Advanced Robotics, Vol. 14, No. 8, pp. 703–715 (2001)

Full paper

Efficient recursive algorithm for the operational space inertia matrix of branching mechanisms

KYONG-SOK CHANG * and OUSSAMA KHATIB

Robotics Laboratory, Computer Science Department, Stanford University, Stanford, CA 94305, USA

Received 25 January 2000; accepted 7 June 2000

Abstract—This article describes an efficient recursive algorithm for the computation of the operational space inertia matrix of an n-link branching robotic mechanism with multiple (m) operational points. The proposed algorithm achieves the complexity of $O(nm + m^3)$. Since m can be considered as a small constant in practice, as the number of links increases, this algorithm performs significantly better than the existing $O(n^3 + m^3)$ symbolic method. The experimental results of this algorithm are presented using real-time dynamic simulation.

Keywords: Branching mechanisms; operational space formulation with multiple operational points; modified spatial notation; operational space inertia matrix; efficient recursive algorithm.

1. INTRODUCTION

The operational space formulation [1] is an approach for the dynamic modeling and control of a complex branching (tree-like) redundant mechanism (Fig. 1) at its task or end-effector level. This formulation is particularly useful for dealing with simultaneous tasks of multiple end-effectors since its basic structure provides dynamic decoupling among end-effectors' tasks and the complex internal dynamics in their associated null space.

In order for this formulation to be usable in real-time control of a complex n-link mechanism, however, the complexity $O(n^3 + m^3)$ of the existing symbolic method [1] is not acceptable when n is large. This $O(n^3)$ complexity comes from the explicit inversion operation of the joint space inertia matrix of size $O(n^2)$, required for the computation of the operational space inertia matrix.

In this article, we propose an efficient recursive algorithm for the computation of the operational space inertia matrix of an n-link branching redundant robotic mechanism with m operational points. The proposed algorithm achieves the

*Present address: 4777 Sawgrass Drive West, Ann Arbor, MI 48108-8612, USA.

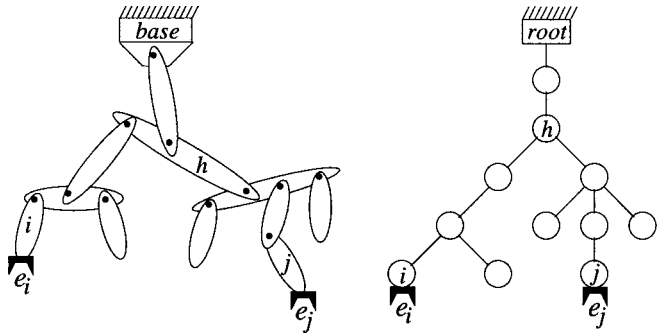

Figure 1. A branching robot with a tree-like topology. In its corresponding tree structure, each link becomes a node and each joint becomes an edge of the tree.

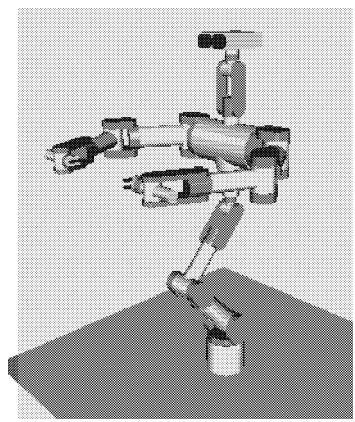

Figure 2. A basic humanoid robotic mechanism.

complexity of $O(nm + m^3)$. Since m can be considered as a small constant in practice, we obtain the linear running time of $O(n)$ for this algorithm. Therefore, the application of this algorithm results in a significant increase in the computational performance.

Section 2 provides background material describing a modified spatial notation and basic spatial kinematic and dynamic quantities using this notation. In Section 3, an efficient recursive algorithm is developed based on these spatial quantities and its $O(nm + m^3)$ complexity is proved. Finally, real-time simulation results with a basic humanoid redundant robotic mechanism with $n = 24$ and $m = 2$ (Fig. 2) are presented to illustrate the efficiency of the proposed algorithm.

2. SPATIAL NOTATION AND QUANTITIES

This section summarizes the *modified spatial notation* used throughout the article. In addition, some of the spatial quantities, which are essential for developing the

proposed algorithm in the next section, are presented using this modified spatial notation.

2.1. Spatial notation

Spatial notation has been widely used in the modeling of kinematics and dynamics of complex robotic mechanisms [2–7]. The modified spatial notation and quantities developed in this section combines various versions of existing spatial notations and conventional vector notations [8–10] in order to utilize the results from various researchers in a unified way. In this modified spatial notation, each quantity incorporates the appropriate linear (placed in upper or upper-left corners) and angular (placed in lower or lower-right corners) components, and results in a concise form (6×1 vector or 6×6 matrix).

