Spring 1995

A general method for the inverse kinematics of rotational displacements in spatial mechanisms

John D. Kliminski
New Jersey Institute of Technology

UMI Number: 1374927

ABSTRACT

A GENERAL METHOD FOR THE INVERSE KINEMATICS OF ROTATIONAL DISPLACEMENTS IN SPATIAL MECHANISMS

by
John D. Kliminski

An iterative technique was developed to solve the inverse kinematics problem for the joint rotations in both closed-loop and open-loop spatial mechanisms and robotic manipulators in any prescribed configuration. The method is based on fixing one link in space and maneuvering the other links to form a closed chain, following an approximation of the actual physical assembly of the mechanism. In order to apply the same principle to both types of mechanisms, an open-loop mechanism was modeled as a closed-loop mechanism by creating a fictitious fixed link in the free space between the base and the end of the chain of links. A computer program was written to test the validity of the algorithm. The results of several examples and comments on the success and limitations of the method are included. Possible applications and suggestions for future work are proposed.

A GENERAL METHOD FOR THE INVERSE KINEMATICS OF ROTATIONAL DISPLACEMENTS IN SPATIAL MECHANISMS

by
John D. Kliminski

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Mechanical Engineering

Department of Mechanical and Industrial Engineering

May 1995

APPROVAL PAGE

A GENERAL METHOD FOR THE INVERSE KINEMATICS OF ROTATIONAL DISPLACEMENTS IN SPATIAL MECHANISMS

John D. Kliminski

Dr. Ian S. Fischer, Thesis Advisor — Date
Associate Professor of Mechanical Engineering, NJIT

Dr. Zhiming Ji, Committee Member — Date
Assistant Professor of Mechanical Engineering, NJIT

Dr. Kwabena A. Narh, Committee Member — Date
Assistant Professor of Mechanical Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: John D. Kliminski

Degree: Master of Science in Mechanical Engineering

Date: May 1995

Undergraduate and Graduate Education:

- Master of Science in Mechanical Engineering
  New Jersey Institute of Technology, Newark New Jersey, 1995

- Bachelor of Science in Mechanical Engineering
  New Jersey Institute of Technology, Newark New Jersey, 1992

Major: Mechanical Engineering

I would like to dedicate this thesis to those special few whom I've truly called my friends for their love, loyalty, and understanding.

ACKNOWLEDGMENT

The author would like to express his sincere gratitude to his advisor, Dr. Ian S. Fischer, for his guidance and assistance in the course of this research.

The author also appreciates Dr. Zhiming Ji and Dr. Kwabena A. Narh for their cooperation in serving as members of the review committee.

The author would like to acknowledge Mr. Stephen Altenderfer for his early encouragement of his pursuit of science.

Finally, the author would like to express his gratitude to his parents, John J. and June M. Kliminski, for their support and patience throughout this endeavor.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Symbol	Meaning
n	The number of links in the mechanism.
α_i	The angular length (twist) of link i.
θ_i	The angular joint displacement at joint i.
$\mathbf{k}_i$	The axis of joint i.
$\mathbf{V}_i$	A vector in the direction of the joint axis of link i.
$\mathbf{V}_{in}$, $\mathbf{V}_{out}$	The terminal vectors.
σ	The miss angle.
σ_c	The correction angle.
$\mathbf{M}$	The miss axis.
ϕ_i	The angle between $\mathbf{V}_i$ and $\mathbf{M}$.
s	The index of the joint axis closest to the miss axis.
U	Coordinate transformation matrix.
R	General rotational transformation matrix.
α, β, γ	ZYX Euler angles.
E	Euler transformation matrix.

CHAPTER 1

INTRODUCTION

1.1 The MAIM Method

This thesis introduces a novel iterative method for determining the rotational displacements in a spatial mechanism. This technique is called the Miss Angle Iteration Method, abbreviated MAIM, and has been developed as the first part of a two-stage solution to the general spatial inverse kinematics problem. The MAIM method subdivides the general problem to remove the effects of the translational displacements from consideration, allowing the determination of the correct rotational joint displacements required to achieve closure of the kinematic chain. Given the desired orientation of one of the links, the method maneuvers the system of unconstrained links to connect to the specified link in an approach paralleling the actual physical assembly of the mechanism in such a configuration. The MAIM method provides the solutions for the rotations at the joints, to be used as knowns in a later routine to compute the translations.

1.2 A Brief Commentary on Kinematics

The science of kinematics comprises the study of mechanisms in all their complex and diverse forms. Even though considerable work has been accomplished, the matter of inverse kinematics, determining the joint displacements corresponding to a desired configuration, continues to confound engineers and resist all but the most complicated and limited solutions. An effective means to overcome the level of difficulty commonly encountered is required.

1.3 Definition of a Mechanism

The term "mechanism" refers to a mechanical device for the purpose of transferring motion or force from a source to an output (Sandor and Erdman, 1984). Mechanisms are comprised of links connected by joints. Presuming that the links are rigid, the displacements at the joints determine the configuration of the mechanism. These joints can take many forms: revolute, prismatic, cylindrical, spherical, and others, allowing motion in one, two, or three directions. In each case, this motion is some combination of rotation and translation. Each of these possible motions at a given joint is referred to as a degree of freedom (Sandor and Erdman, 1984).

Mechanisms are classified according to their construction. Closed-loop mechanisms are arranged such that their links connect to form a closed kinematic chain. Open-loop mechanisms, by comparison, do not form a closed chain, but rather feature a free end able to move to any position within the reach of the linkage. Each of these types can operate in two or three dimensional space (Nikravesh, 1988).

Planar mechanisms, as their name suggests, operate entirely in one plane or in parallel planes in the case of the necessity to overlap links. As such, they are somewhat limited and by those limitations considerably simpler to analyze. Constrained to move in only two dimensions, the mathematical analysis of their displacements can be tedious but is tenable. Most of this area has been worked to satisfaction at this time. A new level of complexity appears when adding the option to move in the third dimension.

Spatial mechanisms are useful for certain applications and are becoming more prominent. With the advent of such machines as robotic manipulators, these mechanisms have proven to be useful due to the freedom and versatility provided by their ability to move in space. The analysis of these three-dimensional linkages is considerably more complicated due to the larger number of independent variables involved - one for as many as six possible degrees of freedom for each joint in a mechanism with potentially unlimited numbers of joints.

1.4 The Positioning Problem

When a mechanism possesses a large number of joints, it becomes more versatile but also more complex, having more degrees of freedom and therefore more variables to work with. Since each joint contributes one or more variables for consideration and the effect of the alteration of each of these elements on the overall position of the mechanism is contingent on the current configuration of the others, the task of setting the joint variables to achieve a desired final position requires the careful adjustment of many mutually influential factors. Thus, it is necessary to properly specify many parameters in order to put a mechanism in a certain position. The problem remains of how to determine the correct joint displacements for such a position.

In the past, the analysis of mechanisms was almost entirely dependent on graphical techniques but recent advances in digital computers have made analytical solutions more practical (Doughty, 1988). Iterative techniques based on matrix algebraic solutions are now commonly used (Fu, Gonzalez, and Lee, 1987). Unfortunately, due to the large number of variables involved, this approach often results in cumbersome equations which are fraught with hindering complications and ultimately possess multiple equally valid solutions to further confound the user (Sandor and Erdman, 1984). Even the most basic mechanisms possess equations which are highly non-linear and transcendental.

1.5 Options to Determine Position

A convention for notation introduced by Denavit and Hartenberg (1955) has become standard. The Denavit-Hartenberg notation uses variables to specify the geometries of the links and the displacements at the joints. Local coordinate frames are established on the end of each link and the joint displacements are measured from their axes. Mathematically, coordinate transformation matrices made up from these parameters can be used to specify the relationship between the frames fixed on neighboring links. Successive multiplication of these matrices will provide a relationship for any or all of the kinematic chain (Fischer; 1993).

The general coordinate transformation matrix is expressed in terms of the Denavit-Hartenberg joint parameters as

$$ {}_{i+1}^{\ i}U = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix} $$

where angle α_i represents the twist of link i and angle θ_i represents the angular displacement at joint i. The overall configuration of a mechanism can be expressed mathematically in terms of its joint variables as a complete product of these transformation matrices.

The resulting equations can be solved in either direction: using known joint displacements to compute the current output position or, given the desired output position, determining the required joint variables. These processes are more commonly known as "forward kinematics" and "inverse kinematics" respectively.

Establishing the chain of transformation matrices and forward-substituting with the Denavit-Hartenberg parameters expediently leads to the forward kinematics solutions. The inverse kinematics solution, however, is considerably more complicated. Unlike forward kinematics, no standard method exists so far for solving the position equations in reverse. This paper endeavors to present a new approach to this problem which has certain advantages over existing techniques.

1.5.1 Forward Kinematics

The forward kinematics solution is relatively simple, although not particularly useful in most applications. Forward kinematics requires only knowing the current joint variables and from those and the chain of transformation matrices it is possible to work through the system of mathematical expressions to find the current final position of the mechanism. All equations lead directly to this solution and all of the necessary variables are provided to carry out the calculations. Even though the algebra may be lengthy, it is guaranteed that it can be performed. What makes forward kinematics so easily solvable is that many variables lead to one result.

1.5.2 Inverse Kinematics

In contrast, inverse kinematics is considerably difficult while the results are essential in many practical applications. The inverse kinematics problem forces a user to start with the final position and try to work back to solve for all of the joint variables. Consequently, little data must be used to solve for many unknowns. In addition to this problem, the equations themselves are extremely complicated, with even the most simple systems being highly non-linear and transcendental. Ultimately there is no guarantee that a solution even exists, if for example the specified position is unreachable in reality, or that it can be obtained using the chosen or any mathematical approach, due to a wide variety of computational problems. A solution generally does not exist in closed form, and the multiple solutions are indistinguishable from each other until the procedure is completed.

All presently proposed approaches to inverse kinematics are lengthy, complicated, and highly specific for each application. Even the solutions for planar mechanisms are tedious enough to be undesirable to work through and performing the inverse kinematics for a spatial mechanism tends to be a complicated affair. Obviously, even when performed by a computer, these calculations cannot be carried out in "real time" as is necessary for many industrial applications. As such, the calculations to obtain highly desirable results are extremely undesirable if not outright impossible to perform.

The inverse kinematics problem has yet to be solved in a satisfactory way. Even the simplest mechanisms represent considerable challenges. Many techniques have been developed over the years for various cases.

Tsai and Morgan (1985) were able to reduce the equations for a five or six degree of freedom mechanism to a simultaneous system of eight second-order polynomials. To solve these they applied a generic continuation computer algorithm.

Pennock and Yang (1985) set up a systematic approach using dual-numbers specifically for mechanisms with specially designed geometries. They proceeded to solve the matrix equation of the kinematic chain for each special case individually. While this method, like the others, is successful, it would seem to be too specific to be useful in general applications.

Lee, Woernle, and Hiller (1991) were able to solve the inverse kinematics problem analytically for the general 6R manipulator. Aside from the fact that their solution required a 16th-degree polynomial, even they admit that each mechanism must be solved uniquely and that their method cannot be applied to all mechanisms. They also refer to the fact that most commercially available robots are designed with special geometry to allow the inverse kinematics problem to be solvable by conventional methods. Aside from the geometry, redundancy is desirable in most designs as a factor of safety, but such additional links bring additional levels of complication to their analysis and so any elements in a mechanism which are not essential are discouraged. From this it seems obvious that the development of a method which is not limited to special cases for convenience would allow more diverse design of manipulators.

Crane, Carnahan, and Duffy (1991) developed an analytical inverse kinematic solution for a seven degree of freedom robotic arm proposed by NASA for use as a manipulator on a space station by specifying one of the joint variables as prescribed to reduce it to one of three sets of six degree of freedom chains, which could be solved by any of the diverse but difficult means available. However, this solution is no more than removing one of the variables by arbitrarily declaring it to be a constant and it is unlikely that such a convenience will be available in its eventual operation.

Manseur and Doty (1992a) reduced the four degree of freedom problem to a system of four linear equations in the sine and cosine of two of the joint variables, thus leaving a fairly simple problem of four equations with four unknowns. As for more advanced and complex mechanisms, they presented an iterative technique for a five degree of freedom problem in a companion paper (1992b) and state that their approach to that for six degrees of freedom was still in development as of that writing.

Another approach is that recognized by Ridley (1994a) who promotes graphical solutions, citing that most conventional solutions shroud the physical simplicity of the mechanism with abstract mathematics. He also criticized most methods for their tendency to limit themselves by using only arms with spherical wrists to simplify the mathematics. In the end, his graphical approach was able to determine all but one joint variable, which he admits would require an analytical solution or physical measurement. However, in a follow-up paper, Ridley (1994b) presents an analytical approach to find all the possible joint positions for a given end position based on his graphical technique. Ultimately, despite his enthusiasm, Ridley admits that actual graphical methods are mostly useful only for visualizing the problem and its manipulation and that analytical solutions are more accurate and hence more useful, if more complex and obscure.

The methods mentioned above and many more not cited here involve complicated mathematical systems of intricate equations which are highly non-linear and transcendental. In the course of solving some or all of these equations, conflicts may arise with singularities. Where the solutions do exist, they are often buried in complicated manipulations of the final individual matrix elements which are unique for each type of mechanism. Even with the considerable insight required to perform these non-obvious matrix element derivation techniques, analytical solutions still present problems. The solution to the inverse kinematics problem for a mechanism with even a modest number of degrees of freedom most likely does not exist in closed form and furthermore, as previously stated, any one of those multiple solutions may or may not exist.

Aside from the mathematical complexities, there are other disadvantages to the majority of current solution schemes. Graphical solutions require great effort and precision and their results are not very accurate. Analytical techniques of any sort tend to be very obscure and involved and suffer from the reality that the more complicated the mathematics, the greater the chance for the introduction of computational errors and the more processing time required to arrive at the solutions. As with any purely mathematical presentation, there is a relatively complete insulation of the actual physical meaning of the problem to the user. Due to the highly complicated nature of the problem, many of the solutions that have been developed are limited to only very specific types of mechanisms.

Due to the deficiency of the current solution techniques in inverse kinematics, more primitive means are often employed in areas where the results cannot wait for a general system to determine them. In most conventional cases now, a robot can be "taught" by manually putting the manipulator arm in the desired position and allowing the computer controller to memorize the required joint positions for each case. However this will not be possible in many of the desired future applications of robotics such as remote operations where the operator cannot physically be present, such as space work, or where conditions are too dangerous, such as working with hazardous materials or in otherwise hostile environments. These existing methods of specifying the joint displacements are clearly inadequate for these purposes. There is a need for a reliable method to determine the configuration of the joints knowing only the desired end position.

1.6 Motivation for this Approach

The trend in inverse kinematics seems to be toward highly mathematical approaches. However, as can be seen, both analytical and graphical approaches have their drawbacks. The method proposed in this paper seeks to combine the better aspects of the two.

This approach, called the "Miss Angle Iteration Method," abbreviated MAIM, is motivated by a physical sense of the mechanism. From this basis, it applies a minimum of mathematics to the analytical problem and translates some of that mathematics into the corresponding physical reality instead of the reverse to work toward the solutions for the correct joint displacements for a given position.

This novel approach offers several advantages over methods which follow only one of the traditional routines. No derivatives are needed. No mathematics more complicated than matrix algebra is required. Comparatively few calculations are made at each iteration, thus reducing the risk of round-off errors and other computational problems. The use of a computer is essential since this method, like most inverse kinematics solutions, is iterative and hence the computations involved are repetitive. The user retains a physical sense of what the method is doing and how it proceeds to reach closure.

1.7 Summary of this Presentation

This paper proposes a relatively simple technique for solving the inverse kinematics problem of general spatial mechanisms, evaluating the joint rotations by a combination of the physical and mathematical approach. It will be shown that the general matrix displacement equation can be partitioned into two distinct problems to effectively remove the translational displacements from consideration. The remaining equation represents a corresponding spherical mechanism which can be analyzed separately.

In practical applications, the final position of one link in a mechanism is known. For the MAIM method, the corresponding positions of the other links can be assumed and the mechanism can be constructed with the links in these incorrect positions. Due to the inaccurate joint displacements, the first and last links will not close at the initial joint, but rather the distal end of the last link will reach a position which leaves some gap between it and the proximal end of the first link. Examining the relationship between the unconnected ends of these links leads to the determination of the angular magnitude of this gap and the evaluation of the axis of that angle. Altering the joint which provides the best approximation of that axis by a portion of that angle will reduce the gap. The mechanism can then be reconstructed with the links in their new positions and the magnitude of the gap checked again. This process is repeated until the gap has been sufficiently diminished.

