David E. Orin
William W. Schrader
Department of Electrical Engineering
The Ohio State University
Columbus, Ohio 43210

Efficient Computation of the Jacobian for Robot Manipulators

Abstract

This paper discusses and compares six different methods for calculating the Jacobian for a general N*-degree-of-freedom manipulator. We enumerate the computational efficiency of each in terms of the total number of multiplications, additions/subtractions, and trigonometric functions required as well as in terms of the number of matrix-vector operations needed. We also give the execution times on a PDP-11/70 minicomputer for determining the Jacobian for an example seven-degree-of-freedom manipulator. This paper formulates one of the best new methods for determining the Jacobian.*

1. Introduction

The control for most state-of-the-art industrial robots is relatively simple, based on servos at each joint. However, the controls for the next generation of robots must allow adaptation to the environment through the use of various kinds of sensory information (Dodd and Rossol 1979). For this case, control is most naturally implemented in Cartesian coordinates at the end-effector.

Some kind of transformation is needed between end effector coordinates and joint coordinates. An inverse kinematics procedure may be used to compute joint angles when the end-effector position and orientation are known. However, this may result in extremely complex equations that are difficult to derive and to implement in real time for control (Duffy 1980). Furthermore, for the case of a redundant manipulator (more than 6 degrees of freedom), the inverse kinematics equations are in general underspecified and other approaches to the control based on the Jacobian have been proposed (Liegeois 1977; Klein and Huang 1983; Trevelyan, Kovesi, and Ong 1983; Yoshikawa 1983).

The Jacobian relates joint rates to end-effector rates and may be used if control is based on resolved rate (Whitney 1969) and if position feedback is closed at the end-effector (Klein and Briggs 1980). Also, the transpose of the Jacobian relates end-effector forces to joint torques and may be used when force control is needed (Wu and Paul 1982).

This paper compares several approaches that have been proposed for computing the Jacobian, including a new approach introduced here. These approaches are compared in terms of their computational efficiency since the Jacobian must be computed in real time for control. An enumeration is made of the total number of multiplications, additions/subtractions, and trigonometric functions that are required to determine the Jacobian. These are given as a function of the total number of degrees of freedom, N. The best approaches give computation that varies linearly with N.

The execution times for determining the Jacobian (using PASCAL on a PDP-11/70 minicomputer) for an example seven-degree-of-freedom manipulator is also presented. Finally, the number of matrix-vector operations involved in computing the Jacobian for each approach is enumerated. These may provide the most important basis of comparison for future matrix-vector processors.

1.1. Notation

The basic parameters used to describe the kinematics of a manipulator, as first presented by Denavit and Hartenberg (1955) and extended and given in detail in Paul (1981) are used throughout. In particular, a coordinate system is attached to each link of the manipulator with the z-axis directed along the joint axis. The links are numbered from 0 at the base to N at the end-effector. A separate coordinate system, labeled with the subscript $N+1$ (or E for end-effector) is also fixed to the end-effector at any desired point.

The International Journal of Robotics Research,
Vol. 3, No. 4, Winter 1984.
0278-3649/84/040066-10 $05.00/00,

Fig. 1. Link parameters associated with link i.

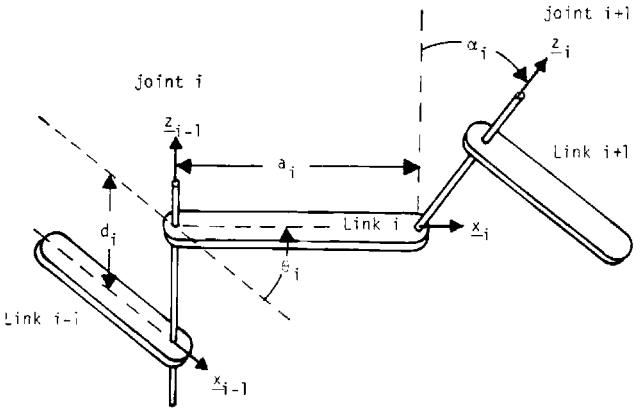
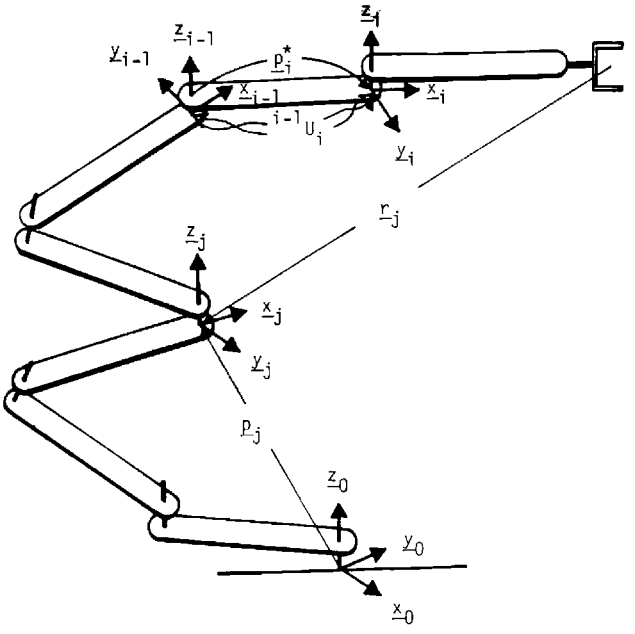


Fig. 2. Depiction of vectors p_j, r_j, p_i^*, and transform ${}^{i-1}U_i$ for a manipulator.