For example, a spatial acceleration, $\mathbf{a}_i$, and a spatial force, $\mathbf{f}_i$, of link i are defined as:

$$\mathbf{a}_i = \begin{bmatrix} \dot{v}_i \\ \dot{\omega}_i \end{bmatrix} \quad \text{and} \quad \mathbf{f}_i = \begin{bmatrix} f_i \\ \mathcal{N}_i \end{bmatrix},$$

where v_i, ω_i, f_i and $\mathcal{N}_i$, are 3×1 linear velocity, angular velocity, force and moment vectors expressed in frame i, respectively. Also, the spatial inertia matrix of link i in frame c_i, $\mathbf{I}_{c_i}$, is a 6×6 symmetric positive definite matrix and defined as:

$$\mathbf{I}_{c_i} = \begin{bmatrix} M_i & \mathbf{0} \\ \mathbf{0} & I_{c_i} \end{bmatrix}, \quad M_i = m_i \, \mathbf{1}_3, \tag{1}$$

where $\mathbf{1}_3$ is a 3×3 identity matrix, m_i is the mass of link i and I_{c_i} is the 3×3 inertia tensor matrix of link i in frame c_i. The origin of frame c_i is located at the center of mass of link i shown in Fig. 3.

Note that in Fig. 3 the origin of each link frame is located at the joint and any variable without the reference frame number (front superscript) is expressed in its own frame. Also, if link i is a leaf (outermost) link, end-effector frame e_i is located at the tip (operational point) of link i (see Fig. 1).

The general joint model, $\mathbf{S}_i$, is a $6 \times n_i$ matrix with full column rank, n_i, when joint i has n_i d.o.f. ($n_i \leqslant 6$) [2, 5]. Its columns (unit vectors) make up a basis for the motion space of joint i. Notice that this matrix is constant since it is expressed in its own frame. For example, if joint i is a prismatic joint along y-axis and joint j is a spherical joint, their corresponding general joint models are:

$$\mathbf{S}_i = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T \quad \text{and} \quad \mathbf{S}_j = \begin{bmatrix} \mathbf{0} & \mathbf{1}_3 \end{bmatrix}^T.$$

The 6×6 spatial transformation matrix, ${}^h_i\mathbf{X}$, transforms a spatial quantity from frame i to frame h:

$${}^h_i\mathbf{X} = \begin{bmatrix} {}^h_i R & \mathbf{0} \\ \widehat{{}^h r_i} \, {}^h_i R & {}^h_i R \end{bmatrix}, \tag{2}$$

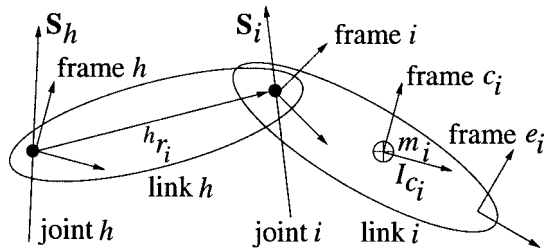

Figure 3. Basic notation.

where ${}^h_i R$ is the 3×3 rotation matrix and $\widehat{{}^h r_i}$ is the cross-product operator (3×3 skew-symmetric matrix) associated with ${}^h r_i$, the 3×1 position vector from the origin of frame h to the origin of frame i expressed in frame h shown in Fig. 3.

For example, the spatial transformations of accelerations and forces between frames i and c_i are:

$$\mathbf{a}_{c_i} = {}^i_{c_i}\mathbf{X}^T \mathbf{a}_i, \tag{3}$$

$$\mathbf{f}_i = {}^i_{c_i}\mathbf{X}\, \mathbf{f}_{c_i}. \tag{4}$$

The 6×6 spatial inertia matrix of link i in frame i can be computed, by spatially transforming $\mathbf{I}_{c_i}$ (1) from frame c_i to frame i, as:

$$\mathbf{I}_i = {}^i_{c_i}\mathbf{X}\, \mathbf{I}_{c_i}\, {}^i_{c_i}\mathbf{X}^T, \tag{5}$$

where $\mathbf{I}_i$ is a symmetric positive definite matrix since (5) encapsulates the spatial counterpart of the similarity transformation and the parallel axis theorem [8, 9] for $\mathbf{I}_{c_i}$.

