Manipulator Inverse Kinematic Solutions Based on Vector Formulations and Damped Least-Squares Methods

CHARLES W. WAMPLER, II

Abstract—**Inverse kinematic solutions are used in manipulator controllers to determine corrective joint motions for errors in end-effector position and orientation. Previous formulations of these solutions, based on the Jacobian matrix, are inefficient and fail near kinematic singularities. Vector formulations of inverse kinematic problems are developed that lead to efficient computer algorithms. To overcome the difficulties encountered near kinematic singularities, the exact inverse problem is reformulated as a damped least-squares problem, which balances the error in the solution against the size of the solution. This yields useful results for all manipulator configurations.**

INTRODUCTION

IN GENERAL terms, an inverse kinematic problem for a serial-link manipulator is the problem of finding joint motions that will produce a given motion of the end-effector. The solution can be used in the control of the manipulator to determine joint motions that will correct measured errors in the position and orientation of the end-effector. For example, resolved-rate control [1] is based on an inverse velocity/angular velocity solution, and resolved-acceleration control [2] is based on an inverse acceleration/angular acceleration solution. An inverse kinematic solution is also useful when a human operator specifies the velocity of a telemanipulator's end-effector via a joystick or other device. Closed-form solutions to the problem of finding joint positions corresponding to a given end-effector position and orientation are known for only a few simple nonredundant manipulator geometries, but for redundant arms and nonredundant arms with complex geometries, the inverse position/orientation problem can be solved iteratively using inverse velocity/angular velocity solutions [3], [4]. To date, inverse kinematic solutions have not found widespread use due to three primary reasons: 1) nonredundant arms with closed-form inverse position/orientation solutions are sufficient for many applications; 2) existing methods for computing inverse kinematic solutions based on the Jacobian matrix are not efficient; and 3) these methods suffer from numerical difficulties near configurations where the Jacobian matrix is ill-conditioned (i.e., at kinematical singularities). The methods presented in this paper address the latter two problems, thereby opening the door to the practical investigation of the advantages offered by redundant arms and nonredundant arms with unconventional geometries.

The following conventions in mathematical notation are used in this paper: the position vector of point P with respect to point Q is denoted by $\boldsymbol{p}^{P/Q}$; vectors are in boldface and scalars are in italics; matrices are underlined and the elements are identified by typeface as either vectors or scalars; the superscript "$'$" is the transposition operator; and the delimiter "$]_\circ$" is an operator which converts a vector to a 3×1 column matrix, where "$\circ$" refers to a set of unit vectors, and the conversion is as follows:

$$\boldsymbol{v}]_\circ = [\boldsymbol{x}_\circ \cdot \boldsymbol{v} \quad \boldsymbol{y}_\circ \cdot \boldsymbol{v} \quad \boldsymbol{z}_\circ \cdot \boldsymbol{v}]'. \tag{1}$$

This paper considers manipulators consisting of a series of rigid links connected by joints to form a single chain with no closed loops. Only two types of single-degree-of-freedom joints will be discussed: rotational, for which all relative motion between two links is restricted to rotation about a line fixed in both links, and translational, for which relative motion is restricted to translation along a line fixed in both links. In accordance with the notation developed by Denavit and Hartenburg [5], the links are numbered sequentially from 0 to n, starting at the base of the manipulator, and the geometry of each link is described as illustrated in Fig. 1. A right-handed set of orthogonal unit vectors, $\boldsymbol{x}_i$, $\boldsymbol{y}_i$, $\boldsymbol{z}_i$, is fixed in each link i ($i = 0,\cdots,n$) with $\boldsymbol{z}_i$ parallel to the line L_i fixed in both links i and $i-1$; that is, parallel to joint axis i. The orientation of $\boldsymbol{z}_0$ is arbitrary. In addition, the common normal N_i between joint axes i and $i+1$ has a length a_i and its intersections with the joint axes labeled O_i and Q_i, respectively. Unit vector $\boldsymbol{x}_i$ is parallel to the common normal, and unit vector $\boldsymbol{y}_i$ completes the orthogonal set. The kinematics of the link are completely characterized by its length a_i, offset b_i, rotation angle θ_i, and twist angle β_i. The motion of link i with respect to link $i-1$ is measured by the generalized coordinate q_i, which is either the rotation θ_i about $\boldsymbol{z}_i$ or the translation b_i along $\boldsymbol{z}_i$ depending on the nature of joint i. Finally, to aid in the description of the motion of the

[1]Manuscript received March 8, 1985; revised August 28, 1985. This work was supported under a Veterans Administration Rehabilitation R & D Merit Review grant.

The author was formerly with the Rehabilitation Research and Development Center of the Veterans Administration Medical Center, Palo Alto, CA 94304, USA. He is now a member of the Mathematics Department, General Motors Research Laboratories, Warren, MI 48090, USA.

IEEE Log Number 8406015.

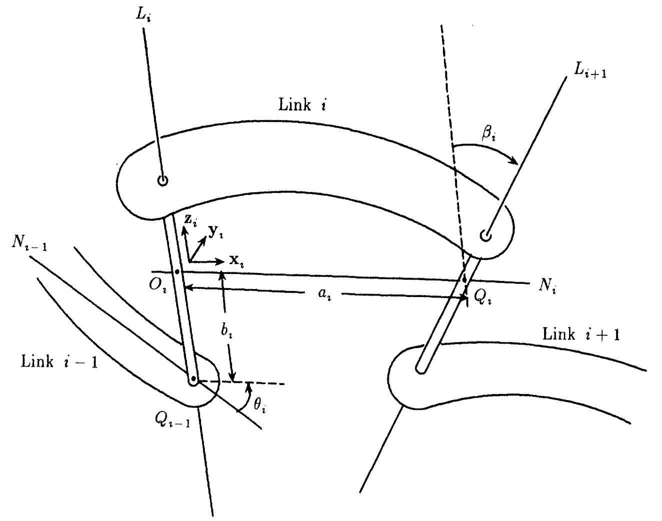

Fig. 1. Geometry of a manipulator link.

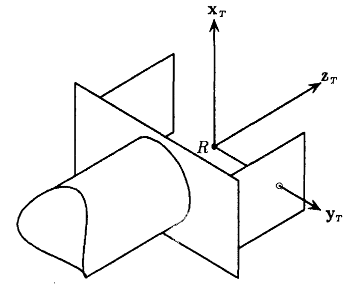

Fig. 2. Tool vectors for a two-fingered gripper.

end-effector, let there be fixed in link n a point R and the right-handed set of orthogonal unit vectors $\boldsymbol{x}_T$, $\boldsymbol{y}_T$, and $\boldsymbol{z}_T$. Point R is to be chosen as the tip of a tool (screw-driver, paint spray nozzle, etc.) or the center of gripping jaws, and the unit vectors are chosen in directions somehow relevant to the function of the tool (see Fig. 2). These *tool vectors* have been called the normal, orientation, and approach vectors, by Paul [6] and sweep, lift, and reach vectors by Whitney [7].