The remainder of this paper presents these concepts in detail. Chapter 2 demonstrates the validity of the partitioning of the general problem into two independent components to establish the justification for solving for the rotational displacements exclusively. Chapter 3 presents the MAIM theory and the iterative method applied to closed-loop mechanisms. Similarly, Chapter 4 presents the theory and the iterative method applied to open-loop mechanisms. Chapter 5 contains representative examples solved by the MAIM method using a test program to prove the validity of the technique. Finally, Chapter 6, the conclusion, discusses the MAIM method, its advantages and limitations, some future developmental work, and several possible applications.

CHAPTER 2

PARTITIONING OF THE GENERAL INVERSE KINEMATICS PROBLEM

2.1 Derivation

One of the most serious problems in the solution of inverse kinematics problems is that such a large number of variables exist. Each of the joint displacements forms one variable, and the more degrees of freedom which the mechanism possesses, the more variables it requires. It would be very convenient if there were some way to reduce the number of variables that must be solved for at one time.

By deriving a general expression for the configuration of a spatial mechanism, an interesting property about the resulting simultaneous equations can be observed. The general matrix equation containing both the rotational and translational joint displacements can be partitioned into two separate equations such that one of these equations contains only the rotational joint variables. Originally presented by Fischer (1988) as part of a paper on the application of Principle of Transference in spatial mechanisms, this theorem has been used as a basis for the development of this work. The relevant sections of the derivation of this principle will be repeated here due to its significance in the usefulness of the theory presented herein.

Fischer's paper presented an approach using dual numbers. A dual number, D, consists of a primary component, A, and a dual component, B, and is represented in the form

$$D = A + \varepsilon B$$

where ε is an arbitrary number such that $\varepsilon \neq 0$ but $\varepsilon^2 = 0$ (Yang and Freudenstein, 1964). As with complex numbers, vector coordinates, and similar orthogonal systems, the primary and dual components are independent. This notation is very convenient for many kinematic analysis operations.

The dual representation of a coordinate transformation matrix consists of the primary component U, which represents a matrix containing only rotational displacements, and the dual component V, which represents a matrix containing a combination of rotational and translational displacements. For a link of length d_m and twist α_m with displacements of angle θ_m and distance s_m with respect to the local joint axis, these components become

$$
{}_{m+1}^{m}U = \begin{bmatrix} \cos\theta_m & -\cos\alpha_m \sin\theta_m & \sin\alpha_m \sin\theta_m \\ \sin\theta_m & \cos\alpha_m \cos\theta_m & -\sin\alpha_m \cos\theta_m \\ 0 & \sin\alpha_m & \cos\alpha_m \end{bmatrix} \tag{2.1}
$$

$$
{}_{m+1}^{m}V = \begin{bmatrix} -s_m \sin\theta_m & \begin{matrix} d_m \sin\alpha_m \sin\theta_m \\ -s_m \cos\alpha_m \cos\theta_m \end{matrix} & \begin{matrix} d_m \cos\alpha_m \sin\theta_m \\ +s_m \sin\alpha_m \cos\theta_m \end{matrix} \\ s_m \cos\theta_m & \begin{matrix} -d_m \sin\alpha_m \cos\theta_m \\ -s_m \cos\alpha_m \sin\theta_m \end{matrix} & \begin{matrix} -d_m \cos\alpha_m \cos\theta_m \\ +s_m \sin\alpha_m \sin\theta_m \end{matrix} \\ 0 & d_m \cos\alpha_m & -d_m \sin\alpha_m \end{bmatrix} \tag{2.2}
$$

By modeling each link with the dual number coordinate transformation matrix

$$
{}_{m+1}^{m}T = {}_{m+1}^{m}U + \varepsilon\, {}_{m+1}^{m}V \tag{2.3}
$$

the kinematic matrix chain for the entire mechanism becomes

$$
{}_{n}^{1}T = {}_{2}^{1}T\, {}_{3}^{2}T \cdots {}_{n}^{n-1}T \tag{2.4}
$$

which can be rewritten in a simplified form as

$$ {}^{1}_{n}T = A + \varepsilon B \tag{2.5} $$

The condition for loop closure is

$$ A + \varepsilon B = I \tag{2.6} $$

Since the primary and dual components are independent by definition, they can be equated separately. Thus,

$$ A = I \tag{2.7} $$

$$ B = [0] \tag{2.8} $$

where I is the 3×3 identity matrix and [0] is the 3×3 null matrix.

Substituting the actual elements into Equations 2.4 and 2.5 above, the primary component becomes

$$ {}^{1}_{2}U\,{}^{2}_{3}U \cdots {}^{n-1}_{n}U = A \tag{2.9} $$

and the dual component becomes

$$ \begin{aligned} &{}^{1}_{2}V\,{}^{2}_{3}U \cdots {}^{n-1}_{n}U + {}^{1}_{2}U\,{}^{2}_{3}V \cdots {}^{n-1}_{n}U \\ &\quad + \cdots + {}^{1}_{2}U \cdots {}^{n-2}_{n-1}U\,{}^{n-1}_{n}V = B \end{aligned} \tag{2.10} $$

Then, substituting the primary expression into its closure value, Equation 2.7, yields

$$ {}^{1}_{2}U\,{}^{2}_{3}U \cdots {}^{n-1}_{n}U = I \tag{2.11} $$

By inspection, it can be observed that this equation contains only variables representing the rotational displacements. Substituting the dual components into its closure relationship, Equation 2.8, provides an equation in both the rotational and translational variables.

Equation 2.11 is of interest as it allows part of the inverse kinematics problem to be simplified to dealing only with the joint rotations. Ultimately, solutions must be found for all of the joint variables, including both the rotational and translational displacements; but this derivation proves that the solution of the inverse kinematics problem can be taken in two parts: solving for the joint rotations independently, then, using those rotations, solving for the remaining translations.

In summary, by modeling a general spatial mechanism with dual numbers and performing the matrix mathematics required to indicate closure, the final matrix product of the kinematic chain is shown to have a primary component and a dual component. By working with symbolic algebra to display the actual terms of the matrices, it can be observed that the primary component involves only the joint rotations as variables. This independent expression contains far fewer variables than that for the entire mechanism. The dual component involves both rotational and translational displacements, but if the correct values for the rotations are known from any means, the dual equation becomes an expression in only the translational unknowns. This reduced equation again presents far fewer variables than that encountered when attempting to solve the entire inverse kinematics problem at once. Fischer presented a method to solve for the translations knowing the rotations. Other matrix algebraic solution routines would also be effective. Since all of the relevant angles in the mechanism are known, all of the trigonometric terms effectively become constants. The remainder of this paper will present a technique for determining the joint rotations of a spatial mechanism for a given desired configuration, thereby attempting to provide the results of the equation represented by the primary component.

CHAPTER 3

PRESENTATION OF THE CLOSED-LOOP MAIM THEORY

3.1 Analyzing the General Spherical Mechanism

With the general inverse kinematics problem partitioned, the present study will deal with the solution of the spherical component of the spatial mechanism. All of the terms used herein refer to the partitioned mechanism described previously and deal exclusively with the rotational displacements of the joints. Spherical mechanisms possess several interesting properties which make their analysis somewhat simpler. Since all of the joint axes in such a mechanism intersect, the overall mechanism can be visualized as though the links were arcs floating on the surface of a sphere. It is possible to travel from any point on the surface of the sphere to any other point by transforming along the radial axis into the center, rotating the necessary amounts, then transforming back out by the original radial distance. Since this radial distance is arbitrary, the entire spherical mechanism can be regarded as collapsing down to a point such that only the rotational quantities matter.

Each link can be modeled using the standard Denavit-Hartenberg convention. A local coordinate frame is attached to the distal end of the previous link and the joint displacements are measured from this origin point. The geometry of a link is expressed as the fixed distances separating adjacent coordinate frames. The Denavit-Hartenberg parameters include the angular twist of the link, α, the rotation of the joint, θ, the linear length of the link, d, and the linear displacement of the joint, s. Due to the spherical nature of the problem being analyzed, only the angular quantities, the link twists and the joint rotations, need to be considered. The translational displacements have been eliminated from consideration and the lengths of the links vanish since the joint axes of each link intersect. Thus, a general link in a spherical mechanism becomes as shown in Figure 3.1.

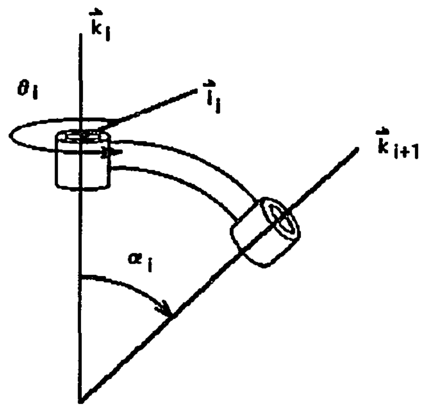


Figure 3.1 The angular Denavit-Hartenberg parameters for a spherical link.

Ideally, if all of the joint variables are set correctly, the chain of links comprising the mechanism closes, as shown in Figure 3.2.

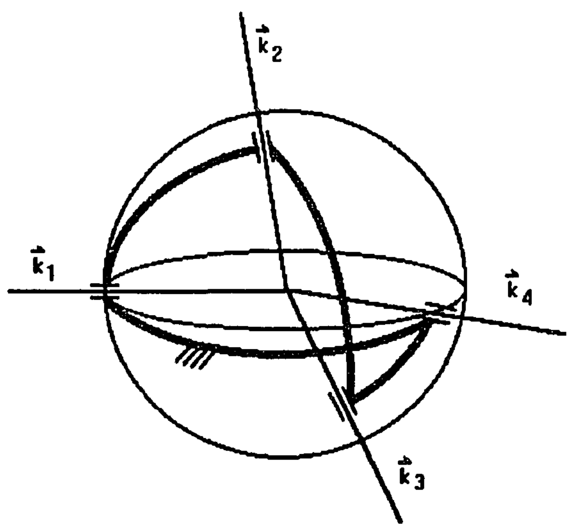


Figure 3.2 Ideal mechanism configuration.

However, in most applications, only the rotation at one joint is initially known. The others must be determined. The MAIM method can be applied to solve this problem.

As the function of the MAIM method is only to improve the accuracy of an existing configuration, some initial configuration must be assumed. Thus, before beginning the method, it is necessary to make initial guesses of the rotational displacements for the remaining joints. These guesses need not be precisely equal to or even relatively close to the correct solutions. The first step in implementing the MAIM method is to assemble the mechanism in the configuration given by the current joint parameters. Since the position of one link must be specified in any practical mechanisms application, this link can be fixed in its known position and the remainder of the linkage assembled using this link as a starting point with the other links in their currently designated positions. Due to the probable errors in the joint variables, the mechanism chain will not close. This imperfect configuration is shown in Figure 3.3.

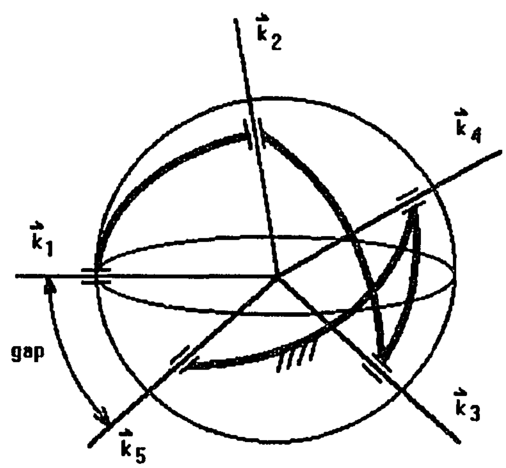


Figure 3.3 Mechanism configuration with errors.

For a closed-loop mechanism, it is important to note that for this method the input crank is fixed and the frame link is allowed to float along with all of the others in order to maneuver toward closure.

3.2 Iterative Procedure to Achieve Loop Closure

As has been stated, the general transformation matrix U relates the joint axes to each other.

$$ {}^{i}_{i+1}U = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{3.1} $$

Since the axis of rotation of a joint is taken to be the **k** axis in the local coordinate frame with its origin at that joint, let $\mathbf{V_i}$ denote a vector in this direction for joint i for simplicity. For convenience, the first joint axis is established as a unit vector in the **k** direction of the proximal joint on the first link in the local coordinates of joint i.

$$ \overline{V}_1 = \begin{Bmatrix} 0 \\ 0 \\ 1 \end{Bmatrix} \tag{3.2} $$

Vectors corresponding to the other joint axes can then be obtained by successive coordinate transformations of this first vector.

$$ \overline{V}_2 = {}^{1}_{2}U\,\overline{V}_1 $$

$$ \overline{V}_3 = {}^{1}_{2}U\,{}^{2}_{3}U\,\overline{V}_1 $$

$$ \cdots $$

$$ \overline{V}_n = {}^{1}_{2}U\,{}^{2}_{3}U \cdots {}^{n-1}_{n}U\,\overline{V}_1 \tag{3.3} $$

If the joint variables deviate from their exact positions required for closure, these computations will result in n+1 axes for a closed-loop mechanism with n joints. The extra axis is that associated with the free distal end of the last link which fails to meet the proximal end of the first link. These two ends should meet to form one joint but, due to the errors in the link orientations, they do not. This concept makes physical sense in terms of the imperfect result obtained by the assembly of the mechanism with its links in incorrect orientations.

The condition for loop closure for the ideal case where all of the joints are in their proper positions is

$$ {}^{1}_{2}U \, {}^{2}_{3}U \cdots {}^{n}_{1}U = I \tag{3.4} $$

However, if any of the joints are improperly aligned, this expression will not be true. Thus,

$$ {}^{1}_{2}U \, {}^{2}_{3}U \cdots {}^{\;\;n}_{n+1}U \neq I \tag{3.5} $$

Hence, due to the errors in the joint variables, the kinematic chain does not close. Physically, this means that instead of closing, the ends of two of the links are left free. Thus, the matrix product represents not a return to the initial coordinate frame but an arrival at some other a point in space.

3.2.1 The Miss Angle and Miss Axis

The joint axes corresponding to these unconnected link ends shall be called the 'terminal vectors," $\mathbf{V_{in}}$ and $\mathbf{V_{out}}$, being $\mathbf{V_1}$ and $\mathbf{V_{n+1}}$ respectively. These vectors represent the current orientations of the two free ends of the linkage. The gap between the terminal vectors represents the angular miss in the closure of the mechanism. The discrepancy between $\mathbf{V_{in}}$ and $\mathbf{V_{out}}$ can be seen in Figure 3.4.

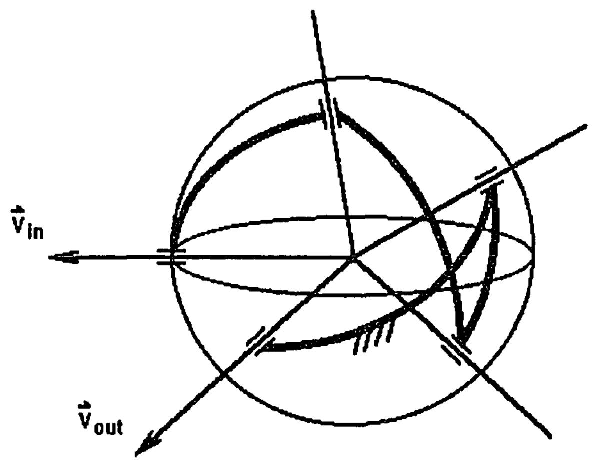


Figure 3.4 The terminal vectors.

Note that, regardless of their orientation, the terminal vectors pass through a common point, the center of the sphere. Therefore, if $\mathbf{V}_{in}$ and $\mathbf{V}_{out}$ are in alignment, the mechanism is closed.

As has been shown, due to errors in the joint variables, the linkage chain does not close. The current configuration does leave the end of the chain at some other point. This position leaves a gap in space between the distal end of the final link and the proximal end of the initial link. The angular magnitude of that gap indicates the severity of the error in closure. To determine the relationship between the current position of the mechanism and the ideal position associated with closure, the final terminal vector could be transformed back to the initial terminal vector in a manner similar to that of transforming between the intermediate joint axes. As each of the terminal vectors represents the **k** direction in their respective local coordinate frame, a general transformation matrix which is capable of rotating one frame into alignment with another is required.

