dot{\mathbf{x}} \cdot \dot{\mathbf{x}} + \frac{1}{2} \boldsymbol{\omega}^T I \boldsymbol{\omega} \tag{9}$$

where I is the constant moment of inertia matrix as seen in the specific body-fixed frame with origin at the center of mass, and $\mathbf{x}$ is the position of the center of mass of the rigid body as seen in a space-fixed frame of reference. The following subsections develop the mathematical framework needed to handle the rotational contribution to kinetic energy in our extension of Fixman's theorem.

3.2. Jacobians associated with parameterized rotations

When a time-varying rotation matrix is parameterized as[4]

$$R(t) = A(q_1(t), q_2(t), q_3(t)) = A(\mathbf{q}(t)),$$

then by the chain rule from calculus, one has

$$\dot{R} = \frac{\partial A}{\partial q_1}\dot{q}_1 + \frac{\partial A}{\partial q_2}\dot{q}_2 + \frac{\partial A}{\partial q_3}\dot{q}_3.$$

Multiplying on the left by R^T and extracting the dual vector from both sides, one finds that:[27]

$$\boldsymbol{\omega} = J(A(\mathbf{q}))\dot{\mathbf{q}} \tag{10}$$

where

$$J(A(\mathbf{q})) = \left[\left(A^T \frac{\partial A}{\partial q_1} \right)^\vee, \left(A^T \frac{\partial A}{\partial q_2} \right)^\vee, \left(A^T \frac{\partial A}{\partial q_3} \right)^\vee \right] \tag{11}$$

which is called the 'body' Jacobian. When using the ZXZ Euler angle parameterization (α, β, γ), the Jacobian is written explicitly as:[27]

$$J = \begin{pmatrix} \sin\beta \sin\gamma & \cos\gamma & 0 \\ \sin\beta \cos\gamma & -\sin\gamma & 0 \\ \cos\beta & 0 & 1 \end{pmatrix}. \tag{12}$$

3.3. Differential operators for SO(3)

Let $A \in SO(3)$ be an arbitrary rotation, and $f(A)$ be a function that assigns a real or complex number to each value of A. In analogy with the definition of the partial derivative (or directional derivative) of a complex-valued function of $I\!R^N$-valued argument, we can define differential operators which act on functions of rotation-valued argument:

$$\frac{\partial}{\partial \xi^{\mathbf{n}}} f(A) = \lim_{\epsilon \to 0} \frac{1}{\epsilon} [f(A \cdot A_{\mathbf{n}}(\epsilon)) - f(A)]$$
$$= \left. \frac{df(A \cdot A_{\mathbf{n}}(t))}{dt} \right|_{t=0} \tag{13}$$

where $A_{\mathbf{n}}(\theta)$ denotes a counterclockwise rotation by an angle θ around an axis defined by the unit vector $\mathbf{n}$. In the above definition, the dummy variable ξ is introduced to emphasize that the derivative is not with respect to $\mathbf{n}$, but rather the derivative along a coordinate defined by the direction $\mathbf{n}$.[5] Note that for a small value of θ,

$$A_{\mathbf{n}}(\theta) = \exp(\theta N) \approx I\!I + \theta N = I\!I + \theta(n_1 E_1 + n_2 E_2 + n_3 E_3).$$

[4] We use the different symbols $R(t)$ and $A(q)$, because these functions have different arguments even though $R(t) = A(q(t))$.
[5] In the Lie theory literature, the derivative $\partial f/\partial \xi^{\mathbf{n}}$ would be denoted Nf, which may be confusing for an engineering audience.

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

We now find the explicit forms of the operators $\frac{\partial}{\partial \xi^{\mathbf{n}}}$ in any 3-parameter description of rotation $A = A(q_1, q_2, q_3)$. Expanding Eq. (13) in a Taylor series in ϵ and using the classical chain rule, one writes

$$\frac{\partial f(A)}{\partial \xi^{\mathbf{n}}} = \sum_{i=1}^{3} \frac{\partial f(A)}{\partial q_i} \frac{\partial q_i^r}{\partial \epsilon}\bigg|_{\epsilon=0}$$

where $\{q_i^r\}$ are the parameters such that $A(q_1, q_2, q_3) \cdot A_{\mathbf{n}}(\epsilon) = A(q_1^r, q_2^r, q_3^r)$. The 'r' denotes the fact that each q_i is perturbed by multiplication of $A(q)$ on the right by $A_{\mathbf{n}}(\epsilon)$. At this point, the coefficients $\frac{\partial q_i^r}{\partial \epsilon}|_{\epsilon=0}$ is not known, but can be determined by observing two different-looking, though equivalent, ways of writing the product $A \cdot A_{\mathbf{n}}(\epsilon)$ for infinitesimally small ϵ:

$$A \cdot A_{\mathbf{n}}(\epsilon) = A + \epsilon A \cdot N = A + \epsilon \sum_{i=1}^{3} \frac{\partial A}{\partial q_i} \frac{\partial q_i^r}{\partial \epsilon}\bigg|_{\epsilon=0}.$$

The first equality results from direct multiplication of A and $A_n(\epsilon) = \mathit{I\!I} + \theta N$, and the second equality results from expanding $A(q^r)$ in a Taylor series about $\epsilon = 0$. From the above equation, we have that

$$N = \sum_{i=1}^{3} A^T \frac{\partial A}{\partial q_i} \frac{\partial q_i^r}{\partial \epsilon}\bigg|_{\epsilon=0},$$

or, using the linearity of the $\vee$ operator,

$$\mathbf{n} = (N)^{\vee} = \sum_{i=1}^{3} \left(A^T \frac{\partial A}{\partial q_i} \right)^{\vee} \frac{\partial q_i^r}{\partial \epsilon}\bigg|_{\epsilon=0},$$

which is rewritten using the definition of the Jacobian (Eq. (11)) as $\mathbf{n} = J \frac{\partial \mathbf{q}^r}{\partial \epsilon}|_{\epsilon=0}$. This allows us to solve for

$$\frac{\partial \mathbf{q}^r}{\partial \epsilon}\bigg|_{\epsilon=0} = J^{-1}\mathbf{n}.$$

For example, if A is parameterized with ZXZ Euler angles, J is the Jacobian calculated in Eq. (12), and its inverse is

$$J^{-1} = \begin{pmatrix} \sin\gamma/\sin\beta & \cos\gamma/\sin\beta & 0 \\ \cos\gamma & -\sin\gamma & 0 \\ -\cot\beta\sin\gamma & -\cot\beta\cos\gamma & 1 \end{pmatrix}.$$

Making the shorthand notation $\frac{\partial}{\partial \xi^{\mathbf{e}_i}} = \frac{\partial}{\partial \xi^i}$, we then write for the ZXZ Euler angles

$$\frac{\partial}{\partial \xi^1} = -\cot\beta\sin\gamma \frac{\partial}{\partial \gamma} + \frac{\sin\gamma}{\sin\beta}\frac{\partial}{\partial \alpha} + \cos\gamma \frac{\partial}{\partial \beta};$$

$$\frac{\partial}{\partial \xi^2} = -\cot\beta\cos\gamma \frac{\partial}{\partial \gamma} + \frac{\cos\gamma}{\sin\beta}\frac{\partial}{\partial \alpha} - \sin\gamma \frac{\partial}{\partial \beta};$$

$$\frac{\partial}{\partial \xi^3} = \frac{\partial}{\partial \gamma}.$$

The exact form of the differential operators will depend on the specific parameterization used. Each different parameterization will result in different concrete forms of these abstract operators.