Jacobian Matrices and Partial Velocity Matrices

The relation between changes in joint coordinates and changes in the position and orientation of the end-effector can be written in terms of a Jacobian matrix. Suppose that the position and orientation of the end-effector are given by an $m \times 1$ column matrix $\underline{x} = [x_1 \cdots x_m]'$, whose elements are continuous differentiable functions of the joint coordinates $\underline{q} = [q_1 \cdots q_n]'$ and time t. Usually, the elements of $\underline{x}$ include three Cartesian position coordinates and three Euler angles, in which case $m = 6$. The Jacobian matrix, $\underline{J}_x$, for $\underline{x}$ with respect to $\underline{q}$ is defined as the $m \times n$ matrix whose ijth element is

$$J_{x,ij} \triangleq \frac{\partial x_i}{\partial q_j}. \tag{2}$$

By the chain rule for differentiation, the total derivative of $\underline{x}$ with respect to time is

$$\dot{\underline{x}} = \underline{J}_x \dot{\underline{q}} + \frac{\partial \underline{x}}{\partial t}. \tag{3}$$

Ordinarily, there is no explicit dependence on time, in which case the last term disappears. Solutions of (3) for $\dot{\underline{q}}$ give joint motions which will produce a given end-effector motion $\dot{\underline{x}}$. Two major difficulties are encountered in evaluating the elements of the Jacobian matrix. First, the use of (2) to obtain expressions for these elements is generally tedious and usually leads to an inefficient computer code. Second, for any set of three orientation parameters, such as Euler angles, there is a configuration of the end-effector at which at least one parameter is a discontinuous function, and therefore the associated elements of $\underline{J}_x$ are not well-defined. The discontinuity can be avoided through the use of a larger set of orientation parameters, such as four Euler parameters or nine elements of a direction cosine matrix [8]. However, this increases the size of the Jacobian matrix, thereby causing further inefficiency.

An alternative that avoids these difficulties is to use the relation between the joint speeds and the velocity and angular velocity of the end-effector. Kane [9] has shown that if a particle P (or rigid body B) is part of a system S that possesses n degrees of freedom in a reference frame N, and $q_1, \cdots, q_n$ form a set of generalized coordinates for S in N, then the velocity of P in N (the angular velocity of B in N) can be expressed as

$$\boldsymbol{v}^P = \sum_{i=1}^{n} \boldsymbol{v}_i^P \dot{q}_i + \boldsymbol{v}_t^P \tag{4}$$

$$\boldsymbol{\omega}^B = \sum_{i=1}^{n} \boldsymbol{\omega}_i^B \dot{q}_i + \boldsymbol{\omega}_t^B \tag{5}$$

where $\boldsymbol{v}_i^P$ $(i = 1, \cdots, n)$, $\boldsymbol{\omega}^B$ $(i = 1, \cdots, n)$, $\boldsymbol{v}_t^P$, and $\boldsymbol{\omega}_t^B$ are vector functions of $q_1, \cdots, q_n$ and t. Since the generalized coordinates are independent, (4) and (5) can be taken as the definitions of $\boldsymbol{v}_i^P$, called the *partial velocity with respect to* q_i *of* P *in* N, and $\boldsymbol{\omega}_i^B$, called the *partial angular velocity with respect to* q_i *of* B *in* N. These relations apply to the motion with respect to the base of the endpoint R and the tool reference frame T of a serial-link manipulator, so after defining the matrices

$$\hat{\underline{J}} \triangleq \begin{bmatrix} \boldsymbol{v}_1^R & \cdots & \boldsymbol{v}_n^R \\ \boldsymbol{\omega}_1^T & \cdots & \boldsymbol{\omega}_n^T \end{bmatrix} \tag{6}$$

$$\hat{\underline{\boldsymbol{v}}} \triangleq \begin{bmatrix} \boldsymbol{v}^R - \boldsymbol{v}_t^R \\ \boldsymbol{\omega}^T - \boldsymbol{\omega}_t^T \end{bmatrix} \tag{7}$$

one has the expression

$$\hat{\underline{J}} \dot{\underline{q}} = \hat{\underline{\boldsymbol{v}}}. \tag{8}$$

In general, none of the joint motions has an explicit dependence on time, so the vectors $\boldsymbol{v}_t^R$ and $\boldsymbol{\omega}_t^T$ are identically zero. The $2 \times n$ vector matrix $\hat{\underline{J}}$ is called the partial velocity matrix for the end-effector. Scalar forms of (8) have appeared previously in robotics literature. An equivalent $6 \times n$ scalar partial velocity matrix, $\hat{\underline{J}}_M \triangleq \hat{\underline{J}}]_M$, can be formed by resolving each of the elements of $\hat{\underline{J}}$ into components along unit vectors fixed in some reference frame M. The matrix $\hat{\underline{J}}_0$ has been called the base Jacobian matrix [8] and more commonly, simply the Jacobian matrix [2], [10], [11], but both are misnomers since for three-dimensional motion the partial angular velocity vectors are not the partial derivatives of any set of orientational parameters. However, as noted in [8], $\hat{\underline{J}}_0$ is a basic building block of the Jacobian matrix for any set of coordinates which describe the position and orientation of the end-effector with respect to the base, for if

$$\dot{\underline{x}} = \underline{\Omega}\hat{\underline{v}}]_0 \tag{9}$$

the Jacobian matrix for $\underline{x}$ with respect to $\underline{q}$ is

$$\underline{J}_x = \underline{\Omega}\hat{\underline{J}}_0. \tag{10}$$

Rates of change of orientation parameters are easily related to angular velocity (except at discontinuities of Euler angles), and this factored form of the Jacobian often reduces computation. Nevertheless, most computation problems involving the Jacobian can be reformulated directly in terms of the partial velocity matrix with a corresponding increase in efficiency. Further, the vector form of the partial velocity matrix allows each element to be expressed in a different reference frame, an approach which leads to more efficient computer algorithms.

Computing the Elements of the Partial Velocity Matrix

The first step in any computation involving the partial velocity matrix is to find its vector elements. To represent these numerically, each one can be converted to a 3×1 scalar matrix according to (1). As shown in the following theorems, the partial velocity $\boldsymbol{v}_i^R$ and the partial angular velocity $\boldsymbol{\omega}_i^T$ are most simply represented in the reference frame of link i.

