A New Composite Body Method for Manipulator Dynamics

Shir-Kuan Lin
Institute of Control Engineering
National Chiao Tung University
Hsinchu 30039, Taiwan, Republic of China
Received October 12, 1989; accepted November 16, 1990

This article presents a new composite body method for numerically forming the inertia matrix and the bias vector of manipulators, which is more efficient than the other two existing types of composite body methods. The main discrepancy of this one from the existing ones is that all points in a manipulator are observed from the origin of the base frame and the distances are all measured from this origin. The required computations of the present method for the inertia matrix and the bias vector of a manipulator with \( n \) rotational joints are

\[
(10.5n^2 + 38.5n - 85)M + (6n^2 + 39n - 70)A
\]

and

\[
(12.5n^2 + 5.5n + 3)M + (9n^2 + n)A,
\]

respectively, where "M" denotes multiplications, "A" does additions. In numerically forming the inertia matrix, the present method is more efficient than other methods in the literature for a manipulator with five or more joints; whereas this method is also superior to the recursive Newton–Euler formulation in computing the bias vector for a manipulator with six or less joints.

INTRODUCTION

It is well known that the dynamic model of a manipulator can be described with

\[
H(q)\ddot{q} + b(q,\dot{q}) = \tau
\]

where

\[
H(q) = \frac{\partial \mathbf{M}}{\partial q},
\]

\[
b(q,\dot{q}) = \frac{\partial \mathbf{b}}{\partial q},
\]

\[
\tau = \frac{\partial \mathbf{T}}{\partial q},
\]

\[
\mathbf{M} = \mathbf{M}^T
\]

and

\[
\mathbf{M} = \mathbf{M}^T = \mathbf{D} + \mathbf{H} + \mathbf{C}
\]

where

\[
\mathbf{D}
\]

is the diagonal inertia matrix, \( \mathbf{H} \) is the Coriolis and centrifugal force matrix, \( \mathbf{C} \) is the centripetal force matrix, and \( \mathbf{T} \) is the torque vector.
where \( H \) is the positive definite symmetric inertia matrix, \( b \) is often called the bias vector, \( \tau \) is the column of actuator forces, and \( q \) is the column of generalized coordinates, i.e., the displacement of joints.

The dynamics of manipulators can be categorized in two parts: the inverse dynamics and the forward dynamics. Many modern control schemes for manipulators require the inverse dynamics that determine the actuator forces for the prescribed joint displacements, velocities and accelerations. Due to the demand of real time control, several efficient algorithms for the inverse dynamics are developed. The recursive Newton–Euler formulation\(^1\) is found the most efficient in the literature.\(^2\) Recently, researchers have been struggling to improve the efficiency of this formulation. Two of the most efficient algorithms based on the recursive Newton–Euler formulation were proposed by Khalil and Kleinfinger\(^3\) and Balafoutis et al.\(^4\)

The forward dynamics are required while the motion simulation of manipulators is performed. The object is to solve the joint accelerations from (1) when the actuator forces are given as input values. The joint velocities and displacements can be obtained by integrating the joint accelerations, and are then used to calculate the bias vector. The research of the forward dynamics focuses on the formulation of the inertia matrix and the bias vector, which is the central topic of this article.

An industrial manipulator is different from a general multibody system in that it mostly has only six links or less. This fact should be taken into account while we investigate the formulation of manipulator dynamics. Our goal is to look for an efficient formulation for a manipulator with six links or less, although this formulation may be less efficient for a manipulator having more than six links.

In the literature, there have been two types of composite body (or generalized body) methods for forming the dynamic model of a manipulator, which are more efficient than the other methods. Walker and Orin\(^5\) developed a composite body method only for numerically forming the inertia matrix \( H \). The bias vector \( b \) was computed using the recursive Newton–Euler formulation\(^1\) by setting \( \dot{q} = 0 \). Another composite body method for forming the inertia matrix was derived from Lagrange’s equations by Renaud.\(^6,7\) He also combined the resulting formulation with the reduced Christoffel symbols symbolically to form the bias vector. Based on this method, Burdick\(^9\) established a LISP-based program, EMDEG, to automatically generate the symbolic dynamic model of manipulators. Vukobratovic et al.\(^10\) extended the Renaud’s formulation to the bias vector, and established an algorithm for numerically forming the inertia matrix and the bias vector of a manipulator with rotational joints. An alternative algorithm of the Renaud’s formulation is recently proposed by Fijany and Bejczy.\(^11,12\) They derived a recursive formulation for the inertia matrix which is similar to that of Walker and Orin\(^5\) in structure, but is equivalent to Renaud’s formulation in substance. Nevertheless, it is not more efficient than the original Renaud’s formulation (see Appendix).

An important discovery by Renaud is that some combinations of the inertia parameters of the links in a composite body are invariant to the manipulator motion (cf. (A5) and (A6) in Appendix). These invariant terms can be inter-
interpreted with the concept of the augmented body\textsuperscript{14} as the first moment and the inertia tensor of an augmented body.\textsuperscript{4,8} Renaud\textsuperscript{8} and Balafoutis et al.\textsuperscript{4} adopted this recognition to reformulate the recursive Newton-Euler formulation in terms of these invariant terms, independently, which is found computationally superior to the original one.\textsuperscript{1}

For numerically forming the inertia matrix, there is another method, which was developed by the author of this article in his earlier work.\textsuperscript{13} The basis of this method is an explicit formulation which relates the entries of the inertia matrix to the partial derivatives of the velocities and angular velocities of links with respect to the joint velocities. The algorithm based on this formulation is only adequate to manipulators with few joints since its computational complexity is of order $n^3$.

This article presents a new type of composite body method for numerically forming the inertia matrix and the bias vector of manipulators. We try to derive a closed-form formulation from the composite body theory of Walker and Orin\textsuperscript{5} since we believe that a closed-form formulation can improve the efficiency. This is achieved by observing all points in a manipulator from the origin of the base frame. The formulations are derived in the second and third sections. An algorithm and a comparison of efficiency with other methods are presented in the fourth section. It is shown that the present method is more adequate to industrial manipulators than other methods.

**FORMULATION OF THE INERTIA MATRIX**

The basic theory of the new composite body method is similar to that of Walker and Orin's. However, the forces and torques and the center of mass of a composite body are all observed in an inertia frame, namely the base frame $E_0$. And we want to derive a closed-form formulation for the inertia matrix, although some terms in the formulation are still computed in a recursive form. Suppose that a manipulator has $n$ low-pair joints, which are labeled as joint 1 to $n$ outward from the base. Assign a body-fixed frame on each joint, i.e., frame $E_i$ is fixed on joint $i$. The distance from the origin of $E_i$ to that of $E_j$ is designated as $\bar{r}_{ij}$, and the distance from the origin of $E_i$ to the center of mass of link $j$ as $\bar{p}_j$ (Fig. 1). Define a composite body $j$ as the union of link $j$ to link $n$. The mass of the composite body $j$ is denoted as $m_j$, and the distance from the origin of the base frame to the center of mass of the composite body as $r_j$. Hence

$$m_j = \sum_{i=j}^{n} m_i$$

$$r_j = \frac{\sum_{i=j}^{n} m_i \bar{p}_i}{m_j}$$

where $m_i$ is the mass link $i$. The inertia tensor of the composite body, $J_j$, results by using Huygeno–Steiner formula\textsuperscript{14} to obtain
Figure 1. A composite body.

\[ \mathbf{J}_j = \sum_{i=1}^{n} \mathbf{I}_i - m_i[(\dot{\mathbf{p}} \times s_j) \times ([\dot{\mathbf{p}} \times s_j] \times)] \]

\[ = \left( \sum_{i=1}^{n} \mathbf{I}_i - m_i[\dot{\mathbf{p}} \times] \right) + m_j[r_j \times][r_j \times] \]

where \( \mathbf{I}_i \) is the inertia tensor of link \( i \), and \( [\mathbf{a} \times] \) denotes a skew-symmetric matrix representing the vector multiplication \( (\mathbf{a} \times) \), i.e., \( [\mathbf{a} \times] \mathbf{b} = \mathbf{a} \times \mathbf{b} \).

