

ON THE METRICS AND COORDINATE SYSTEM INDUCED SENSITIVITY IN COMPUTATIONAL KINEMATICS

Vojin T. Jovanovic

T. Sendzimir, Inc.

Waterbury, Connecticut

and

Kazem Kazerounian

Professor

Department of Mech. Eng.

University of Connecticut

Storrs, Connecticut

key words: sensitivity, singularity, Newton-Raphson method, scaling, chaos, fractals, equations, basins of attraction.

ABSTRACT

This paper is an attempt to investigate the sensitivity to change of units and coordinate systems in computational kinematics when it involves both orientation and displacement in three dimensional space. The focus is on the behavior of the Newton-Raphson iterative technique for solving customary system of equations for Kinematic loop closure. It is shown that with the change of units or coordinate systems for some initial points the method does not converge to the same solution. Such behavior is attributed to the shift of the boundaries of so called basins of attraction which play an important role in the theory of Chaos and Fractals. A number of numerical examples have been investigated and presented in tabulated form. To reduce the effect of sensitivity two procedures are suggested. One of them, Nonlinear Elimination, is a recent development based on numerical elimination of variables in a system of equations. Investigation presented in this paper is the first of its kind and it is hoped that it will initiate further research to treat the problem of sensitivity in computational kinematics.

1. INTRODUCTION

Sensitivity to the change of units or choice of coordinate systems in computational kinematics plays an important role in determining an outcome of the numerical procedure (for example the roots of the systems of equations). This sensitivity is most prominent when resolving the inverse kinematics problem with the Newton-Raphson method is attempted. Unfortunately, even though a large body of methods for solving system of equations exists (Rheinboldt [1974]), studies on the sensitivity problem are virtually nonexistent. The problem was probably destined to wait until development of the Theory of Chaotic Dynamical Systems (Devaney[1989]) which shed some light on the problem of iterating mathematical functions. This subject is relatively new and, with the aid of computers, is becoming a fertile field for scientific endeavors . We will attempt to show that some of the findings of chaos theories can help in understanding and treating the sensitivity problem.

To solve a system of equations there are several methods such as: Continuation methods, Matrix iterative methods and Polynomial elimination (Gupta [1993]), however, sensitivity to the change of units for these methods has not been investigated. In this paper we plan to proceed with such investigation using the Newton-Raphson method, as well as to discuss potential remedies. We propose step cutting in the Newton-Raphson method and Nonlinear Elimination algorithm as good solutions to resolve this issue Some of the developments in this paper are related to the work presented in Jovanovic and Kazerounian [1996]. More detailed discussion of chaotic numerical instabilities with applications is given in Jovanovic [1997].

In this paper we first explain what experiments we performed and how we developed necessary numerical procedures. Then, we demonstrate how the change of units affects final state sensitivity, and, finally, we discuss its cause as well as possible treatments.

2. MODELS EXAMINED

For numerical experiments, Robotic manipulators were selected because they ably demonstrate final state sensitivity when numerical solving is attempted. Such demand is best fulfilled if some of the kinematic equations are scaled by a constant when change of units occurs. It is shown that the governing kinematic equations for these systems include a mixture of unitless terms as well as terms with the unit of length (or higher order). The following are the mechanisms that are the subject of our numerical experimentation.

2.1 2-link manipulator in the plane (2D)

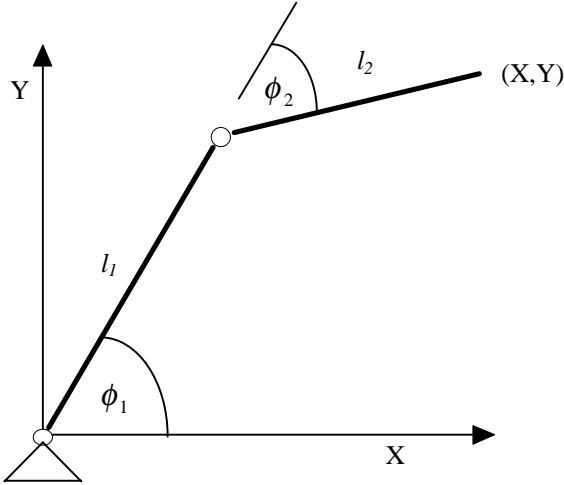


Figure 1. 2-link manipulator

2.2 3-joint revolute plane manipulator (2D3joint)

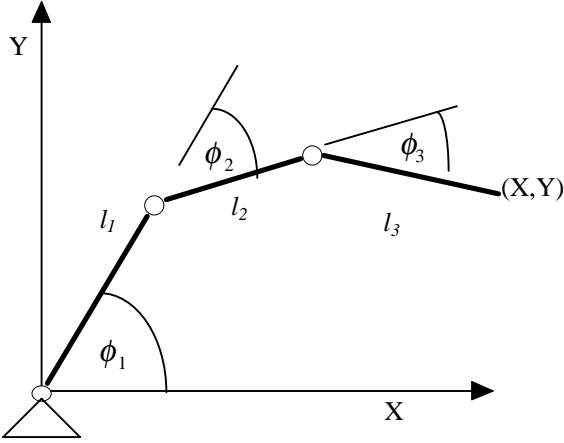


Figure 2. 3-joint revolute manipulator.

2.3 3R manipulator in 3D (3D3joint)

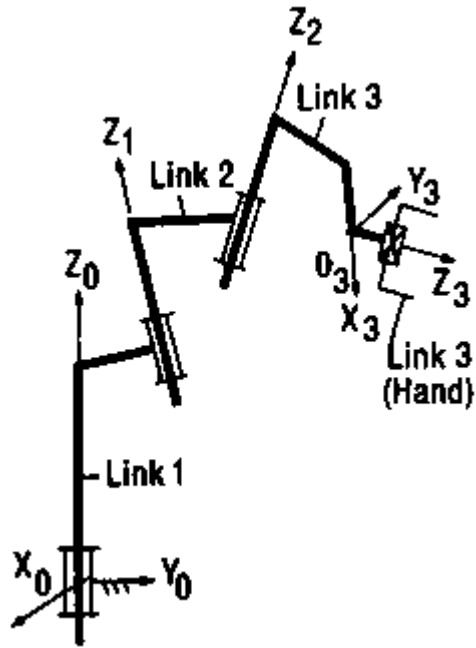


Figure 3. 3R manipulator.