Theorem 1: The partial angular velocity of link r in the base with respect to q_i is

$$\boldsymbol{\omega}_i^r = \begin{cases} \mathbf{0}, & \text{if } r < i \text{ or joint } i \text{ is translational} \\ \boldsymbol{z}_i, & \text{if } r \geqslant i \text{ and joint } i \text{ is rotational,} \end{cases}$$
$$i, r = 1, \cdots, n. \tag{11}$$

Corollary: The partial angular velocity of the end-effector in the base with respect to q_i is

$$\boldsymbol{\omega}_i^T = \begin{cases} \mathbf{0}, & \text{if joint } i \text{ is translational} \\ \boldsymbol{z}_i, & \text{if joint } i \text{ is rotational} \end{cases} \qquad i = 1, \cdots, n. \tag{12}$$

Proof: If joint i is rotational, link i performs a simple rotation about $\boldsymbol{z}_i$ in link $i-1$; but if the joint is translational, there is no relative rotation of the links. Consequently, the angular velocity of link i in link $i-1$ is

$$\boldsymbol{\omega}^{i/i-1} = \begin{cases} \mathbf{0}, & \text{if joint } i \text{ is translational} \\ \dot{q}_i \boldsymbol{z}_i, & \text{if joint } i \text{ is rotational,} \end{cases}$$
$$i = 1, \cdots, n. \tag{13}$$

By the addition theorem for angular velocity, the angular velocity of link r in the base can be expressed as

$$\boldsymbol{\omega}^r = \sum_{i=1}^{r} \boldsymbol{\omega}^{i/i-1}, \qquad r = 1, \cdots, n. \tag{14}$$

After the right-hand member of (13) has been substituted into (14), the theorem follows by comparison with (5).

Theorem 2: The partial velocity in the base with respect to q_i of a point P_r fixed in link r is

$$\boldsymbol{v}_i^{P_r} = \begin{cases} \mathbf{0}, & \text{if } r < i; \\ \boldsymbol{z}_i, & \text{if } r \geqslant i \text{ and joint } i \text{ is translational} \\ \boldsymbol{z}_i \times \boldsymbol{p}^{P_r/Q_{i-1}}, & \text{if } r \geqslant i \text{ and joint } i \text{ is rotational,} \end{cases}$$
$$i, r = 1, \cdots, n \tag{15}$$

Corollary. The partial velocity in the base with respect to q_i of the endpoint R is

$$\boldsymbol{v}_i^R = \begin{cases} \boldsymbol{z}_i, & \text{if joint } i \text{ is translational} \\ \boldsymbol{z}_i \times \boldsymbol{p}^{R/Q_{i-1}}, & \text{if joint } i \text{ is rotational,} \end{cases}$$
$$i = 1, \cdots, n. \tag{16}$$

Proof: Comparison of (4) with the chain-rule expansion of $\boldsymbol{v}^{P_r} = d\boldsymbol{p}^{P_r/Q_0}/dt$ gives

$$\boldsymbol{v}_i^{P_r} = \frac{\partial \boldsymbol{p}^{P_r/Q_0}}{\partial q_i} \tag{17}$$

Due to the open-loop chain structure of the manipulator, motion at joint i will move point P_r only if $r \geqslant i$, otherwise the partial velocity with respect to q_i is zero. When $r \geqslant i$ and joint i is translational, the position vector $\boldsymbol{p}^{P_r/Q_0}$ can be expressed as the sum

$$\boldsymbol{p}^{P_r/Q_0} = \boldsymbol{p}^{P_r/O_i} + q_i \boldsymbol{z}_i + \boldsymbol{p}^{Q_{i-1}/Q_0}. \tag{18}$$

Only the second vector on the right-hand side of this equation depends on q_i, so the partial derivative is simply $\boldsymbol{z}_i$. When $r \geqslant i$ and joint i is rotational, the position vector $\boldsymbol{p}^{P_r/Q_0}$ can be expressed as the sum

$$\boldsymbol{p}^{P_r/Q_0} = \boldsymbol{p}^{P_r/Q_{i-1}} + \boldsymbol{p}^{Q_{i-1}/Q_0}. \tag{19}$$

The second vector is fixed, but the first vector rotates by q_i about $\boldsymbol{z}_i$, so the partial derivative with respect to q_i is $\boldsymbol{z}_i \times \boldsymbol{p}^{P_r/Q_{i-1}}$, and the theorem is proved.

The vector elements of $\hat{\underline{J}}$ are all numerically represented after computing the position vectors $\boldsymbol{p}^{R/Q_{i-1}}]_i$ ($i = 1, \cdots, n$). This is because the scalar representation of $\boldsymbol{z}_i$ in link i coordinates is $\boldsymbol{z}_i]_i = [0 \;\; 0 \;\; 1]'$, so the partial angular velocity with respect to joint i has a trivial representation in reference frame i. The same is true for the partial velocity with respect to a translational joint. Further, the cross-product operation for the partial velocity with respect to a rotational joint i is trivial once the position vector $\boldsymbol{p}^{R/Q_{i-1}}]_i$ has been computed, because

$$\begin{aligned} \boldsymbol{v}_i^R]_i &= \lfloor \boldsymbol{z}_i \times \boldsymbol{p}^{R/Q_{i-1}} \rfloor_i \\ &= \left[-\boldsymbol{y}_i \cdot \boldsymbol{p}^{R/Q_{i-1}} \quad \boldsymbol{x}_i \cdot \boldsymbol{p}^{R/Q_{i-1}} \quad 0 \right]', \end{aligned}$$
$$i = 1, \cdots, n \tag{20}$$

which is merely a rearrangement of the elements of $\boldsymbol{p}^{R/Q_{i-1}}]_i$. Point R is fixed in the last link, so $\boldsymbol{p}^{R/Q_{n-1}}]_n$ is a constant if joint n is rotational, while if joint n is translational one needs only to add $[0\ 0\ q_n]'$ to the constant vector $\boldsymbol{p}^{R/O_n}]_n$. The remaining position vectors can be generated using the recursive formula

$$p^{R/Q_{i-1}}]_i = {}^i\underline{C}^{i+1}\boldsymbol{p}^{R/Q_i}]_{i+1} + \boldsymbol{p}^{Q_i/Q_{i-1}}]_i, \qquad i = n-1,\cdots,1. \quad (21)$$

where the direction cosine matrix relating the unit vectors $\boldsymbol{x}_{i+1}$, $\boldsymbol{y}_{i+1}$, $\boldsymbol{z}_{i+1}$ to $\boldsymbol{x}_i$, $\boldsymbol{y}_i$, $\boldsymbol{z}_i$ is

$${}^i\underline{C}^{i+1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta_i & -\sin\beta_i \\ 0 & \sin\beta_i & \cos\beta_i \end{bmatrix} \cdot \begin{bmatrix} \cos\theta_{i+1} & -\sin\theta_{i+1} & 0 \\ \sin\theta_{i+1} & \cos\theta_{i+1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (22)$$

and the position vector spanning link i is

$$\boldsymbol{p}^{Q_i/Q_{i-1}}]_i = \begin{cases} [a_i\ \ 0\ \ q_i]', & \text{if joint } i \text{ is translational} \\ [a_i\ \ 0\ \ b_i], & \text{if joint } i \text{ is rotational} \end{cases} \qquad i = 1,\cdots,n-1. \quad (23)$$

The algorithm indicated by Eq. (21) uses $(n-1)$ coordinate conversions and $2(n-1)$ scalar additions. In turn, each coordinate conversion requires 8 multiplications and 4 additions, unless the twist angle β_i is an integer multiple of 90°, in which case all the elements of the corresponding rotation matrix are either 0, 1, or -1. Contemporary industrial manipulators commonly have this property, and accordingly the computational cost of the algorithm can be cut almost in half. A by-product of the algorithm is $\boldsymbol{p}^{R/Q_0}$, the position of the endpoint with respect to point Q_0 fixed in the base.