The relative position of joint i, q_i, is with respect to the **z**-axis of the previous link, $\mathbf{z}_{i-1}$. That is, links $i-1$ and i are connected at joint i.

Four parameters are used to describe each successive joint and link pair—the joint angle (θ_i) and offset distance (d_i) as well as the link length (a_i) and twist (α_i) (see Fig. 1). From these parameters, the 4×4 homogeneous transformation, ${}^{i-1}\mathbf{T}_i$, which relates positions in coordinate system i to those in coordinate system $i-1$, may be computed. For either a rotational or a sliding joint the result is as follows (Paul 1981):

$$ {}^{i-1}\mathbf{T}_i = \left[\begin{array}{cccc} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ \hline \mathbf{0} & \mathbf{0} & 0 & 1 \end{array}\right], \qquad (1) $$

where $c\theta_i$ and $s\theta_i$ indicate the cosine and sine, respectively. The transformation between other link coordinate systems may be obtained through multiplication of the intermediate transformation matrices.

The ${}^{i-1}\mathbf{T}_i$ matrix gives both position and orientation changes between successive coordinate systems. The top-left 3×3 part of the matrix gives all orientation information and will be denoted as ${}^{i-1}\mathbf{U}_i$. The first three elements of the right-hand column give all the relative position information and will be denoted as ${}^{i-1}\mathbf{p}_i^*$. Also of value in describing the various approaches to computing the Jacobian are the use of position vectors from the end-effector to link j, $\mathbf{r}_j$, and from the base to link j, $\mathbf{p}_j$. These are all shown in Fig. 2.

2. The Jacobian Matrix

The final form of the Jacobian matrix, as determined in the various approaches, may differ in two important ways:

1. The components may be expressed in any coordinate system from 0 to $N+1$.
2. The reference point on the end-effector (either real or ficticious) for which the translational velocity is instantaneously computed may be chosen somewhat arbitrarily.

These lead to the following notation for the Jacobian matrix, **J**, and the variables it relates:

$$ \begin{bmatrix} \ell_\omega \\ \ell_v \end{bmatrix}_m = {}^{\ell}\mathbf{J}_m[\dot{\mathbf{q}}], \qquad 0 \leq \ell, m \leq N+1, \qquad (2) $$

where

$$
{}^{\ell}\mathbf{J}_m = \begin{bmatrix}
{}^{\ell}\gamma_1^x & {}^{\ell}\gamma_2^x & \cdot & \cdot & \cdot & {}^{\ell}\gamma_N^x \\
{}^{\ell}\gamma_1^y & {}^{\ell}\gamma_2^y & \cdot & & & {}^{\ell}\gamma_N^y \\
{}^{\ell}\gamma_1^z & \cdot & & \cdot & & {}^{\ell}\gamma_N^z \\
{}^{\ell}\beta_1^x & \cdot & & & \cdot & {}^{\ell}\beta_N^x \\
{}^{\ell}\beta_1^y & \cdot & & & & {}^{\ell}\beta_N^y \\
{}^{\ell}\beta_1^z & & & & & {}^{\ell}\beta_N^z
\end{bmatrix}_m \tag{3}
$$

and $\boldsymbol{\omega}$ is the rotational velocity vector of the end-effector, $\mathbf{v}$ is the translational velocity vector of the end-effector, and $\dot{\mathbf{q}}$ is the vector of joint rates. The components of the end-effector velocity and the Jacobian are expressed in the ℓth link coordinate system, as indicated by the leading superscript. The reference point for the end-effector velocity is understood to be instantaneously at the origin of coordinate system m, as indicated by the trailing subscript. For m equal to other than N or $N+1$, the velocity reference point is a fictitious point not physically on the end-effector. However, it may be chosen in this way to reduce the computational complexity of determining the Jacobian.

In the following sections, six different methods for computing the Jacobian matrix are presented. In all cases, the fundamental approach from the appropriate reference is given. The notation, however, is standardized, and nonessential details (for purposes of this comparison) are eliminated.

Several of the methods for computing the Jacobian are based on earlier work by Pieper (1968) and Whitney (1972). In fact, in many cases the latest work is just a detailed elaboration of the previous work. However, since the exact computational approach is not made explicit in Pieper and since closer attention to computational detail has resulted in somewhat greater efficiency than that presented in Whitney, these earlier methods will not be compared in this paper.