3.4. Infinitesimal motions and associated jacobians

For "small" motions, the matrix exponential description of a rigid-body motion is approximated well when truncated at the first two terms:

$$\exp\left[\begin{pmatrix} \Omega & \mathbf{v} \\ \mathbf{0}^T & 0 \end{pmatrix} \Delta t\right] \approx \mathit{I\!I} + \begin{pmatrix} \Omega & \mathbf{v} \\ \mathbf{0}^T & 0 \end{pmatrix} \Delta t. \qquad (14)$$

Here $\Omega = -\Omega^T$ and $\boldsymbol{\omega} = \Omega^{\vee}$ describe the rotational part of the displacement. Since the second term of the right side in Eq. (14) consists mostly of zeros, it is common to extract the information necessary to describe the motion as

$$\begin{pmatrix} \Omega & \mathbf{v} \\ \mathbf{0}^T & 0 \end{pmatrix}^{\vee} = \begin{pmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{pmatrix} \in \mathbb{R}^{6\times 1}.$$

This six-dimensional vector is called an *infinitesimal screw motion* or *infinitesimal twist*. The fact that we have used the $\vee$ operation to extract a six-dimensional vector from 4×4 "screw matrices" as well as using it to extract a three-dimensional vector from 3×3 skew-symmetric matrices should not be a source of concern, since its use will always be clear from the context.

Given a homogeneous transform

$$T(\mathbf{q}) = \begin{pmatrix} R(\mathbf{q}) & \mathbf{b}(\mathbf{q}) \\ \mathbf{0}^T & 1 \end{pmatrix}$$

parameterized with $\mathbf{q} = [q_1, \ldots, q_6]^T \in \mathbb{R}^6$, one can express the homogeneous transform corresponding to a slightly changed set of parameters as the truncated Taylor series

$$T(\mathbf{q} + \delta\mathbf{q}) = T(\mathbf{q}) + \sum_{i=1}^{6} \delta q_i \frac{\partial T}{\partial q_i}(\mathbf{q}).$$

This result can be shifted to the identity transformation by multiplying on the left by T^{-1} to define an equivalent relative infinitesimal motion:

$$\begin{pmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{pmatrix} = \mathcal{J}(\mathbf{q})\dot{\mathbf{q}}$$

$$\text{where} \quad \mathcal{J}(\mathbf{q}) = \left[\left(T^{-1}\frac{\partial T}{\partial q_1} \right)^{\vee}, \ldots, \left(T^{-1}\frac{\partial T}{\partial q_6} \right)^{\vee} \right]. \qquad (15)$$

3.5. Differential operators for SE(3)

The differential operators $\partial/\partial\tilde{\xi}_i$ for $i = 1, \ldots, 6$ acting on functions on $SE(3)$ are calculated similarly with the case of $SO(3)$. For small translational and rotational displacements from the identity along (or about) the ith coordinate axis, the homogeneous transforms representing infinitesimal motions are given by

$$T_i(\epsilon) \stackrel{\triangle}{=} \exp(\epsilon \tilde{E}_i) \approx \mathit{I\!I}_{4\times 4} + \epsilon \tilde{E}_i$$

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

where

$$\tilde{E}_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}; \quad \tilde{E}_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix};$$

$$\tilde{E}_3 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}; \quad \tilde{E}_4 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix};$$

$$\tilde{E}_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}; \quad \tilde{E}_6 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

The "tilde" symbol is used to distinguish the $SE(3)$ basis elements from those for $SO(3)$. It is often convenient to write the $SE(3)$ basis elements in a 6×1 vector form as $(\tilde{E}_i)^\vee = \tilde{\mathbf{e}}_i$, such that all elements are zeros except for the ith element with 1.

Given that elements of $SE(3)$ (viewed as homogeneous transforms) are parameterized as $T = T(\mathbf{q})$, the differential operators take the form

$$\frac{\partial}{\partial \tilde{\xi}^i} f(T) = \lim_{\epsilon \to 0} \frac{1}{\epsilon} [f(T \cdot T_i(\epsilon)) - f(T)] = \left. \frac{df(T \cdot (I\!I + t\tilde{E}_i))}{dt} \right|_{t=0} \tag{16}$$

Since T and $T_i(\epsilon)$ are 4×4 matrices, we henceforth drop the "$\cdot$" notation since it is understood as matrix multiplication.

In analogy with the $SO(3)$ case, we define $q^{r,i}$ such that $T(q)T_i(\epsilon) = T(q^{r,i})$, and we observe for the case of $\frac{\partial}{\partial \tilde{\xi}^i}$ that

$$T + \epsilon T \tilde{E}_i = T + \epsilon \sum_{j=1}^{6} \left. \frac{\partial T}{\partial q_j} \frac{\partial q_j^{r,i}}{\partial \epsilon} \right|_{\epsilon=0}.$$

In analogy with the $SO(3)$ case, these are two equivalent ways of writing $T\, T_i(\epsilon)$. Subtracting T and multiplying T^{-1} on the left of this expression, we then have that

$$\tilde{E}_i = \sum_{j=1}^{6} \left. T^{-1} \frac{\partial T}{\partial q_j} \frac{\partial q_j^{r,i}}{\partial \epsilon} \right|_{\epsilon=0},$$

or

$$\tilde{\mathbf{e}}_i = \sum_{j=1}^{6} \left. \left(T^{-1} \frac{\partial T}{\partial q_j} \right)^\vee \frac{\partial q_j^{r,i}}{\partial \epsilon} \right|_{\epsilon=0},$$

which is written as $\tilde{\mathbf{e}}_i = \mathcal{J}(\mathbf{q}) \frac{d\mathbf{q}^{r,i}}{d\epsilon}|_{\epsilon=0}$ where $\mathcal{J}$ is the $SE(3)$ Jacobian defined in Section 3.4. This allows us to solve for

$$\left. \frac{d\mathbf{q}^{r,i}}{d\epsilon} \right|_{\epsilon=0} = \mathcal{J}^{-1} \tilde{\mathbf{e}}_i,$$

which is used to calculate

$$\frac{\partial f}{\partial \tilde{\xi}^i} = \sum_{j=1}^{6} \left. \frac{\partial f}{\partial q_j} \frac{\partial q_j^{r,i}}{\partial \epsilon} \right|_{\epsilon=0}$$

as

$$\frac{\partial f}{\partial \tilde{\xi}^i} = \left(\frac{\partial f}{\partial \mathbf{q}} \right) \cdot (\mathcal{J}^{-1} \tilde{\mathbf{e}}_i). \tag{17}$$

3.6. Extension to chains of rigid bodies

Eq. (9) can be rewritten as

$$K = \frac{1}{2} (\boldsymbol{\omega}^T, \mathbf{v}^T) \mathcal{I} \begin{pmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{pmatrix} = \frac{1}{2} \dot{\mathbf{q}}^T \mathcal{J}^T \mathcal{I} \mathcal{J} \dot{\mathbf{q}}.$$

The mass matrix is $G(q) = \mathcal{J}^T \mathcal{I} \mathcal{J}$, and thus, the inverse of the mass matrix for a single rigid body is

$$H(\mathbf{q}) = \mathcal{J}^{-1} \mathcal{I}^{-1} \mathcal{J}^{-T}$$

where $\mathcal{I} = I \oplus (m I\!I_3)$ is the 6×6 inertia matrix.