The acceleration \( (\ddot{s}_j) \) of the center of mass and the angular acceleration \( (\dot{\alpha}_j) \) of the composite body \( j \) due to the motion of joint \( j \) only (i.e., the other joints are assumed stationary) are

\[ \ddot{s}_j = (K_j^* \mathbf{u}_j \times (s_j - j s) + K_j \mathbf{u}_j) \ddot{q}_j \]

\[ \dot{\alpha}_j = K_j^* \mathbf{u}_j \ddot{q}_j \]

where \( \mathbf{u}_j \) is the unit vector along joint \( j \), \( q_j \) is the displacement of joint \( j \), and

\[ K_j^* = (1 - K_j) = \begin{cases} 1, & \text{for rotational joint } j, \\ 0, & \text{for translational joint } j. \end{cases} \]

The inertia force \( (\mathbf{f}_{Tj}) \) and torque \( (\mathbf{t}_{Tj}) \) of the composite body \( j \) can be obtained using Newton–Euler equations. According to vectorial mechanics, the inertia force and torque can be represented by an equivalent force with an equivalent torque acting at the origin of the base frame, i.e.,

\[ \mathbf{f}_{Ej} = \mathbf{f}_{Tj} = -m_j \ddot{s}_j \]

\[ \mathbf{t}_{Ej} = \mathbf{t}_{Tj} + r_j \times \mathbf{f}_{Ej} = -J_j \dot{\alpha}_j - m_j r_j \times \ddot{s}_j \]
Under the situation that only joint \( j \) moves and the gravity is neglected, the force and torque exerted on joint \( m \) by link \( m, m \leq j \), are also an equivalent force and an equivalent torque, but acting on joint \( m \), of the inertia force and torque of the composite body \( j \). The actuator force applied on joint \( m \) is to resist the component of the force or torque exerted on joint \( m \) by link \( m \) along the direction of joint \( m \). Therefore, we get the actuator force of joint \( m \) due to \( \dot{q}_j \),

\[
\tau_{m,j} = -u_m \cdot (K_m^+(t_{E_j} - \eta_k s \times t_{E_j}) + K_m t_{E_j}), \quad m \leq j. \tag{10}
\]

Recalling (1) while setting \( b = 0 \) and all \( \dot{q}_i = 0 \), \( i \neq j \), we have

\[
\tau_{m,j} = H_{mj} \dot{q}_j, \quad m \leq j \tag{11}
\]

too, where \( H_{mj} \) is the \((m,j)\)th-entry of \( H \). Since \( H \) is symmetric, we just need to consider the upper triangular matrix, i.e., \( m \leq j \). Substituting (4)–(9) into (10) yields

\[
H_{mj} = K_m^* K_j^* \left\{ u_m \cdot \left( \sum_{i=j}^{n} \mathbf{I}_i - m_i [\mathbf{I}_0 \times [\mathbf{I}_p \times 1]] \mathbf{u}_j + (\mathbf{u}_j \times \hat{\eta}_j) \times (\hat{\mathbf{m}}_j r_j) \right) \\
- (u_m \times \hat{\eta}_j s) \cdot (u_j \times ((\hat{\mathbf{m}}_j r_j) - \hat{\mathbf{m}}_j \hat{\eta}_j s)) \right\} \\
+ K_m K_j^* u_m \cdot (\mathbf{u}_j \times ((\hat{\mathbf{m}}_j r_j) - \hat{\mathbf{m}}_j \hat{\eta}_j s)) \\
+ K_m^* K_j \mathbf{u}_j \cdot (u_m \times (\hat{\mathbf{m}}_j r_j) - \hat{\mathbf{m}}_j u_m \times \hat{\eta}_j s) \\
+ K_m K_j \hat{\mathbf{m}}_j u_m \cdot u_j, \quad m \leq j. \tag{12}
\]

This is the formulation of the inertia matrix based on the new composite body method. All vectors and the inertia tensors in this formulation are represented with respect to the base frame. Since the result of the formulation is a scalar, it is independent of the choice of the coordinate frame for the representation. Therefore, the transformation from the base frame to another frame for the representations of vectors and inertia tensors in the formulation does not change the result. It can be shown that the representations of vectors and inertia tensors with respect to the local frame \( E_j \) can save quite many computations.

For brevity, we denote

\[
\mathbf{j}_j^{(p)} = \sum_{i=j}^{n} \mathbf{j}_i^{(p)} - m_i [\mathbf{I}_0 \times [\mathbf{I}_p \times 1]] \\
\hat{\mathbf{p}}_j^{(p)} = \hat{\mathbf{m}}_j \mathbf{r}_j^{(p)} = \sum_{i=j}^{n} m_i \mathbf{p}_i^{(p)} \\
\hat{\mathbf{s}}_m^{(p)} = \mathbf{u}_m^{(p)} \times \eta_k \mathbf{s}_m^{(p)}, \quad m \leq j. \tag{13}
\]
where superscript "(j)" denotes the representation of a vector or an inertia tensor with respect to frame $E_j$. $\mathbf{J}_j$ is the inertia tensor of the composite body $j$ about the origin of the base frame in contrast to that about the center of mass of the composite body $\mathbf{J}_j$. $\mathbf{p}_j$ is the first moment of the composite body $j$ about the origin of the base frame. Note that

$$\mathbf{a}^{(j)} = j^{+1}_j R \mathbf{a}^{(j+1)}$$

(16)

$$\mathbf{J}_j^{(j)} = j^{+1}_j R \mathbf{J}_j^{(j+1)} j^{+1}_j R^T$$

(17)

where $j^{+1}_j R$ is a $3 \times 3$ matrix representing the coordinate transformation from frame $E_j$ to frame $E_{j+1}$. The proof of (17) is as follows. Suppose $\mathbf{a}^{(j+1)} = A^{(j+1)} \mathbf{b}^{(j+1)}$, where $\mathbf{a}$, $\mathbf{b}$ are vectors, $A$ is a matrix mapping $\mathbf{b}$ to $\mathbf{a}$. Thus,

$$\mathbf{a}^{(j)} = j^{+1}_j R \mathbf{a}^{(j+1)} \mathbf{b}^{(j+1)} = j^{+1}_j R \mathbf{A}^{(j+1)} j^{+1}_j R^T \mathbf{b}^{(j+1)}$$

(18)

since the coordinate transformation matrix is an orthogonal matrix. This is the so-called similarity transformation.

Let

$$\mathbf{I}_i = I_i - \frac{1}{2} \text{tr}[I_i] E$$

(19)

where $E$ is an identity matrix, then

$$\mathbf{I}_i = \tilde{I}_i - \text{tr}[\tilde{I}_i] E$$

(20)

It is easy to show that

$$[\mathbf{a} \times] [\mathbf{b} \times] = \mathbf{b} \mathbf{a}^T - \text{tr}[\mathbf{b} \mathbf{a}^T] E$$

(21)

According to (20) and (21), we can compute

$$\mathbf{J}_j^{(j)} = \sum_{i=1}^{n} \tilde{I}_i^{(j)} - m_i \delta \mathbf{p}(j) \delta \mathbf{p}(j)^T$$

(22)

and then get $\mathbf{J}_j^{(j)}$ by the relation of

$$\mathbf{J}_j^{(j)} = \tilde{\mathbf{J}}_j^{(j)} - \text{tr}[\tilde{\mathbf{J}}_j^{(j)}] E$$

(23)

Such an arrangement can save a few computations (see the fourth section).