2.1. Method I

In the first method, as presented by Vukobratović and Potkonjak (1979), the angular velocity of link i is written as a linear function of the previous joint velocities:

$$
{}^{i}\boldsymbol{\omega}_i = \sum_{j=1}^{i} {}^{i}\gamma_j \dot{q}_j. \tag{4}
$$

For a rotational joint, the angular velocity of link i may be computed from that for link $i-1$ and the relative rate at joint i:

$$
\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{q}_i \mathbf{z}_{i-1}. \tag{5}
$$

The appropriate component equation is

$$
{}^{i}\boldsymbol{\omega}_i = {}^{i-1}\mathbf{U}_i^T \left\{ {}^{i-1}\boldsymbol{\omega}_{i-1} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \dot{q}_i \right\}. \tag{6}
$$

The general form for ${}^{i-1}\boldsymbol{\omega}_{i-1}$,

$$
{}^{i-1}\boldsymbol{\omega}_{i-1} = \sum_{j=1}^{i-1} {}^{i-1}\gamma_j \dot{q}_j, \tag{7}
$$

may be used in Eq. (6), with the following result:

$$
{}^{i}\boldsymbol{\omega}_i = \sum_{j=1}^{i-1} [{}^{i-1}\mathbf{U}_i^T \, {}^{i-1}\gamma_j \dot{q}_j] + {}^{i-1}\mathbf{U}_i^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \dot{q}_i. \tag{8}
$$

Equating the coefficients for Eqs. (4) and (8) results in a recursive method for computing ${}^{i}\gamma_j$. Using a similar approach for the translational velocity (with coefficients ${}^{i}\beta_j$) and considering the case of sliding joints result in the following sets of recursive equations:

$$
{}^{i}\gamma_i = {}^{i-1}\mathbf{U}_i^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \tag{9}
$$

$$
i = 1, 2, \ldots, N\} \quad \text{revolute joint;}
$$

$$
{}^{i}\gamma_i = \mathbf{0}, \tag{10}
$$

$$
i = 1, 2, \ldots, N\} \quad \text{sliding joint;}
$$

$$
{}^{i}\gamma_j = {}^{i-1}\mathbf{U}_i^T \, {}^{i-1}\gamma_j, \quad j = 1, 2, \ldots, i-1, \quad i = 2, \ldots, N+1; \tag{11}
$$

$$
{}^{i}\boldsymbol{\beta}_i = {}^{i}\gamma_i \times {}^{i}\mathbf{p}_i^*, \tag{12}
$$

$$
i = 1, 2, \ldots, N\} \quad \text{revolute joint;}
$$

$$
{}^{i}\boldsymbol{\beta}_i = {}^{i-1}\mathbf{U}_i^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad i = 1, 2, \ldots, N\} \quad \text{sliding joint;} \tag{13}
$$

$$
{}^{i}\boldsymbol{\beta}_j = {}^{i-1}\mathbf{U}_i^T \, {}^{i-1}\boldsymbol{\beta}_j + ({}^{i}\boldsymbol{\gamma}_j \times {}^{i}\mathbf{p}_i^*), \quad j = 1, 2, \ldots, i-1, \quad i = 2, \ldots, N+1. \tag{14}
$$

For $i = N + 1$ in Eqs. (11) and (14), the coefficients as determined are just the elements of the Jacobian matrix, ${}^{N+1}\mathbf{J}_{N+1} = {}^{E}\mathbf{J}_E$. This method is rather inefficient because for $i = N + 1$, $N + 1$ Jacobians are computed. First, the Jacobian is determined for a manipulator consisting of the first link only, then for a manipulator consisting of the first two links, then three links, and so forth up to N links plus the end-effector. The recursions in all cases are from the base to the end-effector. This is a rather inefficient method because all that is needed is the Jacobian for the entire manipulator. It may be very useful, however, in the study of dynamics, where it is necessary to compute the Jacobian for each set of links in the manipulator (Vukobratović and Potkonjak 1979).

2.2. Methods II, III, and IV

The next three methods use the same basic concepts but result in different forms for the Jacobian matrix. They are based on earlier methods by Pieper (1968) and Whitney (1972). With close attention to computational detail, they provide relatively efficient methods for computing the Jacobian.

The angular velocity for the end-effector may be written as a function of the joint rates (for rotational joints):

$$
\boldsymbol{\omega} = \dot{\mathbf{q}}_1 \mathbf{z}_0 + \dot{\mathbf{q}}_2 \mathbf{z}_1 + \cdots + \dot{\mathbf{q}}_N \mathbf{z}_{N-1}. \tag{15}
$$

The translational velocity for the end-effector may be determined by noting that a particular component is just the appropriate cross-product of the individual joint rate vector and position vector from the joint axis to the velocity reference point ($-\mathbf{t}_i$). That is,

$$
\mathbf{v} = -(\dot{\mathbf{q}}_1 \mathbf{z}_0) \times \mathbf{t}_0 - (\dot{\mathbf{q}}_2 \mathbf{z}_1) \times \mathbf{t}_1 - \cdots - (\dot{\mathbf{q}}_N \mathbf{z}_{N-1}) \times \mathbf{t}_{N-1}. \tag{16}
$$

Comparing Eqs. (2) and (3) with Eqs. (15) and (16), we find that the coefficients of the Jacobian matrix, for the case of a rotational joint, may be written in terms of the joint axis vector and relative position vector:

$$
\left.\begin{aligned} \boldsymbol{\gamma}_i &= \mathbf{z}_{i-1} \\ \boldsymbol{\beta}_i &= -(\mathbf{z}_{i-1} \times \mathbf{t}_{i-1}) \end{aligned}\right\} \quad \text{rotational joint.} \tag{17}
$$