Comparison to Other Methods

In a recent paper [11], Orin and Shrader have compared six methods of computing the partial velocity matrix, and these can be compared to the approach of this paper. A major difference between these methods and the present one is that they resolve all of the partial velocities and partial angular velocities into a common reference frame. This is not always necessary; for example, in force-control applications one often wishes to compute

$$\underline{\tau} = \hat{\underline{J}}' \cdot \begin{bmatrix} \boldsymbol{f} \\ \underline{t} \end{bmatrix} \quad (24)$$

where $\underline{\tau}$ is the reactions at the joints due to a torque $\underline{t}$ applied to the end-effector along with a force $\boldsymbol{f}$ applied at the endpoint R. This is most efficiently done by successively converting $\boldsymbol{f}$ and $\underline{t}$ to each link reference frame and taking advantage of the trivial elements in $\boldsymbol{v}_i^R]_i$ and $\boldsymbol{\omega}_i^T]_i$ when forming the dot-products. However, to compute inverse kinematic solutions, it is often convenient to resolve all the vectors to the same reference frame.

Unlike some of the methods outlined by Orin and Shrader, the choice of reference frame only weakly affects the efficiency of the computation about to be described. Ordinarily, the sensors from which a velocity command is derived will be fixed either in the base or the end-effector of the manipulator, so these reference frames are the most expedient choices. To convert the vectors to their representation in the base frame, one may first compute the n direction cosine matrices ${}^0\underline{C}^i$, $i = 1,\cdots,n$, using the recursive formula

$${}^0\underline{C}^i = {}^0\underline{C}^{i-1}\,{}^{i-1}\underline{C}^i, \qquad i = 2,\cdots,n \quad (25)$$

and to work in the end-effector frame, the recursion can be reversed to produce ${}^i\underline{C}^T$ $i = 1,\cdots,n$ as

$${}^i\underline{C}^T = {}^i\underline{C}^{i+1}\,{}^{i+1}\underline{C}^T, \qquad i = n-1,\cdots,1. \quad (26)$$

(An intermediate reference frame can be chosen by splitting the recursion into two chains moving from that frame towards each end of the manipulator.) A by-product of either of these procedures is that one additional stage of the recursion yields the direction cosine matrix for the end-effector with respect to the base, ${}^0\underline{C}^T$. The conversion to representations in the reference frame M is accomplished as

$$\boldsymbol{v}_i^R]_M = {}^M\underline{C}^i\boldsymbol{v}_i^R]_i, \quad \boldsymbol{\omega}_i^T]_M = {}^M\underline{C}^i\boldsymbol{\omega}_i^T]_i, \qquad i = 1,\cdots,n. \quad (27)$$

The computation is reduced by approximately 70 percent by taking advantage of the zero element on the right-hand side of (20) and noting also that $\boldsymbol{\omega}_i^T]_M$ is either the third column of ${}^M\underline{C}^i$ or zero. To compare directly with the results reported by Orin and Shrader, an operation count was performed for a manipulator with right-angle twists, and the end-effector reference frame T was chosen. Additions, subtractions and multiplications involving values that are identically zero were not counted, but multiplications by ± 1 were included. The total count for an implementation of (21), (26), (27) is $26n - 20$ multiplications and $13n - 10$ additions, which compares favorably with the $30n - 25$ multiplications and $15n - 25$ additions for the most effecient scheme of Orin and Shrader. If operations involving ± 1 are excluded, the number of multiplications is reduced to $22n - 16$. The greater efficiency of the present method over that of Orin and Shrader is attributable to the elimination of cross-product operations due to the triviality of (20). Additionally, some algorithms produce the partial velocities of an imaginary point of the end-effector that instantaneously coincides with the origin of another link. If so desired, the recursion of (21) can be rearranged to produce such a result with an insignificant effect on the total operation count. Thus the present method is comparable to the best of those in reference [11] for $n < 6$, and the coefficient of n is the smallest of any of these.

Inverse Velocity / Angular Velocity Solutions

An inverse velocity/angular velocity solution is a set of joint speeds $\dot{\boldsymbol{q}}$ which satisfies Eq. (8) for a given end-effector velocity/angular velocity $\hat{\underline{v}}$ at a given manipulator configuration $\underline{q}$. Depending on the number of joints and the rank of the partial velocity matrix $\hat{\underline{J}}$, this problem may have no solutions, one unique solution, or an infinite number of solutions. In the case that there are no solutions, one might wish to find the set of joint speeds which minimizes the error $\|\hat{\underline{J}}\dot{\underline{q}} - \hat{\underline{v}}\|$, and in the case that there are many solutions, one might wish to find the smallest solution, that is, the solution which minimizes $\|\dot{\underline{q}}\|$. The discussion which follows begins with the case where there is a unique solution and then examines a reformulation of the problem for redundant arms, configurations near kinematical singularities, minimization of side criterion, and solutions subject to joint speed limits.

Unique Solution

A free body in space has six degrees-of-freedom corresponding to the six independent components of velocity and angular velocity in $\hat{\underline{v}}$. While the number of independent components is reduced in certain circumstances, such as planar motion, the discussion here will be limited to the fully independent case; the reduction to the planar case is straightforward. The rank r of $\hat{\underline{J}}$ is the number of linearly independent columns in the matrix, and the matrix is said to be full-rank, if $r = \min(n, 6)$, or rank-deficient, if $r < \min(n, 6)$. When $n = 6$, the manipulator is said to be nonredundant since there is a unique solution to (8) at any configuration where $\hat{\underline{J}}$ is full-rank ($r = 6$). One method of obtaining this solution is to first convert (8) into six equivalent scalar equations by representing all vectors in the same reference frame using the algorithm presented above and then solving the resultant six equations in six unknowns using Gaussian elimination or a variant.