Finally, we establish the following recursive algorithm for the variables in (12)

$$\mathbf{u}_m^{(j)} = j^{-1}_j R \mathbf{u}_m^{(j-1)}, \quad m \leq j$$

(24)
which are in a forward recursive form, and

\[ \dot{\mathbf{m}}_j = m_j + \dot{\mathbf{m}}_{j+1} \]  

(28)

\[ \dot{\mathbf{p}}_j^{(j)} = m_j \dot{\mathbf{p}}_j^{(j)} + j^{(j)} \mathbf{R}\dot{p}_j^{(j+1)} \]  

(29)

\[ \ddot{\mathbf{j}}_j^{(j)} = \mathbf{I}_j^{(j)} - m_j \mathbf{p}_j^{(j)} (\mathbf{p}_j^{(j)})^T + j^{(j)} \mathbf{R}\ddot{\mathbf{j}}_j^{(j+1)} j^{(j)} \mathbf{R}^T \]  

(30)

which are in a backward recursive form.

**FORMULATION OF THE BIAS VECTOR**

A formulation for the bias vector based on the same composite body concept is derived in this section. First, we should know the properties of the bias vector. The \( m \)th component of the bias vector is in the form of

\[ (\mathbf{b})_m = \sum_{j=1}^{n} \sum_{k=1}^{n} D_{mkj} \dot{q}_k \dot{q}_j + \eta_m \]  

(31)

where \( \eta_m \) contains the gravity, and the first term on the right-hand side is composed of Coriolis and centripetal forces and has the following properties:

\[ D_{mkj} = D_{mkj}, \]  

(32)

\[ D_{jkl} = D_{jlk}, \quad j \geq k; \]  

(33)

\[ D_{jkm} = -D_{mkj}, \quad m,j \geq k. \]  

(34)

which can be derived from Christoffel symbol. Thus, we are only concerned with \( D_{mkj}, \ \kappa \leq j \) and \( m < j \); the others can be directly obtained using the above relations (32)–(34). We rewrite (31) as follows,

\[ (\mathbf{b})_m = \sum_{j=1}^{n} \sum_{k=1}^{j-1} 2D_{mkj} \dot{q}_k \dot{q}_j + \sum_{j=1}^{n} D_{mkj} \dot{q}_j \dot{q}_j + \eta_m \]  

(35)

This equation indicates that the bias vector is influenced by joint velocities in pairs. To derive \( D_{mkj} \), we can just assume that joints \( k \) and \( j, k < j \), move at constant speed (i.e., \( \dot{q}_k = \ddot{q}_k = 0 \)) and other joints are kept stationary. Under this situation, the angular velocity and acceleration of the composite body \( j \) are, respectively,
\[ \omega_j = K_k^* u_k \dot{q}_k + K_j^* u_j \dot{q}_j \quad (36) \]

\[ \alpha_j = \frac{d\omega_j}{dt} = K_k^* K_j^* (u_k \dot{q}_k) \times (u_j \dot{q}_j) \quad (37) \]

In (37), the differential rule for a vector in a rotating frame is applied. The equation for the acceleration of point \( p \) fixed in a moving frame is

\[ a_p = a_0 + \omega \times (\omega \times r) + \alpha \times r + 2\omega \times v_0 \quad (38) \]

where \( a_0, v_0 \) are the acceleration and velocity of the origin of the moving frame; \( \omega, \alpha \) are the angular velocity and acceleration of the frame, \( r \) is the distance from the origin of the frame to point \( p \). On the right-hand side of (38), the second term is the centripetal acceleration, the third term is the tangential acceleration and the last term is Coriolis acceleration. Applying this equation, we obtain the acceleration of the origin of joint \( j \),

\[ \frac{d^2 s}{dt^2} = \omega_k \times (\omega_k \times (s_k - s) + 2\omega_k \times (K_j u_k \dot{q}_k)) = K_k^* u_k \times (u_k \times (s_k - s)) \dot{q}_k \dot{q}_k + 2K_k^* K_j u_k \times u_j \dot{q}_j \dot{q}_j \quad (39) \]

and then the acceleration of the center of mass of the composite body \( j \),

\[ \ddot{r}_j = \frac{d^2 s}{dt^2} + \omega_j \times (\omega_j \times (r_j - s) + \alpha_j \times (r_j - s)) = \frac{d^2 s}{dt^2} + \{K_k^* u_k \times (u_j \times (r_j - s)) \dot{q}_k \dot{q}_k + K_j^* u_j \times (u_j \times (r_j - s)) \dot{q}_j \dot{q}_j + K_k^* K_j^* (u_k \times u_j) \times (r_j - s) \dot{q}_k \dot{q}_j + K_k^* K_j^* (u_k \times u_j) \times (r_j - s) \dot{q}_k \dot{q}_j \} \quad (40) \]

It should be remarked that

\[ \alpha_i = 0, \quad i < j \quad (41) \]

\[ \frac{d^2 p}{dt^2} = \begin{cases} 0, & i < k, \\ K_k^* u_k \times (u_k \times (p - s)) \dot{q}_k \dot{q}_k & k \leq i < j. \end{cases} \quad (42) \]

For the present case, the bias vector without the gravity term is

\[ (b)_{mk(j)} = 2D_{mkj} \dot{q}_k \dot{q}_j + D_{kkj} \dot{q}_k \dot{q}_j + D_{mj} \dot{q}_j \dot{q}_j \quad (43) \]

As was mentioned above, we are only concerned with \( D_{mkj} \) and \( D_{mj} \). Therefore, only the terms with \( \dot{q}_k \dot{q}_j \) and \( q_j^2 \) in (37) and (40) are necessary to be taken in
account for the derivation of the bias coefficients. One of Newton–Euler’s equations for composite body \( j \) in the present case is the same as (8), the other is

\[
\mathbf{t}_{ij} = -J_j \mathbf{a}_j - \omega_j (J_j \omega_j)
\]

\[
= K^*_k K^*_j \left(-J_j (\mathbf{u}_k \times \mathbf{u}_j) - \mathbf{u}_k \times (J_j \mathbf{u}_j) - \mathbf{u}_j \times (J_j \mathbf{u}_k)\right) \dot{q}_k \dot{q}_j
\]

\[
- K^*_k \mathbf{u}_k \times (J_j \mathbf{u}_k) \dot{q}_k \dot{q}_j - K^*_j \mathbf{u}_j \times (J_j \mathbf{u}_j) \dot{q}_j \dot{q}_j
\]  

(44)

Although the inertia forces of links \( k \) to \( j - 1 \) are not zero in the present case, they can be neglected since they have only the terms with \( \dot{q}_k \). We use (37), (40), and (44) instead of (5), (6), and (9) to repeat the procedure in the previous section, and finally get

\[
D_{mki} = K^*_k \mathbf{u}_k^{(j)} \cdot \mathbf{d}_{mj}, \quad m < j, k \leq j;
\]  

(45)

\[
\mathbf{d}_{mj} = K^*_m K^*_j \left\{ \mathbf{j}_j^{(j)} (\mathbf{u}_j^{(j)} \times \mathbf{u}_m^{(j)}) + \mathbf{s}_j^{(j)} \times (\mathbf{p}_{j}^{(j)} \times \mathbf{u}_m^{(j)})
\right.
\]

\[
+ \mathbf{s}_m^{(j)} \times (\mathbf{u}_j^{(j)} \times (\mathbf{p}_{j}^{(j)} - \dot{m}_j \mathbf{j}_m^{(j)}))
\]

\[
- K_m K^*_j \mathbf{u}_m^{(j)} \times (\mathbf{u}_j^{(j)} \times (\mathbf{p}_{j}^{(j)} - \dot{m}_j \mathbf{j}_m^{(j)}))
\]

\[
+ K^*_m K^*_j \mathbf{u}_j^{(j)} \times (\mathbf{u}_m^{(j)} \times (\mathbf{p}_{j}^{(j)} - \dot{m}_j \mathbf{s}_m^{(j)}))
\]

\[
- K_m K^*_j \mathbf{r}_j \mathbf{u}_m^{(j)} \times \mathbf{u}_j^{(j)}, \quad m < j.
\]  