2.2. Spatial quantities

In this subsection, we present some of the essential spatial quantities using the modified spatial notation to support the proposed algorithm developed in the next section. The force propagator, ${}^h_i\mathbf{L}$, propagates a spatial force from link i to its parent link h across the actuated joint i in a dynamically consistent manner [5]:

$$\mathbf{f}^*_h = {}^h_i\mathbf{L}\, \mathbf{f}^*_i, \tag{6}$$

where $\mathbf{f}^*_h$ is the resulting propagated spatial force of link h when the propagated spatial force of link i, $\mathbf{f}^*_i$, is propagated across joint i. Force propagation is physically valid only if the spatial force is propagated in inward (tip-to-base) direction. Note that $\mathbf{f}^*_{\text{leaf}} = \mathbf{f}_{\text{leaf}}$ at the beginning of the recursion.

Similarly, the acceleration propagator, shown to be equivalent to the transpose of the force propagator (${}^h_i\mathbf{L}^T$) [5], propagates a spatial acceleration of link h to its child link i across the actuated joint i in a dynamically consistent manner [2, 5]:

$$\mathbf{a}^*_i = {}^h_i\mathbf{L}^T\, \mathbf{a}^*_h, \tag{7}$$

where $\mathbf{a}_i^*$ is the resulting propagated spatial acceleration of link i when the propagated spatial acceleration of link h, $\mathbf{a}_h^*$, is propagated across joint i. Acceleration propagation is physically valid only if the spatial acceleration is propagated in outward (base-to-tip) direction. Note that $\mathbf{a}_{\text{leaf}}^* = \mathbf{a}_{\text{leaf}}$ at the end of the recursion.

The force propagator, ${}_i^h\mathbf{L}$, and the acceleration propagator, ${}_i^h\mathbf{L}^T$, are defined as [2, 5]:

$$ {}_i^h\mathbf{L} = {}_i^h\mathbf{X}\left[\mathbf{1}_6 - \mathbf{S}_i\bar{\mathbf{S}}_i\right]^T, \tag{8} $$

$$ {}_i^h\mathbf{L}^T = \left[\mathbf{1}_6 - \mathbf{S}_i\bar{\mathbf{S}}_i\right]{}_i^h\mathbf{X}^T, \tag{9} $$

where $\mathbf{1}_6$ is a 6×6 identity matrix. $\bar{\mathbf{S}}_i$ is the generalized inverse of $\mathbf{S}_i$ weighted by the corresponding inertia matrix and defined as:

$$ \bar{\mathbf{S}}_i = \mathbf{D}_i^{-1}\mathbf{S}_i^T\mathbf{I}_i^A. $$

The articulated-body inertia matrix of link i, $\mathbf{I}_i^A$, introduced by Featherstone [2], relates the spatial force and acceleration of a link, taking into account the dynamics of the rest of the articulated body [2, 3, 5]. Using the force propagator (8) and the acceleration propagator (9), the articulated-body inertia matrix of link h, $\mathbf{I}_h^A$, can be written as:

$$ \mathbf{I}_h^A = \mathbf{I}_h + \sum_i \left[{}_i^h\mathbf{L}\mathbf{I}_i^A{}_i^h\mathbf{L}^T\right], \tag{10} $$

where $\mathbf{I}_h$ is the spatial inertia matrix (5) of link h and i represents the index of each child link of link h. This recursive equation shows that the articulated-body inertia of a link is the sum of its own spatial inertia and the dynamically consistent projection of the articulated-body inertia of each child link. Note that $\mathbf{I}_h^A = \mathbf{I}_h$ if link h is a leaf link.

$\mathbf{D}_i$ is the $n_i \times n_i$ full rank matrix projecting $\mathbf{I}_i^A$ onto the motion space of joint i with n_i degrees of freedom:

$$ \mathbf{D}_i = \mathbf{S}_i^T\mathbf{I}_i^A\mathbf{S}_i. $$

Notice that the force propagator (8) has the same dynamic property as the dynamically consistent null space projection matrix for redundant robotic systems [11, 10]. Both quantities guarantee that the resulting (propagated or projected) quantity does not produce any coupling effect in their corresponding motion (operational) space.

A 6×6 inertia matrix, Ω_i relates the propagated spatial acceleration of link i and the propagated spatial force of the same link at the joint [3–5]:

$$ \mathbf{a}_i^* = \Omega_i\mathbf{f}_i^*. \tag{11} $$

Finally, the 6×6 operational space inertia matrix, Λ_{e_n}, of a single open-chain mechanism defined as [11]:

$$ \mathbf{a}_{e_n} = \Lambda_{e_n}^{-1}\mathbf{f}_{e_n}, \tag{12} $$

can be related to Ω_n using (2), (3), (4), (11), and (12):

$$\Lambda_{e_n}^{-1} = {}^{n}_{e_n}\mathbf{X}^T \Omega_n {}^{n}_{e_n}\mathbf{X}, \tag{13}$$

where end-effector frame e_n is at the tip of leaf link n.