3.2.1.1 Rotation About a General Axis

A vector in one frame can be rotated to its corresponding position in another by a general rotational transformation which relates the two frames (Craig, 1986). Just as the U_i coordinate transformation matrix allows the transformation from one local joint frame to the next, a rotation matrix can be used to transform between two coincident local coordinate frames separated by a rotation of a general angle, θ, about a general axis, **k**. This effect is shown in Figure 3.5.

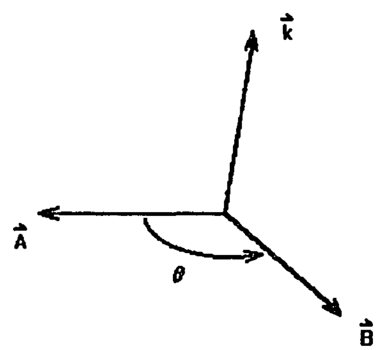


Figure 3.5 Rotation about a general axis.

Mathematically, this is represented by

$$\overline{B} = R(\overline{k}, \theta)\overline{A} \tag{3.6}$$

where

$$R(\overline{k},\theta) = \begin{bmatrix} k_x k_x \text{ver}\theta + \cos\theta & k_x k_y \text{ver}\theta - k_z \sin\theta & k_x k_z \text{ver}\theta + k_y \sin\theta \\ k_x k_y \text{ver}\theta + k_z \sin\theta & k_y k_y \text{ver}\theta + \cos\theta & k_y k_z \text{ver}\theta - k_x \sin\theta \\ k_x k_z \text{ver}\theta - k_y \sin\theta & k_y k_z \text{ver}\theta + k_x \sin\theta & k_z k_z \text{ver}\theta + \cos\theta \end{bmatrix} \tag{3.7}$$

in which the versine function is defined as

$$\text{ver}\theta = 1 - \cos\theta$$

and

$$\overline{k} = \begin{Bmatrix} k_x \\ k_y \\ k_z \end{Bmatrix}$$

3.2.1.2 Computing the Miss Angle and Miss Axis

The mechanism would be closed if $\mathbf{V}_{out}$ were to be transformed from its current position to the position of $\mathbf{V}_{in}$. To simulate this, an imaginary rotational transformation could be introduced to transform $\mathbf{V}_{out}$ to $\mathbf{V}_{in}$ and therefore produce closure. Analysis of this rotation will allow the determination of the angular magnitude of the present gap in the loop and the axis of the rotation ideally required to correct it.

Applying the concept of a general rotation about an axis, $\mathbf{V}_{out}$ can be rotated into alignment with $\mathbf{V}_{in}$. In this case, both vector positions are known. The terminal vectors represent the **k** directions of their respective local coordinate frames, hence both coordinate frames are defined and instead, the axis of rotation and the required angle of rotation between them is to be determined.

From the terminal point of the kinematic chain in its current position, a theoretical rotation can be inserted such that the mechanism closes. This rotation can be symbolized by the matrix R, such that

$$ {}_{2}^{1}\mathrm{U}\,{}_{3}^{2}\mathrm{U}\cdots{}_{n+1}^{\;\;n}\mathrm{U}\mathrm{R}=\mathrm{I} \tag{3.8}$$

For simplicity let

$$ \mathrm{B}={}_{2}^{1}\mathrm{U}\,{}_{3}^{2}\mathrm{U}\cdots{}_{n+1}^{\;\;n}\mathrm{U} \tag{3.9}$$

Then

$$ \mathrm{B}\mathrm{R}=\mathrm{I} \tag{3.10}$$

Hence,

$$ \mathrm{R}=\mathrm{B}^{\mathrm{T}} \tag{3.11}$$

For the vectors $\mathbf{V}_{in}$ and $\mathbf{V}_{out}$, the required angle of rotation is the angular magnitude of the gap in the configuration of the mechanism, called the "miss angle," denoted by σ, and the required axis is the axis of rotation to turn $\mathbf{V}_{out}$ to $\mathbf{V}_{in}$, called the "miss axis," symbolized $\mathbf{M}$. Thus, the rotation matrix R becomes

$$R(\overline{M},\sigma)=\begin{bmatrix} m_x m_x \text{ver}\sigma + \cos\sigma & m_x m_y \text{ver}\sigma - m_z \sin\sigma & m_x m_z \text{ver}\sigma + m_y \sin\sigma \\ m_x m_y \text{ver}\sigma + m_z \sin\sigma & m_y m_y \text{ver}\sigma + \cos\sigma & m_y m_z \text{ver}\sigma - m_x \sin\sigma \\ m_x m_z \text{ver}\sigma - m_y \sin\sigma & m_y m_z \text{ver}\sigma + m_x \sin\sigma & m_z m_z \text{ver}\sigma + \cos\sigma \end{bmatrix} \quad (3.12)$$

The corresponding axis and angle of rotation in terms of the mechanism can be seen in Figure 3.6.

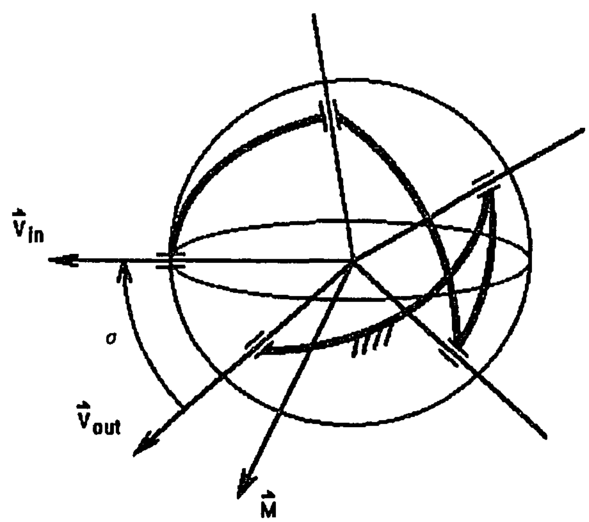


Figure 3.6 The miss angle and miss axis.

The actual matrix R is known by computation from Equation 3.11. Denoting the elements of R by r_{ij}, this matrix can be solved for the miss angle, σ, and miss axis, $\mathbf{M}$.

The magnitude of the miss angle can be computed from the relationship

$$|\sigma| = \cos^{-1}\left(\frac{r_{11} + r_{22} + r_{33} - 1}{2}\right) \tag{3.13}$$

From this calculation, the magnitude of σ will be bounded between 0 and π due to the principle values of $\cos^{-1}$. Thus, in terms of the rotation matrix R, let the miss angle σ be unconditionally defined as the smaller angle between the terminal vectors $\mathbf{V}_{out}$ and $\mathbf{V}_{in}$. The angular magnitude of the miss angle will always be positive. The direction of the required angular correction to the mechanism will be determined later.

The vector coordinates representing the miss axis can be expressed as

$$\overline{M} = \begin{Bmatrix} m_x \\ m_y \\ m_z \end{Bmatrix} = \frac{1}{2\sin\sigma} \begin{Bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{Bmatrix} \tag{3.14}$$

Since the miss angle is defined as the smaller angle between the terminal vectors, let the miss axis be defined as the required axis for the rotation of the one terminal vector toward the other through that angle. Thus, the orientation of **M** will be normal to the plane of the terminal vectors and its direction will be determined by the relative positions of vectors $\mathbf{V}_{out}$ and $\mathbf{V}_{in}$, with **M** being oriented in opposite directions for opposite positions of the terminal vectors.

The magnitude of the miss angle indicates the severity of the gap in the loop. The value of the miss angle σ can be compared to the acceptable level of angular tolerance for the gap. If the miss angle is small enough to fall within a narrow tolerance, this indicates that vectors $\mathbf{V}_1$ and $\mathbf{V}_{n+1}$ are very nearly in alignment and therefore the mechanism is at a very close approximation to closure. If the miss angle is not small enough, the joints in the mechanism require further adjustment to achieve tolerable closure.

3.2.2 Approximating the Miss Axis

Obviously, the vector **M** is the ideal axis about which to rotate by the angle σ in order to close the mechanism. However, in reality, the only rotations possible are those about the joint axes. Therefore, to best approximate the ideal corrective action, the joint axis which is closest in alignment to the miss axis should be selected for a corrective rotation about that joint.

Let ϕ_i be the angle between joint axis i and the miss axis. Treating these axes as vectors in space, the common trigonometric relationship of the vector dot product will determine the magnitude of this angle for each pairing of axes. The axes $\mathbf{V_{in}}$ and $\mathbf{V_{out}}$ should be neglected since these are not subject to corrective rotation: $\mathbf{V_{out}}$ does not really exist and $\mathbf{V_{in}}$ is fixed as an input variable. The most advantageous joint axis to select will be the axis which is closest to being collinear with the miss axis. As previously stated, the value of the angle ϕ for each pair of axes can be obtained from the dot product relationship

$$\overline{V}_i \cdot \overline{M} = |\overline{V}_i||\overline{M}|\cos\phi_i \tag{3.15}$$

where, by definition,

$$\overline{V}_i \cdot \overline{M} = v_{ix}m_x + v_{iy}m_y + v_{iz}m_z$$

Solving for the angle ϕ yields

$$\phi_i = \cos^{-1}\left(\frac{\overline{V}_i \cdot \overline{M}}{|\overline{V}_i||\overline{M}|}\right) \tag{3.16}$$

The physical meaning of these measurements is shown in Figure 3.7.

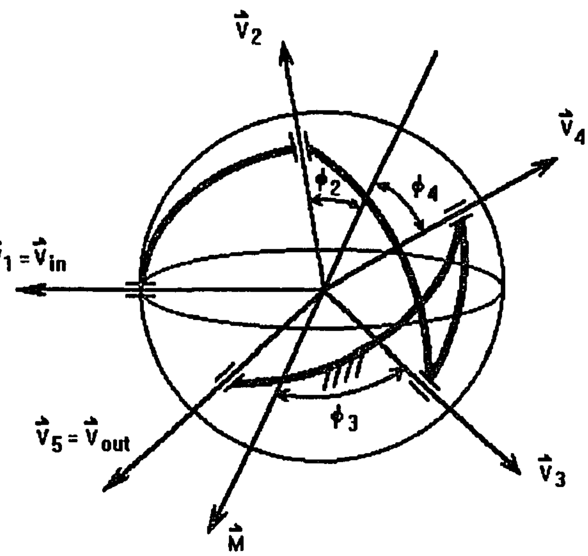


Figure 3.7 Relative alignment of the joint axes and the miss axis.

With the set of angles ϕ thus obtained, it is then necessary to determine the joint axis which is closest to the miss axis. If two vectors are collinear, the angle between them is either 0° or 180°. Whether the relative angle between the miss axis and the joint axis approaches 0° or 180° does not matter at this stage since a positive rotation about an axis in one direction is equivalent to a negative rotation about one in the other. Likewise, the sign of each angle ϕ is not relevant since only the magnitude of the relative orientation between the axes is required. Thus, the joint axis which comes closest to this alignment with the miss axis will be the one desired. Determining the joint axis which makes an angle closest to 0° or 180° to the miss axis is equivalent to finding the joint axis which exhibits the greatest difference between 90° and itself. Denoting the joint axis closest to alignment, in either sense, to the miss axis with the index s, the identity of this axis can be determined by

$$s = \text{index of } \phi \text{ of maximum of} \left(\left| \frac{\pi}{2} - \phi_i \right| \right) \qquad i = 2, \ldots, n \qquad (3.17)$$

3.2.3 Setting the Correction Angle

Knowing the joint axis which can be corrected to provide the most improvement to the closure of the mechanism, it is now necessary to determine the appropriate amount by which to correct the joint displacement. As has been shown, rotating about the miss axis by the miss angle is the ideal way to close the gap. However, since the axis being used for the rotation is not precisely the one for which the miss angle applies, the angle to be used should not be precisely the miss angle. To account for this deviation directly, a relationship between the angles associated with those axes could be paralleled to the relationship between the axes themselves. The angle of correction about a particular joint axis can be taken as some percentage of the full miss angle, where this percentage corresponds to the ratio of the relative alignment between the joint axis and the miss axis. This percentage relationship is chosen to reflect a sense of how accurate the proposed correction would be to the ideal one and to use the same ratio to determine the amount by which to effectively correct the specific joint.

If the full angle of deviation from closure is denoted by σ, let σ_c represent the actual angle of correction for this step. Thus,

$$\sigma_c = \left(\frac{\left| \phi_s - \frac{\pi}{2} \right|}{\frac{\pi}{2}} \right) \tag{3.18}$$

This custom algorithm is a fairly simple method of weighting the data based on the percentage of the measured deviation of the joint axis from a vector normal to the miss axis. The greater the magnitude of the difference between a vector at $\pi/2$ and the joint axis, the closer the joint axis is to alignment with the miss axis, and hence the closer the correction angle will be to the miss angle.

In addition, a safety factor can be applied to prevent accidental over-correction or other errors. This can be any arbitrarily selected fraction. A factor of 1/4 will be used here as this value has been found by experience to be efficient.

$$\sigma_c = \frac{\sigma_c}{4} \tag{3.19}$$

The direction of the correction angle can be determined by examining the joint axis to be corrected. The miss axis **M** has been constructed based on rotating the terminal vector $\mathbf{V}_{out}$ to the terminal vector $\mathbf{V}_{in}$ assuming a positive sense of the miss angle σ. Therefore, if the actual axis of rotation $\mathbf{V}_s$ approximates **M**, that axis should be corrected by $+\sigma_c$, and if the actual axis $\mathbf{V}_s$ approximates **-M**, it should be corrected by $-\sigma_c$. The orientation of $\mathbf{V}_s$ relative to **M** can be determined by whether the angle computed from their dot product, ϕ_s, is less than or greater than $\pi/2$, being close to **M** or **-M** respectively.

3.2.4 Correcting the Mechanism and Re-iterating the Analysis

The joint axis and angle of rotation which will provide the most improvement for the miss in the configuration of the real, physical mechanism are now known. That joint angle must then be adjusted by that amount.

$$\theta_s = \theta_s + \sigma_c \tag{3.20}$$

In terms of the real mechanism, the effect of this correction is shown in Figure 3.8.

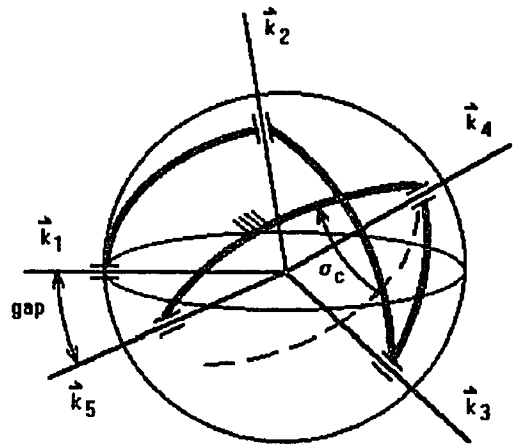


Figure 3.8 The corrected joint angle and the corresponding new position of the mechanism.

With the correction made, the joint axes must be re-evaluated with the new data and the new miss angle computed from these axes in order to determine how close the chain now is to closure. This process is repeated as many times as necessary to reduce the miss to within a tolerable limit and effectively close the mechanism.

CHAPTER 4

PRESENTATION OF THE OPEN-LOOP MAIM THEORY

4.1 Resolving the Open-Loop Mechanism Complications - The Concept of the Virtual Link

Several problems exist in the analysis of open-loop mechanisms which complicate their inverse kinematics. Mathematically, by definition, the open-loop linkage does not close. As a result of this, the kinematic chain forms a different and unique matrix for each position. In addition, the orientation of the end effector, located at the end of the last link, is prescribed but the required orientations of none of the links in the mechanism are initially known and therefore none can be fixed. With closed-loop linkages, the input crank can be considered a fixed link, since its position and orientation are known and constant. This somewhat simplifies the analysis process by both removing one set of the total variables and by providing some fixed point of stability from which to work. For an open-loop mechanism, no real link has a prescribed orientation leaving all of the members yet to be specified and their eventual configuration ambiguous. These complications have made the analysis of open-loop mechanisms considerably more difficult than the closed-loop variety.