For the case of a sliding joint, similar expressions may be derived:

$$
\left.\begin{aligned} \boldsymbol{\gamma}_i &= \mathbf{0}, \\ \boldsymbol{\beta}_i &= \mathbf{z}_{i-1} \end{aligned}\right\} \quad \text{sliding joint.} \tag{18}
$$

The three methods presented in this section differ in the coordinate system used to express the components of the Jacobian and also in the velocity reference point chosen. In Waldron's work (1982), the components of the Jacobian are determined with respect to the base coordinate system 0, and the velocity reference point is chosen to be the origin of the base coordinate system. ${}^{0}\mathbf{J}_0$ is thus computed. Since $\mathbf{p}_i$ is defined as the position vector from the origin of the base coordinate system to that of link i (see Fig. 2), then the following recursive equations may be used:

$$
{}^{0}\mathbf{U}_0 = \mathbf{I}; \tag{19}
$$

$$
{}^{0}\mathbf{U}_i = {}^{0}\mathbf{U}_{i-1} \, {}^{i-1}\mathbf{U}_i, \qquad i = 1, 2, \ldots, N-1; \tag{20}
$$

$$
{}^{0}\boldsymbol{\gamma}_i = {}^{0}\mathbf{U}_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad i = 1, 2, \ldots, N\} \quad \text{revolute joint;} \tag{21}
$$

$$
{}^{0}\boldsymbol{\gamma}_i = \mathbf{0}, \qquad i = 1, 2, \ldots, N\} \quad \text{sliding joint;} \tag{22}
$$

$$
{}^{0}\mathbf{p}_0 = \mathbf{0}; \tag{23}
$$

$$^0\mathbf{p}_i = {}^0\mathbf{p}_{i-1} + {}^0\mathbf{U}_i\, {}^i\mathbf{p}_i^*, \quad i = 1, 2, \ldots, N-1; \tag{24}$$

$$^0\boldsymbol{\beta}_i = {}^0\boldsymbol{\gamma}_i \times (-{}^0\mathbf{p}_{i-1}), \quad i = 1, 2, \ldots, N\} \text{ revolute joint;} \tag{25}$$

$$^0\boldsymbol{\beta}_i = {}^0\mathbf{U}_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad i = 1, 2, \ldots, N\} \text{ sliding joint.} \tag{26}$$

Note in the above equations that the components of $\mathbf{z}_{i-1}$ in coordinate system $i-1$ are $[0 \quad 0 \quad 1]^T$. Also, $\mathbf{I}$ is the identity matrix. In Eqs. (20) and (24), the recursions for computing the orientation matrix and position vector are directed from the base out to the end-effector.

In Olson and Ribble's work (Ribble 1982), the components of the Jacobian are determined with respect to the base coordinate system, and the velocity reference point is chosen to be the center of the end-effector E. ${}^0\mathbf{J}_E$ is thus computed. Since $\mathbf{r}_i$ is defined as the position vector from the origin of the end-effector coordinate system to that of link i, Eq. (25) may be replaced by

$$\left.\begin{aligned} ^0\mathbf{r}_{i-1} &= {}^0\mathbf{p}_{i-1} - {}^0\mathbf{p}_{N+1} && (27)\\ ^0\boldsymbol{\beta}_i &= {}^0\boldsymbol{\gamma}_i \times (-{}^0\mathbf{r}_{i-1}) && (28)\end{aligned}\right\} \quad i = 1, \ldots, N.$$

Also, the indices on Eqs. (20) and (24) must be changed to $N+1$. In Olson and Ribble's work the recursion for computing the orientation matrix is from the base out. The recursion for the position vector is also from the base out; however, Eq. (27) must be added to give the velocity reference point at the center of the end-effector. While this method is slower than that of Waldron (1982), an advantage is that the velocity reference point is a real point on the end-effector and so the values of ${}^0\boldsymbol{\beta}_i$ relate directly to the velocity of the end-effector.

In Renaud's work (1981), the components of the Jacobian and the velocity reference point are both associated with a link ℓ, which is approximately midway between the base and end-effector. ${}^\ell\mathbf{J}_\ell$ is thus computed. The recursive equations for computing γ and β are

$$^\ell\mathbf{U}_\ell = \mathbf{I}; \tag{29}$$

$$^\ell\mathbf{U}_i = {}^\ell\mathbf{U}_{i-1}\, {}^{i-1}\mathbf{U}_i, \quad i = \ell+1, \ell+2, \ldots, N-1; \tag{30}$$

$$^\ell\mathbf{U}_{i-1} = {}^\ell\mathbf{U}_i\, {}^{i-1}\mathbf{U}_i^T, \quad i = \ell, (\ell-1), \ldots, 1; \tag{31}$$

$$^\ell\boldsymbol{\gamma}_i = {}^\ell\mathbf{U}_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad i = 1, 2, \ldots, N\} \text{ revolute joint;} \tag{32}$$