(46)

It follows from (34) that

\[
D_{mki} = -D_{jkm} = -K^*_k \mathbf{u}_k^{(m)} \cdot \mathbf{d}_{jm}, \quad m > j > k.
\]  

(47)

In the derivation of (45) and (46), the following relations were first substituted into (40) and (44),

\[
\mathbf{u}_k \times (\mathbf{u}_j \times (\mathbf{r}_j - \mathbf{i}_s)) + \mathbf{u}_j \times (\mathbf{u}_k \times (\mathbf{r}_j - \mathbf{i}_s)) + (\mathbf{u}_k \times \mathbf{u}_j) \times (\mathbf{r}_j - \mathbf{i}_s)
\]

\[= 2 \mathbf{u}_k \times (\mathbf{u}_j \times (\mathbf{r}_j - \mathbf{i}_s)) \]  

(48)

\[
J_j (\mathbf{u}_k \times \mathbf{u}_j) + \mathbf{u}_k \times (J_j \mathbf{u}_j) + \mathbf{u}_j \times (J_j \mathbf{u}_k) = 2 \mathbf{u}_j \times \left( J_j - \frac{1}{2} \text{tr}[J_j] \mathbf{E} \right) \mathbf{u}_j
\]  

(49)

The final form of (46) is obtained by applying the following further equalities,

\[
(\mathbf{u}_m \times \mathbf{r}_j) \times (\mathbf{u}_j \times \mathbf{r}_j) = -\mathbf{r}_j \mathbf{r}_j^T [\mathbf{u}_j \times ] \mathbf{u}_m
\]  

(50)

\[
J_j - \frac{1}{2} \text{tr}[J_j] \mathbf{E} = \mathbf{j}_j - \dot{m}_j \mathbf{r}_j \mathbf{r}_j^T
\]  

(51)

\[
\mathbf{u}_j \times \left( \left( J_j - \frac{1}{2} \text{tr}[J_j] \right) \mathbf{u}_j \right) = \mathbf{u}_j \times (J_j \mathbf{u}_j)
\]  

(52)
Equation (51) follows from (4), (21), (22) and

$$[a \times ][b \times ] - \frac{1}{2} \text{tr}([a \times ][b \times ]E = ba^T$$ (53)

It is worth mentioning that all variables in (46) can also be computed with the recursive algorithm (24)-(30). That means, $H_{mj}$ and $d_{mj}$ are related to the same variables. However, to compute $D_{mj}$ is still time-consuming. Combining (35), (45), and (47), we can alternatively compute the bias vector without the gravity term in a more efficient form of

$$(b)_m - \eta_m = - \sum_{j=1}^{m-1} (\xi_j^{(m)} \cdot \dot{q}_j) + \sum_{j=m+1}^{n} (\xi_j^{(j)} \cdot \dot{q}_j) + d_{mj}$$ (54)

where

$$\xi_j^{(j)} = \sum_{k=1}^{j-1} 2 K_k u_k^{(j)} \dot{q}_k + K_j^* u_j^{(j)} \dot{q}_j.$$ (55)

which can be calculated in a forward recursive form of

$$\xi_j^{(j+1)} = \frac{1}{j+1} R(\xi_j^{(j)} + K_j^* u_j^{(j)} \dot{q}_j) + K_{j+1} u_{j+1}^{(j+1)} \dot{q}_{j+1}$$ (56)

and

$$\xi_j^{(m)} \dot{q}_j = \frac{1}{m} R(\xi_j^{(m-1)} \dot{q}_j), \quad m > j.$$ (57)

It is apparent that $\xi_j^{(j)} = 0$ for $j < N_{fR}$ if the first $(N_{fR} - 1)$ joints are translational. Therefore, we should keep in mind that $d_{mj}$ for $j < N_{fR}$ is unnecessary to be calculated in forming the bias vector.

It is much easier to derive the gravity term, $\eta_m$, in the bias vector. Assume the manipulator is stationary, i.e., $\dot{q}_i = \ddot{q}_i = 0$ for all $i$. The forces exerted on the manipulator are gravitational forces only. The gravitational force of the composite body $j$ is $m_j g$ acting at the center of mass, where $g$ is the gravitational acceleration. The actuator force applied on joint $j$ is to resist the component of the force or torque exerted on joint $j$ by link $j$ along the direction of joint $j$, i.e.,

$$\eta_j = -K_j^* u_j^{(j)} \cdot ((r_j^{(j)} - j s_j^{(j)}) \times \dot{m}_j g^{(j)}) - K_j u_j^{(j)} \cdot \dot{m}_j g^{(j)}$$ (58)

**ALGORITHM**

The literature13,15 has shown that the choice of a body-fixed coordinate system plays an essential role in the computation efficiency of kinematic and dynamic problems of manipulators. In the work,15 the normal driving-axis coordinate system (also called the modified Denavit–Hartenberg notation16) was
Lin: A New Composite Body Method

used to reformulate the recursive Newton–Euler formulation. The resulting formulation is more efficient than the original one based on the Denavit–Hartenberg notation, and made the implementation on an INTEL 8086/8087 microprocessor (5MHz clock) under 17ms for the Stanford manipulator. It is also shown that the reformulated Walker and Orin’s composite body method based on the normal driving-axis coordinate system can save 45 multiplications and 28 additions for a general open chain in comparison with the one based on the Denavit–Hartenberg notation. This experience is adopted again to establish an algorithm for the formulations derived above.

In the normal driving-axis coordinate system (Fig. 2), the z-axis of a body-fixed frame is the driving axis of the corresponding link, i.e., \( \mathbf{u}^{(i)} = [0 \ 0 \ 1]^T \). And the distance from the origin of frame \( E_{i-1} \) to frame \( E_i \) is \( i^{-1} \mathbf{s}^{(i-1)} = [b_i - d_i \ \sin \beta_i \ \beta_i \ \cos \beta_i]^T \), where \( b_i, d_i, \beta_i \) and \( \theta_i \) are the parameters of the coordinate system and are shown in Figure 2. Note, \( d_i = d_i^* + q_i \), \( \theta_i = \theta_i^* \); if joint i is translational; otherwise \( d_i = d_i^* \), \( \theta_i = \theta_i^* + q_i \); i.e., \( d_i \) and \( \theta_i \) are the null-position values of \( d_i \) and \( \theta_i \), respectively.