2.4 6R manipulator (special configuration Puma)

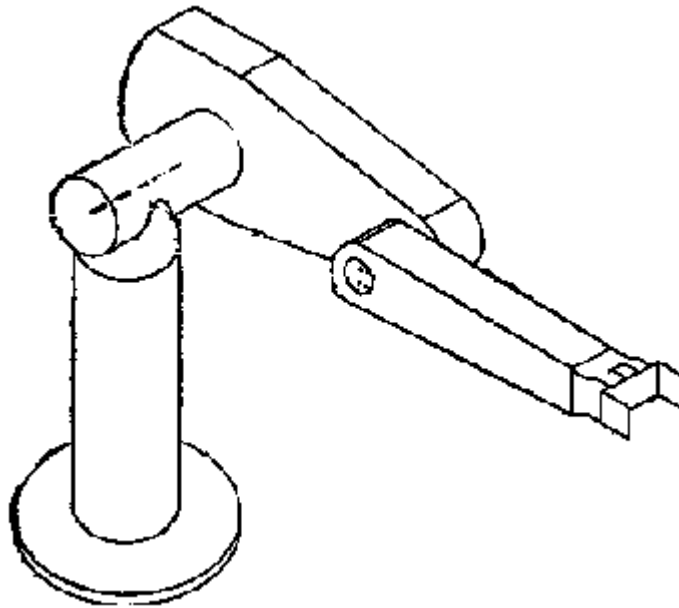


Figure 4. Puma manipulator.

2.5 6R general manipulator

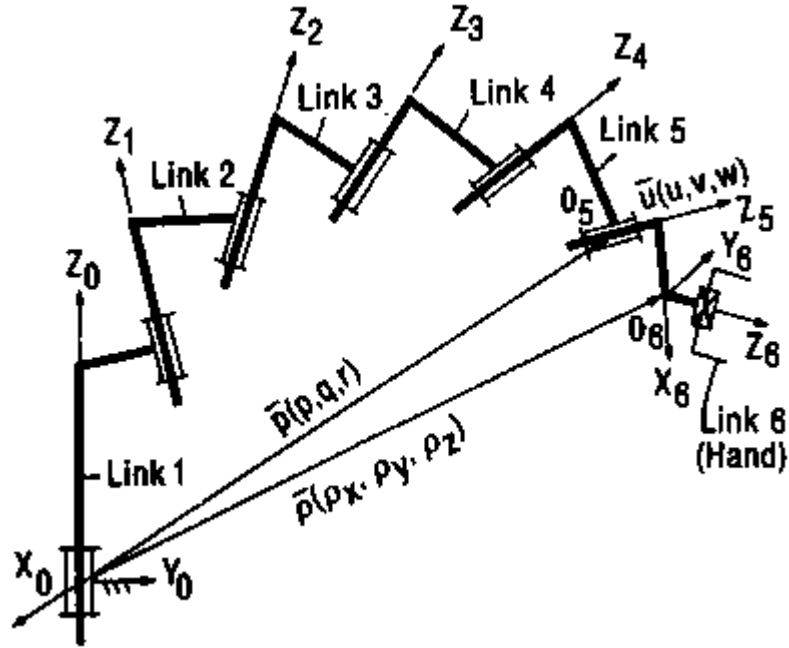


Figure 5. 6R general manipulator

3. NUMERICAL METHODS

For the above examples loop closure equations yield a system of coupled nonlinear equations that completely describe the geometry (position) of the mechanism. Such equations contain rotation as well as the position of the end effector (hand) of the manipulators. We will later show that this mixing is what causes difficulties in numerical computation.

3.1 Loop closure equations

The closure equation for general 6R manipulator is the following matrix equation. Equations for other cases are simple and can be derived from the general case.

$$A_1 A_2 A_3 A_4 A_5 A_6 = A_{hand} \quad (1)$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} l_x & m_x & n_x & \rho_x \\ l_y & m_y & n_y & \rho_y \\ l_z & m_z & n_z & \rho_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

A_{hand} is the 4×4 transformation matrix describing the Cartesian coordinate system 6, attached to the end effector (or last link) with respect to the coordinate system 0, (attached to the base link). A_i are 4×4 transformation matrices relating systems $i+1$ to i and are the commonly used Denavit-Hartenberg matrices. In the inverse kinematics procedure, the

unknown quantities are joint angles, and the above matrix equation must be solved for them. This is the inverse kinematics problem for the 6R series manipulator.

The matrix equation entails 12 scalar equations. Of these 12, only 6 are independent, because minor 3 by 3 principal matrix (rotation matrix) comprised of the 3 rows and columns is orthonormal. There are several methods available for extracting six independent equations (Orin and Schrader [1984]). Here we select the explicit Jacobian formulation as follows.

Joint velocities of the manipulator are related to the hand of the manipulator with

$$J(\boldsymbol{\theta}) \dot{\boldsymbol{\theta}}(t) = \begin{bmatrix} \boldsymbol{\omega}(t) \\ \mathbf{v}(t) \end{bmatrix} \quad (2)$$

where $\boldsymbol{\omega}$ is the angular velocity of the hand of the manipulator, \mathbf{v} is the velocity of the tip of the arm, $\boldsymbol{\theta}$ are joint variables and $\dot{\boldsymbol{\theta}}$ joint velocities respectively. $J(\boldsymbol{\theta})$ is the Jacobian matrix that can be calculated using

$$J_{i+1} = \begin{cases} \begin{bmatrix} \mathbf{z}_i \\ \mathbf{z}_i \times \mathbf{p}_i \mathbf{P}_{Hi} \end{bmatrix} & \text{for revolute joint} \\ \begin{bmatrix} 0 \\ \mathbf{z}_i \end{bmatrix} & \text{for prismatic joint} \end{cases} \quad i = 0, \dots, 5. \quad (3)$$

where J_{i+1} is the column in the Jacobian matrix, \mathbf{z}_{i-1} is the unity vector along the i^{th} joint axis and $\mathbf{P}_i \mathbf{P}_H$ is the vector beginning at i^{th} joint and ending at the tip of the manipulator. Both vectors are written with respect to the base system of coordinates. This representation of Jacobian spares us from differentiating Equation (1). Now the right hand side of the Equation (2) needs to be put into an appropriate form for the iteration. Velocity of the tip of the manipulator can be approximated by subtracting the 4th column of the matrix on the left from the 4th column of the right hand side of the Equation (1) and dividing with the time increment. However, for finding angular velocity of the hand a care must be exercised because angular velocity cannot be integrated due to its nonholonomic nature. One way around that constraint is remembering that