$$^\ell\boldsymbol{\gamma}_i = \mathbf{0}, \quad i = 1, 2, \ldots, N\} \text{ sliding joint;} \tag{33}$$

$$^\ell\mathbf{t}_\ell = \mathbf{0}; \tag{34}$$

$$^\ell\mathbf{t}_i = {}^\ell\mathbf{t}_{i-1} + {}^\ell\mathbf{U}_i\, {}^i\mathbf{p}_i^*, \quad i = \ell+1, \ell+2, \ldots, N-1; \tag{35}$$

$$^\ell\mathbf{t}_{i-1} = {}^\ell\mathbf{t}_i - {}^\ell\mathbf{U}_i\, {}^i\mathbf{p}_i^*, \quad i = \ell, (\ell-1), \ldots, 1; \tag{36}$$

$$^\ell\boldsymbol{\beta}_i = {}^\ell\boldsymbol{\gamma}_i \times (-{}^\ell\mathbf{t}_{i-1}), \quad i = 1, 2, \ldots, N\} \text{ revolute joint;} \tag{37}$$

$$^\ell\boldsymbol{\beta}_i = {}^\ell\mathbf{U}_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad i = 1, 2, \ldots, N\} \text{ sliding joint.} \tag{38}$$

The position vector $\mathbf{t}_i$ in the previous equations is from the origin of coordinate system $m = \ell$ to the origin of coordinate system i. Also note that the recursions are from link ℓ in toward the base and out toward the end-effector.

The main advantage of this method is that the manipulator is broken into two manipulators of approximate length $N/2$, and the Jacobian for a manipulator of length $N/2$ is calculated twice. The matrices at the beginning of the Jacobian calculation are simple, and Renaud's method, starting twice, takes advantage of this.

2.3. Method V

The method described in this section is new, resulting from an effort to compute ${}^E\mathbf{J}_E$ in the most efficient

way possible. We see from Eqs. (9) through (14) that Method I is somewhat inefficient because the elements of N different Jacobians are computed. That is, the Jacobian for a manipulator consisting of the first link only is determined, then for a manipulator consisting of the first two links, then the first three links, and so on. Also, the recursions are from the base to the end-effector.

If the recursion is redirected to be from the end-effector to the base and if values of γ and β not needed are not computed, then the resulting equations are

$$^{N+1}\mathbf{U}_{N+1} = \mathbf{I}; \tag{39}$$

$$^{N+1}\mathbf{U}_{i-1} = {}^{N+1}\mathbf{U}_i \, {}^{i-1}\mathbf{U}_i^T, \quad i = (N+1), \ldots, 2, 1; \tag{40}$$

$$^{N+1}\boldsymbol{\gamma}_i = {}^{N+1}\mathbf{U}_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad i = 1, 2, \ldots, N\} \text{ revolute joint;} \tag{41}$$

$$^{N+1}\boldsymbol{\gamma}_i = \mathbf{0}, \quad i = 1, 2, \ldots, N\} \text{ sliding joint;} \tag{42}$$

$$^{N+1}\mathbf{r}_{N+1} = \mathbf{0}; \tag{43}$$

$$^{N+1}\mathbf{r}_{i-1} = {}^{N+1}\mathbf{r}_i - {}^{N+1}\mathbf{U}_i \, {}^i\mathbf{p}_i^*, \quad i = (N+1), \ldots, 2, 1; \tag{44}$$

$$^{N+1}\boldsymbol{\beta}_i = {}^{N+1}\boldsymbol{\gamma}_i \times (-{}^{N+1}\mathbf{r}_{i-1}), \quad i = 1, 2, \ldots, N\} \text{ revolute joint;} \tag{45}$$

$$^{N+1}\boldsymbol{\beta}_i = {}^{N+1}\mathbf{U}_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad i = 1, 2, \ldots, N\} \text{ sliding joint.} \tag{46}$$

It should be noted that the resulting equations are similar to Waldron's (1982) except that the recursion is in the opposite direction. This was not immediately apparent during the development of the equations. Once the common notation had been applied to all the methods, we recognized the similarity. The velocity reference point is now a real point on the end-effector, and the components of the Jacobian are expressed in the end-effector coordinate system.

2.4. Method VI

The final method comes from Paul's work (1981) and also computes ${}^E\mathbf{J}_E$. Paul, however, uses the 4×4 homogeneous transformation matrix as the basis for the computations. The appropriate equations are

$$^{N+1}\mathbf{T}_{N+1} = \mathbf{I}; \tag{47}$$

$$^{i-1}\mathbf{T}_{N+1} = {}^{i-1}\mathbf{T}_i \, {}^i\mathbf{T}_{N+1}, \quad i = (N+1), \ldots, 2, 1; \tag{48}$$

$$^{N+1}\boldsymbol{\gamma}_i = {}^{i-1}\mathbf{U}_{N+1}^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad i = 1, 2, \ldots, N\} \text{ revolute joint;} \tag{49}$$

$$^{N+1}\boldsymbol{\gamma}_i = \mathbf{0}, \quad i = 1, 2, \ldots, N\} \text{ sliding joint;} \tag{50}$$

$$^{N+1}\boldsymbol{\beta}_i^j = -[{}^{i-1}\mathbf{U}_{N+1}^j \times {}^{i-1}r_{i-1}]^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \left.\begin{matrix} j = 1, 2, 3, \\ i = 1, 2, \ldots, N \end{matrix}\right\} \text{ revolute joint;} \tag{51}$$

$$^{N+1}\boldsymbol{\beta}_i = {}^{i-1}\mathbf{U}_{N+1}^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad i = 1, 2, \ldots, N\} \text{ sliding joint.} \tag{52}$$

