A Derivative-Free Iterative Method for Locating the Hand Position of a Robot Manipulator

Moody T. Chu[1 2]
Mathematics and Computer Science Division
Argonne National Laboratory
9700 South Cass Avenue
Argonne, Illinois 60439-4844

December 1988

[1]This work was supported in part by the Applied Mathematical subprogram of the Office of Energy Research, U.S. Department of Energy, under Contract W-31-109-Eng-38, while the author was spending his sabbatical leave at Argonne National Laboratory.

[2]Permanent Address: Department of Mathematics, North Carolina State University, Raleigh, North Carolina 27695-8205.

Abstract

In attempting to minimize the distance between the hand of a manipulator and the position of a goal, a numerical method that alternately adjusts the configuration of one joint at a time is proposed. The adjustment for either a revolute joint or a prismatic joint can be made effectively without derivative evaluation. This method can be applied to any manipulator of any number of joints with arbitrary geometry. Global convergence is expected in general. Empirical results seem to suggest that the rate of convergence is linear and that for redundant manipulators the error constants are fairly small.

1 Introduction

A manipulator usually is considered to consist of a series of links connected together by joints. The study of kinematic analysis is concerned with the relationship between the end-effector, *the hand*, and the joint displacements. One well-established technique in this field is to use the so called *homogeneous transformations* [3] [12] [4] to effectively describe these relationships. More precisely, by embedding a coordinate frame in each mechanical linkage of the manipulator, we can use homogeneous transformations to describe the relative position and orientation between these coordinate frames. Indeed, each homogeneous transformation is a 4×4 real matrix. Thus, the position and orientation of the hand in terms of base coordinates are given by the matrix product of the homogeneous transformations representing the relative joint displacements. This relationship is called *the kinematic equation.*

One utmost important problem in robot manipulation is to control the relative motion associated with each joint so that the the hand can be positioned in a desired manner. This is because in practice we normally know where we want to move the manipulator and we need to know how to make the move. The problem of determining the joint coordinates, given the position and orientation of the hand, is commonly referred to as *the inverse kinematics problem.* Enormous amount of efforts have been devoted to solving the inverse kinematics problem. Without attempting to be complete, we mention [1], [2], [5], [7], [10], [12], [13], [14], [16] and the references cited therein. It is always desired to obtain a close-form solution for the inverse kinematics problem. However, the ability developed thus far seems to be limited only to manipulators having special geometry [14] [8]. For manipulators with completely general geometry, the analysis of the equation becomes very complicated and can only be solved numerically. Most of the numerical methods adopt the well-known Newton-Raphson iterative technique [14] [17]. Therefore, the success of these methods seems to depend upon the initial guesses. Recently, the homotopy method [16] has been applied successfully to computing all possible solutions of a general 6-R inverse kinematic equation. This involves, however, a nontrivial process of reducing the kinematic equation to a system of polynomials.

In order to manage the hand of a manipulator freely within the range of motion, it is necessary that a manipulator has six degrees of freedom. Three degrees of freedom are required to specify position and three more to specify orientation. In this paper we shall not concern ourselves with the orientation problem. In part this is because, from a theoretical point of view, locating the position of the hand is more interesting than aligning the orientation of the hand. In part this is because, from a practical point of view, the orientation of the hand can be effectively adjusted by attaching an appropriate wrist to the end-effector after the hand position has been located. We present an iterative method for locating the position of the hand only. The method has the advantages of being derivative-free, globally convergent, and easy to program. Our idea is based on the so called *alternating variables methods* [6]. That is, we adjust one joint at a time in an attempt to minimize the distance between the desired position of the object and the current position of the hand while all other joints are kept rigid. When every joint has been adjusted, the whole cycle is repeated until convergence occurs. We note that an algorithm using similar ideas has been developed in [11]. We shall see, however, the calculation in our approach is much simpler. The technique discussed here can be applied to any manipulator of any number of joints with arbitrary geometry. In fact, a general purpose code to which the only input will be the the initial configuration of the manipulator and the data of the goal can easily be developed.

We begin in the next section with a brief introduction of the homogeneous transformations just so that we can conveniently describe our algorithm. In section 3, we show how the adjustment for either a revolute joint or a prismatic joint can be handled effectively. In section 4, we report our numerical experiment with a general 6-R manipulator. We finally point out several problems that deserve further research.