$$[\boldsymbol{\omega}] = \dot{R} R^T = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \quad (4)$$

where R is the rotation matrix at the current instant of time. From this relation we can construct the angular velocity vector $\boldsymbol{\omega}$ from the elements of the matrix product above or below the zero diagonal in Equation (4). For its construction we use product

$$(R_G - R)R^T \quad (5)$$

where R_G is the goal for the manipulator and R is the current rotation matrix for the manipulator. Substitution from Equation (1) yields

$$\left(\begin{bmatrix} l_x & m_x & n_x \\ l_y & m_y & n_y \\ l_z & m_z & n_z \end{bmatrix} - \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \right) \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}^T = \begin{bmatrix} \cdot & \cdot & \cdot \\ \Delta_z & \cdot & \cdot \\ -\Delta_y & \Delta_x & \cdot \end{bmatrix} \quad (6)$$

Finally, Equation (2) can be rewritten as follows

$$J(\theta_{i+1} - \theta_i) = \begin{Bmatrix} \Delta_x \\ \Delta_y \\ \Delta_z \\ \rho_x - a_{14} \\ \rho_y - a_{24} \\ \rho_z - a_{34} \end{Bmatrix} \quad (7)$$

Rearranging the last equation we get

$$\begin{Bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \end{Bmatrix}_{i+1} = \begin{Bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \end{Bmatrix}_i + h J^{-1} \begin{Bmatrix} \Delta_x \\ \Delta_y \\ \Delta_z \\ \rho_x - a_{14} \\ \rho_y - a_{24} \\ \rho_z - a_{34} \end{Bmatrix}_i \quad (8)$$

Equation (8) can be recognized as the Newton-Raphson iteration applied to six independent manipulator equations. h is the step correction factor (usually 1). Note that Jacobian matrix represents a mix of terms of position and orientation

information about the manipulator. This will be shown to have an adverse impact on sensitivity of numerical calculation to the change of units.

3.2 The Newton-Raphson method

For the function of one variable $f(x)$ the Newton-Raphson method is a simple feedback scheme $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$

which is depicted in Figure 6.

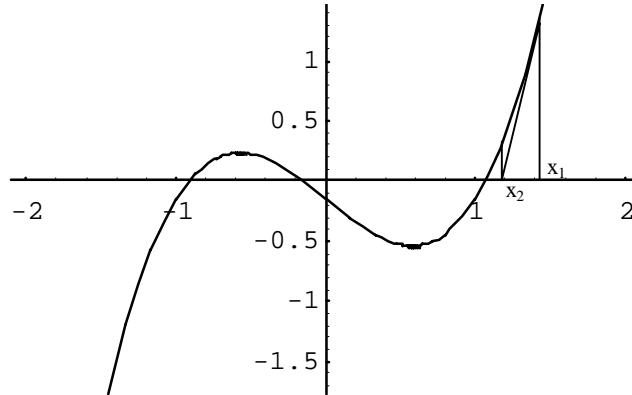


Figure 6. Graphical representation of the Newton-Raphson method.

By locally approximating $f(x)$ at x_1 with the tangent line and finding its intersection x_2 with the axis x for the next local approximation, Newton-Raphson rapidly converges to a “nearest” root. In view of this, Equation (8) is the Newton-Raphson iteration formula for 6 independent equations where the Jacobian matrix represents a “tangent” to hypersurface created by the equations.

3.3 Nonlinear elimination algorithm

Lanzkron et. al. [1996] described a new algorithm for solving systems of nonlinear equations. We believe that this algorithm can be used to reduce sensitivity to units in our inverse kinematics problem. Even though this algorithm is devised for “problems in which convergence seems to be interminably tedious and yet there is no evidence of ill conditioning (or singularity)” it reduces sensitivity to units by a significant margin. With this algorithm we were able to separate information about rotation and position of the Robotic arm in the Jacobian which positively affected the final state sensitivity. Before we describe details we will shortly describe the procedure.

Existing procedures for solving the system of nonlinear equations, besides plain Newton-Raphson method, usually revolve around Powell method (Jacoby et. al. [1972]). Consider the numerical solution of a system of nonlinear equations

$$f(X) = 0 \tag{9}$$

where $X = (x_1, x_2, \dots, x_n)^T$ and $f = (f_1, f_2, \dots, f_n)^T$. To find a solution to this system of equations we need to iterate

$$X_{k+1} = X_k - t_k f_k'^{-1} f_k \quad (10)$$

where $t_k \in (0, 1]$ is chosen to force

$$\|f_{k+1}\| < a \|f_k\| \quad (11)$$

for some $0 < a < 1$. When the norm of f becomes zero we obtain a solution to the system of equations.

To obtain the nonlinear elimination algorithm we need some additional steps. These steps involve computing intermediate solutions of subsystem of equations with the Newton-Raphson method. This is done with keeping some variable constant while iterating the rest. The procedure is reminiscent of eliminating variables in closed form except that we are doing it numerically. Consider $f(X)$ to be partitioned as

$$f(X) = (f_1(u, v), f_2(u, v))^T \quad (12)$$

where $X=(u, v)^T$ are partitioned unknowns. The proposition is to eliminate f_1 as an inner function. This leads to a block Jacobian matrix

$$f'(u, v) = \begin{bmatrix} f_{11}(u, v) & f_{12}(u, v) \\ f_{21}(u, v) & f_{22}(u, v) \end{bmatrix} \quad (13)$$

We first need to solve $f_1(h(v), v) = 0$ for $h(v)$ while keeping v constant. Differentiating f_1 as a function of v we obtain $h'(v) = -f_{11}^{-1} f_{12}$. Existence of the unique mapping $h(v)$ is guaranteed by the implicit function theorem and nonsingularity of f_{11} is required. Assuming now that $u = h(v)$ exists on an appropriate set, we attempt to solve the original system of equations rewritten as $f_2(h(v), v) = 0$ for v . While doing this we make sure that the norm of all of f decreases.

Algorithm can be concisely represented as

Input $X=(u_0, v_0)$ and f .