Figure 2. Illustration of the normal driving-axis coordinate system.
Since it is recommended to assign the base frame $E_0$ coincident with frame $E_1$ in the null-position configuration, the distance between the origins of $E_0$ and $E_1$ is then zero. Thus,

$$
\mathbf{s}_i^{(i)} = \mathbf{u}_i^{(i)} \times \mathbf{s}_i^{(i)} = 0, \quad i \geq 1
$$

$$
\mathbf{p}_i^{(i)} = \mathbf{c}_i^{(i)}
$$

$$
\mathbf{s}^{(2)} = \frac{1}{2} \mathbf{R} \mathbf{s}^{(1)} = \begin{bmatrix}
    b_2 \cos \theta_2 \\
    -b_2 \sin \theta_2 \\
    d_2
\end{bmatrix}
$$

The terms, which are constant and can be calculated in advance, are listed in the part "initialization" of the algorithm shown in Figure 3. Examining (12), we find that $\mathbf{s}_m^{(i)} = \mathbf{u}_m^{(i)} \times \mathbf{s}_m^{(i)}$ is just required for $K_m^* = 1$ in forming $H_{mj}$. If joint $N_R$ is the first rotational joint (i.e., $K_i^* = 0, i < N_R$), only the $(3,3)$th entry of $\mathbf{J}_{N_R}$ is required, and all $\mathbf{J}_i^{(i)}, i < N_R$ are unnecessary since $H_{mj}$ is related to $\mathbf{J}_j^{(i)}$ only in the case of $K_m^* K_j^* = 1, m \leq j$, nor are all $\mathbf{p}_i^{(i)}, i < N_R$. For the case that joint $j$ is a rotational joint, but not the first, we also just need the third column of $\mathbf{s}_j^{(i)}$ because of $\mathbf{u}_j^{(i)} = [0, 0, 1]^T$. This implies that it is unnecessary to convert $\mathbf{J}_j^{(i)}$ to $\mathbf{j}_j^{(i)}$ in the recursive procedure, so that computing $\mathbf{J}_j^{(i)}$ can save a few of additions. All these conclusions apply to the formulation (46) for $d_{mj}$, too.

Let

$$
\mathbf{a}_j = \mathbf{J}_j^{(i)} \mathbf{u}_j^{(i)} + \mathbf{s}_j^{(i)} \times \mathbf{p}_j^{(i)}
$$

$$
= \begin{bmatrix}
    (\mathbf{J}_j^{(i)})_{13} \\
    (\mathbf{J}_j^{(i)})_{23} \\
    -((\mathbf{J}_j^{(i)})_{11} - (\mathbf{J}_j^{(i)})_{22})
\end{bmatrix} + \mathbf{s}_j^{(i)} \times \mathbf{p}_j^{(i)}
$$

$$
\mathbf{e}_j = \mathbf{u}_j^{(i)} \times (\mathbf{p}_j^{(i)} - \hat{\mathbf{m}}_j \mathbf{s}_j^{(i)}) = \mathbf{u}_j^{(i)} \times \mathbf{p}_j^{(i)} - \hat{\mathbf{m}}_j \mathbf{s}_j^{(i)}
$$

$$
\mathbf{B}_j = -\mathbf{J}_j^{(i)} [\mathbf{u}_j^{(i)} \times ] + [\mathbf{s}_j^{(i)} \times ] [\mathbf{p}_j^{(i)} \times ]
$$

where $(\mathbf{J})_{ij}$ denotes the $(i, j)$th-entry of $\mathbf{J}$. The formulations (12), (46), and (58) can be rewritten as

$$
H_{mj} = K_m^* K_j^* (\mathbf{a}_j \cdot \mathbf{u}_m^{(j)} - \mathbf{s}_m^{(j)} \cdot \mathbf{e}_j)
$$

$$
+ K_m^* K_j^* \mathbf{e}_j \cdot \mathbf{u}_m^{(j)}
$$

$$
+ K_m^* K_j ((\mathbf{u}_m^{(j)} \times \mathbf{p}_m^{(j)}) z - \hat{\mathbf{m}}_j (\mathbf{s}_m^{(j)}) z)
$$

$$
+ K_m K_j \hat{\mathbf{m}}_j (\mathbf{u}_m^{(j)}) z \quad m \leq j
$$

$$
d_{mj} = K_m^* K_j^* (\mathbf{B}_j \mathbf{u}_m^{(j)} + \mathbf{s}_m^{(j)} \times \mathbf{e}_j)
$$

$$
+ K_m K_j^* \mathbf{e}_j \times \mathbf{u}_m^{(j)}$$
Lin: A New Composite Body Method

**INITIALIZATION:**

\[ m_n = m_n \]

DO \( i = n \) to 1 with increment of -1

\[ I_i^{(i)} = I_i^{(i)} - \frac{1}{2} \chi [I_i^{(i)}] \]

END DO \( i \)

\( N_{FR} \equiv \) No. of the First Rotational Joint

IF (no Rotational Joints), \( N_{FR} = n + 1 \)

\[ \frac{\partial p^{(i)}}{\partial \theta} = 0 \]

\[ \frac{\partial p^{(i)}}{\partial \theta} = 0 \]

\[ p_i^{(i)} = m_1 \frac{\partial p^{(i)}}{\partial \theta} \]

\( (J_i^{(i)})_{33} \equiv (I_i^{(i)})_{33} + m_1 ((p^{(i)})_x^2 + (p^{(i)})_y^2) \)

IF \( (N_{FR} > 1) \) THEN

DO \( j = 1 \) to \( (N_{FR} - 1) \)

\[ g^{(i)} = \mathbf{R} g^{(i-1)} \]

END DO \( j \)

RETURN

END

**START:**

**Form** \( \mathbf{R}, \frac{\partial \mathbf{p}^{(i)}}{\partial \theta}, \frac{\partial \mathbf{a}^{(i)}}{\partial \theta} \)

<table>
<thead>
<tr>
<th>Number of Operations</th>
<th>Times</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial \mathbf{R}}{\partial \theta} )</td>
<td>K(_i^\theta) : 4M 1s/c</td>
</tr>
<tr>
<td>( \frac{\partial \mathbf{p}^{(i)}}{\partial \theta} )</td>
<td>K(_i) : 2M</td>
</tr>
<tr>
<td>( \frac{\partial \mathbf{a}^{(i)}}{\partial \theta} )</td>
<td>K(_i) : 2M 1A</td>
</tr>
</tbody>
</table>

**FOR THE CASE OF** \( n = 1 \)

IF \( (n = 1) \) THEN

\[ H_{11} = K_1^\theta (J_1^{(1)})_{33} + K_1 m_1 \]

\( (b)_1 = -K_1^\theta (m_1 e_1^{(1)}) x - K_1 \eta_1 \)

RETURN

END IF \( (n = 1) \)

**Form** \( \frac{\partial \mathbf{p}^{(i)}}{\partial \theta}, \frac{\partial \mathbf{a}^{(i)}}{\partial \theta} \)

\( \frac{\partial \mathbf{a}^{(i)}}{\partial \theta} = [b_2 \cos \theta_2 - b_2 \sin \theta_2 \ d_2]^T \)

DO \( i = 2 \) to \( n \)

\( \frac{\partial \mathbf{a}^{(i)}}{\partial \theta} = \mathbf{R} \left( \frac{\partial \mathbf{a}^{(i-1)}}{\partial \theta} + \frac{\partial \mathbf{a}^{(i-1)}}{\partial \theta} \right) \)

\( (b)_i = [b_2 \cos \theta_2 - b_2 \sin \theta_2 \ d_2]^T \)

END DO \( i \)

**Form** \( \mathbf{u}_j^{(i)}, \frac{\partial \mathbf{a}^{(i)}}{\partial \theta} \) FOR \( i > j \)

DO \( j = 1 \) to \( n - 1 \)

\( \mathbf{u}_j^{(i)} = \mathbf{R} \left[ \begin{array}{c} 0 \\ 1 \end{array} \right]^T \)

END IF \( (n = 1) \)

\( 1 \) The terms enclosed within a box should be excluded when it is to form the inertia matrix only.