In some cases, the partial velocity matrix can be decomposed into a form that reduces the number of operations necessary to obtain a solution. One such case is a six-degree-of-freedom manipulator with a spherical wrist, i.e., the last three joints are rotational with axes intersecting at a single point. For such a manipulator, the task of solving a set of six simultaneous linear equations can be replaced by the task of solving two sets of three simultaneous equations: one set involving the velocity produced by the first three joints and the second set involving the angular velocity produced by the last three joints. Solutions of this type have been discussed by Featherstone [12], and Hollerbach and Sahar [13]. In any case, the use of a vector formulation of the problem with a careful consideration of the coordinate frames in which the actual scalar computations are performed will lead to the most efficient solution algorithm.

Redundant Arms

When the number of degrees of freedom of a manipulator is greater than six ($n > 6$) and the partial velocity matrix $\hat{\underline{J}}$ is full-rank ($r = 6$), there are an infinite number of sets of joint speeds which will produce any given end-effector velocity and angular velocity $\hat{\underline{v}}$. Accordingly, the motion of the arm may be improved by finding among these solutions a set of joint speeds that meets some additional criteria. Two criteria have been previously proposed in the literature: minimum-norms [1], [7], [8] and constrained gradient optimization [15], [16]. Both depend on the Moore-Penrose generalized inverse, which is described briefly in the following paragraph.

The singular value decomposition theorem states that for any $m \times n$ matrix $\underline{A}$, there exist orthogonal matrices $\underline{U}$ $(m \times m)$ and $\underline{V}$ $(n \times n)$ such that

$$\underline{A} = \underline{U}\underline{\Sigma}\underline{V}' \tag{28}$$

where the $m \times n$ matrix $\underline{\Sigma}$ has the block matrix form

$$\underline{\Sigma} = \begin{bmatrix} \underline{S} & \underline{0} \\ \underline{0} & \underline{0} \end{bmatrix}, \qquad \underline{S} = \operatorname{diag}(\sigma_1, \cdots, \sigma_r), \quad r = \operatorname{rank}(\underline{A}). \tag{29}$$

For convenience, the diagonal elements of the $r \times r$ matrix $\underline{S}$ can be ordered so that $\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_r > 0$. The numbers $\sigma_1, \cdots, \sigma_r$ along with $n - r$ additional zeros are called the singular values of $\underline{A}$ [14]. If the singular value decomposition of a partial velocity matrix is written in the form of (28), the columns of $\underline{U}$ divide between those directions in which the end-effector can move and those in which it cannot move, while the columns of $\underline{V}$ divide between sets of joint speeds which move the end-effector and those which do not. The singular values are "speed ratios" between joint speeds and end-effector motion. The Moore–Penrose generalized inverse of $\underline{A}$, also called the pseudoinverse, is given by

$$\underline{A}^{+} = \underline{V}\underline{\Sigma}^{+}\underline{U}', \qquad \underline{\Sigma}^{+} = \begin{bmatrix} \underline{S}^{-1} & \underline{0} \\ \underline{0} & \underline{0} \end{bmatrix}. \tag{30}$$

The pseudoinverse has many properties useful in problems involving least squares, minimum norms, and orthogonal projections. Although (30) is useful for expository purposes, there are many methods of computing the inverse without resorting to the singular value decomposition, provided the rank of the matrix is known. For full-rank matrices with $r = m \leqslant n$, the formula

$$\underline{A}^{+} = \underline{A}'(\underline{A}\underline{A}')^{-1} \tag{31}$$

is often used and has been employed previously in connection with inverse velocity/angular velocity solutions for manipulators [1], [7], [8], [15], [16]. It is easy to confirm that the pseudoinverse of a nonsingular square matrix is simply the inverse of the matrix.

The minimum-norm inverse velocity/angular velocity problem is stated as follows: among all of the solutions to $\hat{\underline{J}}\dot{\underline{q}} = \hat{\underline{v}}$, find the unique set of joint speeds $\dot{\underline{q}}_0$ which minimizes $\|\dot{\underline{q}}\| = (\dot{\underline{q}}'\dot{\underline{q}})^{1/2}$. For any reference frame k, the solution to this problem is

$$\dot{\underline{q}}_0 = \hat{\underline{J}}_k^{+}\hat{\underline{v}}\,]_k. \tag{32}$$

The pseudoinverse can be modified to minimize other quadratic functions of $\dot{\underline{q}}$. For example, to minimize power consumption and enhance safety, Whitney [1] suggested the use of the kinetic energy of the arm. An alternative that requires much less computation is the minimization of $\dot{\underline{q}}'\underline{D}\dot{\underline{q}}$, where $\underline{D}$ is a diagonal matrix that approximates the kinetic energy matrix or reflects the maximum allowed joint speeds.

Constrained gradient optimization was first applied to the control of manipulators by Liégeois [15] and further developed by Klein and Huang [16]. In this method, the desirability of a configuration of the arm is measured by a continuous, differentiable function $H(\underline{q})$ which is to be minimized subject to the required motion $\hat{\underline{v}}$. The function chosen by Liégeois was the sum of the squares of the deviations of the joints from some nominal location, but other possibilities include penalty functions which become large as the arm approaches obstacles or joint limits, such as the potential functions proposed by Khatib [8]. One method of minimizing H is to choose joint speeds as close as possible to its negative gradient $\dot{\underline{q}}_g = -\beta \partial H/\partial \underline{q}$; that is, one may choose the set of joint speeds $\dot{\underline{q}}_H$ that minimizes $\|\dot{\underline{q}} - \dot{\underline{q}}_g\|$ subject to $\hat{\underline{J}}\dot{\underline{q}} = \hat{\underline{v}}$. The solution to this constrained least-squares problem is

$$\dot{\underline{q}}_H = \hat{\underline{J}}_k^+ \hat{\underline{v}}]_k + \left(\underline{I} - \hat{\underline{J}}_k^+ \hat{\underline{J}}_k\right) \dot{\underline{q}}_g \tag{33}$$

where $\dot{\underline{q}}_H$ is the minimum-norm solution plus the projection of $\dot{\underline{q}}_g$ onto the null space of $\hat{\underline{J}}$ [17]. It can be shown that the second term always moves the joints in a manner which would decrease the optimization function, so any increase in H is due solely to the necessity of producing the required velocity $\hat{\underline{v}}$. When $\dot{\underline{q}}_g = \underline{0}$, the solution becomes the minimum-norm solution.

Rank-Deficient or Near-Rank-Deficient Configurations

Almost all manipulators have configurations where the number of independent columns in the partial velocity matrix decreases. The matrix is then rank-deficient and the corresponding configuration is said to be a rank-deficient configuration or a kinematical singularity. At such a configuration, the arm cannot produce an arbitrary velocity/angular velocity $\hat{\underline{v}}$, and there is at least one direction in which the end-effector cannot move: a common example is a fully extended elbow arm. It can be shown that every six degree-of-freedom manipulator with a spherical wrist has a rank-deficient partial velocity matrix when either the partial velocities of the wrist intersection point with respect to the first three joints lie in a plane, or the three wrist joint axes lie in a plane [4]. In the neighborhood of a rank-deficient configuration there is a singular value that is very small, which tends to give a very large joint speed solution when the inverse or pseudoinverse of the matrix is computed. Despite this prevalence of kinematical singularities and a recognition of their presence by many researchers e.g., [8], [16], no one has presented an inverse velocity/angular velocity solution algorithm that gives meaningful results at rank-deficient or nearly rank-deficient configurations.