The inverse of the mass matrix can be rewritten using the derivatives defined in the previous section. Particularly, if we define the $SE(3)$ gradient of a function to be

$$\frac{\partial f}{\partial \tilde{\boldsymbol{\xi}}} = \left[\frac{\partial f}{\partial \tilde{\xi}^1}, \ldots, \frac{\partial f}{\partial \tilde{\xi}^6} \right],$$

then we can apply this gradient to the parameters $\mathbf{q} = [q_1, \ldots, q_6]^T$ used to parameterize a motion. Using Eq. (17) and observing that $\frac{\partial \mathbf{q}}{\partial \mathbf{q}} = I\!I$, we find that

$$\left[\frac{\partial \mathbf{q}}{\partial \tilde{\boldsymbol{\xi}}} \right] = \mathcal{J}^{-1}. \tag{18}$$

This means that the inverse of the mass matrix for a single rigid-body can be written in the Fixman-like form:

$$H(\mathbf{q}) = \left[\frac{\partial \mathbf{q}}{\partial \tilde{\boldsymbol{\xi}}} \right] \mathcal{I}^{-1} \left[\frac{\partial \mathbf{q}}{\partial \tilde{\boldsymbol{\xi}}} \right]^T. \tag{19}$$

For a collection of n rigid bodies, the configuration space is $(SE(3))^n = SE(3) \times SE(3) \times \cdots \times SE(3)$. Each rigid body has six degrees of freedom described by twists, the ith of which is $\tilde{\boldsymbol{\xi}}_i \in I\!R^6$, and can be described alternatively by the six parameters $\mathbf{q}_i \in I\!R^6$. Composite vectors $\tilde{\boldsymbol{\xi}} = [\tilde{\boldsymbol{\xi}}_1^T, \ldots, \tilde{\boldsymbol{\xi}}_n^T]^T \in I\!R^{6n}$ and $\mathbf{q} = [\mathbf{s}^T, \mathbf{h}^T]^T \in I\!R^{6n}$ can be formed. Then, the inverse of the Jacobian is computed as

$$\mathcal{J}_\oplus^{-1} = \begin{pmatrix} \left[\frac{\partial \mathbf{s}}{\partial \tilde{\xi}} \right] \\ \left[\frac{\partial \mathbf{h}}{\partial \tilde{\xi}} \right] \end{pmatrix} \in I\!R^{6n \times 6n}.$$

$\mathcal{J}_\oplus^{-1}$ is the inverse of the unconstrained serial chain when the generalized coordinates are partitioned into the soft and hard variables. Hence, it differs from the direct sum of the inverse of Jacobian matrices, $\mathcal{J}_1^{-1} \oplus \cdots \oplus \mathcal{J}_n^{-1}$, where $\mathcal{J}_i^{-1}$ denotes the inverse Jacobian of the ith rigid-body. The inverse of the

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

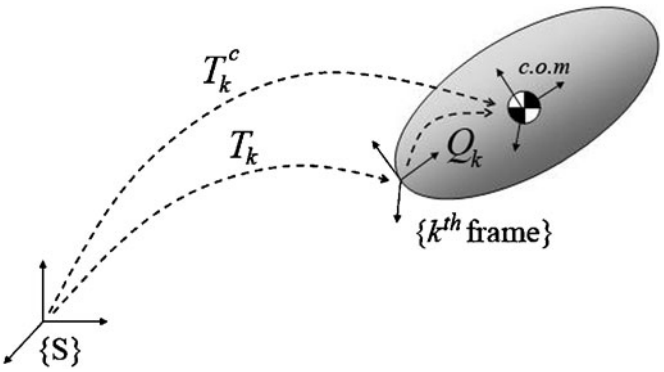

Fig. 1. T_k^c is the homogeneous transformation from the global reference frame {S} to the center of mass of the kth rigid body, and T_k is the one from {S} to the frame k. Q_k is the relative transformation from T_k to T_k^c.

unconstrained mass matrix for this collection of rigid bodies is then of the form

$$H(\mathbf{q}) = \mathcal{J}_{\oplus}^{-1}\left[\mathcal{I}_1^{-1} \oplus \cdots \oplus \mathcal{I}_n^{-1}\right]\mathcal{J}_{\oplus}^{-T}. \tag{20}$$

Everything then follows using the extension of Fixman's theorem as in the point-mass case, with Eq. (20) replacing Eq. (4). The same partitioning into soft and hard variables and the same $O(n)$ performance results.

4. Examples

Examples of an n-link planar revolute manipulator and a polymer chain composed of point masses at each joint are presented in our earlier papers.[22,23] In this section, we describe how to use our extension of Fixman's algorithm for chains of rigid-bodes and demonstrate with a PUMA 560 robot arm and a polypeptide chain.

The homogeneous transformations, T_k, T_k^c and Q_k, are defined as shown in Fig. 1. Using the facts that $T_k^c = T_k \cdot Q_k$ and $T_k = T_1^0 \cdot T_2^1 \cdots T_k^{k-1}$ due to the serial nature, we have that

$$T_k^{k-1} = (T_{k-1})^{-1} \cdot T_k = Q_{k-1}\left(T_{k-1}^c\right)^{-1} T_k^c Q_k^{-1}. \tag{21}$$

Since T_k^{k-1} is written as a function of T_{k-1}^c and T_k^c, in order to differentiate T_{k-1}^c, we use the differential operators for $SE(3)$ described in Section 3.5. Then, we have the Lie derivatives of Eq. (21) given by[6]

$$\frac{\partial T_k^{k-1}}{\partial \tilde{\xi}_j^i} = \begin{cases} -Q_{k-1}\tilde{E}_i\left(T_{k-1}^c\right)^{-1} T_k^c Q_k^{-1}, & j = k-1 \\ Q_{k-1}\left(T_{k-1}^c\right)^{-1} T_k^c \tilde{E}_i Q_k^{-1}, & j = k \\ 0, & \text{otherwise.} \end{cases} \tag{22}$$

[6] For $T \in SE(3)$,

$$T \cdot T^{-1} = I\!I_4 \Rightarrow \frac{dT}{dt}T^{-1} + T\frac{d(T^{-1})}{dt} = 0_4 \Rightarrow \frac{d(T^{-1})}{dt} = -T^{-1}\frac{dT}{dt}T^{-1}.$$

In our algorithm, we compute the inverted Jacobian of an unconstrained system to get the inverse of an unconstrained mass matrix, i.e., $G^{-1} = \mathcal{J}_{\oplus}^{-1}\mathcal{I}^{-1}\mathcal{J}_{\oplus}^{-T}$. However, most manipulators have singularities where the Jacobian matrix is not invertible. In contrast, the mass matrix for most manipulators is invertible at all values of the generalized coordinates, and therefore, there always exists G^{-1}. The problems related with singularities can be mostly eliminated by choosing the parameterization carefully or using more than one method for assigning frames, because the Jacobians computed for different parameterizations may have singularities in different locations. We begin by describing rigid-body motions using D–H framework in the following section.

4.1. Denavit–Hartenberg parameterizations

A screw transformation is a combined rotation and translation along a common axis. In particular,

$$\text{screw}(n, d, \theta) = \begin{pmatrix} A_{\mathbf{n}}(\theta) & d\mathbf{n} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix}$$

where $\mathbf{n} \in I\!R^3$ is any unit vector. D–H parameterization is a method for assigning frames of reference to a robot arm constructed of rotational joints connected with rigid links. The relative transformation from the D–H frame $i-1$ to the D–H frame i appear as two screw motions, such that

$$T_i^{i-1} = \text{screw}(\mathbf{e}_1, b_i, \beta_i) \cdot \text{screw}(\mathbf{e}_3, c_i, \gamma_i) = \begin{pmatrix} c\gamma_i & -s\gamma_i & 0 & b_i \\ s\gamma_i c\beta_i & c\gamma_i c\beta_i & -s\beta_i & -c_i s\beta_i \\ s\gamma_i s\beta_i & c\gamma_i s\beta_i & c\beta_i & c_i c\beta_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{23}$$

where $c(\cdot) = \cos(\cdot)$ and $s(\cdot) = \sin(\cdot)$ are used for a shorthand writing. Here $\mathbf{e}_i$'s are as defined in Section 3.1. The link parameters are defined as follows:[28]

- b_i, the distance from $\hat{Z}_{i-1}$ to $\hat{Z}_i$ measured along $\hat{X}_{i-1}$,
- β_i, the counter clockwise measured angle between $\hat{Z}_{i-1}$ and $\hat{Z}_i$ measured about $\hat{X}_{i-1}$,
- c_i, the distance from $\hat{X}_{i-1}$ to $\hat{X}_i$ measured along $\hat{Z}_i$, and
- γ_i, the counter clockwise measured angle between $\hat{X}_{i-1}$ and $\hat{X}_i$ measured about $\hat{Z}_i$.