The links of an open-loop mechanism are modeled exactly the same way as for a closed-loop mechanism (see Figure 3.1). When constructed, an open-loop linkage extends from its base to its end position in space. This end is usually comprised of an end effector for the purpose of object manipulation in the case of the most common modern form of open-loop mechanism, the robotic manipulator. For the purpose of this paper, this end effector will be referred to as the hand, although it could take any form. Thus, an open-loop mechanism in its desired configuration could be represented as shown in Figure 4.1.

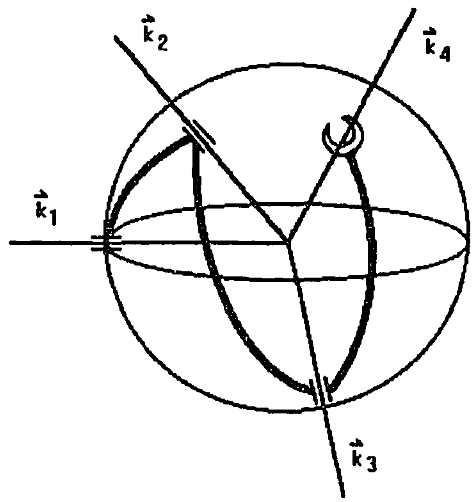


Figure 4.1 An open-loop mechanism.

The problems with the analysis of open-loop mechanisms arise from the fact that the linkage chain does not close. Also, normally no link in the configuration is fixed and thus all of the links are free to move to reach the desired hand position so there is no data initially specified about the orientations of any of the links. To solve these problems, a fixed link can be created as a part of the existing configuration and thereby provide closure for the chain.

To understand this approach, the basic concept of a link must be examined. A link in its simplest form can be considered to be a fixed relationship between one coordinate frame and another. This is obvious in the case of physical links. However, any two points in space separated by fixed dimensions also satisfies this requirement. In the case of the open-loop mechanism, this relationship obviously exists between the ends of each physical link. It must be noted that such a relationship also exists between the position of the base and the ideal position of the hand. By definition, the desired position of the hand is established in space and is fixed relative to the base frame, which is also fixed. This relationship can be seen in Figure 4.2.

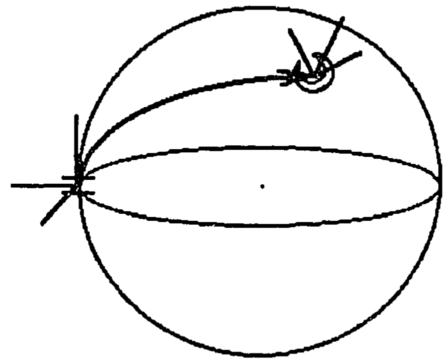


Figure 4.2 Vector representing the position of the hand relative to the base.

The relationship of the hand and the base satisfies the fundamental criterion for a link and thus simulates such a link. Although this relationship fully qualifies as a link in definition, it has no physical form and thus shall be called a "virtual link."

Using this known relationship, it is possible to create such a virtual link in the open-loop linkage which provides the qualities missing from the basic problem and effectively reduces the complicated open-loop model to a comparatively simpler closed-loop model. The problem is then to mathematically specify the relationship between the base and the hand positions.

One common approach to spatial relationships is to state relative coordinate orientations in terms of a set of Euler angles. Euler angles are the specific measurements in a predetermined series of rotations to relate one orientation in space to another, thereby completely specifying their relative position and orientation with only three variables. One such set in particular is the ZYX series of axes. In the case of an open-loop mechanism, the ZYX Euler angles can be used to specify the orientation of the hand frame relative to the base frame. This can be represented as a coordinate transformation using the Euler transformation matrix (Paul, 1981).

The Euler matrix E can be used to represent the hand in terms of the base frame. This relationship is given by

$$
{}_{0}^{1}E = \begin{bmatrix} \cos\alpha\cos\beta & \begin{matrix}\cos\alpha\sin\beta\sin\gamma \\ -\sin\alpha\cos\gamma\end{matrix} & \begin{matrix}\cos\alpha\sin\beta\cos\gamma \\ +\sin\alpha\sin\gamma\end{matrix} \\ \\ \sin\alpha\cos\beta & \begin{matrix}\sin\alpha\sin\beta\sin\gamma \\ +\cos\alpha\cos\gamma\end{matrix} & \begin{matrix}\sin\alpha\sin\beta\cos\gamma \\ -\cos\alpha\sin\gamma\end{matrix} \\ \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \\ \end{bmatrix} \tag{4.1}
$$

where α, β, γ are the ZYX Euler angles of the position in space. The ZYX Euler angles for the desired end effector orientation must be specified by the user. The transpose of the Euler matrix represents the inverse transform, giving the base in terms of the hand frame.

$$
{}_{1}^{0}E = {}_{0}^{1}E^{T}
$$

Mathematically, the Euler transformation matrix serves the same function as the general coordinate transformation matrix used for the real physical links. Thus, by means of the Euler angles and the Euler matrix, the known and fixed dimension of the space between the hand and the base becomes the virtual link.

Note that, in Figure 4.2, the virtual link has been drawn curved for easier visualization of the spherical model. Since it has no real physical form, it really can be considered to be any shape.

With the realization of the virtual link, the mathematical model for an open-loop mechanism becomes identical to that for a closed-loop mechanism. Thus, the MAIM method applied to an open-loop mechanism exactly parallels that for a closed-loop mechanism.

By including the virtual link in the configuration, it can be noted that when all of the joint angles are correctly assigned, the real links meet both ends of the virtual link and the mechanism technically closes. The physical appearance of this concept is shown in Figure 4.3 for aid in visualization.

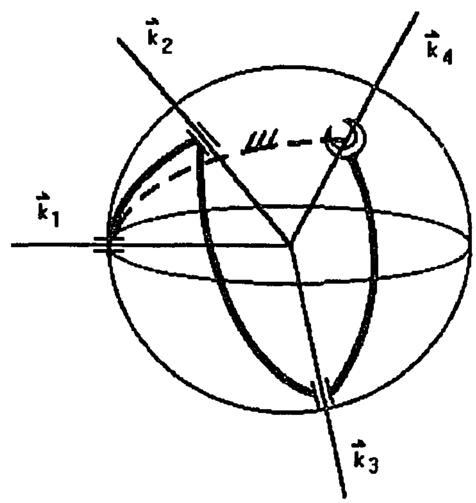


Figure 4.3 Ideal mechanism configuration.

However, the correct rotations of the joints required for the given position are usually not known. These many interdependent variables must be determined knowing only the desired position of the hand and the geometry of the links. The MAIM method can be applied to solve this problem and determine the correct joint rotations. If any of the joints are in incorrect positions, the end of the chain of real links does not meet the beginning of virtual link, leaving a gap, as shown in Figure 4.4.

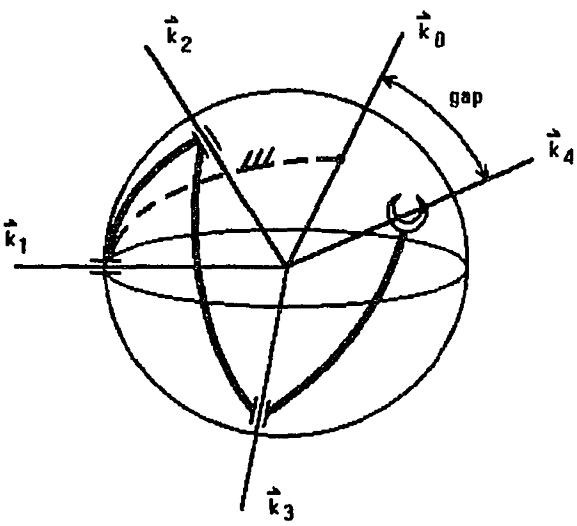


Figure 4.4 Mechanism configuration with errors.

The virtual link represents the ideal hand position relative to the base of the mechanism and is considered prescribed and fixed. The real links of the mechanism can now be adjusted to connect them with the virtual link in that position. The MAIM method can be applied to achieve this.

4.2 Iterative Procedure to Reach the Correct Hand Position

With the ideal hand position known, it is necessary to determine the relationship between the desired hand position and the actual one resulting from the current joint displacements. Since the goal of the linkage is to reach the specified hand position, the ideal hand frame can be used as a starting point. The virtual link can be constructed based on the fixed position of the base relative to the hand and the remainder of the links assembled in their current configuration from the base.

Let the subscript 0 denote the parameters associated with the hand. Let a unit vector be defined in the direction of the **k** axis of the hand frame.

$$\overline{V}_0 = \begin{Bmatrix} 0 \\ 0 \\ 1 \end{Bmatrix} \tag{4.2}$$

Vectors corresponding to the other joint axes can be developed from this starting point. The base axis can be expressed in terms of the hand frame as

$$\overline{V}_1 = {}^{1}_{0}E^{T}\,\overline{V}_0 \tag{4.3}$$

Thereafter, starting from the base, successive general coordinate transformations can be used to develop vectors aligned with the remaining joint axes. The general coordinate transformation matrix is

$${}^{i}_{i+1}U = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{4.4}$$

Therefore, the remaining vectors become

$$\overline{V}_2 = {}^{1}_{0}E^{T}\,{}^{1}_{2}U\,\overline{V}_0$$

$$\overline{V}_3 = {}^{1}_{0}E^{T}\,{}^{1}_{2}U\,{}^{2}_{3}U\,\overline{V}_0$$

$$\cdots$$

$$\overline{V}_n = {}^{1}_{0}E^{T}\,{}^{1}_{2}U\,{}^{2}_{3}U \cdots {}^{n-1}_{n}U\,\overline{V}_0 \tag{4.5}$$

If the joint variables are not in their correct positions to reach the desired hand position, these computations will result in n+2 vectors for an open-loop mechanism with n links. The vectors $\mathbf{V_1}$ through $\mathbf{V_n}$ are the axes of the physical joints in the mechanism. The vector $\mathbf{V_0}$ is the axis associated with the ideal hand position and the vector $\mathbf{V_{n+1}}$ is the axis of the current end position of the linkage. These two should coincide for the ideal closure of the mechanism with its desired position.

With the inclusion of the virtual link, the open-loop mechanism can be regarded as a closed-loop mechanism for purposes of the analysis of its closure. Thus, ideally, if all of the joints are in their correct positions to reach desired hand position, the kinematic equation chain becomes the identity matrix.

$$ {}^{1}_{0}E^{T}\,{}^{1}_{2}U\,{}^{2}_{3}U \cdots {}^{n}_{0}U = I \tag{4.6} $$

However, if any of the joint displacements deviate from their correct values, the mechanism will fail to reach the desired hand position and the kinematic chain will not result in the identity matrix.

$$ {}^{1}_{0}E^{T}\,{}^{1}_{2}U\,{}^{2}_{3}U \cdots {}^{\;n}_{n+1}U \neq I \tag{4.7} $$

In this case, the mechanism fails to reach its desired position, but instead arrives at some other position in space, as shown in Figure 4.4. The discrepancy between those positions is represented by the gap indicated in the figure.

4.2.1 The Miss Angle and Miss Axis

The relationship between the desired position and the current position can be seen by comparing the associated local coordinate frames of these positions. Let the vectors established in the direction of the joint axes of the first joint, denoted $\mathbf{V_0}$, and the last joint, denoted $\mathbf{V_{n+1}}$, be called the terminal vectors $\mathbf{V_{in}}$ and $\mathbf{V_{out}}$ respectively. The terminal vectors are shown in Figure 4.5.

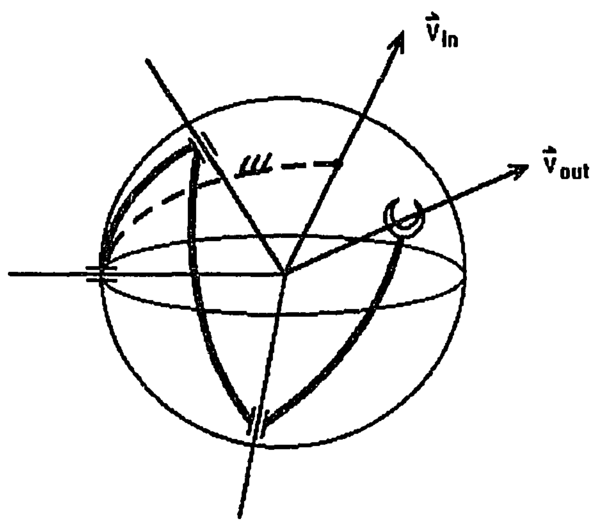


Figure 4.5 The terminal vectors.

Clearly, the mechanism would be in its desired position if the terminal vectors were in alignment. The discrepancy between the vectors is due to the effects of the errors in the joint rotations. To transform from the current mechanism position to the desired one, it would be necessary to rotate $\mathbf{V}_{out}$ from its current position into alignment with $\mathbf{V}_{in}$. In other words, an operation must be performed such that the kinematic chain which includes the virtual link be closed.

4.2.1.1 Rotation About a General Axis

As previously stated in Section 3.2.1.1, a vector in one frame can be rotated to its corresponding position in another by a general rotational transformation matrix (see Figure 3.5). The transformation matrix required to perform the rotation by a general angle, θ, about a general axis, **k**, is

$$R(\bar{k}, \theta) = \begin{bmatrix} k_x k_x \mathrm{ver}\theta + \cos\theta & k_x k_y \mathrm{ver}\theta - k_z \sin\theta & k_x k_z \mathrm{ver}\theta + k_y \sin\theta \\ k_x k_y \mathrm{ver}\theta + k_z \sin\theta & k_y k_y \mathrm{ver}\theta + \cos\theta & k_y k_z \mathrm{ver}\theta - k_x \sin\theta \\ k_x k_z \mathrm{ver}\theta - k_y \sin\theta & k_y k_z \mathrm{ver}\theta + k_x \sin\theta & k_z k_z \mathrm{ver}\theta + \cos\theta \end{bmatrix} \quad (4.8)$$

in which the versine function is defined as

$$\mathrm{ver}\theta = 1 - \cos\theta$$

and

$$\bar{k} = \begin{Bmatrix} k_x \\ k_y \\ k_z \end{Bmatrix}$$

4.2.1.2 Computing the Miss Angle and Miss Axis

Since the vectors $\mathbf{V_{in}}$ and $\mathbf{V_{out}}$ represent the **k** directions of their respective local coordinate frames, the mechanism would be in its desired position if $\mathbf{V_{in}}$ and $\mathbf{V_{out}}$ were in alignment. Using the concept of a general rotation about an axis, it is possible to create an imaginary rotation to align $\mathbf{V_{out}}$ with $\mathbf{V_{in}}$ and thereby close the linkage with its desired position. A corresponding rotational transformation matrix can be inserted into the kinematic chain to compensate for the deviation from closure. The closure expression then becomes

$$ {}_0^1E^T \, {}_2^1U \, {}_3^2U \cdots {}_{n+1}^{\;\;n}U R = I \tag{4.9}$$

For simplicity, let

$$B = {}_0^1E^T \, {}_2^1U \, {}_3^2U \cdots {}_{n+1}^{\;\;n}U \tag{4.10}$$

Then

$$BR = I \tag{4.11}$$

Hence,

$$R = B^T \tag{4.12}$$

Analysis of this simulated rotation to align $\mathbf{V}_{out}$ with $\mathbf{V}_{in}$ from their current positions will indicate the angle and axis of rotation ideally required to correct for the errors in the joints. This angle is called the miss angle, σ, and the associated axis is called the miss axis, **M**. Thus, the rotation matrix R becomes

$$R(\overline{M},\sigma)=\begin{bmatrix} m_x m_x \mathrm{ver}\sigma+\cos\sigma & m_x m_y \mathrm{ver}\sigma-m_z\sin\sigma & m_x m_z \mathrm{ver}\sigma+m_y\sin\sigma \\ m_x m_y \mathrm{ver}\sigma+m_z\sin\sigma & m_y m_y \mathrm{ver}\sigma+\cos\sigma & m_y m_z \mathrm{ver}\sigma-m_x\sin\sigma \\ m_x m_z \mathrm{ver}\sigma-m_y\sin\sigma & m_y m_z \mathrm{ver}\sigma+m_x\sin\sigma & m_z m_z \mathrm{ver}\sigma+\cos\sigma \end{bmatrix} \quad (4.13)$$

The corresponding axis and angle of rotation in terms of the mechanism can be seen in Figure 4.6.