2 Preliminaries

The main purpose of this section is to introduce the notations of the homogeneous transformations so that we can conveniently describe our algorithm. We shall state enough facts without giving too much reasoning. Readers are referred to [3],[9] and [12] for more detailed descriptions.

The *homogeneous coordinate* representation of a point vector

$$v = a\vec{i} + b\vec{j} + c\vec{k} \tag{1}$$

in R^3 is given by a 4×1 column vector

$$v = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \tag{2}$$

where

$$\begin{aligned} a &= x/w \\ b &= y/w \\ c &= z/w \end{aligned} \tag{3}$$

and x, y, z, w can be arbitrary real numbers. Vectors of the form $[a, b, c, 0]^T$ are understood to represent directions only. The vector $[0, 0, 0, 0]^T$ is undefined.

With the homogeneous representations, it is easy to see that the transformation H corresponding to a translation by a vector $a\vec{i} + b\vec{j} + c\vec{k}$ is represented as

$$T = Trans(a, b, c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4}$$

That is, given a vector $u = [x, y, z, w]^T$, the translated vector v is given by

$$v = Tu = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}. \tag{5}$$

Similarly, transformations corresponding to rotations about coordinate axes can easily be formulated. For example, the transformation R corresponding to a rotation about the x-axis by an angle θ is represented by

$$R = Rot(x, \theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6}$$

The combination of a sequence of translations and rotations amounts to the product of the corresponding 4×4 matrix representations. In this way we obtain a general *homogeneous transformation* H represented by the matrix

$$H = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{7}$$

For convenience, we define $\vec{n} = [n_x, n_y, n_z]^T$ and similar notations for $\vec{p}$ and $\vec{a}$. We remark that the three vectors $\vec{n}, \vec{p}$ and $\vec{a}$ are mutually orthonormal. It is not difficult to show that the inverse of H is given by

$$H^{-1} = \begin{bmatrix} n_x & n_y & n_z & -\vec{p} \cdot \vec{n} \\ o_x & o_y & o_z & -\vec{p} \cdot \vec{o} \\ a_x & a_y & a_z & -\vec{p} \cdot \vec{a} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{8}$$

The elements of a homogeneous transformation (7) may be interpreted as four vectors representing a second coordinate frame. That is, the first three column vectors describe the three new axis-directions, and the fourth column vector describes the position of the new origin in terms of the reference coordinate frame. This important fact can be realized simply by applying the homogeneous transformation to the origin $[0, 0, 0, 1]^T$ and the three vectors $[1, 0, 0, 1]^T, [0, 1, 0, 1]^T, [0, 0, 1, 1]^T$. From this viewpoint, when a vector is transformed, the original vector can be considered as a vector in the new coordinate frame whereas the transformed vector is the same vector described with respect to the reference coordinate frame.

The order of matrix multiplication has special physical meanings in homogeneous transformations. Let H_1 and H_2 be two homogeneous transformations. Then the new frame $H_1 H_2$ is obtained by making the translation or rotations represented by H_2 with respect to the frame axes described by H_1. The same coordinate frame $H_1 H_2$ may also be obtained by translating or rotating the frame axes described by H_2 according to the translation or rotations represented by H_1 with respect to the base reference coordinate frame. We shall see in the sequel that this understanding facilitates the derivation of the kinematic equation.

According to [12], we now define linkage parameters for joints as shown in Figure 1. Joint n is connected to joint $n + 1$ by link n. Joint 0 is fixed

Figure 1: The basic notation

to the ground. Link 0 is designated as the base. Depending upon whether joint n is a revolute or a prismatic joint, we define the z_{n-1} axis to be either the axis of rotation or the direction in which the joint moves. The following notations are relevant to the formation of two consecutive joints:

$a_n =$ The distance of the common normal H_nO_n between joint n and joint $n+1$.

$\alpha_n =$ The angle to rotate the axis z_{n-1} about the common normal H_nO_n so that z_{n-1} is parallel to z_n.

$d_n =$ The distance between the two common normals $H_{n-1}O_{n-1}$ and H_nO_n measure along the axis z_{n-1}.

$\theta_n =$ The angle to rotate the extended line of $H_{n-1}O_{n-1}$ about the axis z_{n-1} so that the extended line of $H_{n-1}O_{n-1}$ is parallel to H_nO_n.