Since an arbitrary velocity cannot be produced when the manipulator is at a singularity, one may wish to find sets of joint speeds that minimize the error $\|\hat{\underline{J}}\dot{\underline{q}} - \hat{\underline{v}}\|$ and select among them the minimum-norm set or the set nearest to a preferred solution $\dot{\underline{q}}_p$, as above. The solutions to these problems are exactly as given in (32), (33), but the formula given in (31) is no longer applicable. Klein and Huang [16] and Khatib [8] have suggested removing dependent rows to obtain a full-rank matrix, but this is an unacceptable approach because in some instances the direction of the resultant velocity may change 180° depending on which row is removed [4]. Algorithms for solving the rank-deficient problem exist [18], but the use of these for manipulator control is problematic as described below.

The principal difficulty in computing the pseudo-inverse is the determination of the rank of the matrix [19]. The most effective method is to find the number of nonzero singular values in the singular value decomposition of the matrix. For this purpose, one must establish a value of σ_r, or better, a value of σ_r/σ_1, below which a singular value is declared zero [14]. The value selected should depend on the precision of the computer and the error in the measurement of the joint coordinates used in computing $\hat{\underline{J}}$. The partial velocity matrix is a continuous function of the joint coordinates and so are its singular values, so near any singularity are configurations where at least one singular value is small. As the configuration changes and the smallest singular value crosses the chosen threshold, the effect on the joint speed solution may be considerable, for at stake is the component $\sigma_r^{-1}\underline{v}_r \underline{u}_r' \hat{\underline{v}}]_k$, where $\underline{v}_r$ and $\underline{u}_r$ signify the columns of $\underline{V}$ and $\underline{U}$ corresponding to the singular value in question. In contrast, the effect on the residual velocity error is only moderate, being $\underline{u}_r' \hat{\underline{v}}]_k$. The quality of the motion of the arm will be more acceptable if discontinuities and deadzones in the velocity commands are eliminated, and this can be accomplished through the use of a damped least-squares reformulation of the problem.

Damped Least Squares

One means of settling the dilemma of whether or not to include a component solution associated with a small singular value is to balance the cost of a large solution against the cost of a large residual error by minimizing the sum

$$\|\hat{\underline{J}}\dot{\underline{q}} - \hat{\underline{v}}\|^2 + \alpha^2 \|\dot{\underline{q}}\|^2. \tag{34}$$

This modification of the original problem is known as Levenberg–Marquardt stabilization or damped least-squares [18]. Since the sum in (34) can be written as

$$\left\| \begin{bmatrix} \hat{\underline{J}} \\ \alpha \underline{I} \end{bmatrix} \dot{\underline{q}} - \begin{bmatrix} \hat{\underline{v}} \\ \underline{0} \end{bmatrix} \right\|^2 \tag{35}$$

the unique minimizer $\dot{\underline{q}}_\alpha$ is given by

$$\dot{\underline{q}}_\alpha = \begin{bmatrix} \hat{\underline{J}} \\ \alpha \underline{I} \end{bmatrix}^+ \begin{bmatrix} \hat{\underline{v}} \\ \underline{0} \end{bmatrix} \tag{36}$$

$$= \left(\hat{\underline{J}}' \cdot \hat{\underline{J}} + \alpha^2 \underline{I}\right)^{-1} \hat{\underline{J}}' \cdot \hat{\underline{v}} \tag{37}$$

where the second equality depends on the fact that the composite matrix is full-rank. The solution is written in a vector form, which emphasizes that the individual dot-products can be computed in the most convenient reference frame. A slight variation of the damped least-squares formulation that incorporates gradient minimization and weighted norms is the minimization of

$$\|\underline{\hat{J}}\underline{\dot{q}} - \underline{\hat{v}}\|^2 + \alpha_1^2\|\underline{\dot{q}} - \underline{\dot{q}}_g\|^2 + \alpha_2^2\underline{\dot{q}}'\underline{A}\underline{\dot{q}} \tag{38}$$

where $\underline{A}$ is a positive definite symmetric matrix. This gives the set of joint speeds

$$\underline{\dot{q}}_{\alpha 1\alpha 2} = \left(\underline{\hat{J}}' \cdot \underline{\hat{J}} + \alpha_1^2\underline{I} + \alpha_2^2\underline{A}\right)^{-1}\left(\underline{\hat{J}} \cdot \underline{\hat{v}} + \alpha_1\underline{\dot{q}}_g\right). \tag{39}$$

The effect that damped least-squares has on the solution can be seen using the singular value decomposition of $\underline{\hat{J}}_k$, with which $\underline{\dot{q}}_\alpha$ can be written as

$$\underline{\dot{q}}_\alpha = \underline{V}\underline{\Sigma}_\alpha\underline{U}'\underline{\hat{v}}]_k \tag{40}$$

where the $n \times 6$ matrix $\underline{\Sigma}_\alpha$ is

$$\underline{\Sigma}_\alpha = \begin{bmatrix} \underline{S}_\alpha \\ \underline{0} \end{bmatrix}, \qquad \underline{S}_\alpha = \operatorname{diag}\left(\frac{\sigma_1}{\sigma_1^2 + \alpha^2}, \cdots, \frac{\sigma_6}{\sigma_6^2 + \alpha^2}\right). \tag{41}$$

If α is much less than the smallest nonzero singular value of $\underline{\hat{J}}$, then $\underline{\dot{q}}_\alpha$ is approximately the minimum-norm, least-squares solution $\underline{\dot{q}}_0$. As a singular value approaches zero, the associated component of $\underline{S}_\alpha$ reaches a maximum when $\sigma = \alpha$ and then decreases rapidly to zero. The size of the solution, $\|\underline{\dot{q}}_\alpha\|$, decreases monotonically as α increases, a fact that can be exploited to find solutions subject to joint speed limits.

One would like α to be large enough to ensure good numerical stability as measured by the condition number, which indicates the maximum amplification of noise incurred in solving a set of linear equations [20]. For the $n \times n$ matrix $(\underline{\hat{J}}' \cdot \underline{\hat{J}} + \alpha^2\underline{I})$, the condition number is

$$\kappa^2 \triangleq \operatorname{cond}\left(\underline{\hat{J}}' \cdot \underline{\hat{J}} + \alpha^2\underline{I}\right) = \frac{\sigma_1^2 + \alpha^2}{\sigma_n^2 + \alpha^2} \tag{42}$$

which shows that a large α brings the condition number arbitrarily close to unity [4]. In contrast, α should be small enough that the residual error is not greatly increased when $\underline{\hat{J}}$ is full-rank. A bound on the change in residual error is established in the following theorem [4].