The above four parameters describe constrained motions in $SE(3)$. In other words, we need six parameters to describe a full rigid-body motion in $SE(3)$, but we only have four D–H parameters. Therefore, we impose two dummy variables (which will be set to be zeros as constraints) to the D–H parameters as follows. Equation (23) can be obtained by setting $\alpha_i = a_i = 0$ from either

$${}^A T_i^{i-1} = \text{screw}(\mathbf{e}_3, a_i, \alpha_i) \cdot \text{screw}(\mathbf{e}_1, b_i, \beta_i) \cdot \text{screw}(\mathbf{e}_3, c_i, \gamma_i), \tag{24}$$

$${}^B T_i^{i-1} = \text{screw}(\mathbf{e}_1, b_i, \beta_i) \cdot \text{screw}(\mathbf{e}_3, \mathrm{c_i}, \gamma_i) \cdot \text{screw}(\mathbf{e}_1, a_i, \alpha_i), \tag{25}$$

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

or

$$ {}^{C}T_i^{i-1} = \text{screw}(\mathbf{e}_1, b_i, \beta_i) \cdot \text{screw}(\mathbf{e}_2, a_i, \alpha_i) \cdot \text{screw}(\mathbf{e}_3, c_i, \gamma_i). \quad (26) $$

If $\mathcal{J}^A$, $\mathcal{J}^B$ and $\mathcal{J}^C$ are the Jacobian matrices of Eq. (24), (25), and (26) respectively, then the determinant of each Jacobian is calculated as

$$ \det(\mathcal{J}^A) = \sin^2\beta_i; \quad \det(\mathcal{J}^B) = \sin^2\gamma_i; \quad \det(\mathcal{J}^C) = \cos^2\alpha_i. $$

$\mathcal{J}^A$ becomes singular when $\sin\beta_i = 0$, $\mathcal{J}^B$ becomes singular when $\sin\gamma_i = 0$, and $\mathcal{J}^C$ becomes singular when $\cos\alpha_i = 0$. In the D–H parameterization, we set $\alpha_i = 0$, and therefore, $\cos\alpha_i \neq 0$ for all i. Hence, we choose Eq. (26) to describe homogeneous transformations of rigid bodies in $SE(3)$.

We first need to find analytical expressions for generalized coordinates in terms of homogeneous transformations. If the relative transformation from the frame $k-1$ to the frame k is represented by three screw motions as Eq. (26), it can be written as

$$ {}^{C}T_k^{k-1} = \begin{pmatrix} c\alpha_k c\gamma_k & -c\alpha_k s\gamma_k & s\alpha_k & b_k + c_k s\alpha_k \\ s\beta_k s\alpha_k c\gamma_k + c\beta_k s\gamma_k & c\beta_k c\gamma_k - s\beta_k s\alpha_k s\gamma_k & -s\beta_k c\alpha_k & a_k c\beta_k - c_k s\beta_k c\alpha_k \\ -c\beta_k s\alpha_k c\gamma_k + s\beta_k s\gamma_k & s\beta_k c\gamma_k + c\beta_k s\alpha_k s\gamma_k & c\beta_k c\alpha_k & a_k s\beta_k + c_k c\beta_k c\alpha_k \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (27) $$

Making the shorthand notation, ${}^{C}T_k^{k-1} = T$ and denoting the (k,l)th element of ${}^{C}T_k^{k-1}$ by T_{kl}, when $\alpha_k \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, $\beta_k \in [-\pi, \pi]$ and $\gamma_k \in [-\pi, \pi]$, the generalized coordinates can be extracted as:[7]

$$ \alpha_k = \text{atan}\left(\frac{T_{13}}{\sqrt{T_{11}^2 + T_{12}^2}}\right); $$

$$ \beta_k = \text{atan2}\left(-\frac{T_{23}}{\sqrt{T_{11}^2 + T_{12}^2}}, \quad \frac{T_{33}}{\sqrt{T_{11}^2 + T_{12}^2}}\right); $$

$$ \gamma_k = \text{atan2}\left(-\frac{T_{12}}{\sqrt{T_{11}^2 + T_{12}^2}}, \quad \frac{T_{11}}{\sqrt{T_{11}^2 + T_{12}^2}}\right); $$

$$ a_k = \frac{T_{24}T_{33} - T_{34}T_{23}}{\sqrt{T_{11}^2 + T_{12}^2}}; $$

$$ b_k = T_{14} - T_{13}\frac{T_{24}T_{23} + T_{34}T_{33}}{T_{11}^2 + T_{12}^2}; $$

$$ c_k = \frac{T_{24}T_{23} + T_{34}T_{33}}{T_{11}^2 + T_{12}^2}. $$

[7] In Matlab™, two functions compute the inverse of the tangent, 'atan' and 'atan2'. atan(z) returns the inverse tangent of z. For real z, atan(x) is in the range $[-\pi/2, \pi/2]$. atan2(y, x) gives the value of θ, such that $\sin\theta = y$ and $\cos\theta = x$. The value of θ lies in the interval $[-\pi, \pi]$. In Mathematica™, the same functions are defined as ArcTan[z] and ArcTan[x, y], respectively.

If $\tilde{T}_{kl}$ denotes the (k,l)th element of $\frac{\partial T}{\partial \tilde{\xi}_j^i}$, the Lie derivatives of the above equations are computed as:

$$ \frac{\partial \alpha_k}{\partial \tilde{\xi}_j^i} = \frac{(T_{11}^2 + T_{12}^2)\tilde{T}_{13} - T_{13}(T_{11}\tilde{T}_{11} + T_{12}\tilde{T}_{12})}{\sqrt{T_{11}^2 + T_{12}^2}}; $$

$$ \frac{\partial \beta_k}{\partial \tilde{\xi}_j^i} = \frac{T_{23}\tilde{T}_{33} - T_{33}\tilde{T}_{23}}{T_{23}^2 + T_{33}^2}; $$

$$ \frac{\partial \gamma_k}{\partial \tilde{\xi}_j^i} = \frac{T_{12}\tilde{T}_{11} - T_{11}\tilde{T}_{12}}{T_{11}^2 + T_{12}^2}; $$

$$ \frac{\partial a_k}{\partial \tilde{\xi}_j^i} = \frac{T_{24}\tilde{T}_{33} + T_{33}\tilde{T}_{24} - T_{34}\tilde{T}_{23} - T_{23}\tilde{T}_{34}}{\sqrt{T_{11}^2 + T_{12}^2}} - \frac{(T_{24}T_{33} - T_{34}T_{23})(T_{11}\tilde{T}_{11} + T_{12}\tilde{T}_{12})}{(T_{11}^2 + T_{12}^2)^{3/2}}; $$

$$ \frac{\partial b_k}{\partial \tilde{\xi}_j^i} = \tilde{T}_{14} - (T_{24}T_{23} + T_{34}T_{33}) \times \left[\frac{(T_{11}^2 + T_{12}^2)\tilde{T}_{13} - 2T_{13}(T_{11}\tilde{T}_{11} + T_{12}\tilde{T}_{12})}{(T_{11}^2 + T_{12}^2)^2}\right] - \frac{T_{13}(T_{24}\tilde{T}_{23} + T_{23}\tilde{T}_{24} + T_{34}\tilde{T}_{33} + T_{33}\tilde{T}_{34})}{T_{11}^2 + T_{12}^2}; $$

$$ \frac{\partial c_k}{\partial \tilde{\xi}_j^i} = \frac{T_{24}\tilde{T}_{23} + T_{23}\tilde{T}_{24} + T_{34}\tilde{T}_{33} + T_{33}\tilde{T}_{34}}{T_{11}^2 + T_{12}^2} - \frac{2(T_{23}T_{24} + T_{33}T_{34})(T_{11}\tilde{T}_{11} + T_{12}\tilde{T}_{12})}{(T_{11}^2 + T_{12}^2)^2}. \quad (28) $$

These are elements of $\mathcal{J}_{\oplus}^{-1}$. Soft and hard variables are defined differently depending on each system. In the following sections, we present specific examples of the PUMA 560 and a polypeptide chain.