The configuration of a manipulator with mixed prismatic and revolute joints is completely determined by values of these linkage parameters. In robot manipulation, usually only parameters θ_n and parameters d_n are the variables for revolute joints and prismatic joints, respectively, while all other

Figure 2: A general 6-link manipulator

parameters should be kept constant.

In order to describe the relationship between links, we now assign coordinate frames to each link according to the following rules:

1. The coordinate system for link 0 is fixed to the ground and is considered as the base coordinate system.

2. The origin of the coordinate frame of link n is set to be at the point On. In the case of intersecting axes, the origin is at the point of intersection fo the joint axes.

3. The z-axis for link n is the axis of rotation z_n of joint $n+1$.

4. The origin of the coordinate frame for the last link is set to be at the hand of the manipulator. The z-axis for the last link is chosen to lie in the direction from which the hand would approach an object. The coordinate systems for a general 6-link manipulator is shown in Figure 2.

5. The x-axis for link n is aligned with the direction from H_n to O_n. In the case of intersecting axes, the direction of x-axis is parallel to the

cross product $z_{n-1} \times z_n$.

6. The y-axis is determined according to the right-hand screw rule.

It is obvious that frame $n-1$ can be transformed to frame n through the following four steps:

- Rotate an angle θ_n about the axis z_{n-1}.
- Translate a distance d_n along the axis z_{n-1}.
- Translate a distance a_n along the axis x_n.
- Rotate an angle α_n about the axis x_n.

This sequence of operations may be expressed as

$$\begin{aligned} A_n &= \begin{bmatrix} \cos\theta_n & -\sin\theta_n & 0 & 0 \\ \sin\theta_n & \cos\theta_n & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_n \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_n & -\sin\alpha_n & 0 \\ 0 & \sin\alpha_n & \cos\alpha_n & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta_n & -\sin\theta_n\cos\alpha_n & \sin\theta_n\sin\alpha_n & a_n\cos\theta_n \\ \sin\theta_n & \cos\theta_n\cos\alpha_n & -\cos\theta_n\sin\alpha_n & a_n\sin\theta_n \\ 0 & \sin\alpha_n & \cos\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{9}$$

According to (8), the inverse of A_n is given by

$$A_n^{-1} = \begin{bmatrix} \cos\theta_n & \sin\theta_n & 0 & -a_n \\ -\sin\theta_n\cos\alpha_n & \cos\theta_n\cos\alpha_n & \sin\alpha_n & -d_n\sin\alpha_n \\ \sin\theta_n\sin\alpha_n & -\cos\theta_n\sin\alpha_n & \cos\alpha_n & -d_n\cos\alpha_n \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{10}$$

Now that A_1 describes the position and orientation of the first link with respect to the base coordinate frame and that A_2 describes the position and orientation of the second link with respect to the first, the position and orientation of the second link in base coordinates are given by the matrix product

$$T_2 = A_1 A_2. \tag{11}$$

Continuing in this way, we obtain the kinematic equation

$$T_k = A_1 A_2 \cdots A_k \tag{12}$$

representing the position and orientation of the hand of a general k-link manipulator. It is worth noting that the description of link coordinate frame i with respect to link coordinate frame j, $j < i$, is given by

$${}^{j}T_i = A_{j+1} \cdots A_i. \tag{13}$$

The inverse kinematics problem is, given T_n, determine the variables in $A_1, \cdots, A_n$ so that the equation (12) is satisfied.

3 Algorithm

The basic idea of our algorithm is to adjust one joint at a time in an attempt to minimize the distance between the goal and the the hand while all other joints are kept rigid. In this section we explain how this idea can be carried out effectively.