1. Chose f_1 equations splitting $f(x)$ into $f_1(u, v)$ and $f_2(u, v)$.
2. Given the initial v_0 . Solve $f_1(h(v_0), v_0) = 0$ for $h(v_0)$.
3. $k=0$
4. Repeat until convergence
5. Solve

$$\begin{bmatrix} f_{11} & f_{12} \\ f_{21} & f_{22} \end{bmatrix} \begin{bmatrix} \Delta u_k \\ \Delta v_k \end{bmatrix} = \begin{bmatrix} -f_1 \\ -f_2 \end{bmatrix}, \quad (14)$$

for Δv_k with f_{11}, f_{12}, \dots evaluated at $(h(v_k), v_k)$.

6. Find $t_k \in (0, 1]$ such that

$$\|f(h(v_k + t_k \Delta v_k), v_k + t_k \Delta v_k)\| < a \|f(h(v_k), v_k)\| \quad (15)$$

solving for $u = h(v_k + t_k \Delta v_k)$ of $f_I(u, v_k + t_k \Delta v_k) = 0$ is needed for every t_k .

7. $v_{k+1} = v_k + t_k \Delta v_k$

The authors of this algorithm believe that it will be helpful when the system of equations has some badly scaled components. The poor scaling is not in terms of normal linear miscalling, rather it is due to nonlinearity of the problem. However, our investigation shows that this algorithm is also successful in reduction of sensitivity to units. For details discussion of this algorithm refer to Lanzkron et. al. [1996].

4. SOLVING INVERSE KINEMATICS WITH NONLINEAR ELIMINATION

To implement nonlinear elimination algorithm on inverse kinematics problem of 6R robot manipulator we first need to choose the subsystem of equations and variables for inner loop solving with the Newton-Raphson method. This choice is an important one and this separation of variables is the reason for using this method. The main problem with applying the Newton-Raphson method to kinematic problems, as it will be seen from our experimental results, is that it is sensitive to partial scaling of the equations. Such scaling is changing the condition number(s) of the Jacobian matrix at each step which also changes a final state sensitivity of the method. This is undesirable and Nonlinear Elimination removes to large degree this problem.

We believe that good candidates for inner loop solving are the last three joint variables and the first three equations in the system. This choice is motivated by an attempt to remove mixing of scaled equations in the Jacobian matrix. By doing this, we ensure that troublesome scaling does not affect sensitivity when solving the subsystem of equations with the Newton-Raphson method. This is due to the fact that with the change of units the first three equations do not modify condition numbers of the block matrices in the Jacobian matrix.

Let the joint variables be denoted with θ , the first three equations with f_1 and the last three with f_2 . The algorithm steps would proceed as follows.

1. Given initial angles θ_1^0, θ_2^0 and θ_3^0 solve $f_1(\theta_1^0, \theta_2^0, \theta_3^0, h(\theta_1^0, \theta_2^0, \theta_3^0)) = 0$ for $h(\theta_1^0, \theta_2^0, \theta_3^0)$
2. $k = 0$
3. Repeat until convergence. Solve

$$J^k \begin{Bmatrix} \Delta\theta_1^k \\ \Delta\theta_2^k \\ \Delta\theta_3^k \\ \Delta\theta_4^k \\ \Delta\theta_5^k \\ \Delta\theta_6^k \end{Bmatrix} = \begin{Bmatrix} \Delta_x^k \\ \Delta_y^k \\ \Delta_z^k \\ \rho_x - a_{12}^k \\ \rho_y - a_{24}^k \\ \rho_z - a_{34}^k \end{Bmatrix} \quad (16)$$

for $\Delta\theta_i^k$ where J^k is Jacobian matrix; f_1 and f_2 are evaluated at $(\theta_1^k, \theta_2^k, \theta_3^k, h(\theta_1^k, \theta_2^k, \theta_3^k))$.

4. Find $t_k \in (0, 1]$ such that

$$\|f(\theta_i^k + t_k \Delta\theta_i^k, h(\theta_i^k + t_k \Delta\theta_i^k))\| < a \|f(\theta_i^k, h(\theta_i^k))\| \quad i = 4..6 \quad (17)$$

solving for $u = h(\theta_i^k + t_k \Delta\theta_i^k)$ of $f_1(\theta_i^k + t_k \Delta\theta_i^k, u) = 0$ is needed for every t_k .

5.

$$\begin{Bmatrix} \theta_1^{k+1} \\ \theta_2^{k+1} \\ \theta_3^{k+1} \end{Bmatrix} = \begin{Bmatrix} \theta_1^k \\ \theta_2^k \\ \theta_3^k \end{Bmatrix} + t_k \begin{Bmatrix} \Delta\theta_1^k \\ \Delta\theta_2^k \\ \Delta\theta_3^k \end{Bmatrix} \quad (18)$$

The computation in step 4 implicitly requires a solve of the f_1 equation. This step is potentially expensive, since every new choice of t_k requires a solve of the f_1 equations. It should be noted that in this step we are trying to reduce overall norm of f . This is necessary since $f_1 \neq 0$. Long time should not be spent trying to get solutions for t_k 's because smaller t_k will get an answer faster.

5. EXPERIMENTAL RESULTS

In this section we present a series of numerical experiments designed to demonstrate sensitivity to change of units. The experiments were done using Turbo C++ 3.0 on 486DX2-66MHz Windows 95 environment.

5.1 Sensitivity to change of units

For the manipulators in 2.1 and 2.2 the links are chosen to be the same lengths. We have later varied them accordingly in the programs. For the other two manipulators we borrowed data from Tsai and Morgan [1985] (Denavit-Hartenberg notation).

i	a_i (unit)	d_i (unit)	α_i (deg)
1	0.0	0	-90
2	1	0.345	0
3	-0.047	0	90
4	0	1	-90
5	0	0	90
6	0	0.13	0

Table 1. Linkage parameters (Puma)

i	a_i (unit)	d_i (unit)	α_i (deg)
1	0.5	0.1875	80
2	1	0.375	15
3	0.125	0.25	120
4	0.625	0.875	75
5	3.125	0.5	100
6	0.25	0.125	60

Table 2. Linkage parameters (6R gen. man.)

The hand position and orientation for these two examples are given in Table 3. ρ is the position vector and \mathbf{l} , \mathbf{m} , and \mathbf{n} are the orientation direction vectors.

Given vect.	x - comp.	y - comp.	z - comp.
ρ	0.22441776	0.71549788	0.79551628
\mathbf{l}	-0.71511545	0.6515032	0.25328538
\mathbf{m}	-0.69899036	-0.66895464	-0.25280857
\mathbf{n}	0.00473084	-0.35783135	0.93377425

Table 3. Hand position and orientation.