Theorem 3: If $\boldsymbol{r} = \underline{\hat{v}} - \underline{\hat{J}}\underline{\dot{q}}_0$ and $(\boldsymbol{r} + \delta\boldsymbol{r}) = \underline{\hat{v}} - \underline{\hat{J}}\underline{\dot{q}}_\alpha$ where $\underline{\dot{q}}_0$ is the minimum-norm minimizer of $\|\underline{\hat{J}}\underline{\dot{q}} + \underline{\hat{v}}\|$ and $\underline{\dot{q}}_\alpha$ is the minimizer of the sum $\|\underline{\hat{J}}\underline{\dot{q}} - \underline{\hat{v}}\|^2 + \alpha^2\|\underline{\dot{q}}\|^2$, then

$$\frac{\|\delta\boldsymbol{r}\|}{\|\underline{\hat{v}}\|} \leqslant \frac{\alpha^2}{\sigma_r^2 + \alpha^2} \tag{43}$$

where σ_r is the smallest nonzero singular value of $\underline{\hat{J}}$.

If the units of length are scaled by the maximum reach of a manipulator, the singular values of the partial velocity matrix will range approximately from zero to unity. Then a damping factor $\alpha = 0.03$ will keep the condition number under 1200 while the magnitude of the residual at well-conditioned configurations, say $\sigma_6 > 0.1$, will increase by at most $0.08\ \|\underline{\hat{v}}\|$. The damping factor could also be adjusted according to estimates of the largest and smallest singular values, which is especially attractive in cases where these estimates can be computed inexpensively [4].

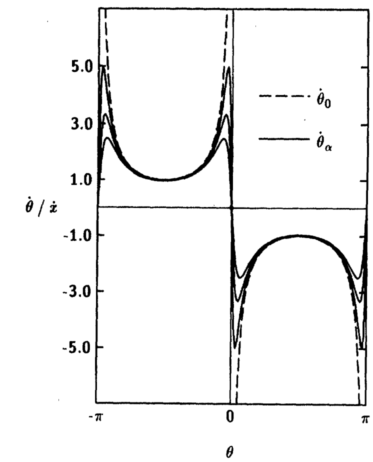

Fig. 3. Joint velocity solutions to approximate a Cartesian speed specification for a one-link arm. The pseudoinverse solution $\dot{\theta}_0$ and three damped least-squares solutions, $\dot{\theta}_\alpha(\alpha = 0.10, 0.15, 0.20)$, are shown.

The simplest example of a manipulator with a singularity consists of a single link of unit length with a rotational joint. Suppose that one cares only about the position of the endpoint along a horizontal line. Then if θ measures the angle of rotation of the link from the horizontal, the relations of interest are

$$\cos\theta = x, \qquad -\sin\theta\dot{\theta} = \dot{x} \tag{44}$$

and a solution for $\dot{\theta}$ given $\dot{x}$ is sought. The pseudo-inverse with a threshold of ϵ is

$$\dot{\theta}_\epsilon = \begin{cases} -\dot{x}/\sin\theta, & \text{if } |\sin\theta| > \epsilon \\ 0, & \text{otherwise} \end{cases} \tag{45}$$

and the damped least-squares solution is

$$\dot{\theta}_\alpha = \frac{-\sin\theta}{\sin^2\theta + \alpha^2}\dot{x}. \tag{46}$$

These solutions approach the exact pseudo-inverse solution in the limit as ϵ or α go to zero, but the damped least-squares solution is continuous and avoids placing a deadzone at the origin [see Fig. 3].

Computing the Damped Least-Squares Solution

The first steps in computing the damped least-squares solution are to form the partial velocity matrix $\underline{\hat{J}}$ and the $n \times n$ symmetric matrix $\underline{P} \triangleq \underline{\hat{J}}' \cdot \underline{\hat{J}}$. After using the method outlined above for computing $\underline{\hat{J}}_0$, one may form $\underline{P}$ as the matrix product $\underline{\hat{J}}_0'\underline{\hat{J}}_0$. (Since $\underline{P}$ is symmetric, only the elements in the upper triangle need to be computed.) By this method, each element requires six multiplications and five additions to compute, but a few operations can be saved in some cases by noting that the ijth element of $\underline{P}$ is given by

$$P_{ij} = \boldsymbol{v}_i^R \cdot \boldsymbol{v}_j^R + \boldsymbol{\omega}_i^T \cdot \boldsymbol{\omega}_j^T. \tag{47}$$

Consequently, the diagonal elements P_{ii} can be computed using only 2 multiplications and 2 additions by performing the computation in reference frame i, noting that $z_i \cdot z_i = 1$ and that $[z_i \times \boldsymbol{p}_i^{R/Q_{i-1}}]_i$ has a zero element (20). Also the operations for the superdiagonal elements $P_{i(i+1)}$ can be cut in half by using $z_i \cdot z_{i+1} = \cos\beta_i$ (22). After the dot-products with $\hat{\underline{v}}$ are computed, the equation

$$(\underline{P} + \alpha^2 \underline{I})\dot{\underline{q}} = \hat{\underline{J}}' \cdot \hat{\underline{v}} \tag{48}$$

can be solved using Cholesky factorization.

In solving (48) using floating point arithmetic, the relative error due to roundoff may be amplified by as much as the condition number κ^2, but the amplification of errors in $\hat{\underline{J}}$ and $\hat{\underline{v}}$ due to other sources (such as measurement error) is limited by the condition number of

$$\begin{bmatrix} \hat{\underline{J}} \\ \alpha \underline{I} \end{bmatrix}$$

with respect to pseudo-inversion (35), which is κ [21]. Thus, if the roundoff error is less than κ times the measurement error, the error in the solution is not likely to be affected by the precision of the computer. If this is not the case, the computation can be done while maintaining the condition number of κ by using Householder transformations to perform a QR factorization of the nonsymmetric matrix

$$\begin{bmatrix} \hat{\underline{J}} \\ \alpha \underline{I} \end{bmatrix}$$

[18]. Forming the normal equations and using a Cholesky factorization as outlined above requires $\frac{1}{6}n^3 + 3\frac{1}{2}n^2 + 2\frac{1}{3}n$ multiplications, while a QR-factorization with column pivoting can be done in $8n^2 + 19n$ multiplications. Thus for small n, the QR-factorization method is twice as expensive as the Cholesky method, and it remains more expensive for $n < 30$.