Suppose the joint to be adjusted is a revolute joint. Since all other joints are kept rigid, the current situation may be simply represented as shown in Figure 3. The origin O stands for the current revolute joint position at which a coordinate frame has been assigned. The unit vector $\vec{n}$ is aligned with the axis of rotation. The three-dimensional vectors $\vec{a}$ and $\vec{b}$ measure, respectively, the positions of the hand and the goal with respect to the current coordinate frame. It is not difficult to see from the geometry that in order to minimize the distance between the hand and the goal subject to the rotation about $\vec{n}$, we should align the projections of $\vec{a}$ and $\vec{b}$ in the plane perpendicular to $\vec{n}$. These projections can easily be calculated to be $\vec{n} \times \vec{a} \times \vec{n}$ and $\vec{n} \times \vec{b} \times \vec{n}$, respectively. Therefore, the joint should be revolved about $\vec{n}$ by an angle

$$\theta = \arccos \frac{(\vec{n} \times \vec{a} \times \vec{n}) \cdot (\vec{n} \times \vec{b} \times \vec{n})}{\|\vec{n} \times \vec{a} \times \vec{n}\| \|\vec{n} \times \vec{b} \times \vec{n}\|}. \tag{14}$$

If all the vectors are written in terms of the current coordinate frame, say, $\vec{a} = [a_1, a_2, a_3]^T$ and $\vec{b} = [b_1, b_2, b_3]^T$. Since $\vec{n} = [0, 0, 1]^T$, the angle can easily

Figure 3: Adjustment of a revolute joint

be calculated to be

$$\theta = \arccos \frac{a_1 b_1 + a_2 b_2}{\sqrt{a_1^2 + a_2^2}\sqrt{b_1^2 + b_2^2}}. \tag{15}$$

Suppose the joint to be adjusted is a prismatic joint. The situation is represented as shown in Figure 4. The origin O stands for the current prismatic joint position at which a coordinate frame has been assigned. The unit vector $\vec{n}$ points to the direction in which the joint moves. The vectors $\vec{a}$ and $\vec{b}$ measure, respectively, the positions of the hand and the goal with respect to the current coordinate frame. When the joint is being adjusted, the vector $\vec{a}$ stays rigid and is shifted up or down along the $\vec{n}$ axis. It is not difficult to see that in order to minimize the distance between the hand and the goal subject to the movement along $\vec{n}$, the vector $\overline{GH'}$ should be perpendicular to the vector $\overline{HH'}$. Therefore, if the goal has coordinates $[g_1, g_2, g_3]^T$ and the hand has coordinates $[h_1, h_2, h_3]^T$, then the joint should be moved in the $\vec{n}$ direction by a distance

$$d = g_3 - h_3. \tag{16}$$

Give a general k-link manipulator, we shall refer to each set of k operations mentioned above as a *sweep*. Depending upon the type of joints, in a

Figure 4: Adjustment of a prismatic joint

given sweep there is one rotation (15) or one translation (16) for each joint. We emphasize that there are different ways to order the operations within a sweep.

Is is evident now that we need an effective way to obtain the coordinates of a given point relative to different coordinate systems. For a robot manipulator, the homogeneous transformation (9) introduced in the preceding section can facilitate this computation. In particular, we know for $n = 0, \ldots, k-1$, the vector $\overline{O_n H}$ (refer to Figure 2) from joint $n+1$ to the hand in the coordinate frame n is given by

$$\overline{O_n H} = A_{n+1} \cdots A_k \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \tag{17}$$

while for $n = 1, \ldots, k-1$, the vector $\overline{O_n G}$ from joint $n+1$ to the goal in the coordinate frame n is given by

$$\overline{O_n G} = (A_1 \cdots A_n)^{-1} \overline{O_0 G} \tag{18}$$

where $\overline{O_0G}$ is the position of the goal in the base coordinate frame. We note that the inverse involved in (refeq:goal) can be obtained easily from (10).

For illustration purpose, we now consider a sweep with the monotone increasing ordering. For convenience, we introduce notations $\vec{e}_4 = [0, 0, 0, 1]^T$, and the homogeneous coordinate $\vec{g} = [g_1, g_2, g_3, 1]^T$ of the goal with respect to the base coordinate frame. When it is not ambiguous, we shall not distinguish a three-dimensional vector from its own homogeneous representations. Let $A_n, n = 1, \ldots, k$ be the matrix representing the current configuration of link n. Let $B_n = {}^{n-1}T_k = A_n \cdots A_k, n = 1, \ldots, k$ be the matrix representing the coordinate frame k with respect to coordinate frame $n-1$. Then the following summarizes one typical sweep of our algorithm:

For $n = 1, \ldots, k$**, do:**

$\vec{a} := B_n \vec{e}_4$

If $n = 1$

then

$\vec{b} := \vec{g}$

else

$\vec{b} := C_{n-1} \vec{g}$

If *revolute*

then

compute angle θ_n **by (15)**

If *prismatic*

then

compute displacement d_n **by (16)**

Update A_n

If $n = 1$

then

$C_1 := A_1^{-1}$

else

$C_n := A_n^{-1} C_{n-1}$

link	θ_n(degree)	d_n	a_n	α_n(degree)
1	20	0.1875	0.5000	80
2	20	0.3750	1.0000	15
3	20	0.2500	0.1250	120
4	30	0.8750	0.6250	75
5	10	0.5000	0.3125	100
6	16	0.1250	0.2500	60

Table 1: Linkage parameters for a 6-R manipulator

We expect our method to be globally convergent in general because at each step of a sweep the distance between the hand and the goal is reduced monotonously. Although in the literature pathological examples [15] have been constructed to show that the alternating variables method may converge to a closed loop, we have not observed any occurrence of cycling in our numerical experiment with robot manipulators. In the case that the proposed goal position lies outside the workspace of a robot manipulator, the iterates of our method converge to a point that is nearest to the goal.

4 Numerical Experience

Our algorithm can be implemented easily. Since most computations are matrix operations, the coding is particularly easier if a matrix-oriented language, such as MATLAB, is used. In this section we report some typical results from our numerical experience.

sweep	x	y	z	e_s
0	2.9366	1.0122	0.8039	2.7284e+00
1	-0.0370	0.6772	0.7877	2.6435e-01
2	0.2356	0.7149	0.7949	1.1187e-02
3	0.2237	0.7156	0.7955	7.3127e-04
4	0.2245	0.7155	0.7955	3.8316e-05
5	0.2244	0.7155	0.7955	2.0515e-06

Table 2: Iterates of the hand position

The linkage parameters given in Table 1 represents the initial configuration of a 6-R manipulator with general geometry. By using our algorithm, we obtain the iterates of the hand position as listed in Table 2. The last column e_s in Table 2 measures the distance from the goal to the hand after the $s-th$ sweep. As is seen, the convergence is quite fast. To further analyze the convergence, we assume the relationship

$$||e_{s+1}|| = c||e_s||^p \tag{19}$$

for some constants c and p. The graph of $\log e_{s+1}$ versus $\log e_s$ is shown in Figure 5. It appears from Figure 5 that *our method converges linearly.* By fitting the data in Table 2, it is interesting to find that $p \approx 1.0319$ whereas $c \approx e^{-2.6784} \approx 0.0687$. We note it is the small value of c that makes the linear convergence fast.

By keeping the values of a_n, d_n and α_n fixed, we investigate numerically the values of error parameters c and p corresponding to different initial values of θ_n. A partial result is listed in Talbe 3. These empirical results seem unanimously in support that our method converges linearly with a fairly small error constant. We have tested many other 6-link manipulators with completely different linkage configurations. We obtain similar fast linear convergence results.

Figure 5: The plot of $\log e_{s+1}$ versus $\log e_s$

θ_1	-81.0406	20.0000	10.0000	81.4389	-90.0000
θ_2	137.0435	20.0000	20.0000	145.3491	0
θ_3	55.3546	20.0000	30.0000	86.9705	90.0000
θ_4	-3.6365	30.0000	40.0000	110.4291	83.0000
θ_5	98.6737	10.0000	50.0000	49.4797	-90.0000
θ_6	166.5527	15.0000	60.0000	158.5217	0
p	1.1106	1.0319	0.9740	1.0245	1.0350
$\log c$	-1.5759	-2.6784	-3.5219	-1.9879	-2.0944

Table 3: Relations between error parameters and initial configurations

goal	$[0.2500, 0.8000, 1.0000]^T$		$[0.2622, 0.8390, 1.1235]^T$	
links	p	$\log c$	p	$\log c$
4	0.9968	-0.6802	0.9981	-0.6443
5	1.0085	-0.2852	0.9960	-0.6752
6	1.0329	-1.9346	0.9990	-1.2614

Table 4: Relations between error parameters and numbers of links

In general we only need three degrees of freedom to locate a position. Thus we suspect that the small error constants c shown above for a 6-link manipulator would be mainly due to the redundancy of the links needed to locate the hand position. Numerical experience, however, seems to indicate that there may be other factors involved. For example, by experimenting with manipulators consisting of the first 4, 5 and 6 links configured in Table 1 to reach the points $[0.2500, 0.8000, 1.0000]^T$ or $[0.2622, 0.8390, 1.1235]^T$, respectively, we obtain error parameters as listed in Table 4. It is quite a surprise to see that a 5-link manipulator may not always converge faster than a 4-link manipulator.