3. EFFICIENT RECURSIVE ALGORITHM

This section describes an efficient recursive algorithm to compute the operational space inertia matrix Λ_e for branching robotic mechanisms without any explicit computation of the inverse of the joint space inertia matrix. We will develop this algorithm from the basic analysis of the physical properties of and the relationships among forces, accelerations, and inertia matrices. Note that since Λ_e is a function of configuration only (14), $\dot{\mathbf{q}} = \mathbf{0}$ can be assumed for the analysis of Λ_e without loss of generality. The proposed algorithm is shown to be of complexity $O(nm + m^3)$.

3.1. Analysis of operational space inertia matrix

The operational space inertia matrix, Λ_e, of an n-link N-d.o.f. branching redundant mechanism with m operational points is defined as [1]:

$$\Lambda_e^{-1} = \mathbf{J}_e \mathbf{A}^{-1} \mathbf{J}_e^T, \tag{14}$$

where Λ_e is an $6m \times 6m$ symmetric positive definite matrix, $\mathbf{J}_e$ is the $6m \times N$ Jacobian matrix and $\mathbf{A}$ is the $N \times N$ joint space inertia matrix. Note that m cannot be greater than n and since each joint can have only up to 6 d.o.f., $N = O(n)$ and the size of $\mathbf{A}$ is $O(n^2)$.

As in the case of a single operational point (12), Λ_e^{-1} relates the forces at the end-effectors to the accelerations at the end-effectors:

$$\mathbf{a}_e = \Lambda_e^{-1} \mathbf{f}_e, \tag{15}$$

where $\mathbf{a}_e$ and $\mathbf{f}_e$ are $6m \times 1$ vertically concatenated vectors of the accelerations and forces of each end-effector:

$$\mathbf{a}_e = \begin{bmatrix} \mathbf{a}_{e_1} \\ \vdots \\ \mathbf{a}_{e_m} \end{bmatrix} \quad \text{and} \quad \mathbf{f}_e = \begin{bmatrix} \mathbf{f}_{e_1} \\ \vdots \\ \mathbf{f}_{e_m} \end{bmatrix}. \tag{16}$$

Also, since Λ_e^{-1} is symmetric, it can be expressed in terms of its 6×6 block matrix components as:

$$\Lambda_e^{-1} = \begin{bmatrix} \Lambda_{e_1,e_1}^{-1} & \cdots & \Lambda_{e_1,e_m}^{-1} \\ \vdots & \ddots & \vdots \\ \Lambda_{e_1,e_m}^{-T} & \cdots & \Lambda_{e_m,e_m}^{-1} \end{bmatrix}. \tag{17}$$

From (15)–(17), the additive property of the coupling effect on the ith end-effector (of leaf link i) can be written as:

$$\mathbf{a}_{e_i} = \sum_{j=1,\ldots,m} \mathbf{a}_{e_i,e_j}, \tag{18}$$

$$\mathbf{a}_{e_i,e_j} = \Lambda^{-1}_{e_i,e_j} \mathbf{f}_{e_j}, \tag{19}$$

where $\mathbf{a}_{e_i,e_j}$ is the coupling acceleration on the ith end-effector by the force of the jth end-effector. This additive property of the coupling effect shows that the resulting acceleration of an end-effector is not only dependent on its own force, but also on the forces of all other end-effectors in the system.

Notice that when the jth end-effector produces the only non-zero force in the system, we can isolate the coupling effect on the ith end-effector by the force of the jth end-effector. This can be written, from (18) and (19), as:

$$\mathbf{a}_{e_i} = \mathbf{a}_{e_i,e_j} = \Lambda^{-1}_{e_i,e_j} \mathbf{f}_{e_j}, \tag{20}$$

when $\mathbf{f}_{e_j} \neq \mathbf{0}$ and $\mathbf{f}_{e_k} = \mathbf{0}$ for all $k \neq j$.

Then, similarly to (11), a 6×6 inertia matrix, $\Omega_{i,j}$ relates the propagated spatial acceleration of link i and the propagated spatial force of the link j at the corresponding joints. This relationship can be written as:

$$\mathbf{a}^*_i = \Omega_{i,j} \mathbf{f}^*_j. \tag{21}$$

Also, similarly to (13), the 6×6 block inverse operational space inertia matrix, $\Lambda^{-1}_{e_i,e_j}$, can be related to $\Omega_{i,j}$ using (2), (3), (4), (20) and (21):

$$\Lambda^{-1}_{e_i,e_j} = {}^{i}_{e_i}\mathbf{X}^T \Omega_{i,j} {}^{j}_{e_j}\mathbf{X}, \tag{22}$$

where end-effector frames e_i and e_j are at the tips of leaf links i and j. Note that this relationship is necessary since the inertial properties are desired at the tips instead of at the joints.