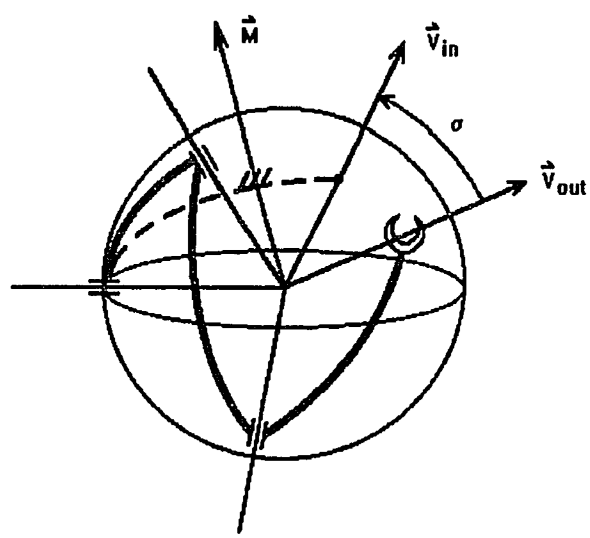


Figure 4.6 The miss angle and miss axis.

The actual matrix R is known by computation from Equation 4.12. Denoting the elements of R by r_{ij}, this matrix can be solved for the miss angle, σ, and the miss axis, **M**.

The magnitude of the miss angle can be computed from the relationship

$$|\sigma| = \cos^{-1}\left(\frac{r_{11} + r_{22} + r_{33} - 1}{2}\right) \tag{4.14}$$

From this calculation, the magnitude of σ will be bounded between 0 and π due to the principle values of $\cos^{-1}$. Thus, in terms of the rotation matrix R, let the miss angle σ be unconditionally defined as the smaller angle between the terminal vectors $\mathbf{V}_{out}$ and $\mathbf{V}_{in}$. The angular magnitude of the miss angle will always be positive. The direction of the required angular correction to the mechanism will be determined later.

The miss axis can be expressed in vector coordinates as

$$\overline{M} = \begin{Bmatrix} m_x \\ m_y \\ m_z \end{Bmatrix} = \frac{1}{2\sin\sigma}\begin{Bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{Bmatrix} \tag{4.15}$$

Since the miss angle is defined as the smaller angle between the terminal vectors, let the miss axis be defined as the required axis for the rotation of the one terminal vector toward the other through that angle. Thus, the orientation of **M** will be normal to the plane of the terminal vectors and its direction will be determined by the relative positions of $\mathbf{V}_{out}$ and $\mathbf{V}_{in}$, with **M** being oriented in opposite directions for opposite positions of the terminal vectors.

The magnitude of the miss angle represents the severity of the gap between the current and desired positions. The value of the miss angle σ can be compared to the acceptable level of angular tolerance for the deviation. If the miss angle is small enough to fall within a narrow tolerance, this indicates that vectors $\mathbf{V}_0$ and $\mathbf{V}_{n+1}$ are very nearly in alignment and therefore the mechanism is very nearly in its correct position. If the miss angle is not small enough, the joints in the mechanism require further adjustment to achieve a tolerable hand position.

4.2.2 Approximating the Miss Axis

With the miss angle and miss axis determined, the ideal way to close the real mechanism is now known. Therefore, for the most effective correction to the position of the mechanism, it is necessary to simulate this ideal action as closely as possible in the reality of the mechanism. Since the only rotations possible are those about the joint axes, the joint axis which is closest to alignment with the miss axis should be selected and a corrective rotation performed about that axis.

Let ϕ_i be the angle between joint axis i and the miss axis. Treating these axes as vectors in space, the trigonometric relationship of the vector dot product can be used to determine the magnitude of this angle for each pairing of axes. Only the vectors corresponding to joint axes which can actually be adjusted need to be considered, that is the vectors $\mathbf{V_1}$ through $\mathbf{V_n}$. The vectors corresponding to the axes of joints 0 and n+1 can be neglected. The axis denoted by $\mathbf{V_0}$ is the ideal hand axis, which is known and fixed, and that denoted by $\mathbf{V_{n+1}}$ is the axis representing the current end position of the mechanism, which does not really exist and cannot be rotated about. The most advantageous joint axis in the mechanism to correct will be the axis which is closest to being collinear with the miss axis. The value of the angle ϕ for each relevant axis pair can be found from dot product relationship

$$\overline{V}_i \cdot \overline{M} = \left|\overline{V}_i\right|\left|\overline{M}\right|\cos\phi_i \tag{4.16}$$

where, by definition,

$$\overline{V}_i \cdot \overline{M} = v_{ix} m_x + v_{iy} m_y + v_{iz} m_z$$

Solving for the angle ϕ yields

$$\phi_i = \cos^{-1}\left(\frac{\overline{V}_i \cdot \overline{M}}{\left|\overline{V}_i\right|\left|\overline{M}\right|}\right) \tag{4.17}$$

In physical terms, these measurements are shown in Figure 4.7.

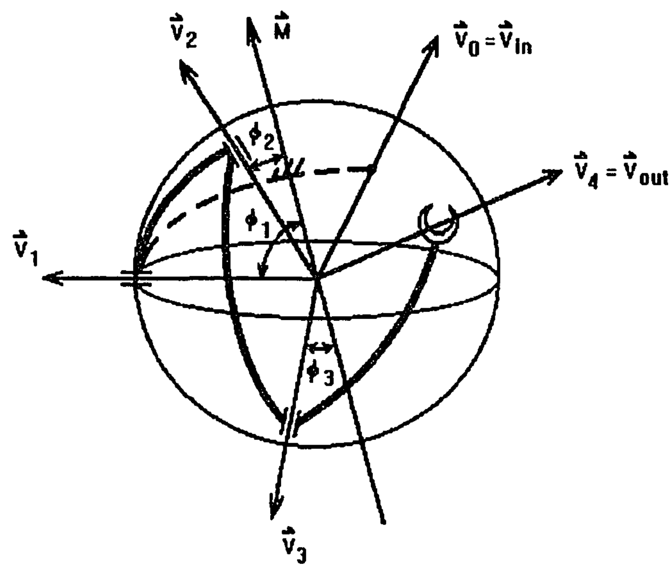


Figure 4.7 Relative alignment of the joint axes and the miss axis.

With the set of angles ϕ thus evaluated, it is necessary to determine the joint axis which is closest in alignment to the miss axis. This relationship is represented by angle ϕ which is closest to 0° or 180°. Whether the joint axis is close to 0° or 180° to the miss axis is not relevant at this point, since a positive rotation about one axis is equivalent to a negative rotation about its opposite. Likewise, the sign of each angle ϕ is not relevant since only the magnitude of the relative orientation between the axes is required. Denoting the joint axis which is closest to alignment, in either sense, to the miss axis with the index s, the identity of this axis can be determined by

$$s = \text{index of } \phi \text{ of maximum of} \left(\left| \frac{\pi}{2} - \phi_i \right| \right) \qquad i = 1, \ldots, n \qquad (4.18)$$

4.2.3 Setting the Correction Angle

With the best approximate for the miss axis selected, the angle by which to rotate that joint in order to correct the position of the mechanism must be determined. As has been shown, performing a rotation about the miss axis by an amount equal to the miss angle would ideally correct the mechanism to obtain closure with the desired hand position. Since the joint axis being corrected in the real mechanism is not precisely the one for which the miss angle applies, the most efficient angle of correction associated with that particular joint axis will not be precisely the miss angle. Hence, an appropriate value for the angle of correction must be selected. To account for the effect of the deviation of the joint axis from the miss axis, a relationship can be developed for the angles of correction associated with those axes based on the relative alignment of the axes themselves. The percentage of alignment between the joint axis and the miss axis can be easily determined and the same percentage can be applied to relate the angle of correction to the full miss angle.

If the full angle of deviation is denoted by σ, let σ_c represent the actual angle of correction for this iteration. Thus,

$$\sigma_c = \left(\frac{\left| \phi_s - \frac{\pi}{2} \right|}{\frac{\pi}{2}} \right) \tag{4.19}$$

This custom algorithm is a simple method of weighting the data based on the percentage of the deviation of the joint axis measured from a vector normal to the miss axis. Thus, the greater the magnitude of the angle between a vector at $\pi/2$ to the miss axis and the joint axis, the closer the joint axis is to alignment with the miss axis, and hence the closer the correction angle will be to the miss angle.

In addition, a safety factor can be applied to prevent accidental over-correction or other errors. This can be any arbitrarily selected fraction. A factor of 1/4 will be used here as this value has been found to be efficient.

$$\sigma_c = \frac{\sigma_c}{4} \tag{4.20}$$

The direction of the correction angle can be determined by examining the joint axis to be corrected. The miss axis **M** has been constructed based on rotating the terminal vector $\mathbf{V}_{out}$ to the terminal vector $\mathbf{V}_{in}$ assuming a positive sense of the miss angle σ. Therefore, if the actual axis of rotation $\mathbf{V}_s$ approximates **M**, that axis should be corrected by $+\sigma_c$, and if the actual axis $\mathbf{V}_s$ approximates **-M**, it should be corrected by $-\sigma_c$. The orientation of $\mathbf{V}_s$ relative to **M** can be determined by whether the angle computed from their dot product, ϕ_s, is less than or greater than $\pi/2$, indicating that the vector is close to **M** or **-M** respectively.

4.2.4 Correcting the Mechanism and Re-iterating the Analysis

The joint axis and angle of rotation which will provide the most improvement in bringing the end of the mechanism toward its desired position are now known. That joint angle must then be adjusted by that amount.

$$\theta_s = \theta_s + \sigma_c \tag{4.21}$$

In terms of the real mechanism, the overall effect of this correction is represented in Figure 4.8.

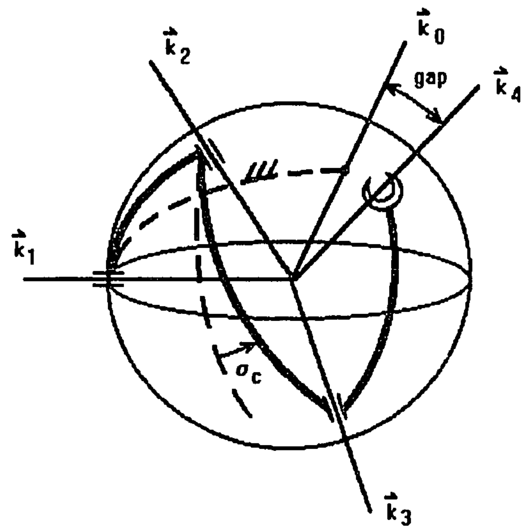


Figure 4.8 The corrected joint angle and the corresponding new position of the mechanism.

The mechanism should then be re-analyzed with the corrected joint angles to determine the resulting miss angle. This process is repeated as many times as necessary to reduce the miss to within a tolerable limit and effectively close the mechanism.

CHAPTER 5

RESULTS

5.1 Overview

This chapter presents the results of examples analyzed to establish the validity of the MAIM method. A computer implementation of this method was coded in the FORTRAN language to demonstrate the effectiveness and accuracy of this technique. Two examples were chosen as representative of the capabilities of the method in general. Sample closed-loop and open-loop mechanisms were selected, in order to show the applicability of the method to both types. The same basic code was adapted to each type of mechanism with only minor modifications. Flowcharts for the respective methods appear in Appendices A and D. A copy of the code for each type is included in Appendices B and E. For both types of examples, a simple forward kinematic computation using the computed joint angles in the matrices will show that these results are valid solutions to within the specified tolerance.

5.2 Closed-Loop Example

A universal joint was selected to demonstrate the MAIM method applied to a closed-loop mechanism. In particular, the Cardan-type universal joint has been used extensively in industry as a shaft coupling (Fischer, 1989). Thus, this represents a real and common spherical mechanism which would require analysis. Figure 5.1 shows a typical Cardan joint.

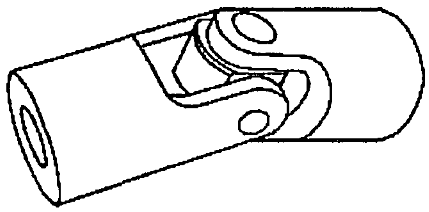


Figure 5.1 The Cardan joint.
Source: Fischer, 1989.

Coordinate systems are attached to the links according to the standard Denavit-Hartenberg convention. In terms of these coordinates, the Denavit-Hartenberg parameters for this four-link spherical mechanism are given in Table 5.1. Only the angular quantities are presented since the linear parameters are not necessary for the MAIM method.

Table 5.1 Angular Denavit-Hartenberg parameters for the Cardan joint.

i	α_i	θ_i
1	90°	θ_1
2	90°	θ_2
3	90°	θ_3
4	150°	θ_4

Initial guesses for the joint displacements of $\theta_1 = 0°$, $\theta_2 = 50°$, $\theta_3 = 320°$, $\theta_4 = 120°$ were chosen and a tolerance of 0.01° was specified for the miss angle. MAIM produced results of $\theta_1 = 0°$, $\theta_2 = 89.996°$, $\theta_3 = 300.005°$, $\theta_4 = 90.004°$ in 75 iterations.

The exact solutions for closure for this mechanism are known, as they can be obtained from a conventional inverse kinematics solution process (Fischer, 1989). These exact solutions are $\theta_1 = 0°$, $\theta_2 = 90°$, $\theta_3 = 300°$, $\theta_4 = 90°$.

A forward kinematic computation using the MAIM results for the joint angles shows that the matrix chain given by Equation 3.4 forms the identity matrix, deviating at most by the allowed tolerance for the miss angle. The joint angles computed by the MAIM method cause this equation to be a valid expression, indicating that the mechanism is closed to within the prescribed tolerance when the joints displacements are set to those values.

The convergence of the miss angle is shown graphically in Figure 5.2. A complete tabulation of the results at each iteration is presented in Appendix C.

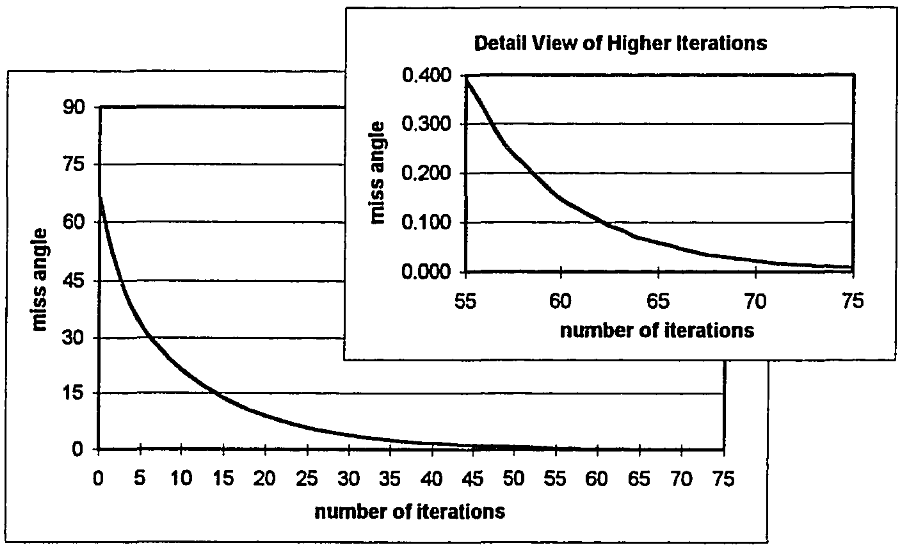


Figure 5.2 Convergence of the miss angle for the Cardan joint.

5.3 Open-Loop Example

The open-loop mechanism chosen for use as an example comes from an actual NASA project, the Space Station Remote Manipulator System (SSRMS). This mechanism is a manipulator arm of the sort which would be used for performing a variety of tasks on a space station. This particular type of arm was previously analyzed by Crane, Carnahan, and Duffy (1991). An arm of this complexity typically represents a considerable challenge for an inverse kinematics solution.

A drawing of the SSRMS is shown in Figure 5.3. Coordinate systems can be attached at the joints according to the standard Denavit-Hartenberg convention.