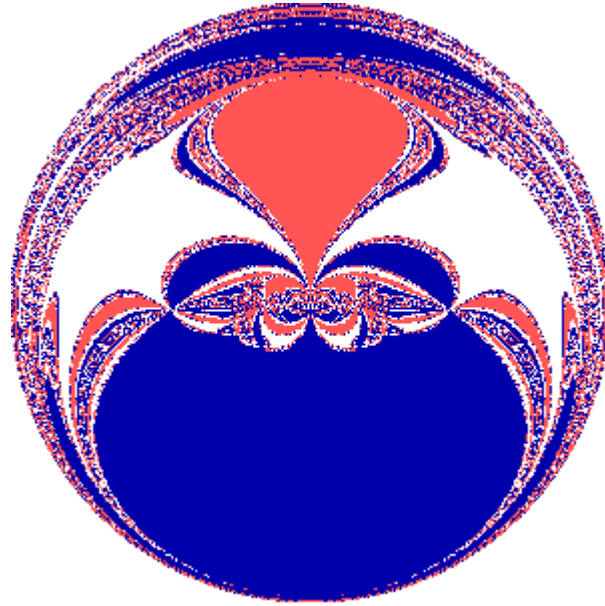


Figure 7. The basins of attraction for 2-link manipulator.

Primary method of our investigation was “numerically” comparing “basins of attraction” (Jovanovic and Kazerounian [1996]) using the Newton-Raphson and Nonlinear Elimination methods. Basins of attraction are the sets of all initial points that lead to any of the configurations by means of numerical iteration. A typical basin of attraction for 2D manipulator is shown in Figure 7. For details of its construction refer to Jovanovic and Kazerounian [1996]. Analysis of the basins of attraction is a standard procedure in Dynamical Systems Theory which produces interesting pictures depending on the sets of equations that is being solved. To determine sensitivity to change of units we proceeded as follows. For each initial point that was taken from the joint space of the manipulator, we determined its destiny through iteration. Then we compared this final state of iteration to a final state of iteration for the same manipulator but with different units. If the two final states were not the same, sensitivity had an effect. We recorded such occasions as well as the average calculation execution time. For the unit change we have chosen to multiply all the lengths of the manipulator with the multiplication factors 10, 1000 and 100000 because such numbers significantly change properties of Jacobian matrix due to excessive scaling of its three last rows. The referent unit factor to which all other unit factors were compared was 1.

The following tables show our experimental results.

Mult. Factor	10	1000	100000
2D	0	0	0
3D3joint	0	0	0

Table 4. Number of different states. Newton-Raphson step 1.

Step=1	fact. 1 time in s	factor 10	factor 1000	factor 10000	factor 100000
2D3joint	.0033	41	.0033	35	.0033
6R gen. man.	.074	3319	.0750	3324	.0769
Puma	.0573	381	.0560	401	.0554

Table 5. Number of different states reached with times of calc. NR step 1.

step=.5	fact. 1 time in s	factor 10	factor 1000	factor 10000	factor 100000
6R gen. man.	.081	664	.0795	649	.0792
Puma	.072	49	.0703	55	.0696

Table 6. Number of different states reached with times of calc. NR step 0.5.

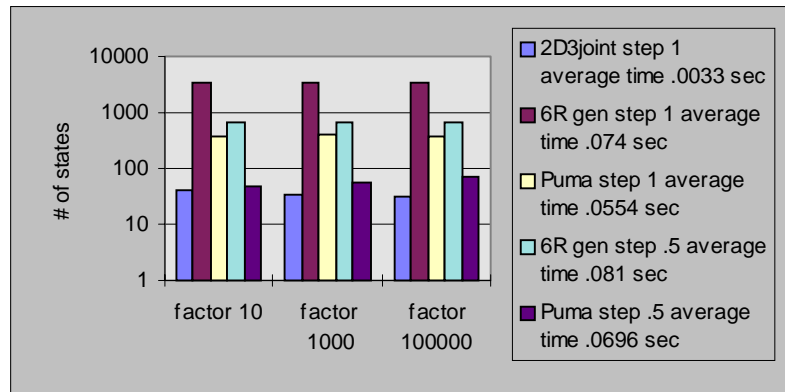


Figure 8. Graphical representation of Table 5 and Table 6.

factor 1000 for	Puma		6R gen. man.	
with step 1.	401	.0554	3324	.0769
with step .9	276	.0536	2582	.0698
with step .8	165	.0502	1919	.0657
with step .7	92	.0524	1373	.0663
with step .6	51	.0580	928	.0706
with step .5	55	.0696	649	.0792
with step .4	45	.0939	417	.0912
with step .3	20	.117	275	.1249
with step .2	5	.168	106	.1824

Table 7. Trend in change of different states due to change in NR step.

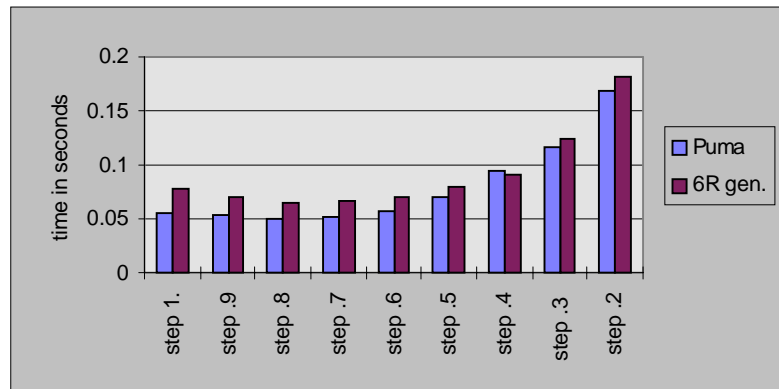


Figure 9. Graphical representation of converging time in Table 7.