**Figure 3.** Algorithm of the new composite body method.
IF \((j \geq 2 \text{ AND } K_j = 0)\), \(\delta j^{<j+1>}\) = \(\delta j^{<j+1>} = j+1R[-(dA^{<j>})v \ (dA^{<j>})e 0]^T\)

IF \((j < n - 1)\) THEN
DO \(i = j + 2\) to \(n\)

\[ u_j^{<j>} = -1R u_j^{j-1<} \]

IF \((i \geq 2 \text{ AND } K_i = 0)\), \(\delta i^{<i>} = -1R \delta i^{j-1<} \)

END DO \(i\)

END IF \((j < n - 1)\)

END DO \(j\)

- **FORM \(\delta j^{<j<} \), \(\delta j^{<j+1>} \)**

IF \((N_{HR} > n)\), GO TO "calculating \(H_{mj}\) and \(d_{mj}\)"

\[ \hat{p}^{<n>}_n = m_n \hat{p}^{<n>} \]

IF \((n > N_{HR})\), \(\hat{J}^{<n>}_n = I_n^{<n>} - (m_n \hat{p}^{<n>})(\hat{p}^{<n>})^T\) \(3M 6A\)

IF \((n = N_{HR})\), \((\hat{J}^{<n>}_n)_{33} = (I_n^{<n>})_{33} + (m_n \hat{p}^{<n>})_e (\hat{p}^{<n>})_e + (m_n \hat{p}^{<n>})_v (\hat{p}^{<n>})_v \) \(2M 2A\)

IF \((N_{HR} = n)\), GO TO "calculating \(H_{mj}\) and \(d_{mj}\)"

DO \(i = n + 1\) to \(N_{HR}\) with increment of \(-1\)

IF \((i \neq 1)\), \(p_i^{<i>} = m_i \hat{p}^{<i>} + \hat{p}^{<i>}R \hat{p}_n^{<j+1>} \)

IF \((i = 1)\), \(p_i^{<i>} = R \hat{p}_n^{<j+1>} \)

IF \((i > N_{HR})\), \(\delta i^{<i>}_{i+1} = I^{<i>}_{i+1} - (m_i \hat{p}^{<i>})(\hat{p}^{<i>})^T + \hat{p}^{<i>}R \hat{J}^{<j+1>} + \hat{p}^{<i>}R^T \)

IF \((i = N_{HR})\) THEN

\(\delta i^{<i>}_{i+1} = \hat{J}^{<j+1>}_{i+1} - \hat{p}^{<i>}(\hat{J}^{<j+1>})_{i+1}R^T \)

IF \((i \neq 1)\), \((\hat{J}^{<i>}_{i})_{33} = (I^{<i>})_{33} + (m_i \hat{p}^{<i>})_e (\hat{p}^{<i>})_e + (m_i \hat{p}^{<i>})_v (\hat{p}^{<i>})_v \)

IF \((i = 1)\), \((\hat{J}^{<i>}_{i})_{33} = (I_{i})_{33} + \hat{p}^{<i>}(\hat{J}^{<j+1>})_{i+1}R^T_{i+1} \)

END IF \((i = N_{HR})\)

END DO \(i\)

- **CALCULATE \(H_{mj}\) AND \(d_{mj}\)**

\(H_{11} = K_i^2 \hat{(J}^{<1>})_{33} + K_1 m_1\)

DO \(j = 2\) to \(n\)

IF \((K_j = 0)\) THEN

\(e_j = \left[ \begin{array}{c} -\hat{p}^{<j>}_j e_f - m_j (\hat{e}^{<j>} f) e_f \\ (\hat{p}^{<j>}_j e_f - m_j (\hat{e}^{<j>} f) e_f) \end{array} \right] \)

\(K_j^2 = 2M 2A\)

IF \((j > N_{HR})\) THEN

\(A = \delta_j^{<j>}(\hat{p}_j^{<j>})^T\)

\(B = \left[ \begin{array}{ccc} -\hat{J}^{<j>}_{11} & \hat{J}^{<j>}_{21} & \hat{J}^{<j>}_{31} \\ -\hat{J}^{<j>}_{12} & \hat{J}^{<j>}_{22} & \hat{J}^{<j>}_{32} \\ -\hat{J}^{<j>}_{13} & \hat{J}^{<j>}_{23} & \hat{J}^{<j>}_{33} \end{array} \right] \)

\(K_j^2 = 7M 4A\)

END IF \((j > N_{HR})\)

END DO \(j\)

Figure 3. continued
$$a_j \equiv \begin{bmatrix} (J_j^{(i)})_{13} \\ (J_j^{(i)})_{23} \\ -(J_j^{(i)})_{11} - (J_j^{(i)})_{22} \end{bmatrix} + \begin{bmatrix} (A)_{23} \\ -(A)_{13} \\ (A)_{12} - (A)_{21} \end{bmatrix} K_j^n; 5A \quad n - N_{JR}
$$

ELSE ($j = N_{JR}$)

$$(a_j)_i \equiv (J_j^{(i)})_{33} + (\hat{s}_j^{(i)})_y (\hat{p}_j^{(i)})_y - (\hat{s}_j^{(i)})_y (\hat{p}_j^{(i)})_y; 2M 2A \quad 1 \text{ for } 2 \leq N_{JR} \leq n$$

END IF ($j > N_{JR}$)

END IF ($K_j = 0$)

$$H_{ij} = K_j ((a_j)_i - \hat{s}_j^{(i)} \cdot e_j) + K_i \tilde{m}_j K_j; 2M 2A \quad n - 1$$

DO $m = 1$ to $j - 1$

$$H_{mj} = K_m K_j (u_m^{(i)} + K_m K_j e_j \cdot u_m^{(i)}) + K_m K_j m_j (u_m^{(i)} - (u_m^{(i)}))$$

END DO

$$H_{mj} = K_m K_j; 3M 2A \quad \forall m \leq N_{JR}$$

$$K_j; 2M 1A \quad n(n - 1)/2$$

$$K_j; 1M \quad \forall m > N_{JR}$$

END DO

DO $m = 1$ to $j - 1$

$$H_{mj} = K_m K_j (u_m^{(i)} + K_m K_j e_j \cdot u_m^{(i)})$$

END DO

$$H_{mj} = K_m K_j; 3M 2A \quad \forall m \leq N_{JR}$$

$$K_j; 2M 1A \quad (n - N_{JR} + 1) \times$$

$$K_j; 4M 2A \quad (n + N_{JR} - 2)/2$$

$$K_j; 2M \quad \forall m > N_{JR}$$

END DO

IF ($j > N_{JR}$), $m_{mj} = K_m K_j B u_m^{(i)} + K_m K_j e_j \times u_m^{(i)}$

$$+ K_m K_j (u_m^{(i)} x p_m^{(i)}) + K_m K_j m_j (u_m^{(i)} + K_m K_j e_j \cdot u_m^{(i)})$$

END IF ($m \neq 1$)

IF ($m \neq 1$)

$$H_{mj} = H_{mj} - K_m K_j \hat{s}_m^{(i)} \cdot e_j - K_m K_j m_j (\hat{s}_m^{(i)}), 2M 2A$$

$$K_m K_j; 2M 1A \quad (n - N_{JR} - 1)$$

$$K_m K_j; 1M 1A \quad \forall m > N_{JR}$$

END DO

IF ($j > N_{JR}$), $m_{mj} = m_{mj}$

$$+ K_m K_j (u_m^{(i)} x p_m^{(i)}) + K_m K_j m_j (u_m^{(i)} + K_m K_j e_j \cdot u_m^{(i)})$$

END IF ($m \neq 1$)

END DO

END IF ($N_{JR} \leq n$)

END DO

END DO

END IF ($N_{JR} \leq n$)

END IF ($N_{JR} = n + 1$) GO TO "calculating ($b_m$)"

$$\langle \zeta_{N_m+1}^{N_{m+1}} \rangle_r \equiv \langle \zeta_{N_m+1}^{N_{m+1}} \rangle_r \quad 3M$$

$$\langle \zeta_{N_m+1}^{N_{m+1}} \rangle_r \quad 3M$$

Figure 3. continued
IF ($N_{FR} = n - 1$), GO TO “calculating $\xi_j^{<m>}$”

DO $j = N_{FR} + 2$ to $n$

\[
\xi_j^{<i>} = \frac{1}{j-1}R_j(\xi_j^{<i-1>} + K_j^{<i>} u_j^{<i-1>} \dot{q}_j^{-1}) \\
+ K_j u_j^{<i>} \dot{q}_j
\]

\[
\xi_j^{<j>} = \xi_j^{<j-1>} \dot{q}_j
\]