4.2. PUMA 560 robot arm

D–H parameters for the PUMA 560 arm are shown in Table I. The vector of generalized coordinates is defined as $\mathbf{q} = [\mathbf{s}^T; \mathbf{h}^T]^T$ where $\mathbf{s} = [\gamma_1, \ldots, \gamma_6]^T$ and $\mathbf{h} = [\mathbf{h}_1^T, \mathbf{h}_2^T, \ldots, \mathbf{h}_5^T, \mathbf{h}_6^T]^T$ with $\mathbf{h}_i = [\alpha_i, \beta_i, a_i, b_i, c_i]^T$ for $i = 1, \ldots, 6$. Here γ_i denotes the ith joint angle of the manipulator. All necessary constants including the link mass values, the moment of inertia about the center of mass (c.o.m) and the location of c.o.m of each link (Q_k's) were adapted

Table I. D–H parameters of PUMA 560 Arm.[29]

i	β_i[°]	b_i[m]	c_i[m]	γ_i[°]
1	90	0	0	γ_1
2	0	0.4318	0	γ_2
3	−90	0.0191	0.1254	γ_3
4	90	0	0.4318	γ_4
5	−90	0	0	γ_5
6	0	0	0	γ_6

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

from Ref. [29]. The soft variables are arbitrarily chosen as $\gamma_k = \pi/3$ for $k = 1, \ldots, 6$ for the purpose of testing our algorithm. Based on these parameters, the homogeneous transformations from the global reference frame to the center of mass of each link are computed as

$$T_1^c = \begin{pmatrix} 0.500 & -0.866 & 0 & -0.069 \\ 0 & 0 & -1 & 0 \\ 0.866 & 0.5 & 0 & 0.04 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_2^c = \begin{pmatrix} -0.5 & -0.866 & 0 & 0.108 \\ 0 & 0 & -1 & 0.026 \\ 0.866 & -0.5 & 0 & 0.561 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_3^c = \begin{pmatrix} -0.25 & 0.433 & -0.866 & 0.004 \\ 0.866 & 0.5 & 0 & -0.108 \\ 0.433 & -0.75 & -0.5 & 0.490 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_4^c = \begin{pmatrix} -0.875 & -0.217 & -0.433 & -0.081 \\ 0.433 & -0.75 & -0.5 & -0.206 \\ -0.217 & -0.625 & 0.75 & 0.637 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_5^c = \begin{pmatrix} -0.063 & 0.974 & -0.217 & -0.089 \\ 0.650 & -0.125 & -0.75 & -0.216 \\ -0.758 & -0.188 & -0.625 & 0.652 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_6^c = \begin{pmatrix} 0.813 & 0.541 & -0.217 & -0.091 \\ 0.217 & -0.625 & -0.75 & -0.223 \\ -0.541 & 0.563 & -0.625 & 0.645 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Given the homogeneous transformations, nonzero elements of the following matrices can be computed using in Eq. (28):

$$\left[\frac{\partial \mathbf{s}}{\partial \tilde{\xi}}\right] = \begin{pmatrix} \frac{\partial \gamma_1}{\partial \tilde{\xi}_1} & 0_{1\times 6} & \cdots & \cdots & \cdots & 0_{1\times 6} \\ \frac{\partial \gamma_2}{\partial \tilde{\xi}_1} & \frac{\partial \gamma_2}{\partial \tilde{\xi}_2} & \ddots & \ddots & \ddots & \vdots \\ 0_{1\times 6} & \frac{\partial \gamma_3}{\partial \tilde{\xi}_2} & \frac{\partial \gamma_3}{\partial \tilde{\xi}_3} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \frac{\partial \gamma_4}{\partial \tilde{\xi}_3} & \frac{\partial \gamma_4}{\partial \tilde{\xi}_4} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \frac{\partial \gamma_5}{\partial \tilde{\xi}_4} & \frac{\partial \gamma_5}{\partial \tilde{\xi}_5} & 0_{1\times 6} \\ 0_{1\times 6} & \cdots & \cdots & 0_{1\times 6} & \frac{\partial \gamma_6}{\partial \tilde{\xi}_5} & \frac{\partial \gamma_6}{\partial \tilde{\xi}_6} \end{pmatrix} \in I\!R^{6\times 36};$$

$$\left[\frac{\partial \mathbf{h}}{\partial \tilde{\xi}}\right] = \begin{pmatrix} \frac{\partial \mathbf{h}_1}{\partial \tilde{\xi}_1} & 0_{5\times 6} & \cdots & \cdots & \cdots & 0_{5\times 6} \\ \frac{\partial \mathbf{h}_2}{\partial \tilde{\xi}_1} & \frac{\partial \mathbf{h}_2}{\partial \tilde{\xi}_2} & \ddots & \ddots & \ddots & \vdots \\ 0_{5\times 6} & \frac{\partial \mathbf{h}_3}{\partial \tilde{\xi}_2} & \frac{\partial \mathbf{h}_3}{\partial \tilde{\xi}_3} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \frac{\partial \mathbf{h}_4}{\partial \tilde{\xi}_3} & \frac{\partial \mathbf{h}_4}{\partial \tilde{\xi}_4} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \frac{\partial \mathbf{h}_5}{\partial \tilde{\xi}_4} & \frac{\partial \mathbf{h}_5}{\partial \tilde{\xi}_5} & 0_{5\times 6} \\ 0_{5\times 6} & \cdots & \cdots & 0_{5\times 6} & \frac{\partial \mathbf{h}_6}{\partial \tilde{\xi}_5} & \frac{\partial \mathbf{h}_6}{\partial \tilde{\xi}_6} \end{pmatrix} \in I\!R^{30\times 36}$$

Table II. D–H parameters of a polypeptide chain.[30]

i	β_i[°]	b_i[Å]	c_i[Å]	γ_i[°]
1	β_1	0	0	70.5
2	β_2	1.525	0	296.2
3	β_3	1.329	0	58.3
4	β_4	1.458	0	289.5
5	β_5	1.525	0	63.8
6	β_6	1.329	0	301.7
7	β_7	1.458	0	70.5

where

$$\frac{\partial \gamma_i}{\partial \tilde{\xi}_j} = \left[\frac{\partial \gamma_i}{\partial \tilde{\xi}_j^1}, \cdots \frac{\partial \gamma_i}{\partial \tilde{\xi}_j^6}\right] \in I\!R^{1\times 6};$$

$$\frac{\partial \mathbf{h}_i}{\partial \tilde{\xi}_j} = \begin{pmatrix} \frac{\partial \alpha_i}{\partial \tilde{\xi}_j^1} & \cdots & \frac{\partial \alpha_i}{\partial \tilde{\xi}_j^6} \\ \vdots & \ddots & \vdots \\ \frac{\partial c_i}{\partial \tilde{\xi}_j^1} & \cdots & \frac{\partial c_i}{\partial \tilde{\xi}_j^6} \end{pmatrix} \in I\!R^{5\times 6}.$$

As shown above, $[\frac{\partial \mathbf{s}}{\partial \tilde{\xi}}]$ and $[\frac{\partial \mathbf{h}}{\partial \tilde{\xi}}]$ are sparse and band-limited. Once all nonzero entries are computed, H_{11}, H_{12}, and H_{22} can be obtained using Eq. (9). We recall that the matrix multiplications for sparse matrices can be done in $O(n)$ for matrices with $O(n)$ nonzero elements by extracting all zero entries. Finally, we can solve Eq. (7) in $O(n)$ by following the steps described in Section 2.2. Numerical results of M and M^{-1} of the PUMA 560 arm are provided in the Appendix for verification of the result.