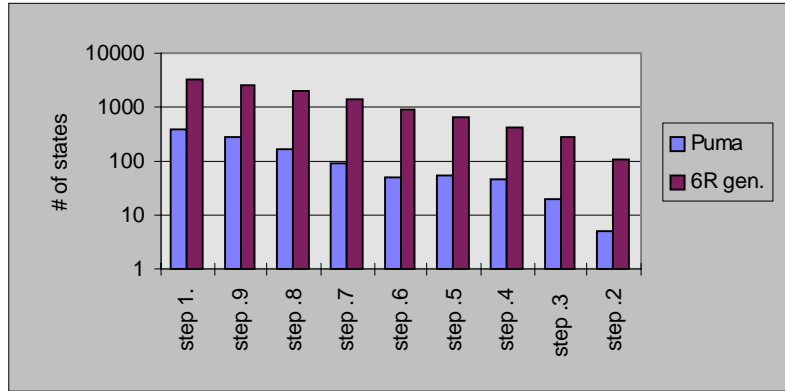


Figure 10. Graphical representation of number of states in Table 7.

Mult. Factor	10		1000	
	6R gen. man.	460	.75	433
Puma	0	.38	0	.38

Table 8. Number of different states using Nonlinear Elimination.

For each manipulator in the multiplication factor column there are two numbers. The first represents the number of different states reached and the second is the average time of convergence in seconds for the particular method.

The calculation was done with the Newton-Raphson method where maximum number of loops was 100 and tolerance was 10^{-5} . For 2D manipulator 10000 points were distributed in its joints $[0, 2\pi]$ covering both its independent angles. For the manipulators which had more than two degrees of freedom, initial 100 by 100 points were distributed along two joint space angles (second and fourth for Puma and 6R gen. man, and first and second for 2D3joint and 3D3joint) while the rest of the joint angles always started iteration with value 0.5. 6R and Puma manipulator were examined in this way because a large amount of time is needed to examine all the points in the space of all joint angles. We believe that with this choice we obtained good enough picture of the sensitivity problems. Note that these results exclude sensitivity due to differentiation since the exact form of Jacobian was used in each case.

5.2 Sensitivity to change of base coordinate system

In some instances it may be desirable to describe a mechanism in a different base coordinate system. Such description certainly has no effect on physics of the problem or the set of solutions for that mechanism: however, a change in sensitivity will occur for the same basic reasons as in the case of change of units. To investigate this claim experimentally, we described the first linkage and the hand of our manipulators in a new coordinate system (Figure 11). Then we again attempted solving inverse kinematics problem.

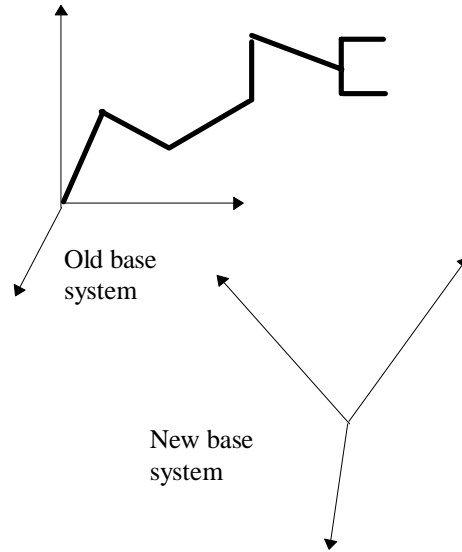


Figure 11. Description of the manipulator in new base coordinate system.

Calculations were performed using the Newton-Raphson method. Once again we compared the final state convergence of the method between the same mechanisms described in different base coordinate systems. For each initial point taken from the joint space for the manipulator, the destiny of its numerical iteration was determined. The final state of iteration was then compared between the manipulator described in the old base system and the manipulator described in the new base system. If the two final states were not the same, sensitivity had an effect. These occasions were recorded. Naturally, for the same manipulator the same set of solutions must be obtained regardless of which base coordinate systems are used; however, as we expected, sensitivity was far from been predictable. The results can be depicted in Table 9 and Table 10.

rotation angle about the screw axis	30 deg	60 deg
2D	0	0
2D3joint	0	0
3D3joint	0	0

Table 9. Number of different states with the new base system.

Angle	30 deg		60 deg	
NR step	1	.5	1	.5
6R gen. man.	7690	6172	7670	6421
Puma	1306	575	1395	584

Table 10. Number of different states with the new base system.

The new base coordinate system was obtained by rotating and translating the old coordinate system about an arbitrary screw axis positioned through $p_x=1$, $p_y=2$ and $p_z=1$ in the direction of $u_x=1$, $u_y=1$ and $u_z=1$. The angle of rotation was taken to be an independent parameter.

It should be noted that, in this case, because of our direct kinematic formulation (Equation (2)), translation of the coordinate system along the screw axis does not affect the sensitivity. Of only significance is the rotation. This is because Equation (2), which was used to form the Newton-Raphson method, involves only free body vectors for which the relative locations of the origins does not enter into calculation. Therefore any description of the free body vectors in the new base system requires only rotation matrix that was used to rotate the old coordinate system about the screw axis.

6. ANALYSIS OF RESULTS

Our primary goal was to demonstrate, explain and reduce sensitivity to change of units when using numerical methods for solving the inverse kinematics problem. This sensitivity is undesirable and in short it is due to chaotic nature of iterations. It basically means that we get different behavior of our numerical method when representing the lengths of linkages of the manipulator in different unit systems (for example, British system versus SI system).

We first applied the above mentioned procedure to 2-link and 3D3joint manipulators examining 10000 initial points in the $[0, 2\pi]$ joint space (Table 4), and we determined that there was absolutely no sensitivity to change of units for the manipulators. Mathematical models for these manipulators do not involve rotation information. Why this is significant it will be explained in the following section. Quite a different result was obtained for the other manipulators, as is depicted in Table 5 and Table 6. Mathematical models for these manipulators do involve rotation information and a jump in sensitivity to units is apparent. By using the Newton-Raphson method with step equal to 1 we obtained approximately .3% for 2D3join, 33% for 6R gen. man and 3.8% for Puma of changes of final solutions as the scale changed. Average converging time for iterations in seconds was .0033 for 2D3join, .074 for 6R gen. man and .056 for Puma. When the Newton-Raphson step was set to .5 we obtained approximately 6.6% for 6R gen. man and .6% for Puma of changes of final solutions. Reduction in changes of final solutions is significant and it will be explained in the following section. The observation should be made that the trend of changes remained constant for all three multiplication factors. Table 8 shows that the biggest reduction in sensitivity was obtained when we used the Nonlinear Elimination algorithm. For 6R general and Puma manipulators the sensitivity to units was 4.5% and 0% respectively with the average calculation time of .75 and .38 seconds. Again, the trend remained constant.