END DO $j$

■ CALCULATE $\xi_j^{<m>}$ ■

\[
\dot{q}_{Nm} = \dot{q}_{Nm} \times \dot{q}_{Nm} 
\]

DO $m = N_{FR} + 1$ to $n$

\[
\xi_j^{<m>} = u_j^{<m>} \dot{q}_{Nm}
\]

END DO $m$

IF ($N_{FR} = n - 1$), GO TO “calculating $(b)_m$”

DO $j = N_{FR} + 1$ to $n - 1$

DO $m = j + 1$ to $n$

\[
\xi_j^{<m>} = m^{-1}R_j \xi_j^{<m-1>}
\]

END DO $m$

■ CALCULATE $(b)_m$ ■

DO $m = 1$ to $n$

\[
(b)_m = \eta_m
\]

IF ($N_{FR} \leq n$) THEN

IF ($N_{FR} < m$), $(b)_m = (b)_m - \xi_j^{<m>} \cdot d_{N(m)}m$

\[
3M 3A \quad n - N_{FR}
\]

IF ($N_{FR} > m$), $(b)_m = (b)_m + \xi_j^{<m>} \cdot d_{m(N(m))}$

\[
1M 1A \quad N_{FR} - 1
\]

END IF ($N_{FR} \leq n$)

IF ($N_{FR} < n$) THEN

DO $j = N_{FR} + 1$ to $n$

IF ($j < m$), $(b)_m = (b)_m - \xi_j^{<j>} \cdot d_{jm}$

\[
3M 3A \quad (n - 1)(n - N_{FR})
\]

IF ($j > m$), $(b)_m = (b)_m + \xi_j^{<j>} \cdot d_{mj}$

END DO $j$

END IF ($N_{FR} < n$)

END DO $m$

RETURN

END

Figure 3. continued

\[
\begin{align*}
K_j^{*}K_j \left[ \begin{array}{c} -(u_j^{(j)})_y \\
(u_j^{(j)})_y \\
\end{array} \right] + \dot{m}_j(g_j^{<j>})_y \\
+ K_j^{*} \dot{m}_j \left[ \begin{array}{c} -(u_j^{(j)})_y \\
(u_j^{(j)})_y \\
\end{array} \right] \\
= 0 \\
\end{align*}
\]

\[
\begin{align*}
\eta_m = -K_j^{*}e_m \cdot (g_j^{<m>})_z \\
K_j^{*} \dot{m}_m(g_j^{<m>})_z \\
\end{align*}
\]

where ($\cdot$) denotes the x-component and etc.

Using (25)–(30), (54), (56), (57), and (62)–(67), we establish the algorithm for numerically forming the dynamic equations of a general manipulator and show it in Figure 3. It should be noted that the computations of the matrix product of
Table I. Comparison of the efficiency of five algorithms for calculating the inertial matrix.a

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of Rotational Joints</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Walker and M</td>
<td>$11n^2 + 53n - 74$</td>
<td>76</td>
<td>184</td>
<td>314</td>
<td>466</td>
<td>640</td>
<td>836</td>
<td>1054</td>
</tr>
<tr>
<td>Orin$^{1,b}$</td>
<td>$8n^2 + 52n - 62$</td>
<td>74</td>
<td>166</td>
<td>274</td>
<td>398</td>
<td>538</td>
<td>694</td>
<td>866</td>
</tr>
<tr>
<td>Renaud$^{6,7,c}$</td>
<td>$13.5n^2 + 22.5n - 68$</td>
<td>31</td>
<td>121</td>
<td>238</td>
<td>382</td>
<td>553</td>
<td>751</td>
<td>976</td>
</tr>
<tr>
<td>Lin$^{13}$</td>
<td>$(7n^3 + 99n^2 - 214n)/6 + 19$</td>
<td>23</td>
<td>92</td>
<td>215</td>
<td>399</td>
<td>651</td>
<td>978</td>
<td>1387</td>
</tr>
<tr>
<td>Present</td>
<td>$10.5n^2 + 38.5n - 85$</td>
<td>34</td>
<td>125</td>
<td>237</td>
<td>370</td>
<td>524</td>
<td>699</td>
<td>895</td>
</tr>
<tr>
<td></td>
<td>$6n^2 + 39n - 70$</td>
<td>32</td>
<td>101</td>
<td>182</td>
<td>275</td>
<td>380</td>
<td>497</td>
<td>626</td>
</tr>
</tbody>
</table>

a M: Multiplication, A: Addition/Subtraction.

The estimate of the required computations is based on the reformulated one in the work$^{13}$ but the estimated computations herein are $2(3M + 3A)(n - 1)$ less than those in the work$^{13}$ since $b\mathbf{a}^T - (a \cdot b) \mathbf{E} = [a \times][b \times]$ was not recognized.

The computational complexity is reestimated in Appendix.

The other one$^{12}$ is somewhat less efficient.

A comparison of the efficiency of this algorithm with the others in numerically forming the inertia matrix is shown in Table I. It is apparent that the present method is the most efficient for computing the inertia matrix of a manipulator with five or more joints, whereas the algorithm described in the work$^{13}$ is preferable for a manipulator with 4 or less joints. The present method is also superior to the most efficient algorithms of the recursive Newton–Euler formulation$^{3,4}$ in numerically forming the bias vector for a manipulator with six or less joints, which can be seen from Table II.

CONCLUSION

A new efficient composite body method has been derived. It is found that the algorithm of this method to numerically form the inertia matrix is more efficient
Table II. Comparison of the efficiency of three algorithms for calculating the bias vector.

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of Rotational Joints, ( n \geq 2 )</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Khalil and Kleinfinger(^3)</td>
<td>M: (101n - 129)</td>
<td>73</td>
<td>174</td>
<td>275</td>
<td>376</td>
<td>477</td>
<td>578</td>
<td>679</td>
</tr>
<tr>
<td></td>
<td>A: (90n - 118)</td>
<td>62</td>
<td>152</td>
<td>242</td>
<td>332</td>
<td>422</td>
<td>512</td>
<td>602</td>
</tr>
<tr>
<td>Balafoutis et al.(^4)</td>
<td>M: (93n - 69)</td>
<td>117</td>
<td>210</td>
<td>303</td>
<td>396</td>
<td>489</td>
<td>582</td>
<td>675</td>
</tr>
<tr>
<td></td>
<td>A: (81n - 66)</td>
<td>96</td>
<td>177</td>
<td>258</td>
<td>339</td>
<td>420</td>
<td>501</td>
<td>582</td>
</tr>
<tr>
<td>Present</td>
<td>M: (12.5n^2 + 5.5n + 3)</td>
<td>64</td>
<td>132</td>
<td>225</td>
<td>343</td>
<td>486</td>
<td>654</td>
<td>847</td>
</tr>
<tr>
<td></td>
<td>A: (9n^2 + n)</td>
<td>38</td>
<td>84</td>
<td>148</td>
<td>230</td>
<td>330</td>
<td>448</td>
<td>584</td>
</tr>
</tbody>
</table>

\(a\) M: Multiplication, A: Addition/Subtraction.

Table II. Comparison of the efficiency of three algorithms for calculating the bias vector.

than the earlier works for a manipulator with five or more joints. For a manipulator with six or less joints, it is also recommended to use the present method to compute the bias vector. The secret of the present method lies in that all points are referred to the base frame. Since the distances are all measured from the same point (the origin of the base frame), the coefficients in second-order part \((n^2)\) of the required computations are reduced in comparison with the other types of composite body methods.