4.3. A polypeptide chain

The D–H parameters for a polypeptide chain are shown in Table II. Frames are attached to each atom in the backbone structure. The main chain atoms are represented as rigid peptide units, linked through the C_α atoms. The parameters, such as bond-length, b_i's, and bond-angle values, γ_i's, for a polypeptide chain are adapted from Ref. [30]. The offset values, $c_i = 0$, for all i, and torsion angles along each bond link, denoted by β_i for all $i = 1, \ldots, n$, are viewed as only soft variables. In fact, the torsion angles along $C'-N$ bonds are fixed to about 180° in polypeptide chains. As shown in Fig. 2, each C_α atom is connected to four atoms, C', N, H, and R (= CH_3 for a polyalanine chain). We assume that this structure is a Tetrahedron with C_α at the center, and CH_3 is considered as a point mass at the location of C. The c.o.m of l point masses at the frame k can be computed as

$${}^c\mathbf{p}_k = \frac{1}{\tilde{M}} \sum_{i=1}^{l} m_i \mathbf{r}_i$$

where $\tilde{M}$ is the sum of all point masses and $\mathbf{r}_i$ is the position of the ith particle seen from the origin of the frame k. Q_k is given by pure translation, ${}^c\mathbf{p}_k$. When ${}^c\mathbf{p}_k$ for all $k = 1, \ldots, n$ are calculated, the moment of inertia at the c.o.m can be computed, respectively. The atomic masses[8] used for numerical examples are

[8] amu = Atomic mass unit, defined to be 1/12 of the mass of a C-12 atom.

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

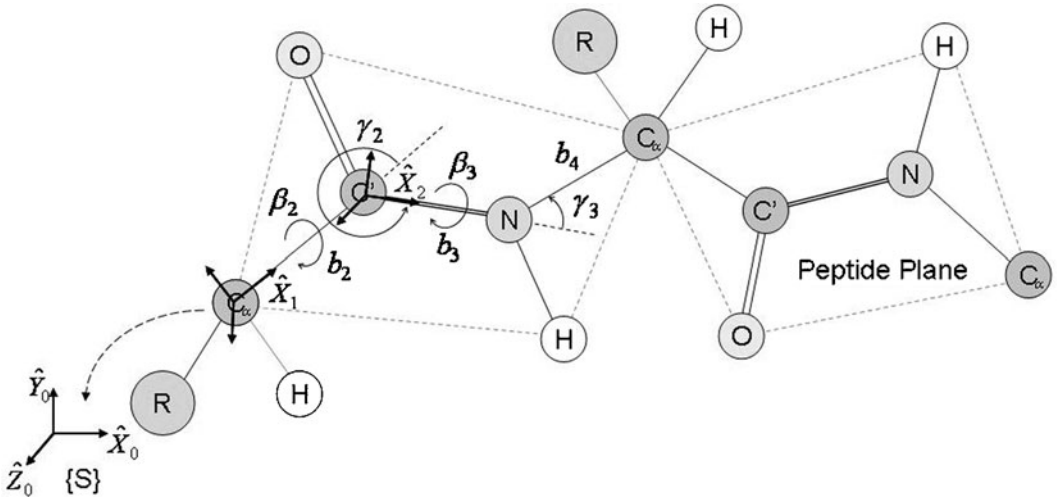

Fig. 2. A polypeptide chain with alanine side chains.

C = 12.0107 amu, N = 14.00674 amu, H = 1.00794 amu, and O = 15.9994 amu.

We first consider that all torsion angles, β_i for $i = 1, \ldots, n$, are soft variables. Then, the vector of generalized coordinates is defined to be $\mathbf{q} = [\mathbf{s}^T; \mathbf{h}^T]^T$ where $\mathbf{s}^T = [\beta_1, \ldots, \beta_n]^T$ and $\mathbf{h}^T = [\alpha_1, \gamma_1, a_1, b_1, c_1, \ldots, \alpha_n, \gamma_n, a_n, b_n, c_n]^T$. For $n = 7$ (containing two rigid peptide planes), we arbitrarily choose $\beta_1 = 0$ and $\beta_i = \pi/3$ for $i = 2, \ldots, 7$. Then, T_k^c's are given by

$$T_1^c = \begin{pmatrix} 0.334 & -0.943 & 0 & 0.285 \\ 0.943 & 0.334 & 0 & -0.403 \\ 0 & 0 & 1 & 0.634 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_2^c = \begin{pmatrix} 0.570 & 0.091 & 0.816 & 0.343 \\ 0.266 & 0.920 & -0.289 & 1.875 \\ -0.777 & 0.382 & 0.5 & 0.525 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_3^c = \begin{pmatrix} 0.940 & -0.090 & 0.329 & 1.241 \\ 0.318 & -0.117 & -0.941 & 1.788 \\ 0.123 & 0.989 & -0.081 & -1.095 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_4^c = \begin{pmatrix} -0.968 & -0.065 & 0.242 & 3.093 \\ -0.154 & 0.917 & -0.369 & 1.696 \\ -0.198 & -0.395 & -0.897 & -1.207 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_5^c = \begin{pmatrix} -0.268 & 0.947 & 0.177 & 1.821 \\ 0.056 & 0.199 & -0.978 & 2.116 \\ -0.962 & -0.253 & -0.107 & -0.933 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_6^c = \begin{pmatrix} -0.674 & 0.101 & -0.732 & 0.821 \\ 0.666 & -0.345 & -0.662 & 2.094 \\ -0.319 & -0.933 & 0.165 & -2.368 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$T_7^c = \begin{pmatrix} -0.782 & 0.428 & -0.453 & -0.415 \\ -0.465 & -0.885 & -0.032 & 3.535 \\ -0.415 & 0.186 & 0.891 & -2.292 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Given the homogeneous transformation matrices, we can compute $[\frac{\partial \mathbf{s}}{\partial \xi}]$ and $[\frac{\partial \mathbf{h}}{\partial \xi}]$ using the analytical expressions of generalized coordinates derived in Section 4.1. H_{11}, H_{12} and H_{22} can be obtained similarly as described for the PUMA 560 arm.

As mentioned earlier, the torsion angles along the C′−N bonds are fixed to be about 180°. Therefore, some of the β_i's should not be treated as soft variables. We now set $\beta_3 = \beta_6 = 180°$. The vectors of soft and hard variables can be redefined as $\mathbf{s}^T = [\beta_1, \beta_2, \beta_4, \beta_5, \beta_7]$ and $\mathbf{h}^T = [\alpha_1, \ldots, \alpha_3, \beta_3, \gamma_3, \ldots, \alpha_6, \beta_6, \gamma_6, \ldots, b_7, c_7]$. H_{11}, H_{12} and H_{22} can be computed accordingly for given $\mathbf{q} = [\mathbf{s}^T, \mathbf{h}^T]^T$. Numerical results are shown in Section 4.4 and the Appendix.