Inverse Acceleration / Angular Acceleration Solutions

Differentation of (4, 5) with respect to time gives

$$\boldsymbol{a}^P = \sum_{i=1}^{n} \boldsymbol{v}_i^P \ddot{q}_i + \boldsymbol{a}_t^P \tag{49}$$

$$\boldsymbol{\alpha}^B = \sum_{i=1}^{n} \boldsymbol{\omega}_i^B \ddot{q}_i + \boldsymbol{\alpha}_t^B. \tag{50}$$

In general, for manipulators, the vectors $\boldsymbol{a}_t^R$ and $\boldsymbol{\alpha}_t^T$ are not zero, as were $\boldsymbol{v}_t^R$ and $\boldsymbol{\omega}_t^T$, because they contain derivatives of the partial velocities and partial angular velocities. However, it is inefficient to compute these vectors by taking derivatives. Rather, one should follow the procedures for acceleration and angular acceleration used in the recursive Newton–Euler formulation of the equations of motion for manipulators [22], with terms involving $\ddot{\underline{q}}$ left out. An inverse acceleration/angular acceleration solution is then computed using the methods outlined above to solve the equation

$$\hat{\underline{J}}\ddot{\underline{q}} = \begin{bmatrix} \boldsymbol{a}^R - \boldsymbol{a}_t^R \\ \boldsymbol{\alpha}^T - \boldsymbol{\alpha}_t^T \end{bmatrix}. \tag{51}$$

In a similar fashion, these methods also apply to inverse kinematic solutions of any higher-order derivative.

Remarks

In practice, inversion of the kinematic equations for the Stanford Arm using the decomposition mentioned above for arms with a spherical wrist requires 42 multiplications and 28 additions, but the method fails near singularities. In comparison, the damped least-squares formulation requires 210 multiplications and 140 additions, but this can be reduced to only 66 multiplications and 38 additions by independently applying damped least-squares to the two three-degree-of-freedom problems encountered when the spherical wrist decompositon is used. Numerical experiment shows that the two least-squares formulations give approximately the same answer as long as $\alpha \geqslant 0.1$ with lengths scaled to the maximum reach of the arm [4]. However, the real cost of the exact damped least-squares method is greatly reduced in the case that the position of the endpoint, $\boldsymbol{p}^{R/Q_0}$, and the direction cosine matrix for the end-effector, ${}^0\underline{C}^T$, are desired, since the computation of these are major components of the formation of the partial velocity matrix.

Summary

An efficient vector formulation of inverse kinematic problems is presented in terms of a partial velocity matrix. It is shown that the ith column of that matrix has a simple expression in link i coordinates and that the most efficient algorithm results from a careful consideration of the reference frames in which the computation is performed. An algorithm whose computation count has a lower coefficient of n than any other of its kind in the literature is given for computing the partial velocity matrix in the reference frame of the end-effector. Problems of rank-deficiency in formulations of inverse kinematics are discussed and a damped least-squares method that is well-behaved even near kinematical singularities is proposed. This suppression of singularities eliminates a source of catastrophic failure present in previous formulations of resolved-rate control. The existence of efficient, well-behaved algorithms for inverse kinematic solutions may lead to the practical use of redundant arms and nonredundant arms lacking closed-form position solutions.

References

[1] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Machine Systems*, vol. MMC-10, pp. 47–53, June 1969.

[2] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, "Resolved-acceleration control of mechanical manipulators," *IEEE Trans. Automatic Control*, vol. AC-25, pp. 468–474, June 1980.

[3] D. L. Peiper, "The kinematics of manipulators under computer control," Ph.D. dissertation, Computer Science Department, Stanford University, 1968.

[4] C. W. Wampler, "Computer methods in manipulator kinematics, dynamics, and control: A comparative study," Ph.D. dissertation, Mechanical Engineering Department, Stanford University, 1984.

[5] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *ASME J. of Applied Mechanics*, vol. 22, pp. 215–221, June 1955.

[6] R. Paul, "Manipulator Cartesian path control," *IEEE Trans. Syst., Man, and Cyber.* vol. SMC-9, pp. 702–711, 1979.

[7] D. E. Whitney, "The mathematics of coordinated control of pros-

thetic arms and manipulators," *ASME J. of Dynamic Systems, Measurement, and Control*, pp. 303–309, Dec. 1972.

[8] O. Khatib, "Commande dynamique dans l'espace opérationnel des robots manipulateurs en présence d'obstacles," Ph.D. dissertation, ENSAE, Toulouse, France, 1980.

[9] T. R. Kane, *Dynamics*. Stanford, CA: Stanford University, 1972.

[10] M. Renaud, "Calcul de la matrice Jacobienne nécessaire à la commande coordonnée d'un manipulateur," *Mechanism and Machine Theory*, vol. 15, No. 2, pp. 81–91, 1980.

[11] D. E. Orin and W. W. Shrader, "Efficient computation of the Jacobian for robot manipulators," *International J. of Robotics Research*, vol. 3, No. 4, pp. 66–75, Winter 1984.

[12] R. Featherstone, "Calculation of robot joint rates and actuator torques from end effector velocities and applied forces," *Mechanism and Machine Theory*, vol. 18, No. 3, pp. 193–198, 1983.

[13] J. M. Hollerbach and G. Sahar, "Wrist-partitioned, inverse kinematic accelerations and manipulator dynamics," *International J. of Robotics Research*, vol. 2, No. 4, pp. 61–76, Winter 1983.

[14] V. C. Klema and A. J. Laub, "The singular value decomposition: its computation and some applications," *IEEE Trans. on Automatic Control*, vol. AC-25, pp. 164–176, April 1980.

[15] A. Liégeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-7, pp. 868–871, Dec. 1977.

[16] C. A. Klein and C.-H. Huang, "Review of psuedoinverse control for use with kinematically redundant manipulators," *IEEE Trans. Syst., Man, Cybern.* vol. SMC-13, pp. 245–250, Mar. 1983.

[17] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications*. New York: Wiley, 1974.

[18] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*. Englewood Cliffs, N. J.: Prentice-Hall, 1974.

[19] B. Noble, "Methods of computing the Moore-Penrose generalized inverse, and related matters," *Generalized Inverses and Applications*, M. Z. Nashed, Ed. New York: Academic, pp. 245–301, 1975.

[20] G. Forsythe and C. B. Moler, *Computer Solution of Linear Algebraic Systems*. Englewood Cliffs, N. J.: Prentice-Hall, 1967.

[21] G. W. Stewart, "On the perturbation of pseudo-inverses, projections, and linear least squares problems," *SIAM Review*, Vol. 19, pp. 634–662, Oct. 1977.

[22] J. Y. S. Luh, M. W. Walker and R. P. C. Paul, "On-line computational scheme for mechanical manipulators," *ASME J. of Dynamic Systems, Measurement, and Control*, vol. 102, pp. 69–76, June 1980.