5 Conclusion Remarks

In the above we have discussed an alternating variables method to locate the hand position of a robot manipulator. The method can be applied to any manipulator of any number of joints with arbitrary geometry. The amount of adjustment for each joint can be easily calculated. Thus far the orientation of the hand has not been considered in the computation. Therefore, a 6-link manipulator would be regarded as redundant. However, our method applied to this kind of redundant manipulators seems to converge fairly fast because of small error constants.

We conclude this paper by pointing out more questions that remain to be further investigated:

1. The study of generalizing the idea of this paper to include the orientation of the hand should be an interesting research. The problem is harder because the simple geometry as that in Figure 3 or Figure 4 no longer exists.

2. The effect of orderings of a sweep on the convergence and the final configuration of the manipulator should be another interesting study. We think the interplay of the initial configuration and the ordering determines the final configuration. The theory is not completely understood yet.

3. In practice, joints may have physical constraints. A prismatic joint, for example, cannot be arbitrarily extendable. When joint constraints exist, what can be said about the performance of our method?

4. It would be desirable that based on the information provided by a sweep, a more substantial adjustment might be made at the next sweep. The acceleration process, of course, would make our algorithm more complicated.

References

[1] H. Albala, Displacement analysis of the general n-bar, single-loop, spatial linkage (two parts), ASME Journal of Mechanical Design, 104(1982), 504-525.

[2] J. Angeles, On the numerical solution of the inverse kinematics problem, The International Journal of Robotics Research, 4(1985), 21-37.

[3] J. Denavit and R. S. Hartenberg, A kinematic notation for lower-pair mechanisms based on matrices, ACME Journal of Applied Mechanics, 22(1955), 215-221.

[4] J. Duffy, Analysis of Mechanisms and Robots Manipulators, Wiley, New York, 1980.

[5] J. Duffy and C. Crane, A displacement analysis of the general spatial, 7R mechanism, Mechanisms and Machine Theory, 15(1980), 153-169.

[6] R. Fletcher, Practical Methods of Optimization, 2nd ed., Wiley, New York, 1987.

[7] A. A. Goldenberg and D. L. Lawrence, A generalized solution to the inverse kinematics of robotic manipulators, ASME Journal of Dynamic Systems, Measurements and Control, 107(1985), 103-106.

[8] K. C. Gupta and B. Roth, Design considerations for manipulator workspace, ASME Journal of Mechanical Design, 104(1982), 704-711.

[9] R. S. Hartenberg and J. Denavit, Kinematic Synthesis of Linkages, McGraw-Hill, New York, 1964.

[10] K. Kazerounian, On the numerical inverse kinematics of robotic manipulators, ASME Journal of Mechanisms, Transmissions, and Automation in Design, 109(1987), 8-13.

[11] J. Linares and A. Page, Position analysis of spatial mechanisms, ASME Journal of Mechanisms, Transmissions, and Automation in Design, 106(1984), 252-255.

[12] R. P. Paul, Robot Manipulators: Mathematics, Programming, and Control, MIT Press, Cambridge, 1984.

[13] G. R. Pennock and A. T. Yang, Application of dual-number matrices to the inverse kinematics problem of robot manipulators, ASME Journal of Mechanisms, Transmissions, and Automation in Design, 107(1985), 201-208.

[14] D. L. Pieper, The Kinematics of Manipulators under Computer Control, Stanford Artificial Intelligence Laboratory, Stanford University, AIM 72, 1968.

[15] M. J. D. Powell, On search directions for minimization algorithms, Math. Programming, 4(1973), 193-201.

[16] L. W. Tsai and A. P. Morgan, Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods, ASME Journal of Mechanisms, Transmissions, and Automation in Design, 107(1985), 189-200.

[17] J. J. Uicker, Jr., J. Denavit and R. S. Hartenberg, An iterative method for the displacement analysis of spatial mechanisms, ASME Journal of Applied Mechanics, 86(1964), 309-316.