3.2. Derivation of the recursive algorithm

In this subsection, we will develop a recursive algorithm by separately analyzing the inertial effects of the block diagonal matrices, $\Lambda^{-1}_{e_i,e_i}$, and of the block off-diagonal matrices, $\Lambda^{-1}_{e_i,e_j}$ $(i \neq j)$, of Λ^{-1}_{e} in (17).

3.2.1. Block diagonal matrices. Each block diagonal matrix, $\Lambda^{-1}_{e_i,e_i}$, is the inertia matrix that would occur if link i is the only leaf link with an end-effector. With this physical insight, $\Lambda^{-1}_{e_i,e_i}$ can be computed using a trivial extension of the Force Propagation Method, an $O(n)$ recursive algorithm to compute the 6×6 inverse operational space inertia matrix of a single open-chain mechanism defined as [3–5]:

$$\Omega_i = \mathbf{S}_i \mathbf{D}_i^{-1} \mathbf{S}_i^T + {}_i^{h_i}\mathbf{L}^T \Omega_{h_i} {}_i^{h_i}\mathbf{L}, \tag{23}$$

where link i is the only child link of its parent link h_i and $\Omega_{\text{root}} = \mathbf{0}$.

Using the relationships from (11) and (21), the Force Propagation Method (23) can be extended immediately for a branching robot by replacing Ω_i with $\Omega_{i,i}$. This extension enables the outward recursion to pass through all children instead of a single child:

$$\Omega_{i,i} = \mathbf{S}_i \mathbf{D}_i^{-1} \mathbf{S}_i^T + {}_i^{h_i}\mathbf{L}^T \Omega_{h_i,h_i} {}_i^{h_i}\mathbf{L}, \tag{24}$$

where h_i is the parent link of link i. Note that this recursion starts from the root link with $\Omega_{\text{root,root}} = \mathbf{0}$ and ends at the leaf links with end-effectors. Then, the block diagonal matrices, Λ_{e_i,e_i}, can be computed by transforming $\Omega_{i,i}$ of leaf links i using (22).

3.2.2. Block off-diagonal matrices. The block off-diagonal matrices, $\Lambda^{-1}_{e_i,e_j}$ $(i \neq j)$, may be regarded as cross-coupling inertias that are a measure of the inertia coupling to the ith end-effector from the force of the jth end-effector via the *nearest common ancestor* of leaf links i and j. The *nearest common ancestor* of links i and j is the first common link in two paths: one from link i to the root link and the other from link j to the root link. For example, in Fig. 1, link h is the nearest common ancestor of leaf links i and j.

From this physical property of block off-diagonal matrices, we can conceptually view $\Lambda^{-1}_{e_i,e_j}$ as an inertial quantity which propagates the spatial force from the jth end-effector to link h (the nearest common ancestor of leaf links i and j) and then propagates the resulting spatial acceleration of link h to the ith end-effector.

With this conceptual view, we will develop a recursive algorithm to compute $\Lambda^{-1}_{e_i,e_j}$ by finding the propagation of the spatial force from the jth end-effector to link h and the propagation of the resulting spatial acceleration from link h to the ith end-effector. Then, $\Lambda^{-1}_{e_i,e_j}$ can be computed by relating the resulting spatial acceleration of link h to the propagated spatial force from the jth end-effector.

First, using (4), (6) and (8), we can propagate the spatial force $\mathbf{f}_{e_j}$ at the jth end-effector to any of its ancestor h:

$$\mathbf{f}_h^* = {}_j^h\mathbf{L}^* {}_{e_j}^j \mathbf{X} \mathbf{f}_{e_j}, \tag{25}$$

$${}_j^h\mathbf{L}^* = \prod_k {}_k^{h_k}\mathbf{L}, \tag{26}$$

where link k is the descendent links of link h in the path from link h to link j and link h_k is the parent link of link k. ${}_j^h\mathbf{L}^*$ results a compound propagation of the spatial force from link j to link h [5].