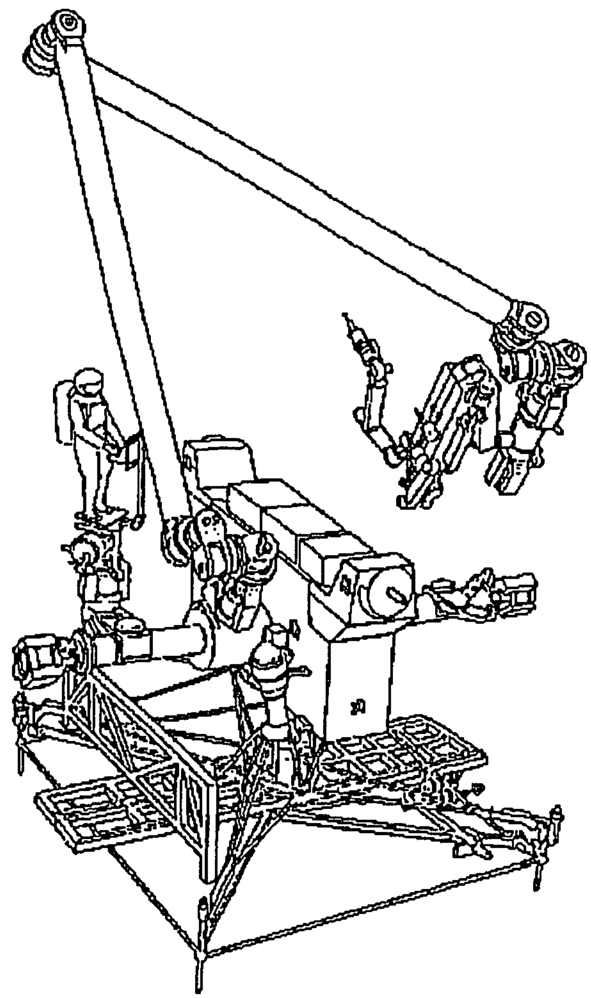


Figure 5.3 The SSRMS arm.
Source: Crane, Carnahan, and Duffy, 1991.

In terms of the specified coordinate systems, the Denavit-Hartenberg parameters for this seven-link open-loop mechanism are listed in Table 5.2. Only the angular quantities are presented since the linear parameters are not required for the MAIM method.

Table 5.2 Angular Denavit-Hartenberg parameters for the SSRMS arm.

i	α_i	θ_i
1	90°	θ_1
2	270°	θ_2
3	0°	θ_3
4	0°	θ_4
5	90°	θ_5
6	90°	θ_6
7	0°	θ_7

For this example, Euler angles of $\alpha = 80°$, $\beta = 30°$, $\gamma = 50°$ were specified for the end effector orientation relative to the base. Initial guesses for the joint displacements of $\theta_1 = 10°$, $\theta_2 = 0°$, $\theta_3 = 11°$, $\theta_4 = 0°$, $\theta_5 = 0°$, $\theta_6 = 0°$, $\theta_7 = 2°$ were selected and a tolerance of 0.01° was prescribed for the miss angle. For this configuration, the MAIM method yielded results of $\theta_1 = 354.979°$, $\theta_2 = 125.000°$, $\theta_3 = 359.947°$, $\theta_4 = 0.000°$, $\theta_5 = 89.565°$, $\theta_6 = 14.702°$, $\theta_7 = 332.612°$ in 66 iterations.

The position resulting from these angles can be confirmed by one of two approaches. A forward kinematics computation using the computed joint displacements in the coordinate transformation matrices will construct the matrix for the base-frame to hand-frame transformation which can then be solved for the ZYX Euler angles. In this way, the joint displacements resulting from the MAIM solution were found to produce Euler angles of $\alpha = 80.007°$, $\beta = 30.005°$, $\gamma = 50.007°$.

Alternately, the prescribed Euler angles and the MAIM results for the joint angles can be used in the elements of the kinematic chain in Equation 4.6. If the angles are valid solutions for closure, the product of these matrices should produce the identity matrix to within the tolerance specified for the miss angle.

The convergence of the miss angle is shown graphically in Figure 5.4. A complete tabulation of the data is presented in Appendix F.

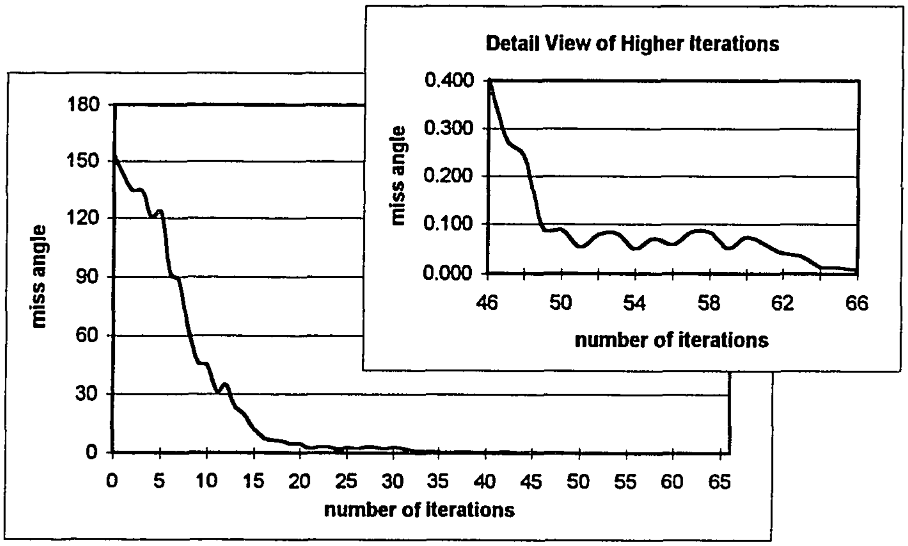


Figure 5.4 Convergence of the miss angle for the SSRMS arm.

CHAPTER 6

CONCLUSION

6.1 Conclusions

This thesis presents a new technique for solving the inverse kinematics problem of general spatial mechanisms and establishes its feasibility. The Miss Angle Iteration Method, abbreviated MAIM, incorporates a mixture of some of the successful elements from previous inverse kinematics solution approaches while avoiding many of the unfavorable ones. The MAIM method offers several advantages over others, including versatility in its application, a sense of physical understanding for the user, and relative mathematical simplicity.

The MAIM approach utilizes a mathematical simulation of the physical problem of correctly positioning the joints in a spatial mechanism for closure. It has been shown that the general problem can be partitioned to remove the translational displacements from consideration. The MAIM method deals solely with the solution for the rotational displacements of the joints. The method serves to improve the existing configuration of a mechanism toward closure. The approach is based on a theory that parallels the actual physical assembly of the links. The orientation of one link in the mechanism is normally specified and as such is established as fixed in the method. Initially, guesses must be made for the orientations of the other links. These links are then assembled in those positions by attaching them in succession onto the known link. A mathematical model to simulate this process is developed by establishing a fixed axis in the position of the one specified joint in the mechanism and using coordinate transformation matrices to develop the subsequent joint axes based on the current positions of the joint angles. Because the joint angles are incorrect, the linkage chain will not close and a gap will exist between the proximal end of the first link and the distal end of the last link. Through analysis, the angular magnitude of this miss in closure and its associated axis of rotation can be determined. A real joint axis is then chosen which best approximates this ideal axis and that joint angle is corrected by a portion of the full miss angle. This adjustment should have the effect of reducing the gap. The links are then re-assembled with the joints in their new positions and the miss in closure re-evaluated. This procedure is repeated until the miss is within tolerable limits, at which point the gap between the initial and final links is considered negligible and the mechanism is effectively closed.

The MAIM method is an inverse kinematics solution technique applicable to both open-loop and closed-loop spatial mechanisms with any number of links. The theory has been proven to be valid. The approach to the problem is sound and the solutions it can obtain are numerically correct. A computer code has been written which has proven the validity of the MAIM method. The code presented herein was intended solely to demonstrate the feasibility of the theory. This code may be used as a framework for developing a program suitable for practical applications.

6.2 Discussion of Results

The MAIM theory was tested by using a variety of numerical examples to verify its feasibility and accuracy. The method was implemented with a computer program designed to demonstrate the basic operation of the technique. Of the numerous examples and cases tested, two were presented in Chapter 5 to be representative of the capabilities of the method in general. A closed-loop and an open-loop mechanism were chosen to show the applicability of the method to both types of linkages.

The Cardan joint was selected as a common four-link closed-loop spherical mechanism for which accurate inverse kinematics results are available for comparison. The MAIM method satisfactorily paralleled these established results.

The analysis of a seven-link open-loop mechanism such as the SSRMS represents an extremely difficult challenge in inverse kinematics, one which is nearly impossible to solve with standard techniques. The SSRMS arm was previously analyzed by Crane, Carnahan, and Duffy (1991) who dealt with the seven degree-of-freedom system as three six degree-of-freedom subchains. These sub-chains were formed by declaring one of the seven joints to be known and therefore fixed, thereby removing it from the computations. In comparison, the MAIM method was able to deal with the mechanism as a whole, in its complete form. No special reductions or restrictions were necessary to solve for the joint angles. With the MAIM method, for any mechanism, closure can be obtained from knowing only the geometry of the links and the desired final point of the configuration.

While the inverse kinematics solution of a four-link mechanism is merely tedious, linkages with five, six, or more members are considerably more difficult to solve by standard methods, if not beyond their capabilities altogether. Hence, a complete analysis of the seven-link mechanism presented in Section 5.3 would be an extremely complicated procedure using most conventional techniques. The MAIM method was able to produce the correct joint displacements in under 100 iterations. The number of iterations required to produce closure in the various tests varied depending on the quality of the initial joint displacement guesses among other factors. In many cases, for both types of mechanisms, more or fewer iterations were achieved than are presented in these samples. The lengthier situations were deliberately chosen for inclusion as examples to demonstrate that the MAIM method can solve any case, regardless of the accuracy of the initial guesses, and to prove that the miss angle will eventually converge by this technique. Due to the simplicity of the computations involved, the actual processing time to perform the method and arrive at these solutions was minimal, even for the cases requiring an extremely high number of iterations.

The basic code for the MAIM method was easily adapted to accommodate closed-loop and open-loop mechanisms. This similar structure is deliberate in order to take optimal advantage of the generality of the method. The method itself is fundamentally the same for each type of mechanism, only the specific identities of the individual joints differ.

The accuracy of the solutions for these or any examples can be confirmed by performing forward kinematics with the results of the method as inputs and comparing the resulting matrix to the identity matrix. For a closed-loop mechanism, inserting the joint angles into Equation 3.4 should produce the identity matrix. Similarly, for an open-loop mechanism, inserting the prescribed Euler angles and the joint angle results into Equation 4.6 should also yield the identity matrix. Alternately, a standard forward kinematics computation can be performed with the results of the method inserted as the joint displacements to arrive at the transformation matrix for the base-frame to the hand-frame, the ZYX Euler matrix. This matrix can then be solved for Euler angles α, β, γ. The resulting Euler angles should be equal to the specified hand position for an open-loop mechanism or precisely 0°, 0°, 0° for a closed-loop mechanism, indicating closure of the linkage chain. By any approach, the results of a closure analysis, either the elements of the identity matrix or the Euler angles, should be approximately equal to the ideal values within the specified tolerance for the miss angle. This computation can be used to confirm the validity of the MAIM method.

Many tests were run with various mechanisms in a variety of configurations. Closed-loop and open-loop mechanisms with a wide range of links were used for trial solutions. The vast majority of these tests were successful. A few problems were encountered, though these were mostly due to the simplicity of the code, not any deficiency in the method.

The sample computer implementation developed here has demonstrated that the MAIM method works successfully. The program was able to solve an adequate variety of test problems to sufficiently prove the feasibility of the theory. In most cases, this program achieved remarkable precision in a comparatively small number of iterations, commonly being able to produce accurate closure to within 0.01° within 100 iterations. This computer version of the MAIM technique was capable of closing the mechanism regardless of the values of the initial guesses of the joint displacements, although the particular set of the multiple solutions thus obtained did vary from case to case. Even under the worst conditions, closure was finally achieved after several hundred iterations.

The code has proven itself to be an effective simulation of the MAIM method. The variety of tests performed with it have been successful enough to confirm that the MAIM theory is sound and the method reliable. Several noteworthy comments regarding the theory and its implementation as well as the difficulties encountered with the sample code will be discussed in the following section.

6.3 Commentary

The MAIM method is a novel approach to solving the inverse kinematics problem of a general spatial mechanism. This new method offers some interesting advantages over previous procedures. As such, several comments regarding the foundation and operation of this alternative approach are worthy of mentioning in closing.

One of the significant innovations presented here is the concept that an open-loop mechanism can be modeled as a closed-loop mechanism by the inclusion of a "virtual link" in its configuration. The recognition that the relationship between the hand and base positions forms a fixed link can be exploited to mathematically close the mechanism model and thereby simplify the analysis of the linkage. This simplification allows an open-loop mechanism to be analyzed with the MAIM method or any other closed-loop kinematics theory. This presentation uses ZYX Euler angles and the Euler transformation matrix to specify the base frame in terms of the hand frame, although any means to relate the two coordinate frames in matrix form could be used.

The MAIM method is a technique for improving the guesses of the rotational joint displacements in an existing mechanism configuration to obtain loop closure. As such, some initially guessed values are needed for the joint angles before beginning the method. While the specific values chosen may have some effect on which one of a set of multiple possible solutions is found or on how rapidly the method converges to that solution, they are otherwise arbitrary. These guesses are merely starting points for the procedure to adjust, thus the ultimately reaching a solution regardless of how inaccurate or extreme the initial guesses may be.

It is important to note that the joint angles computed by the MAIM method are definitely a solution. However, in cases where multiple solutions exist, there is no way to control which of these will be reached. Hence, the solution obtained may not be the one expected or desired. As with any iterative technique, intuition and experience may be used to choose the initial guesses to attempt to influence the method toward a specific solution.

The MAIM method is based on the discrepancy in closure in a mechanism with improperly oriented joints. The miss angle, σ, and the miss axis, **M**, are the ideal means by which to correct the mechanism configuration to produce closure. To simulate the ideal solution with the reality of the mechanism, the joint axis closest to alignment with the miss axis should be adjusted. Whether the joint axis approaches the miss axis closest to 0 or π radians is irrelevant, since a positive rotation about one axis in space is equivalent to a negative rotation about an axis oriented in the opposite direction. Since the miss axis has been assumed based on a positive sense of the miss angle from $\mathbf{V}_{out}$ to $\mathbf{V}_{in}$, if the joint axis approximates the miss axis directly, the joint can be corrected by a positive angle, and if the joint axis approximates the negative of the miss axis, the joint can be corrected by a negative angle.

The joint axis which best approximates the miss axis will be chosen for correction in this manner. However, since the axis being used for the correction is not precisely the one for which the miss angle applies, the angle of correction should not be precisely the miss angle. To account for the deviation in the axes, the relationship between the correction angle and the miss angle can be paralleled to the relationship between the joint axis and the miss axis. The optimal angle by which to correct a specific joint can be taken as a percentage of the full miss angle based on how close of an approximation that joint axis is to the miss axis. A simple way to quantify that relationship is to measure the angle the joint axis makes to a line normal to the miss axis. Forming a ratio of that magnitude to $\pi/2$ will yield the percentage alignment of the joint axis to the miss axis. That percentage can then be applied to the value of the miss angle to produce an appropriate correction angle. Thus, the better the approximation to the miss axis, the larger the correction to the joint angle and the more the improvement of the closure of the mechanism.

The application of a factor of safety is a common practice in engineering. The safety factor of 1/4 was assigned to reduce the size of the correction angle into smaller steps to prevent any potential numerical problems such as over-correction. This value was chosen conservatively since an algorithm which produces slower convergence is preferable to one with larger steps which may generate errors. Experimentation has indicated that this factor seems to be satisfactory.

The MAIM method is based on the concept of the gap in an improperly aligned mechanism represented by the miss angle. Ultimately, the iterative process will reduce the miss angle to a tolerable amount. Usually 0.001° or 0.0001° is well within the operating limits of most conventional mechanisms, so joint configurations which result in a miss angle of that order of magnitude or less will possess a negligible gap in their closure. The MAIM method cannot be in error if it indicates that the gap has effectively been closed since the miss angle at each stage is computed from the results of a normal forward kinematics analysis of the mechanism with its joints in their current positions, a process which is known to be an accurate representation of the overall configuration of a mechanism.

The test program developed represents the MAIM method well enough to demonstrate its validity. While the general results of the sample computer implementations have been successful, some matters remain to be resolved in order to create a universally applicable code. A brief summary of them is presented here to aid future development.