We have also tried to reformulate the new composite body formulations by changing the reference point from the origin of the base frame to that of any body-fixed frame. However, the required computations increase. Someone would suggest to apply the concept of the identical reference point to Renaud's formulation. Unfortunately, the invariant property of the inertia parameters in Renaud's formulation will be destroyed.

To accomplish the dynamic simulation, we still need a linear equation solver and an integration technique. According to the numerical experiment,\(^1\) it is preferable to use the LDL\(^T\) decomposition to solve the linear equations and to apply the fourth order Adams–Bashforth integration method for the dynamic simulation of manipulators.

Kasahara et al.\(^1\) have applied the parallel computation process to Walker and Orin's method.\(^5\) The parallel computation process can also apply to the present method.

This article was supported in part by the National Science Council, Taiwan under grant No. NSC79-0422-E009-02.

**APPENDIX: ALGORITHM OF RENAUD'S FORMULATION**

Renaud's formulation for generating the inertia matrix of a manipulator with only rotational joints is presented in references 6 and 7. The extension to a general manipulator is derived by Burdick.\(^9\) Vukobratovic et al.\(^10\) extended Renaud's formulation to the bias vector for a manipulator with only rotational joints, and proposed an algorithm of Renaud's formulation based on the modified Denavit–Hartenberg notation for numerically forming the inertia matrix and the bias vector. However, the distance between the origins of two adjacent
frames was mistaken to \( \mathbf{J}_n^{(i-1)} = [b_i, 0, d_i]^T \) (cf. (5.2) in the work\(^1\) and the fourth section of this article). The required computations were, thereafter, underestimated. For the purpose of comparing the efficiency of several methods for numerically forming the inertia matrix, we re-establish an algorithm of Renaud’s formulation in a natural programming language in Figure A1.

![Algorithm of Renaud’s formulation for the inertia matrix.](image)
The general form of Renaud’s formulation is\(^9\)

\[
H_{mj} = K_n^* K_j^* \{ u_m^{(j)} \cdot (\hat{j}_x^{(j)} u_j^{(j)}) + m\hat{\mathbf{s}}^{(j)} \cdot (u_j^{(j)} \times \hat{e}_j^{(j)}) \} + K_n^* K_j^* u_m^{(j)} \cdot (u_j^{(j)} \times \hat{e}_j^{(j)}) + K_n^* K_j^* [\hat{\mathbf{m}}_j \times u_j^{(j)}] \cdot (\hat{u}_j^{(j)} \times u_j^{(j)}) + K_n K_j m_j u_m^{(j)} \cdot u_j^{(j)}, \quad m \leq j. \tag{A1}
\]

where the notation is identical to that in the above text except that

\[
\hat{\mathbf{e}}_j^{(j)} = \sum_{i=j}^n m_i j^p_i \tag{A2}
\]

\[
\hat{j}_x^{(j)} = \sum_{i=j}^n \mathbf{T}_i^{(j)} - m_i [j^p_i \times [j^p_i \times \mathbf{j}^{(j)}]] \tag{A3}
\]

\[
j_m \hat{\mathbf{s}}^{(j)} = u_m^{(j)} \times j_m \hat{\mathbf{s}}^{(j)} \tag{A4}
\]

\(\hat{\mathbf{e}}_j\) and \(\hat{j}_x\) are the first moment and the inertia tensor of the composite body \(j\) about the origin of frame \(E_j\), respectively.
Defining the constants concerning the inertia property,
\[ \ddot{c}_j^{(j)} = m_j \dot{c}_j^{(j)} + K_j^{*} \dot{m}_j + K_j^{*} \dot{s}_j^{(j)} + 1 
+ \text{constant terms} \]
(5)

\[ \ddot{j}_j^{(j)} = \dot{j}_j^{(j)} + \dot{m}_j \dot{s}_j^{(j)} + \dot{m}_j \dot{j}_j^{(j)} \times [\dot{s}_j^{(j)} \times \dot{j}_j^{(j)}] 
- K_j^{*} \dot{m}_j + K_j^{*} \dot{s}_j^{(j)} \times [\dot{j}_j^{(j)} \times \dot{s}_j^{(j)}] \]
(6)

then we get
\[ \ddot{\dot{c}}_j^{(j)} = \ddot{c}_j^{(j)} + \dot{m}_j \dot{\dot{c}}_j^{(j)} + K_j^{*} \dot{\dot{m}}_j + K_j^{*} \dot{\dot{s}}_j^{(j)} \]
(7)

\[ \ddot{j}_j^{(j)} = \ddot{j}_j^{(j)} + \dot{m}_j \ddot{j}_j^{(j)} + \dot{m}_j \ddot{j}_j^{(j)} \times [\ddot{s}_j^{(j)} \times \ddot{j}_j^{(j)}] 
- K_j^{*} \dot{m}_j + K_j^{*} \ddot{s}_j^{(j)} \times [\ddot{j}_j^{(j)} \times \ddot{s}_j^{(j)}] \]
(8)

\[ \dot{m}_j \dddot{s}_j^{(j)} = \dot{m}_j \dddot{s}_j^{(j)} + \dddot{c}_j^{(j)} \times \dddot{j}_j^{(j)} \times \dddot{j}_j^{(j)} \]
(9)

from (5)–(7). It should be remarked that most of works dealing with
Renault’s invariant terms made a mistake that (6) and (7) with “1” replacing
K_j^{*} j + 1 were still seen as invariant to the manipulator motion for translational joint
j + 1. In fact, for translational joint j + 1 varies with d_j + 1, the joint
displacement of translational joint j + 1. The required computations of (6) for
a manipulator with n rotational joints are (13.5n^2 + 22.5n - 68)M + (9n^2 + 26n
- 64)A.

The algorithm proposed by Fijany and Bejczy uses the recursive forms
(7) and (8) to calculate the equivalent force (\textbf{f}_E), and couple (\textbf{t}_E), at the
origin of frame j, of the inertia force and torque of the composite-body j due to
\( \ddot{q}_j \),
\[ -\textbf{f}_E = \dot{m}_j \dddot{q}_j + K_j^{*} \dddot{u}_j \times \dddot{q}_j \]
(10)

\[ -\textbf{t}_E = \dot{m}_j \dddot{q}_j + K_j^{*} \dddot{u}_j \times \dddot{q}_j \]
(11)

The first terms on the right-hand sides of (10) and (11) were erroneously
ignored in the work (cf. (54) and (55) in reference 12).

The actuator force applied on joint m is to resist the component of the force
or torque exerted on joint m by link m along the direction of joint m. The fact
that the (m, j)th entry of the inertia matrix is the part of the actuator force of
joint m due to \( \ddot{q}_j \) results in that
\[ H_{ij} = K_j \dot{m}_j + K_j^{*} \dot{u}_j \cdot (\dddot{q}_j \dot{u}_j) \]
(12)

\[ H_{m} = K_m \dot{u}_m \cdot \dot{u}_m + K_m^{*} \dot{u}_m \cdot \dot{u}_m, \quad m < j \]
(13)
where

$$f^{(k)}_{m,j} = -j_k R^j_{E_{ij}}$$  \hspace{1cm} (A14)$$

$$t^{(k)}_{m,j} = t^{(k)}_{m+1,j} + m_{m+1}^{(k)} \times f^{(k)}_{m+1,j}$$ \hspace{1cm} (A15)$$

Note that \( t^{(k)}_{j,j} = t^{(k)}_{E_{ij}} \).

Fijany and Bejczy found that the selection of frame \( k = 2 \) or the frame of the end-effector\(^{12}\) to represent \( f^{(k)}_{j,j} \) and \( t^{(k)}_{m,j} \) can make the algorithm more efficient than the selection of other frame. However, \( H_{ij} \) and \( H_{mj}, m < j \) can be expanded explicitly to be (A1), which is more efficient than Fijang and Bejczy’s algorithm (Table I).

References