4.4. Computational time

The computational time is highly dependent on the computer in which the program runs, such as the memory size, the type of processor, the operating system, etc. Therefore, one should be careful when interpreting the result of computational times required to run the algorithm. The program to test the running times in different sizes of serial chains is written in Matlab version 6.5 and runs in a $2.8GHz$ Pentium4 computer with $1Gb$ RAM. The operating system is Window XP Home edition.

The polypeptide chain example in Section 4.3 is revisited, while all torsion angles are considered as soft variables. Figure 3 shows the computational time to invert the mass matrix for $n = 1, \ldots, 400$. The dashed line indicates the time to solve $\mathbf{x} = M^{-1}\mathbf{b}$ by inverting the $n \times n$ mass matrix, M, using the 'inv(M)' function in Matlab, and the solid line shows one to compute the equation, $\mathbf{x} = (H_{11} - H_{12}H_{22}^{-1}H_{12}^T)\mathbf{b}$. $H_{22}^{-1}\mathbf{c}$ (in the computational step V in Section 2.2) is calculated using the LU decomposition. The time is counted using the 'tic-toc' function in Matlab. Since

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

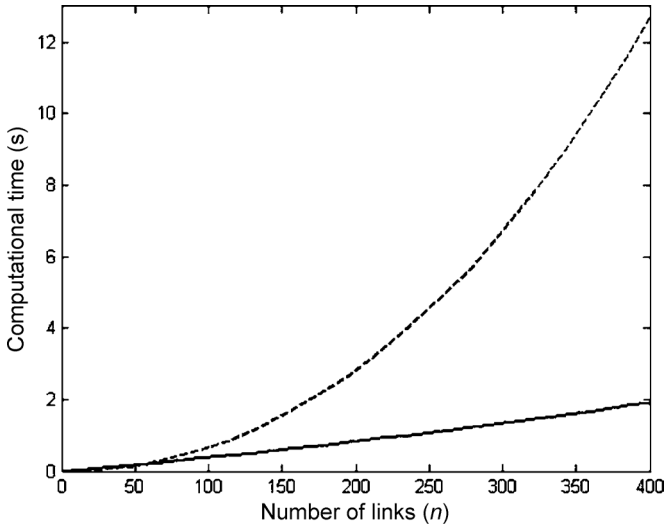

Fig. 3. Computational time (s) vs. the number of links (n): the time required for direct inversion of M (dashed line) and the time required when using the extended Fixman's method (solid line), for $n = 1, \ldots, 400$.

the program can be optimized in many different ways, the running time cannot be viewed as an absolute measure of the computational speed. However, it can be used as a measure of speed or efficiency of an algorithm with provided specifications of the computer and software in which the program runs.

The graph shows that the computational time linearly increases. Instead of counting real time, we can count the number of operations to verify the $O(n)$ computational complexity. We analytically proved that our algorithm requires $O(n)$ operations, but did not include an operation count in this paper. The number of mathematical operations will vary according to different systems and depends on how to optimize the program. In,[31] the actual time required to compute the forward dynamics for an open chain using the recursive Hamiltonian method is provided. For $n = 400$, the running time is about 4.7 s. Our algorithm to solve $\mathbf{x} = M^{-1}\mathbf{b}$ for $n = 400$ requires about 1.9 s as shown in the graph. We note that these numbers are not directly comparable because we do not compute the whole forward dynamic equations as done in.[31] Also, the specification of the computer and software (C^{++}) used to run the algorithm in Ref. [31] differ from ours.

5. Conclusions

More than 30 years ago, a method for $O(n)$ computation of the determinant of the mass matrix for a chain of point masses constrained with rigid bonds was developed by Professor Marshall Fixman. Whereas this theorem apparently has remained unknown to the multibody and robotics literature, we have applied it to develop $O(n)$ forward dynamics algorithms, especially to compute the inverse of the mass matrix, in a series of papers.[22–24] The specific contribution of this paper is the extension of Fixman's theorem to the case of serial chains of rigid bodies. We demonstrate it on two examples: the 6 DOF PUMA 560 manipulator and polypeptide chain on several lengths. The associated mathematics required for this extension have also been developed and presented.

Acknowledgments

This work was supported by National Institutes of Health Grant R01GM075310 and National Science Foundation Information Technology Research 0205466. This paper is dedicated to the memory of Eun-Jung Rhee, who as a promising doctoral candidate in theoretical particle physics at Johns Hopkins University would have come to use Lie-theoretic methods extensively.

References

1. Vereshchagin, A. F., "Gauss principle of least constraint for modeling the dynamics of automatic manipulators using a digital computer," Sov. Phys.-Doklady **20**(1), pp. 33 (1975).
2. J. Y. S. Luh, M. W. Walker and R. P. Paul, "On-line computational scheme for mechanical manipulators," Trans. ASME J. Dy. Sys. Meas. Control **102**: 69–76 (1980).
3. J. M. Hollerbach, "A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity," *In*: Tutorial on Robotics (C. S. G. Lee, R. C. Gonzalez and K. S. Fu, ed.) (IEEE Computer Society Press, Silver Spring, Maryland, 1983), pp. 111–117.
4. G. Rodriguez, "Kalman Filtering, smoothing, and recursive robot arm forward and inverse dynamics'," IEEE J. Robot. Automat. **RA-3** (6), 624–639 (1987).
5. G. Rodriguez and K. Keutz-Delgado, "Spatial operator factorization and inversion of the manipulator mass matrix," IEEE Trans. Robot. Automat. **8**(1), 65–76 (1992).
6. A. Jain and G. Rodriguez, "Recursive flexible multibody system dynamics using spatial operators," J. Guid. Control Dyn. **15**(6), 1453–1466 (1992).
7. S. K. Saha, "A decomposition of the manipulator inertia matrix," IEEE Trans. Robot. Automat **13**(2), 301–304 (1997).
8. S. K. Saha, "Analytical expression for the inverted inertia matrix of serial robots," Int. J. Robotics Research **18**(1), 116–124 (1999).
9. J. Angeles and O. Ma, "Dynamic simulation of n-axis serial robotic manipulators using a natural orthogonal complement," Int. J. Robot. Res. **7**(5), 32–47 (1998).
10. K. S. Anderson and S. Duan, "Highly Parallelizable Low Order Dynamics Algorithm for Complex Multi-Rigid-Body Systems," AIAA J. Guid. Control Dy. **23**(2), 355–364 (2000).
11. K. S. Anderson and S. Duan, "A Hybrid Parallelizable Low Order Algorithm for Dynamics of Multi-Rigid-Body Systems: Part I, Chain Systems," J. Math. Comput. Mod. **30**, 193–215 (1999).
12. K. S. Anderson and S. Duan, "Parallel Implementation of a Low Order Algorithm for Dynamics of Multibody Systems on a Distributed Memory Computing System," J. Eng. Comput. **16**(2), 96–108 (2000).
13. K. S. Anderson, "An Order-N Formulation for the Motion Simulation of General Constrained Multi-Rigid-Body Systems," J. Comput. Str. **43**(3), 565–579 (1992).
14. K. S. Anderson, "An Order-N Formulation for the Motion Simulation of General Multi-Rigid-Body Tree Systems," J. Comput. Str. **46**(3), 547–559 (1991).
15. K. S. Anderson and J. H. Critchley, "Improved 'Order-N' performance algorithm for the simulation of constrained multi-rigid-body dynamic systems," Multibody syst. dyn. **9**, 185–212, (2003).
16. R. Featherstone, "The calculation of robot dynamics using articulated-body inertia," Int. J. Robot. Res. **2**(1), 13–30 (1983).
17. R. Featherstone, "Efficient factorization of the joint space inertia matrix for branched kinematic trees," submitted to Intl. Journal of Robotics Research.