Note that ${}^{i-1}\mathbf{U}_{N+1}^j$ is just column j of the 3×3 orientation part of ${}^{i-1}\mathbf{T}_{N+1}$. Also note that ${}^{i-1}\mathbf{r}_{i-1}$ is just the first three elements of the fourth column of ${}^{i-1}\mathbf{T}_{N+1}$. Furthermore, ${}^{N+1}\beta_i^j$ is just the jth component of the vector ${}^{N+1}\boldsymbol{\beta}_i$.

It may be noted that the results for Methods V and VI are quite similar except that in Method VI, the 3×3 orientation matrix and 3×1 position vector of Method V are grouped into the 4×4 homogeneous transformation matrix form. If the bottom row of the homogeneous transformation matrix is understood to be composed of three 0's and a 1, this seems to make little if any difference in the overall computational complexity. However, another difference is that the ordering of matrix multiplication in Eq. (48) is reversed from that of Eq. (40), and this does result in

Table 1. Number of Computations for Each Method Maintaining the Natural Jacobian Form (N = Number of degrees of freedom)

	Method	J	Multiplication	Addition/ Subtraction	Sines/ Cosines
I	Vukobratović/Potkonjak	${}^{E}\mathbf{J}_{E}$	$10N^2 - 15N + 9$	$N^2 + 7N - 2$	$2N$
II	Waldron	${}^{0}\mathbf{J}_{0}$	$30N - 55$	$15N - 38$	$2N - 2$
III	Olson/Ribble	${}^{0}\mathbf{J}_{E}$	$30N - 11$	$18N - 20$	$2N$
IV	Renaud	${}^{\ell}\mathbf{J}_{\ell}$	$30N - 87$	$15N - 66$	$2N - 2$
V	Orin/Schrader	${}^{E}\mathbf{J}_{E}$	$30N - 25$	$15N - 25$	$2N$
VI	Paul	${}^{E}\mathbf{J}_{E}$	$30N - 18$	$14N - 15$	$2N$

increased complexity in Eq. (51) as opposed to that of Eq. (45). It appears then that some computational savings may be made in Paul's method if the order of matrix multiplication in Eq. (48) is changed.

3. Comparison of the Methods

We now assess the numerical efficiency of computing the Jacobian according to each of the six methods just described. In particular, we enumerate the total number of additions/subtractions, multiplications, and trigonometric functions performed in the process of evaluating the Jacobian. The results, given as a function of the total number of degrees of freedom N, are shown in Table 1.

The results in Table 1 were generated through the use of a program that considered the components of matrices and vectors to be composed entirely of four symbols. The symbol x indicated a nonzero value other than ± 1; + indicated a value of $+1$, $-$ indicated a value of -1, and 0 indicated a value of zero. The program produced a symbolic Jacobian while counting the number of multiplications, additions/subtractions, and trigonometric functions used to produce the Jacobian. Additions, subtractions, and multiplications that included zero were not counted. Multiplications that included ± 1 were totaled separately but have been added into the multiplication formulas of Table 1. It has also been assumed that the twist angle of each link is $0°$ or $\pm 90°$ since most manipulators have this type of kinematic configuration.

From Table 1, we see that the best method is that of Renaud (1981). It should be understood, however, that to use this form of the Jacobian for resolved rate control it is necessary to associate the angular and translational velocity of the end-effector with coordinate system ℓ, where $\ell \simeq (N + 1)/2$.

Table 2. Execution Time for Each Method on the PDP-11/70

	Method	J	Execution Time (ms)
I	Vukobratović/Potkonjak	${}^{E}\mathbf{J}_{E}$	23.0
II	Waldron	${}^{0}\mathbf{J}_{0}$	12.8
III	Olson/Ribble	${}^{0}\mathbf{J}_{E}$	16.2
IV	Renaud	${}^{\ell}\mathbf{J}_{\ell}$	12.9
V	Orin/Schrader	${}^{E}\mathbf{J}_{E}$	15.5
VI	Paul	${}^{E}\mathbf{J}_{E}$	16.9

Methods I, V, and VI all result in the same form for the Jacobian, ${}^{E}\mathbf{J}_{E}$. Of these, the new method proposed in this paper requires the least amount of computation.

All of the methods discussed in this paper have been used to simulate a seven-degree-of-freedom manipulator that has a somewhat general kinematic configuration (nonzero joint offsets and link lengths). A PDP-11/70 minicomputer with floating-point hardware and the PASCAL language were used, and the results are given in Table 2. The results show a close correlation with those of Table 1.