Table 9 and Table 10 represent the result of the experiments designed to test the sensitivity to change of base coordinate system. In Table 9 it could be seen that there is absolutely no existence of sensitivity for 2D, 2D3joint and 3D3joint manipulators. However, Table 10 shows that for 30 and 60 degree angle 6R gen. manipulator gives 77% sensitivity for full and 61% sensitivity for half NR step, while Puma gives 13% sensitivity for full and 6% for half NR step. Again the reduction of sensitivity is apparent and it will be explained in the following sections.

6.1 Reasons for sensitivity

What is the reason for different behavior of the same numerical procedure when, for example, using meters instead of centimeters to quantitatively define the parameters? We believe that the cause for this behavior is rooted in inherited chaotic nature of iterations (Jovanovic and Kazerounian [1996]). When a mathematical model is subject to iteration, basins of attraction can assume a very peculiar shape. Sometimes they exhibit quite an amazing structure depending on certain conditions and, on the other hand, sometimes they could look quite random. However, there is nothing random here and the whole process is a result of a deterministic procedure. This process shows characteristics of chaos for which the main landmark is sensitivity to initial conditions. Such sensitivity is most prominent at the boundaries of basins of attraction. In other words, when the iteration is carried over for two initial points located at the boundary of basins of attraction and on a very small distance from each other, it may happen that the final points reached through iteration are on a very great distance. This phenomenon has been investigated in the past two decades with the aid of computers and it is only now better understood (Devaney [1989]). When examined from this point of view our results are easier to explain. With the change of units in the mathematical model nothing really physically changes, but properties of Jacobian matrix are altered as well as the scaling of lower three equations. This certainly does not have an effect on the set of all possible solutions of the system, but it has an effect on the shape of boundaries of basins of attraction. It shifts the boundaries in such a way that certain number of points would not yield the same structure of configurations as it would before the change of units occurred. Why? Because the *highest sensitivity* of iterations is located at these boundaries, as we already explained before, and any slight change in our mathematical model will change the shape of these sensitive areas. The number of points that gives different final configurations is hard to predict and it should be dependent on the amount of shift of the boundaries. Close examination of Table 5 will reveal how sensitivity affects three different manipulators. The highest sensitivity is obtained for the general 6R manipulator and it is about 33% of the scanned work space area. On the other hand Puma manipulator gives only about 4% sensitivity to units. These numbers are quite different from each other and remains for future investigations to determine the cause for this discrepancy. One question we could not answer at this stage which is why we obtained similar level of sensitivity for different factors. In fact sensitivity was not dependent on the size of the multiplication factor and even factor 1.00001 would give the same level of sensitivity. Also sensitivity remains “constant” between any factors. At this moment we can only speculate about such a result and we leave it for further investigation.

The above reasoning could also be used to interpret sensitivity due to the change of the base coordinate system. The same principle about shift of the boundaries of basins of attraction applies here. The only difference is that in this case this shift, as we already mentioned, occurs due to multiplication of vectors in Equation (2) by the rotation matrix instead of the

partial scaling of the equations. That this action is more drastic is apparent by observing the levels of sensitivity in Table 9 and Table 10 and comparing them to the levels of sensitivity in Table 5 and Table 6. The reason for no sensitivity in Table 9 is because the rotation matrices cancel in the Newton-Raphson method due to a reduced number of equations for 2D, 2D3joint and 3D3joint manipulators.

6.2 Treatment of sensitivity

To deal with the issue of sensitivity in computational mechanics it is necessary to understand the method used for solving systems of equations and that is certainly the Newton-Raphson numerical procedure. It occupies a very significant role in almost every numerical solver due to its simplicity and ease of application. However, its simplicity is misleading. The theory of Chaotic Dynamical Systems has shown that we had known little about this powerful method. Its chaotic nature is now being thoroughly investigated among mathematicians and it is just a beginning of understanding of the method. Nevertheless, today we understand some of its intricacies and it will help us deal with the sensitivity in computational mechanics.

From the classical books on Numerical methods it is well known that the Newton-Raphson is quadratically convergent method and that if initial points are chosen “sufficiently close” to a solution the method will converge to the solution. However, this is just the basic behavior of the method and there are much more finer points to study. In fact, every root in the Newton-Raphson method is a so called super attractive point (Jovanovic and Kazerounian [1996]) to which iteration converges. This is the reason for quadratic convergence of the Newton-Raphson method.

The attractiveness of the roots in the Newton-Raphson method can be best viewed graphically by construction of a cobweb depicted in Figure 12. Here only one degree of freedom system is shown and N represents an iterative Newton-Raphson function $N(x_k)=x_k-f'(x_k)^{-1}f(x_k)$.

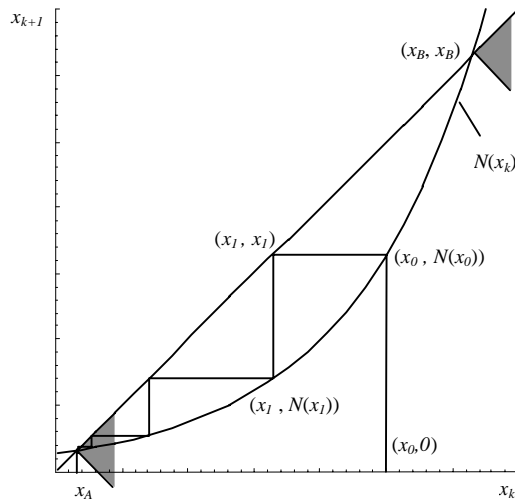


Figure 12. Graphical representation of iterating process.

The cobweb is a set of connected line segments beginning at an initial point x_0 . It consists of one vertical line segment from $(x_0, 0)$ to $(x_0, N(x_0))$ and one horizontal line segment from $(x_0, N(x_0))$ to (x_1, x_1) . Its further parts also consist first of one vertical and then one horizontal line segments which all end after infinitely many iterations in the fixed point. The graph reveals that point x_A is attractive and x_B is repelling under N , because $|(N)'(x_A)| < 1$ (vicinity of $N(x)$ around x_A is within shaded region, Figure 12) and $|(N)'(x_B)| > 1$ (vicinity of $N(x)$ around x_B is outside of shaded region). In fact, every intersection of N with 45 degrees line within the shaded region represents a root of f . Further, the slopes at these intersections must be equal to zero. This is the reason for quadratic convergence of the method (for details refer to Jovanovic and Kazerounian [1996]). Now, what is really important here is that this quadratic convergence or super attractiveness of the roots has a very high impact on the sensitivity to change of units in our examples. This is depicted in Table 6 and Table 7. We repeated our examples with different Newton-Raphson steps and a decrease of sensitivity was apparent. Why would that be the case? With the step h less than 1, the Newton-Raphson is not a quadratically convergent method anymore. At best we have linear convergence. In other words, the roots are not super attractive. They are now only attractive points in the discrete state space. Such reduction of the step smoothes the boundaries of attraction and consequently reduces sensitivity which now serves as a verification to our statement that the sensitivity is rooted in the chaotic behavior at the boundaries of attraction.