The program has been found to be fairly sensitive to the selection of the initial guesses. For some examples, any values chosen will lead to successful solutions. For others, some guesses will be effective and some will lead to a non-convergence. It is impossible for the process to diverge, since by definition all of the joint variables and the miss angle itself are bounded between 0 and 2π radians. However, in some cases the computational process failed to converge to a solution. This condition would seem to indicate that passing through some mechanism configurations may cause this program to drift toward unrecoverable positions. At this point there is no way to speculate on any formal relationship regarding the effects of the initial guesses on the solution process. These types of sensitivities are typical of those encountered in any iterative scheme and cannot be avoided.

Due to the simplicity of the MAIM method, the only problem that can possibly be encountered with the method itself is a failure to converge to a solution. Assuming an adequate number of iterations is allowed, the only cause for the method to fail to converge is if toggling occurs. A common problem in iterative schemes, toggling can have causes which are very difficult to isolate and can delay or prevent convergence to a solution. In this program, several different instances of toggling have been observed, particularly of the miss angle or of one particular joint angle around a value or pair of values. One case involved the correction angle being successively evaluated as an angle with the same magnitude but an alternating sign, with the joint angle thereby reciprocating about the same joint axis indefinitely while the corresponding miss angle toggled between two values. At some points, the magnitude of the miss angle would toggle between complements of 180°, correcting itself back and forth to either side of a semi-circle. In other instances, the miss axis itself would alternately flip between the positive and negative directions, as measured relative to the fixed hand axis. Some of the toggling cases that have been observed seem to suggest evidence of patterns, although the meaning of these has yet to be fully interpreted. Likewise, the cause of these toggling problems is as yet undetermined. In general, until such time as this complication can be resolved, attempts to retry the method with different initial guesses may be found to be more successful.

Some problems with the program may be due to the fact that certain basic concepts cannot be accurately simulated due to the limitations of the computer processor. The miss angle is unconditionally defined as the smaller angle between the terminal vectors. Thus, it is essential to unconditionally guarantee that the miss axis will be constructed using the smaller angle between the terminal vectors. If the computer solution for the vector **M** should be the axis associated with the larger angle of rotation to align the terminal vectors, then the correction will be in the opposite direction and will likely result in an unstable if not detrimental corrective action. Problems such as these which have been encountered in this specific code remain to be resolved by experienced programmers in order develop a program to effectively implement the method.

The computer code written for this demonstration was intended only to prove the feasibility of the method, not to provide universal solutions. As such, some deficiencies exist in the program as it is presented here. Suggestions for possible ways to correct some of the problems encountered with this particular program are included in the following section.

6.4 Future Work and Applications

The MAIM method has been proven to be able to successfully solve for the joint rotations in a spatial mechanism. As has been shown in Chapter 2, the solution for the translational displacements can be obtained after the values for the rotational displacements are ascertained. The results for the rotational displacements provided by the MAIM method can be used as constants in the later computations for the translational displacements of the joints. Since all of the joint rotations in the mechanism are then known, all of the trigonometric terms in the dual component of the kinematic chain equation effectively become constants. Once the problem is thus reduced, the remaining matrix can be solved for the translational displacements by any applicable means. A method for the computation of the translational displacements could be appended to the MAIM code for use after the rotational displacements are determined or applied separately using the results of the MAIM method as input data.

There are many potential practical uses for the MAIM method. For a closed-loop mechanism, if the orientation of the input crank is specified, the MAIM method can be used to return the orientations of the rest of the links. These results can be useful for determination of the position of the output or intermediate links for a given position of the input crank or to construct the positions of all the links for a full revolution of the input crank. The latter case, however, would require some careful work to generate a smooth, continuous display, avoiding any discontinuous jumps between possible alternate multiple solutions for consecutive positions.

For open-loop mechanisms such as any number of a variety of robotic arms, the MAIM method can be used to determine the required joint angles for a desired orientation of the end effector. The method requires knowing only the geometry of the links and the Euler angles specifying the end effector orientation to determine the necessary rotational joint displacements. The user can specify the end effector orientation and use arbitrary or educated guesses of the joint values and the MAIM method will then compute the required joint variables. A potentially practical variation on this theme is that if the arm is presently in one orientation and is desired to be in a different orientation, it is possible to specify the Euler angles of the new orientation and use the joint values of the present configuration as the initial guesses in the calculations. The MAIM method will then return the new angles required for the new orientation. Given the two desired consecutive orientations, another program could be developed which could direct each of the joints to move from its initial orientation to its final orientation, possibly even optimizing the path to move along the most efficient or least interfering route. It would be prudent, however, to allow the opportunity to manually confirm the new joint displacements before their use, in case the results are somehow not acceptable.

The MAIM method as it is represented in the code presented here does not take into account the existence of the possibility of restricted motion of a joint, that is if a joint is constrained to rotate only over a limited angular range. Similarly, the code does not have any contingency for prismatic joints, which are constrained not to rotate at all. The current code has a provision only to select the joint axis closest to the miss axis for adjustment. When performing the MAIM method manually, it will be obvious when such a case causes a problem and the user can correct for it by choosing a different joint to adjust. The computer code could likewise be modified to determine when it is appropriate to reject the closest axis in favor of the second closest, or the third, and so on. Without these measures, the method works fully for mechanisms which have joints possessing a full range of motion while the results must be manually checked for discrepancies for those with joints having limited ranges. Future improvements of the computer implementation of the MAIM method could contain options to declare certain joints to be unable to rotate, eliminating them from consideration for correction by the program, or to take into account joints which are restricted to within a certain range, perhaps opting to correct a different joint entirely if the indicated correction to a joint would place it outside its allowed range.

One area which stands to benefit from a more detailed investigation is the matter of the step-size for adjusting the joint variables. Presently the method uses a step-size large enough to make a steady, controlled change in a joint variable but small enough to not overshoot or otherwise potentially interfere with the natural solution process. The magnitude of this increment could be optimized by whatever statistical method a mathematician or programmer judges to be satisfactory. With regard to the currently suggested step-size, the safety factor of one quarter was chosen almost arbitrarily and is very likely not the most efficient. However, the axis ratio factor was developed and seems effective and so should be used unless much work is put into proving why it is not adequate.

In addition to the safety factor, it may also desirable to improve the step-size algorithm in order to increase the rate of convergence for smaller miss angles. Presently, the program seems to converge almost exponentially slower in terms of the number of iterations performed as the miss angle diminishes. Perhaps this effect is simply due to the fact that the miss angle is very small already and then the joint angle is being corrected by a portion of a percentage fraction of that angle.

While considering small miss angles, it can be shown that the accuracy of the rotation axis determined by Equations 3.14 and 4.15 degrades when the angle of rotation is small. The effect of this on the solution process may deserve exploration due to the potential problems that can be caused by deviations in the miss axis.

The exact effects of the initial guesses on the final solution might be studied for possible patterns in order to develop a way to target a specific set of angles to aim towards for the solution and thereby develop a system to somehow control the multiple solutions problem.

Although two separate programs were created to demonstrate the MAIM method, one unified program could easily be developed to handle the application of the method to both closed-loop and open-loop mechanisms. Some of the routines common to both techniques could be recycled and most of the others could be easily adapted to be flexible enough to work for either type of mechanism. After allowing the user to choose the type of mechanism being analyzed, the program could branch to the relevant sections for the type indicated. In particular, the proper kinematic chain would have to be specified for each option: beginning at the base and transforming to the end of the last link for a closed-loop mechanism and starting at the hand and transforming to the base and then through the links and back to the hand for an open-loop mechanism. Also, the first link in each of these transformations would have to specified as fixed in each case and left out of consideration from correction, but otherwise this merger of the methods seems fairly simple and completely viable.

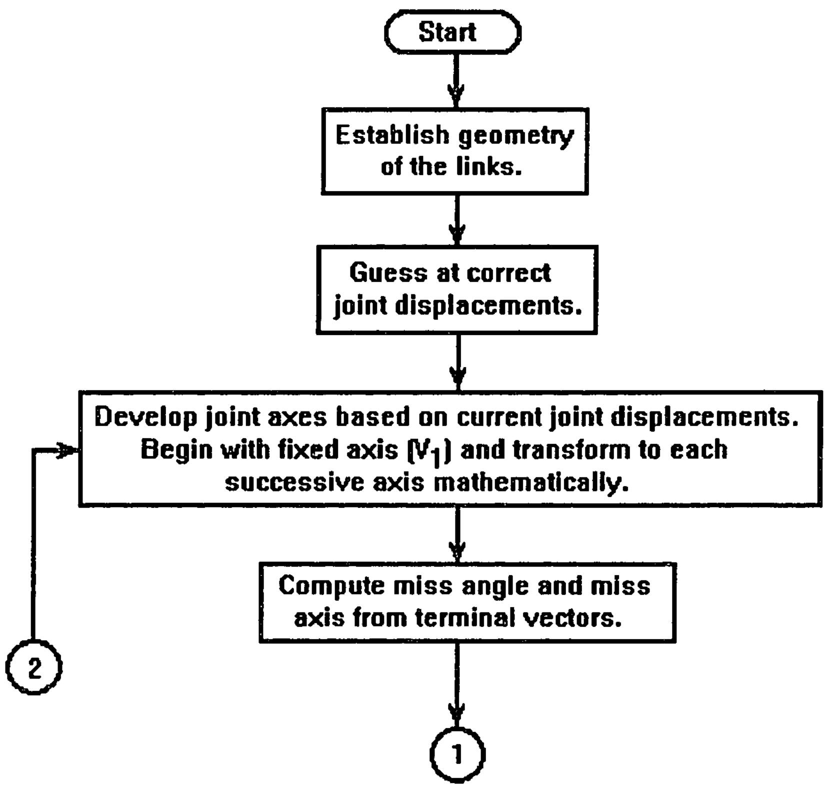
The MAIM method could easily lend itself to being linked with a graphical simulation program to draw the current link positions, the joint axes, the miss axis, and other elements of interest at each iteration for convenient visualization of exactly how the method works and what the adjustment process looks like. To add this feature, either the code for the MAIM method could be expanded to include graphical capability for such a display process or the results from the program for the iterations of interest could be down-loaded to a computerized modeling system such as conventional CAD software and the coordinates processed into graphics there.

If the MAIM method is being used to control the movement of a robotic manipulator of any sort, a step could be added to the computer code to allow the user or another control program to check the newly-solved joint angles before moving the arm there to avoid possible problems. The new arm and joint positions could be checked mathematically by comparison to a range of known allowed positions. Alternately, by linking the MAIM method with a CAD program as previously discussed, the final configuration of the arm could also be checked in a model, either mathematically by a computer controller or visually by the user. The model could ultimately include the obstacles in the work space to confirm that no collisions or impossible or even inconvenient positions occur either in the final position or in the course of moving to it from the previous one.

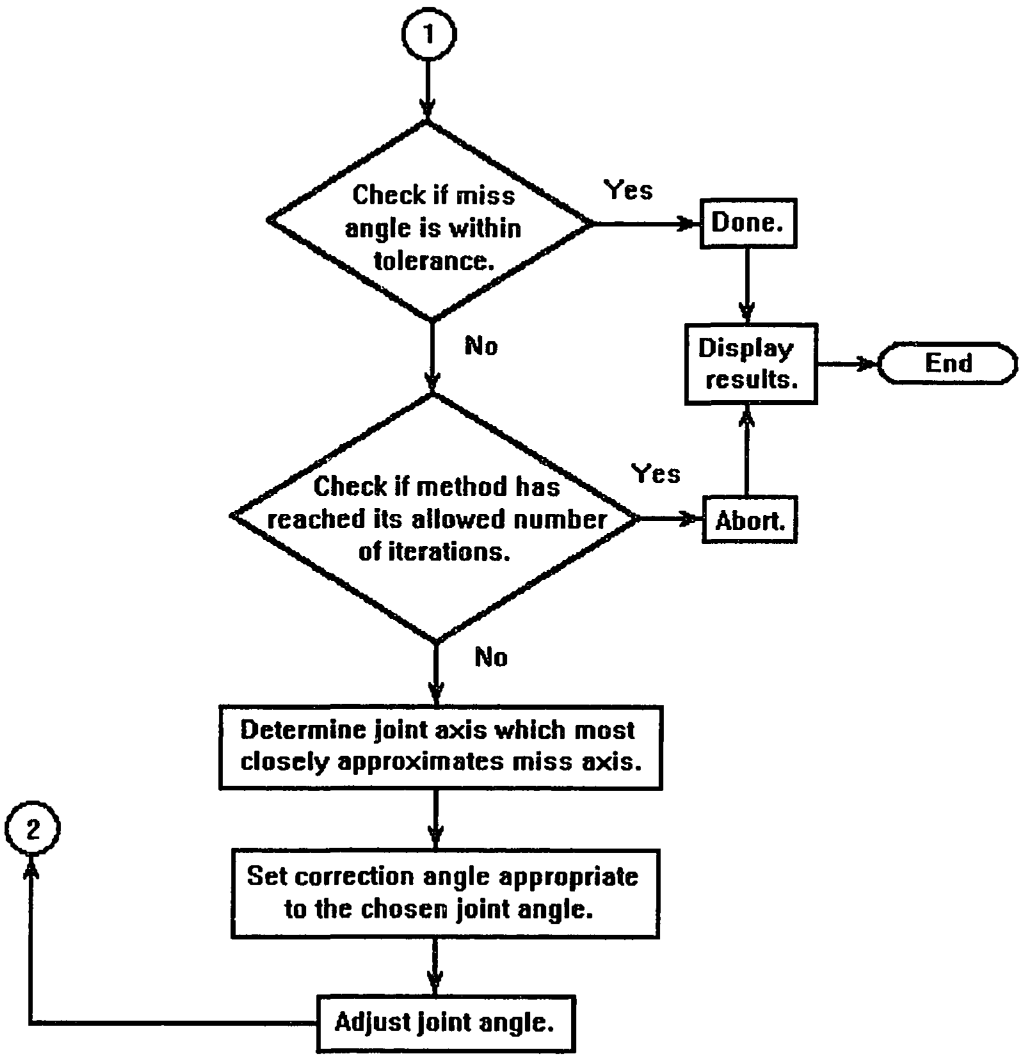
Clearly, the MAIM method has many merits. This new method is very versatile and obviously has many potential applications. The theory has been presented in its entirety and a method has been developed to apply that theory to realistic examples. A sample program demonstrating the feasibility of the method has been developed, however effective implementation still needs considerable refining and fine-tuning. The overall concept of the MAIM method has been shown to have plenty of areas for further development and its uses have just begun to be tapped.