Similarly, using (3), (7), (9) and (26), the propagated spatial acceleration $\mathbf{a}_h^\star$ of link h can be propagated to the end-effector of any of its descendant leaf link i:

$$\mathbf{a}_{e_i} = {}_{e_i}^i \mathbf{X}^T {}_i^h \mathbf{L}^{*T} \mathbf{a}_h^*. \tag{27}$$

Table 1.
Recursive algorithm for Λ_e

1. **Outward Recursion:** Compute the spatial transformation matrices:
$$ {}^{h_i}_{i}\mathbf{X} = \begin{bmatrix} {}^{h_i}_{i}R & \mathbf{0} \\ \widehat{{}^{h_i}r_i}\,{}^{h_i}_{i}R & {}^{h_i}_{i}R \end{bmatrix}. \tag{2} $$
2. **Inward Recursion:** Compute the force propagators:
$$ {}^{h_i}_{i}\mathbf{L} = {}^{h_i}_{i}\mathbf{X}[\mathbf{1}_6 - \mathbf{S}_i\bar{\mathbf{S}}_i]^T. \tag{8} $$
3. **Outward Recursion:** Compute the block diagonal matrices starting with $\Omega_{\text{root,root}} = \mathbf{0}$:
$$ \Omega_{i,i} = \mathbf{S}_i\mathbf{D}_i^{-1}\mathbf{S}_i^T + {}^{h_i}_{i}\mathbf{L}^T\Omega_{h_i,h_i}\,{}^{h_i}_{i}\mathbf{L}. \tag{24} $$
4. **Outward Recursion:** Compute the block off-diagonal matrices with nearest common ancestor h of links i and j:
$$ \Omega_{i,j} = \begin{cases} \text{return}, & \text{if } i = j = h \\ {}^{h_i}_{i}\mathbf{L}^T\Omega^T_{j,h_i}, & \text{else if } j = h \\ \Omega_{i,h_j}\,{}^{h_j}_{j}\mathbf{L}, & \text{otherwise.} \end{cases} \tag{29} $$
5. **Spatial Transformation:** Compute $\Lambda^{-1}_{e_i,e_j}$ from $\Omega_{i,j}$ of leaf links with end-effectors:
$$ \Lambda^{-1}_{e_i,e_j} = {}^{i}_{e_i}\mathbf{X}^T\Omega_{i,j}\,{}^{j}_{e_j}\mathbf{X}. \tag{22} $$
6. **Matrix Inversion:** Compute the operational space inertia matrix, Λ_e, by inverting Λ_e^{-1} (17).

Now, combining (21), (25) and (27), we can relate $\mathbf{a}_{e_i}$ and $\mathbf{f}_{e_j}$ as:

$$ \mathbf{a}_{e_i} = {}^{i}_{e_i}\mathbf{X}^T\,{}^{h}_{i}\mathbf{L}^{*^T}\Omega_{h,h}\,{}^{h}_{j}\mathbf{L}^{*}\,{}^{j}_{e_j}\mathbf{X}\mathbf{f}_{e_j}. \tag{28} $$

Then, from (20)–(22) and (28), we can derive $\Omega_{i,j}$ for the block off-diagonal matrices:

$$ \Omega_{i,j} = {}^{h}_{i}\mathbf{L}^{*^T}\Omega_{h,h}\,{}^{h}_{j}\mathbf{L}^{*}, \tag{29} $$

where h is the nearest common ancestor of leaf links i and j. Note that the recursive version of (29) is presented in Table 1 (step 4). As for the block diagonal matrices, the block off-diagonal matrices, $\Lambda^{-1}_{e_i,e_j}$ $(i \neq j)$, can be computed by transforming $\Omega_{i,j}$ of leaf links i and j to the corresponding tips using (22).

Finally, we can compute the operational space inertia matrix, Λ_e, by inverting Λ_e^{-1} in (17). Table 1 summarizes the recursive algorithm developed in this section. Note that although most processing occurs along the path from the root link to the leaf links with end-effectors, the effects of the other links enter through the articulated-body inertias (10) of the links in the path. Figure 4 illustrates the recursion processes

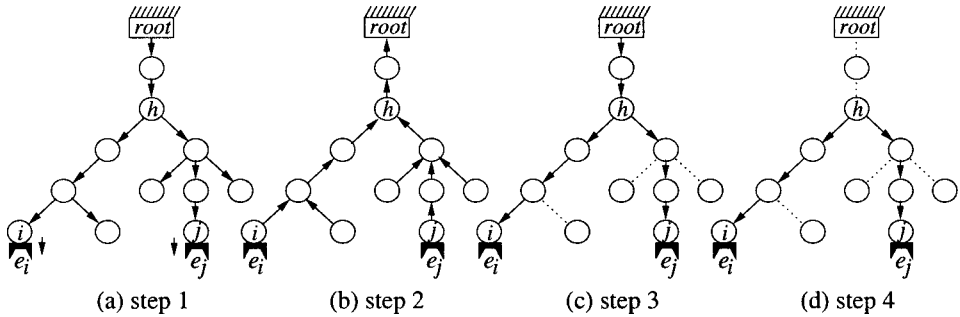

Figure 4. Recursion processes of steps in Table 1: (a) outward recursion for ${}^{h_i}_{i}\mathbf{X}$, (b) inward recursion for ${}^{h_i}_{i}\mathbf{L}$, (c) outward recursion for $\Omega_{i,i}$ and (d) outward recursion for $\Omega_{i,j}$ $(i \neq j)$.

of the proposed algorithm for the branching robot shown in Fig. 1. Arrows indicate the direction of the recursion. Note that there is no computation required among the nodes connected by dotted lines.