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press

18. S. R. Ploen and F. C. Park, "Coordinate-Invariant Algorithms for Robot Dynamics," IEEE Trans. Robot. Automat. **15**(6) 1130–1135 (1999).
19. J. Naudet and D. Lefeber, "General Formulation of an Efficient Recursive Algorithm based on Canonical Momenta for Forward Dynamics of Closed-loop Multibody Systems," Proceedings of IDETC/CIE, Long Beach, CA, USA (Sep. 2005) DETC 2005-84917.
20. J. Naudet, D. Lafeber and Z. Terze, "General Formulation of an Efficient Recursive Algorithm based on Canonical Momenta for Forward Dynamics of Open-loop Multibody Systems," Multibody Dynamics, 2003.
21. M. Fixman, "Classical statistical mechanics of constraints: A theorem and application to polmers," *Proc. Nat. Acad. Sci.* **71**(8), 3050–3053 (Aug. 1974).
22. K. Lee and G. S. Chirikjian, "A New Perspective on O(n) Mass-Matrix Inversion for Serial Revolute Manipulators," *IEEE International Conference on Robotics and Automation*, Barcelona, Spain (Apr. 2005).
23. K. Lee, G. S. Chirikjian, "A New Method for O(n) Inversion of the Mass Matrix," *International Symposium on Multibody Systems and Mechatronics*, Uberlandia, Brazil (Mar. 2005).
24. Y. Wang and G. S. Chirikjian, "A New O(n) Method for Inverting the Mass-Matrix for Serial Chains Composed of Rigid Bodies," *Proceedings of IDETC/CIE, Long Beach, CA, USA* (Sep. 2005) DETC 2005-84365.
25. A. Fijani and G. M. T. D'Eleuterio, "Parallel O(log N) Algorithms for Computation of Manipulator Forward Dynamics," *IEEE Trans. Robot. Automat.*, **11**(3), (1995) 389–400.
26. G. H. Golub and C. F. Vam Loan, *Matrix Computations* (Johns Hopkins University Press, Baltimore, MD, USA and London, UK, 1996).
27. G. S. Chirikjian and A. B. Kyatkin, *Engineering Applications of Noncommutative Harmonic Analysis* (CRC Press, Boca Raton, FL, 2001).
28. J. J. Craig, *Introduction to Robotics*, 2nd ed. (Addison-Wesley, Reading, MA, 1989).
29. R. P. Paul and H. Zhang, "Computationally efficient kinematics for manipulators with spherical wrists," *International Journal of Robotic Research*, **5**(2), (1986) 32–44.
30. R. V. Pappu, R. Srinivasan and G. D. Rose, "The Flory isolated-pair hypothesis is not valid for polypeptide chains: Implications for protein folding," *Biophysics* **97**(23), pp. 12565–12570, (2000).
31. J. Naudet, "Forward Dynamics of Multibody Systems: A Recursive Hamiltonian Approach *Ph.D. Thesis* (Brussels, Belgium Vrije Universiteit Brussel, 2005).

Appendix

The mass matrix of the PUMA 560 arm in Section 4.2 is given by

$$M_{puma} = \begin{pmatrix} 7.0719 & 3.6036 & 0.3138 & -0.0723 & -0.0004 & -0.0002 \\ 3.6036 & 2.8533 & 0.3379 & 0.0722 & -0.0004 & -0.0006 \\ 0.3138 & 0.3379 & 0.3259 & 0.1446 & 0.0002 & -0.0005 \\ -0.0723 & 0.0722 & 0.1446 & 0.2883 & 0.0003 & -0.0003 \\ -0.0004 & -0.0004 & 0.0002 & 0.0003 & 0.0007 & 0.0002 \\ -0.0002 & -0.0006 & -0.0005 & -0.0003 & 0.0002 & 0.0004 \end{pmatrix}.$$

The inverse of the constrained mass matrix using the extended Fixman's algorithm in Eq. (6) is computed as

$$M_{puma}^{-1} = \begin{pmatrix} 0.4130 & -0.5346 & 0.0655 & 0.2043 & -0.0775 & -0.3333 \\ -0.5346 & 1.0959 & -0.5659 & -0.1247 & 0.4489 & 0.3780 \\ 0.0655 & -0.5659 & 4.537 & -2.1106 & -2.288 & 4.0852 \\ 0.2043 & -0.1247 & -2.1106 & 4.6143 & -2.0481 & 1.6933 \\ -0.0775 & 0.4489 & -2.288 & -2.0481 & 1810.6 & -908.69 \\ -0.3333 & 0.3780 & 4.0852 & 1.6933 & -908.69 & 2682.8 \end{pmatrix}.$$

Matrix multiplication of the above two matrices yields that $M_{puma} \cdot M_{puma}^{-1} = \mathbb{I}_6$. We note that Eq. (7) can be computed in $O(n)$, but not the matrix M^{-1} (while each column of M^{-1} can be obtained in $O(n)$). We provide the numerical results of M^{-1} for a verification purpose.

The constrained mass matrix of 7-link polypeptide chain (when all β_i's are viewed as soft variables) in Section 4.2 is given by

$$M_{poly} = \begin{pmatrix} 870.7768 & 54.2187 & 161.5036 & 234.8087 & -176.7587 & 16.9382 & -1.8702 \\ 54.2187 & 434.7277 & 58.8923 & 157.5614 & -93.0816 & -41.1893 & -10.7686 \\ 161.5036 & 58.8923 & 189.6189 & 103.2236 & -79.4282 & 53.5038 & 20.2896 \\ 234.8087 & 157.5614 & 103.2236 & 163.9448 & -113.3383 & -13.8750 & -10.3724 \\ -176.7587 & -93.0816 & -79.4282 & -113.3383 & 95.9587 & 13.0774 & 11.6638 \\ 16.9382 & -41.1893 & 53.5038 & -13.8750 & 13.0774 & 63.0571 & 29.1266 \\ -1.8702 & -10.7686 & 20.2896 & -10.3724 & 11.6638 & 29.1266 & 23.1711 \end{pmatrix}.$$

The inverse of the constrained mass matrix for the 7-link polypeptide chain computed from $H_{11} - H_{12}H_{22}^{-2}H_{12}^T$ is given by

$$M_{poly}^{-1} = \begin{pmatrix} 0.0025 & 0.0014 & 0.0016 & -0.0047 & 0.0022 & -0.0021 & -0.0012 \\ 0.0014 & 0.0049 & 0.0010 & -0.0089 & -0.0021 & 0.0026 & -0.0047 \\ 0.0016 & 0.0010 & 0.0194 & -0.0078 & 0.0137 & -0.0202 & -0.0015 \\ -0.0047 & -0.0089 & -0.0078 & 0.0536 & 0.0389 & 0.0064 & -0.0013 \\ 0.0022 & -0.0021 & 0.0137 & 0.0389 & 0.0735 & -0.0127 & -0.0164 \\ -0.0021 & 0.0026 & -0.0202 & 0.0064 & -0.0127 & 0.0629 & -0.0510 \\ -0.0012 & -0.0047 & -0.0015 & -0.0013 & -0.0164 & -0.0510 & 0.1140 \end{pmatrix}.$$

Matrix multiplication of the above matrices yields $M_{poly} \cdot M_{poly}^{-1} = \mathbb{I}_7$.

https://doi.org/10.1017/S0263574707003852 Published online by Cambridge University Press