In Table 2, any changes in ordering from that anticipated by reviewing Table 1 probably result from differences in the amount of overhead involved in indexed addressing. Also, in evaluating the execution times given in Table 2, it should be understood that

Table 3. Number of Computations for Each Method Including Conversion to ${}^{E}\mathbf{J}_{E}$ (N = Number of degrees of freedom)

Method	Multiplication	Addition/ Subtraction	Sines/ Cosines
I Vukobratović/Potkonjak	$10N^2 - 15N + 9$	$N^2 + 7N - 2$	$2N$
II Waldron	$54N - 31$	$33N - 37$	$2N$
III Olson/Ribble	$48N - 23$	$30N - 32$	$2N$
IV Renaud	$54N - 81$	$30N - 79$	$2N$
V Orin/Schrader	$30N - 25$	$15N - 25$	$2N$
VI Paul	$30N - 18$	$14N - 15$	$2N$

Table 4. Execution Time for Each Method on the PDP-11/70 (All Forms Further Converted to ${}^{E}\mathbf{J}_{E}$).

Method	Execution Time (ms)
I Vukobratović/Potkonjak	23.0
II Waldron	20.1
III Olson/Ribble	19.1
IV Renaud	20.1
V Orin/Schrader	15.5
VI Paul	16.9

the absolute magnitudes have little significance. That is, little effort was made to reduce these times. Instead, much effort was made to ensure "fairness" so that relative magnitudes are significant.

Perhaps a more appropriate way to compare the various methods is to transform the Jacobian produced by each method to a common form. In this case, ${}^{E}\mathbf{J}_{E}$ will be used as the final form. To move the velocity reference point from the origin of coordinate system m to the end-effector, the following transformation is appropriate:

$$ {}^{\ell}\mathbf{J}_{E} = \left[\begin{array}{ccc|c} & \mathbf{I} & & \mathbf{0} \\ \hline 0 & -{}^{\ell}r_{m}^{z} & {}^{\ell}r_{m}^{y} & \\ {}^{\ell}r_{m}^{z} & 0 & -{}^{\ell}r_{m}^{x} & \mathbf{I} \\ -{}^{\ell}r_{m}^{y} & {}^{\ell}r_{m}^{x} & 0 & \end{array}\right] {}^{\ell}\mathbf{J}_{m}. \quad (53) $$

To convert the components of ${}^{\ell}\gamma_{iE}$ and ${}^{\ell}\beta_{iE}$ from the ℓth coordinate system to the end-effector coordinate system, the following transformation may be used:

$$ {}^{E}\mathbf{J}_{E} = \left[\begin{array}{c|c} {}^{\ell}\mathbf{U}_{E}^{T} & \mathbf{0} \\ \hline \mathbf{0} & {}^{\ell}\mathbf{U}_{E}^{T} \end{array}\right] {}^{\ell}\mathbf{J}_{E}. \quad (54) $$

The total transformation is just a projective transformation of screw coordinates (Sugimoto and Matsumoto 1983) and is given as follows:

$$ {}^{E}\mathbf{J}_{E} = \left[\begin{array}{c|c} {}^{\ell}\mathbf{U}_{E}^{T} & \mathbf{0} \\ \hline {}^{\ell}\mathbf{U}_{E}^{T} * \begin{bmatrix} 0 & -{}^{\ell}r_{m}^{z} & {}^{\ell}r_{m}^{y} \\ {}^{\ell}r_{m}^{z} & 0 & -{}^{\ell}r_{m}^{x} \\ -{}^{\ell}r_{m}^{y} & {}^{\ell}r_{m}^{x} & 0 \end{bmatrix} & {}^{\ell}\mathbf{U}_{E}^{T} \end{array}\right] {}^{\ell}\mathbf{J}_{m}. \quad (55) $$

The results for each method after conversion to ${}^{E}\mathbf{J}_{E}$ are given in Tables 3 and 4. From these tables, we see that the last two methods are the most efficient, with the one proposed in this paper being the best.

It should also be understood that the results presented in these tables are not completely definitive. Each method has its own natural output form for the Jacobian, and to depart from this to transform it to a common form biases the results toward those that naturally produce this common form. However, whether the natural or common form of the Jacobian is used, Methods V and VI are especially computationally efficient.

Table 5. Number of Matrix-Vector Computations for Each Method

	Method	$\mathbf{J}$	Cross-Product	Matrix-Vector Product	Matrix Product (3×3)	Vector Addition/ Subtraction	Matrix Product (4×4)
I	Vukobratović/Potkonjak	${}^{E}\mathbf{J}_{E}$	$1/2(N^2+3N)$	N^2+N		$1/2(N^2+N)$	
II	Waldron	${}^{0}\mathbf{J}_{0}$	$N-1$	$N-1$	$N-2$	$N-2$	
III	Olson/Ribble	${}^{0}\mathbf{J}_{E}$	N	$N+1$	N	$2N$	
IV	Renaud	${}^{i}\mathbf{J}_{i}$	$N-1$	$N-2$	$N-3$	$N-3$	
V	Orin/Schrader	${}^{E}\mathbf{J}_{E}$	N	N	N	N	
VI	Paul	${}^{E}\mathbf{J}_{E}$	$3N$				N

One final comparison among the methods is given in Table 5. In this case, the basic arithmetic operations are for vectors and matrices. This is appropriate because processors of the future should be equipped with such instructions. The results are similar to those obtained previously, in that Renaud's method is the best. Of those that result in ${}^{E}\mathbf{J}_{E}$, the method proposed in this paper is the best. (A 4×4 matrix product of homogeneous transforms is basically equivalent to a 3×3 matrix product along with a matrix-vector product and vector add.)