Chaotic behavior seems to be absent in 2D and 3D3joint examples, the truth is quite the opposite. The chaotic behavior close to the boundaries is still there. However, it is only that the mathematical equations for these manipulators do not experience partial scaling of the equations (mixing of orientation and position does not occur) and the boundaries remain unchanged. Therefore, we do not see sensitivity to change of units (or, in other words, shift of the boundaries). This property provides a lead for other means to reduce sensitivity besides cutting the step in the Newton-Raphson method. It suggests that by separating orientation and position information in the mathematical model we could reduce the effect of partial scaling of equations problem. For this reason we selected the Nonlinear Elimination method as an alternative to the Newton-Raphson method.

To understand how mixing has an adverse effect it is necessary to examine Equation (8). Here it is necessary to find J^{-1} . This is an involved mathematical operation which scrambles two types of information about the manipulator - position and orientation. Due to a nonlinear nature of the problem separation is not possible in the closed form. It is only possible to numerically separate the equations which can be done with Nonlinear Elimination procedure. This procedure, as explained in section 4, is based on inner loop solving of the subsystem of equation and globally reducing the norm of the whole system of equations. If for the subsystem of equations we select those that do not have partially scaled components a certain degree of separation should occur. A good candidate is the upper right 3 by 3 matrix in J because it does not contain the position information. Now, solving this subsystem of equations is an attempt in orienting the manipulator in the work space without paying attention to positioning. A degree of success of such an approach is depicted in Table 8. The sensitivity to units for Puma manipulator was completely removed while for 6R general manipulator it was reduced to about 4% from its previous 33%. These results fulfill the expectations and serve as verification to existence of the mixing

problem in the computation mechanics. However, one problem with this method was an increased number of initial conditions that did not lead to a solution, about 15%-17%, while with the Newton-Raphson method was not more than 5%. We cannot explain this here, and in fact nonconvergence remains one of the major problems in solving systems of equations. We believe that it is also related to super attractiveness of the Newton-Raphson method.

Our experimental results contain the average calculation time for each numerical procedure. We provided it in view of possibility for practical application so that a user can weigh the tradeoffs. It appears that the simple step cutting in the Newton-Raphson method is the most successful in terms of sensitivity and computational speed. It also significantly reduces the number of nonconverging initial conditions due to smoothening of the boundaries of basins of attraction. One should remember, when using the step cutting procedure, to give the Newton-Raphson method enough time to converge by increasing the maximum number of loops.

7. CONCLUSION

We have presented a study on the sensitivity to units and change of coordinate systems in computational kinematics where both rotation and displacement are included. We demonstrated clearly that this sensitivity is present as well as how it affects a numerical procedure when solving the system of loop closure equations is attempted. The contributions of this paper are 1) discovery that the metric and coordinate system induced sensitivity is rooted in the chaotic nature of iterations close to the boundaries of basins of attraction and 2) developed methods to reduce this sensitivity. A number of numerical experiments were used to demonstrate the chaotic behavior at boundaries. The results led to conclusion about possibility of smoothening the boundaries or separating information of position from that of rotation in Jacobian matrix. Two methods proved to be successful, cutting step in the Newton-Raphson method and the Nonlinear Elimination. The former is an attempt in smoothening the boundaries of attraction, while the latter is an attempt in separating orientation and position information. As usual, there are questions that remain unresolved. Why did the level of sensitivity remain more or less constant for different multiplication factors? How can degree to sensitivity be predicted? These and other questions require further investigation.

8. REFERENCES

- Devaney, R. L. (1989), *An Introduction to Chaotic Dynamical Systems*, 2nd ed. , Addison-Wesley, Redwood City, CA.
- Gupta, K. C., (1993), "Computational Kinematics", chapter of the book: *The first 40 years of modern Kinematics*, J. Wiley, book editor: A. Erdman
- Jacoby, S.L.S., Kowalik, J.S., and Pizzo, J.T., (1972), *Iterative methods for Nonlinear Optimization Problems*, Prentice-Hall, Englewood Cliffs, New Jersey.
- Jovanovic, V., (1997), "*Identifying, Utilizing and Improving Chaotic Numerical Instabilities in Computational Kinematics*", Ph.D. Thesis, The University of Connecticut.
- Jovanovic, V. T. and Kazerounian, K., (1995), "Chaos in Computational Kinematics", Proceedings of 4th National Applied Mechanisms and Robotics Conference, Ohio, Cincinnati, December 1995.

Jovanovic, V. T. and Kazerounian, K., (1996), "Using Chaos to Obtain Global Solutions in Computational Kinematics", Proceedings of The 1996 ASME Design Eng. Tech. Conferences, August 18-22, 1996, Irvine, California. Also to appear in *ASME Journal of Mechanical Design*.

Lanzkron, P. J., Rose D. J., and Wilkes J.T., (1996), "An Analysis of Approximate Nonlinear Elimination", *SIAM Journal of Scientific Computing*, Vol. 17, No. 2, pp. 538-559, March 1996

Orin, D. E. and Schrader, W. W., (1984), "Efficient Computation of the Jacobian for Robot Manipulators," *International J. Robotics Research*, vol. 3, no. 4, pp. 66-75.

Raghavan, M., and Roth, B., (1995), "Solving Polynomial Systems for the Kinematic Analysis and Synthesis of Mechanisms and Robot Manipulators", 50th Anniversary Design Issue of *ASME Journal of Mechanical Design*, Vol. 117, pp. 71-79

Rheinboldt, W. C., (1974), *Methods for Solving Systems of Nonlinear Equations*, SIAM Philadelphia, PA.

Tsai, L.-W., and Morgan, A. P., (1985), "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*, Vol. 107, pp. 189-200.