3.3. Computational complexity

This subsection presents the computational complexity of the proposed algorithm for an n-link branching mechanism with m operational points. From Table 1, steps 1 and 2 can be computed in $O(n)$ since there are n links in the system. Also, since step 3 requires one outward recursion involving at most n links, all $\Omega_{i,i}$ can be computed in $O(n)$. In step 4, since there are at most n links to propagate to compute all $m-1$ $\Omega_{k,j}$ $(k \neq j)$ from each of m end-effector k, all $\Omega_{i,j}$ can be computed in $O(mn)$. Spatial transformations of $m(m+1)/2$ block matrices cost $O(m^2)$ in step 5. In step 6, an inversion of Λ_e^{-1} of size $O(m^2)$ requires $O(m^3)$. Then, Λ_e can be computed in $O(nm + m^3)$. Note that since m can be considered as a small constant for any realistic robotic mechanism, $m = O(1)$, the overall running time of the proposed algorithm is $O(n)$ in practice. Thus, as the number of links in a mechanism increases, the proposed algorithm performs significantly better than the existing $O(n^3 + m^3)$ symbolic method [1] which still requires $O(n^3)$ inversion operations for $\mathbf{A}^{-1}$ (14).

4. EXPERIMENTAL RESULTS

Using the proposed algorithm, we were able to perform the computation of the operational space inertia matrix for a complex branching robotic mechanism (Figure 2) in less than 1 ms using a PC with a 266 MHz Pentium II running under the QNX real-time operating system. This branching robot has an operational point at each of its two end-effectors and 24 links connected by 1-d.o.f. joints. This result implies that the proposed algorithm enables highly redundant robotic mechanisms such as a humanoid robotic mechanism with multiple operational points to be efficiently controlled at a high servo rate in a low-cost hardware environment.

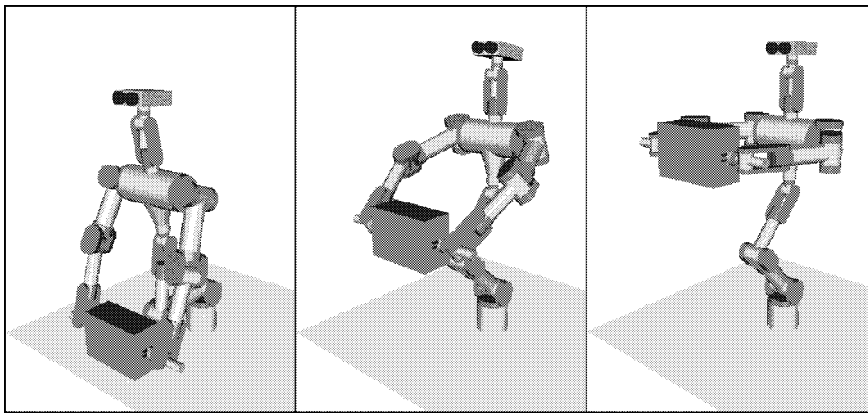

Figure 5. Motion sequence — putting a box on the floor.

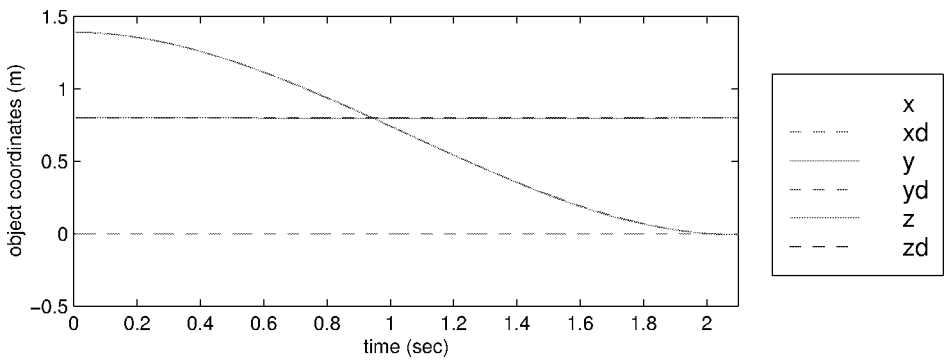

Figure 6. Plot — putting a box on the floor in Fig. 5.