4. Summary

Six different methods for computing the Jacobian have been presented and compared for computational efficiency. The results indicate that the new method presented in this paper is the most computationally efficient when the Jacobian is based on end-effector coordinates. A method proposed by Renaud (1981) gives the best results when an arbitrary coordinate system is used.

To use a Jacobian of any particular form for resolved rate control or force control, the end-effector rates or forces must be associated with the same coordinate system. The total amount of computation required often depends on the specifics of the application for which the Jacobian is used. At a minimum, though, the results of this paper should give a general indication of the total computation required.

The results presented here assume the use of a serial processor. A pipelined linear array of processors may also be used but requires a new formulation of the problem. This is the subject of present investigations at The Ohio State University (Schrader 1983), and the present paper provides the foundation for this work. Advances in both algorithms and architectures promise to allow a control based on the Jacobian to be more viable for real-time implementation in the future.

Acknowledgment

This work was supported by the National Science Foundation under Grant No. ECS-8312677 to The Ohio State University.

References

Denavit, J., and Hartenberg, R. B. 1955. A kinematic notation for lower-pair mechanisms based on matrices. *ASME J. Appl. Mech.*, vol. 23, pp. 215–221.

Dodd, G. G., and Rossol, L., eds. 1979. *Computer vision and sensor-based robots.* New York: Plenum Press.

Duffy, J. 1980. *Analysis of mechanisms and robot manipulators.* London: Edward Arnold.

Klein, C. A., and Briggs, R. L. 1980. Use of active compliance in the control of legged vehicles. *IEEE Trans. Sys. Man Cyber.* SMC-10(7):393–400.

Klein, C. A., and Huang, C. H. 1983. Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Trans. Sys. Man Cyber.* SMC-13(3):245–250.

Liegeois, A. 1977. Automatic supervisory control of the configuration and behaviour of multibody mechanisms. *IEEE Trans. Sys. Man Cyber.* SMC-7(12):868–871.

Paul, R. P. 1981. *Robot manipulators: Mathematics, programming, and control.* Cambridge: MIT Press.

Pieper, D. L. 1968. The kinematics of manipulators under computer control. Ph.D. thesis, Stanford University, Department of Computer Science.

Renaud, M. 1981. (Oct., Tokyo). Geometric and kinematic models of a robot manipulator: Calculation of the Jacobian matrix and its inverse. *Proc. 11th Int. Symp. Industr. Robots.*

Ribble, E. A. 1982. Synthesis of human skeletal motion and the design of a special-purpose processor for real-time animation of human and animal figure motion. M.S. thesis, Ohio State University, Department of Electrical Engineering, Columbus, Ohio.

Schrader, W. W. 1983. Efficient Jacobian computation for robot manipulators on serial and pipelined processors. M.S. thesis, Ohio State University, Department of Electrical Engineering, Columbus, Ohio.

Sugimoto, K., and Matsumoto, Y. 1983 (Bretton Woods, N.H.). Kinematic analysis of manipulators by means of the projective transformation of screw coordinates. *Proc. 1st Int. Symp. Robotics Res.* Cambridge: M.I.T. Artificial Intelligence Laboratory.

Trevelyan, Kovesi, and Ong 1983; Yoshikawa 1983). (Bretton Woods, N.H.). Motion control for a sheep shearing robot. *Proc. 1st Int. Symp. Robotics Res.* Cambridge: M.I.T. Artificial Intelligence Laboratory.

Vukobratović, M., and Potkonjak, V. 1979. Contribution of the forming of computer methods for automatic modelling of spatial mechanisms motions. *Mechanism and Machine Theory,* vol. 14, pp. 179–200.

Waldron, K. J. 1982. Geometrically based manipulator rate control algorithms. *Mechanism and Machine Theory* 17(6):379–385.

Whitney, D. E. 1969. Resolved motion rate control of manipulators and human prostheses. *IEEE Trans. Man-Mach. Sys.* MMS-10(2):47–53.

Whitney, D. E. 1972. The mathematics of coordinated control of prosthetic arms and manipulators. *J. Dyn. Sys. Measurement, Contr.,* vol. 94, pp. 303–309.

Wu, C. H., and Paul, R. P. 1982. Resolved motion force control of robot manipulators, *IEEE Trans. Sys. Man Cyber.* SMC-12(3):266–275.

Yoshikawa, T. 1983 (Bretton Woods, N.H.). Analysis and control of robot manipulators with redundancy. *Proc. 1st Int. Symp. Robotics Res.* Cambridge: M.I.T. Artificial Intelligence Laboratory.