In order to show the effectiveness of the proposed algorithm, we also have controlled this robot under the operational space formulation for branching mechanisms [1] using the proposed algorithm with Coriolis, centrifugal and gravitational forces [12]. In the real-time dynamic simulation environment developed in our laboratory, we have achieved a servo rate of 300 Hz with the set-up above.

Figure 5 shows the motion sequence that occurred when this robot was commanded to put the box on the floor while being advised to keep its posture the same as the initial configuration. The dynamics of the resulting closed-chain mechanism is computed using the *augmented object model* [13]. Notice that the robot had to adjust its advised posture behavior in the null space without producing any coupling acceleration at both end-effectors in order not to violate the primary task behavior in the operational space. This was done automatically without any additional commands. Figure 6 shows the cubic spline motion of the augmented object (box).

5. CONCLUSION

We have proposed an efficient $O(nm + m^3)$ recursive algorithm for the operational space inertia matrix of an n-link branching robotic mechanism with m operational points. Since m can be considered as a small constant in practice, we obtain the linear running time of $O(n)$ for this algorithm. Therefore, as the complexity of a robotic mechanism increases, the proposed algorithm performs significantly better than the traditional $O(n^3 + m^3)$ symbolic method. The real-time simulation results with a complex redundant robotic mechanism ($n = 24$, $m = 2$) illustrate the efficiency of this algorithm.

Acknowledgements

We thank Oliver Brock, Robert Holmberg, Oscar Madrigal, Diego Ruspini, Luis Sentis, H. F. Machiel Van der Loos and Kazuhito Yokoi for their comments and support during the preparation of this manuscript.

REFERENCES

1. J. Russakow, O. Khatib and S. M. Rock, Extended operational space formulation for serial-to-parallel chain (branching) manipulators, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Nagoya, Japan, pp. 1056–1061 (1995).
2. R. Featherstone, *Robot Dynamics Algorithms*. Kluwer, Boston, MA (1987).
3. G. Rodriguez, K. Kreutz and A. Jain, A spatial operator algebra for manipulator modeling and control, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 1374–1379 (1989).
4. K. Kreutz-Delgado, A. Jain and G. Rodriguez, Recursive formulation of operational space control, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 1750–1753 (1991).
5. K. W. Lilly, *Efficient Dynamic Simulation of Robotic Mechanisms*. Kluwer, Boston, MA (1992).
6. K. W. Lilly and D. E. Orin, Efficient O(n) recursive computation of the operational space inertia matrix, *IEEE Trans. Syst. Man Cybernet.* **23**, 1384–1391 (1993).
7. B. V. Mirtich, Impulse-based dynamic simulation of rigid body systems, PhD thesis, University of California at Berkeley (1996).
8. D. T. Greenwood, *Principles of Dynamics*, 2nd edn. Prentice-Hall, Reading, MA (1988).
9. J. J. Craig, *Introduction to Robotics*, 2nd edn. Addison-Wesley, Reading, MA (1989).
10. O. Khatib, The impact of redundancy on the dynamic performance of robots, *Laboratory Robotics Automat.* **8**, 37–48 (1996).
11. O. Khatib, A unified approach to motion and force control of robot manipulators: the operational space formulation, *IEEE J. Robotics Automat.* **3** (1), 43–53 (1987).
12. K.-S. Chang and O. Khatib, Operational space dynamics: Efficient algorithms for modeling and control of branching mechanisms, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, USA, Vol. 1, pp. 850–856 (2000).
13. K.-S. Chang, R. Holmberg and O. Khatib, The augmented object model: Cooperative manipulation and parallel mechanism dynamics, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, USA, Vol. 1, pp. 470–475 (2000).

ABOUT THE AUTHORS

Kyong-Sok Chang received his BSE and MSE in Aerospace Engineering from the University of Michigan, Ann Arbor in 1991 and 1992, respectively. He then went on to receive his MS in 1994 and his PhD in 2000 in Computer Science from Stanford University, with a specialization in robotics. His research focuses on efficient algorithms involving intuitive, interactive and fast dynamic modeling, simulation and control of highly redundant articulated branching mechanisms. He is currently the President and co-founder of Arachi, Inc., where he is developing a real-time dynamic simulation and control commercial software package.

Oussama Khatib received his PhD in 1980 from Sup'Aero, in Toulouse, France. He is presently a Professor of Computer Science at Stanford University leading a research group in Robotics and Haptics. His research focuses on autonomous robots, human-centered robotics, robot design, virtual dynamic environments and haptic interactions, emphasizing methodologies and technologies that address the intricate dynamic nature of these systems, provide the capabilities needed for their action and interaction with the environment, and cope with their real-time requirement. The practical implications of such research span a variety of topics ranging from the autonomous ability of a robot to cooperate with a human to the haptic interaction of a user with a virtual prototype, an animated character or a surgical instrument.