APPENDIX A

CLOSED-LOOP METHOD FLOWCHART



CLOSED-LOOP METHOD FLOWCHART
(Continued)



APPENDIX B

CLOSED-LOOP METHOD PROGRAM CODE (FORTRAN)

```
* Miss Angle Iteration Method (MAIM)
* A Spherical Mechanism Orientation Program
* Closed-loop version

* Method developed by John D. Kliminski,
* incorporating concepts proposed by Dr. Ian S. Fischer
* Program written by John D. Kliminski
* Via NJIT 1994                Updated: 10/19/94


*   Variable List:
* n          - number of links in the mechanism.
* th(n)      - joint angles, theta.
* alph(n)    - twist angles of the links, alpha.
* U(3,3)     - transformation matrix for a given set of coordinate
*              frames.
* V(i,k)     - array of joint axes; the i index represents the x,y,z
*              vector coordinates of the axis and k is the joint index.
*              V is therefore a matrix of column vectors of dimension
*              (3,n+1).
* M(3)       - miss axis in x,y,z vector coordinates.
* sig        - miss angle, sigma.
* stol       - the acceptable tolerance of the magnitude of the miss
*              angle.
* phi(n)     - element phi(i) is the angle between joint axis i and the
*              miss axis.
* s          - index of the joint axis which is nearest to collinear with
*              the miss axis.
* sigcor     - the amount of correction to a joint angle on a given
*              iteration.
* R(3,3)     - general rotational transformation matrix.
* it         - counter for the number of iterations performed.
* itmax      - prescribed maximum number of iterations allowed.
* d2r,r2d    - conversion factors for degrees to/from radians.
* Y(i),Z(i)  - temporary storage arrays for vectors.

*                          --- Main Program ---

* Declare variables.
      integer n, it, itmax, s
      double precision alph(12), th(12), V(3,13),
     & M(3), sig, stol, sigcor, phi(12), p, pi, d2r, r2d

* Declare fundamental trigonometric and mechanism data as common.
      common /tm/ n, pi
      pi=4.d0*datan(1.d0)
      d2r=pi/180.d0
      r2d=180.d0/pi

      write(6,*)
```

```
      write(6,*) '  Miss Angle Iteration Method (MAIM) '
     & 'for Closed-Loop Mechanisms'
      write(6,*)
      write(6,*)
      write(6,*) '(Enter angles on one line separated by commas '
     & 'or on separate lines.)'
      write(6,*)

* Input the specific data for the mechanism and convert all angles
* from degrees to radians.
      write(6,*) 'Enter the number of links in the mechanism: '
      read(5,*) n
      write(6,*) 'Enter the twist of each link (degrees): '
      read(5,*) (alph(i), i=1,n)
      do 12 i=1,n
12      alph(i)=alph(i)*d2r
      write(6,*) 'Enter the initial guesses for the joint',
     & ' angles (degrees): '
      read(5,*) (th(i), i=1,n)
      do 14 i=1,n
14      th(i)=th(i)*d2r
      write(6,*) 'Enter the tolerance for the miss angle (degrees): '
      read(5,*) stol
      stol=stol*d2r
      write(6,*) 'Enter the maximum number of iterations: '
      read(5,*) itmax
      write(6,*)
      write(6,*) 'Ok.'
      write(6,*)
      write(6,*) 'Working...'

* Initialize iteration counter.
      it=0
* Begin iterative procedure.
111   continue

* Develop joint axes.
      call axes (th, alph, V)

* Calculate the miss angle and the miss axis vector.
      call mangle (th, alph, M, sig)

* If current miss angle is within tolerance, end program.
      if (sig.le.stol) then
        goto 999
      endif

* If limit of iterations is exceeded, end program.
      if (it.ge.itmax) then
        goto 999
      endif

* Determine which joint angle to adjust to improve the miss angle.
* Calculate the angle between each joint axis, V, and the miss axis, M.
      do 4 i=2,n
        call anglevec (M, V, i, p)
4       phi(i)=p
* Select the joint axis closest to the miss axis.
      call angcomp (phi, s)
```

```
* Determine the magnitude and direction of the correction angle.
      call setcorr (phi, s, sig, sigcor)

* Adjust the appropriate joint angle by the correction angle.
      th(s)=th(s)+sigcor

* Reset all joint variables to range from 0 to 2*pi.
      call corrth (th)

* Increment iteration counter.
      it=it+1

* Repeat method for next iteration.
      goto 111

999   continue
* Display final results.
      write(6,*)
      if (it.ge.itmax) then
        write(6,*) '          *** Maximum iterations reached --- ',
     &   'method aborted ***'
        write(6,*)
      endif
      write(6,*)
      if (sig.le.stol) then
        write(6,*) '              Method successfully completed.'
        write(6,*)
      endif
      write(6,*) 'The Miss Angle was reduced to ',sig*r2d,' (degrees)'
      write(6,*) 'after',it,' iterations.'
      if (sig.le.stol) then
        write(6,*)
        write(6,*) 'This Miss Angle is within the specified',
     & ' tolerance of ',stol*r2d,' (degrees).'
      endif
      write(6,*)
      write(6,*) 'The final results (in degrees) are: '
      do 91 i=1,n
91    write(6,95) i,th(i)*r2d
95    format(' theta ',i2,' = ',f10.6)
      write(6,*)

      end

*                           --- Subroutines ---

      Subroutine axes (th, alph, V)
      double precision th(n), alph(n), V(3,n+1), pi,
     & U(3,3), Z(3), Y(3)
      common /tm/ n, pi
* Develop joint axes, V1 through Vn+1.
* Define initial axis, V(i,1) (of unit length).
      V(1,1)=0.d0
      V(2,1)=0.d0
      V(3,1)=1.d0
* Obtain other axes, V(i,2) through V(i,n+1), by successive
* post-multiplication of V(i,1) by the U transformation matrix.
      do 2 j=1,n
* Initialize temporary storage array Z as V(i,1).
        do 22 i=1,3
```

```
22        Z(i)=V(i,1)
* Post-multiply by the appropriate transformation matrices.
        do 24 i=j,1,-1
          call makeU (th, i, alph, U)
          call mat31mult (U, Z, Y)
          do 23 ii=1,3
23          Z(ii)=Y(ii)
24        continue
* Return resulting matrix product Z to V(i,j+1), rounding off
* elements to eliminate multiplication precision errors.
        do 21 i=1,3
          Z(i)=(dint(Z(i)*1.d12))/1.d12
21        V(i,j+1)=Z(i)
2       continue
      return
      end

      Subroutine makeU (th, i, alph, U)
* Creates appropriate U transformation matrices.
      double precision th(n), alph(n), U(3,3), pi
      integer i
      common /tm/ n, pi
      U(1,1)=dcos(th(i))
      U(1,2)=-dcos(alph(i))*dsin(th(i))
      U(1,3)=dsin(alph(i))*dsin(th(i))
      U(2,1)=dsin(th(i))
      U(2,2)=dcos(alph(i))*dcos(th(i))
      U(2,3)=-dsin(alph(i))*dcos(th(i))
      U(3,1)=0.d0
      U(3,2)=dsin(alph(i))
      U(3,3)=dcos(alph(i))
      return
      end

      Subroutine magnvec (G, mG)
* Returns the magnitude of a vector.
      double precision G(3), mG
      mG=dsqrt(G(1)**2+G(2)**2+G(3)**2)
      return
      end

      Subroutine anglevec (A, B, k, ang)
* Computes the angle between two vectors from the dot-product of
* the vectors.
      double precision A(3), B(3,n+1), ang, D(3), mA, mD, dp,
     & pi, rdp
      integer k
      common /tm/ n, pi
      dp=A(1)*B(1,k)+A(2)*B(2,k)+A(3)*B(3,k)
      call magnvec (A, mA)
      do 41 j=1,3
41      D(j)=B(j,k)
      call magnvec (D, mD)
      rdp=(dint(dp/(mA*mD)*1.d12))/1.d12
      ang=dacos(rdp)
      return
      end

      Subroutine mangle (th, alph, M, sig)
* Compute the miss angle and the miss axis based on the imaginary screw
```

```
* motion of vector V(i,n+1) rotating to coincide with vector V(i,1).
      double precision th(n), alph(n), M(3), sig, B(3,3), R(3,3),
     & pi, U(3,3), sc
      common /tm/ n, pi

* Develop transformation matrix B.
* Initialize B as I.
      do 30 i=1,3
      do 30 j=1,3
        if (i.eq.j) then
          B(i,j)=1.d0
          else
            B(i,j)=0.d0
        endif
30    continue
* Perform successive multiplications of U matrices to construct B.
      do 31 i=1,n
        call makeU (th, i, alph, U)
        call mult33mat (B, U, B)
31    continue

* Obtain R by transposing B.
      call transp (B, R)

* Compute angle of rotation.
      sc=(R(1,1)+R(2,2)+R(3,3)-1.d0)/2.d0
* Correct for possible propagation of errors.
      if ((dabs(sc)).gt.1d0) then
        sc=dint(sc)
      endif
      sig=dacos(sc)

      if (sig.ne.0.d0) then
* Develop miss axis from rotation matrix and angle of rotation.
        M(1)=(R(3,2)-R(2,3))/(2.d0*dsin(sig))
        M(2)=(R(1,3)-R(3,1))/(2.d0*dsin(sig))
        M(3)=(R(2,1)-R(1,2))/(2.d0*dsin(sig))
      endif

      return
      end

      Subroutine transp (B, R)
* Transposes a 3x3 matrix B to make R.
      double precision B(3,3), R(3,3), pi
      common /tm/ n, pi
      do 38 i=1,3
      do 38 j=1,3
38      R(i,j)=B(j,i)
      return
      end

      Subroutine mult33mat (A, B, C)
* Multiplies 3x3 matrices A and B to produce C.
      double precision A(3,3), B(3,3), C(3,3), pi
      common /tm/ n, pi
      do 35 ir=1,3
      do 35 ic=1,3
        C(ir,ic)=0.d0
35      C(ir,ic)=A(ir,1)*B(1,ic)+A(ir,2)*B(2,ic)+A(ir,3)*B(3,ic)
```

```
      return
      end

      Subroutine angcomp (phi, s)
* Determine the joint axis which is closest to being collinear with
* the miss axis (represented by the phi angle furthest from pi/2.)
      double precision phi(n), phi0, phi1, pi
      integer s
      common /tm/ n, pi
      phi0=0.d0
      s=0
      do 5 i=2,n
        phi1=dabs((pi/2.d0)-phi(i))
        if (phi1.ge.phi0) then
          s=i
          phi0=phi1
        endif
5       continue
      return
      end

      Subroutine setcorr (phi, s, sig, sigcor)
* Determine the desired magnitude and direction of the correction angle.
      double precision phi(n), sig, sigcor, pi
      integer s
      common /tm/ n, pi
* Determine the magnitude based on a percentage of the miss angle.
      if ((phi(s).gt.(0.9*pi/2.d0)).and.(phi(s).lt.(1.1*pi/2.d0))) then
* Establish minimum correction angle (in case phi(i) is very close
* to pi/2).
        sigcor=sig*0.1
          else
            sigcor=(dabs(pi/2.d0-phi(s))/(pi/2.d0))*sig/4.d0
      endif
* Accommodate step size for the case of a very small angle.
      if (sig.lt.1.d-2) then
        sigcor=sigcor*2.d0
      endif
* Determine the sign of the correction angle based on the relative
* orientation of the joint axis and the miss axis.
      if (phi(s).gt.(pi/2.d0)) then
        sigcor=-sigcor
      endif
      return
      end

      Subroutine mat31mult (A, B, C)
* Multiplies a 3x3 matrix A and a 3x1 matrix B to produce a 3x1
* matrix C.
      double precision A(3,3), B(3), C(3), pi
      common /tm/ n, pi
      do 2 ir=1,3
        C(ir)=0.d0
        C(ir)=A(ir,1)*B(1)+A(ir,2)*B(2)+A(ir,3)*B(3)
2       continue
      return
      end

      Subroutine corrth (th)
* Routine to correct all joint angles to be between 0 and 2*pi.
```

```
      double precision th(n), pi
      common /tm/ n, pi
      do 7 i=2,n
        if (th(i).lt.0.d0) then
          th(i)=th(i)+2.d0*pi
        endif
        if (th(i).gt.2.d0*pi) then
          th(i)=th(i)-2.d0*pi
        endif
7       continue
      return
      end
```

APPENDIX C

RESULTS FOR A CLOSED-LOOP EXAMPLE

The following is the complete list of intermediate data from the example of the MAIM method applied to a closed-loop mechanism presented in Section 5.2. This table contains, for each iteration, the iteration number, 'it. #', the current joint angles at that iteration, $\theta_1, \ldots, \theta_n$, and the miss angle, σ, for the mechanism in the configuration resulting from those joint angles. The joint angle shown in boldface indicates the joint axis which is closest to the miss axis in that configuration. This joint angle is the one to be adjusted. The adjustment step-size is not shown in the table but can easily be found by determining the difference between two successive values of an adjusted joint angle.

Table C.1 Results for the MAIM method applied to the Cardan joint example.

it. #	θ_1	θ_2	θ_3	θ_4	σ
0	0.000	**50.000**	320.000	120.000	66.354
1	0.000	61.367	320.000	**120.000**	56.578
2	0.000	**61.367**	320.000	110.935	49.137
3	0.000	69.284	320.000	**110.935**	42.616
4	0.000	**69.284**	320.000	104.678	37.809
5	0.000	**74.778**	320.000	104.678	33.624
6	0.000	79.147	320.000	**104.678**	30.583
7	0.000	79.147	**320.000**	100.880	28.038
8	0.000	79.147	316.291	**100.880**	25.419
9	0.000	79.147	**316.291**	97.805	23.405
10	0.000	**79.147**	313.303	97.805	21.357
11	0.000	81.899	**313.303**	97.805	19.457
12	0.000	**81.899**	310.880	97.805	17.831
13	0.000	84.126	310.880	**97.805**	16.332
14	0.000	84.126	**310.880**	95.844	15.056
15	0.000	**84.126**	308.891	95.844	13.653
16	0.000	85.726	**308.891**	95.844	12.635
17	0.000	85.726	307.283	**95.844**	11.535
18	0.000	85.726	**307.283**	94.391	10.552

Table C.1 (continued)

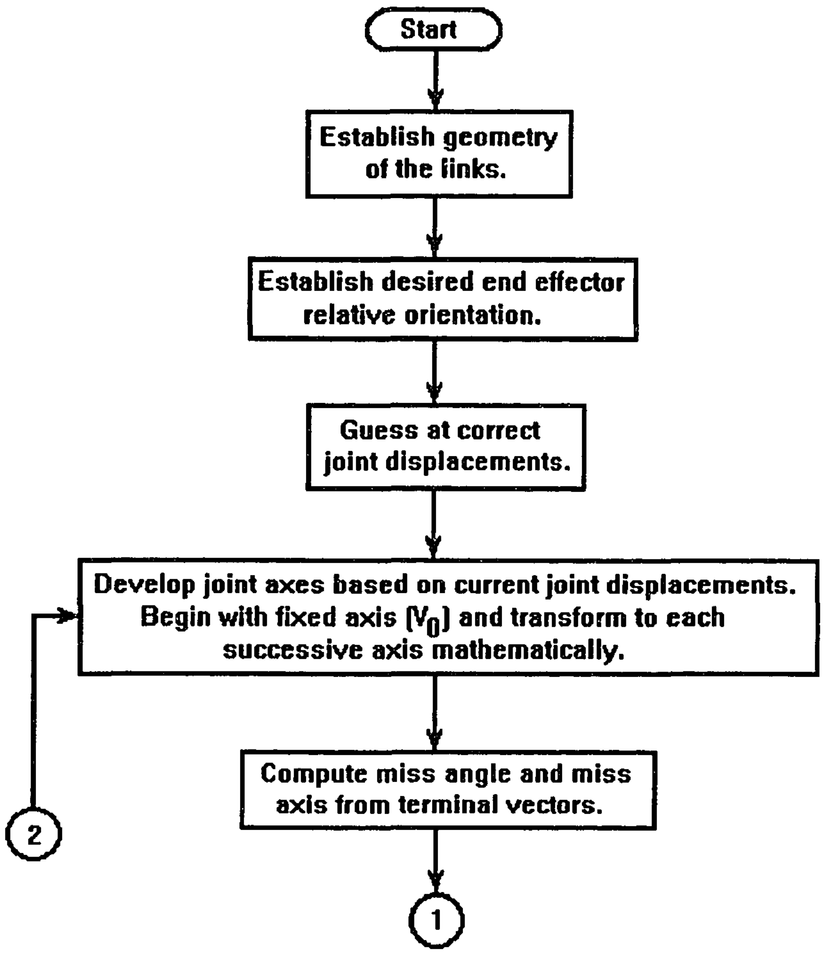
it. #	θ_1	θ_2	θ_3	θ_4	σ
19	0.000	85.726	305.974	**94.391**	9.677
20	0.000	85.726	**305.974**	93.210	8.899
21	0.000	**85.726**	304.911	93.210	8.210
22	0.000	86.795	**304.911**	93.210	7.465
23	0.000	86.795	304.042	**93.210**	6.915
24	0.000	**86.795**	304.042	92.342	6.328
25	0.000	87.579	**304.042**	92.342	5.804
26	0.000	**87.579**	303.321	92.342	5.322
27	0.000	88.219	**303.321**	92.342	4.905
28	0.000	88.219	302.734	**92.342**	4.524
29	0.000	88.219	**302.734**	91.760	4.124
30	0.000	**88.219**	302.253	91.760	3.819
31	0.000	88.694	302.253	**91.760**	3.501
32	0.000	88.694	**302.253**	91.332	3.218
33	0.000	88.694	301.853	**91.332**	2.950
34	0.000	88.694	**301.853**	90.982	2.724
35	0.000	**88.694**	301.527	90.982	2.512
36	0.000	89.016	**301.527**	90.982	2.292
37	0.000	**89.016**	301.259	90.982	2.122
38	0.000	89.277	301.259	**90.982**	1.948
39	0.000	89.277	**301.259**	90.744	1.792
40	0.000	89.277	301.036	**90.744**	1.642
41	0.000	89.277	**301.036**	90.550	1.517
42	0.000	**89.277**	300.854	90.550	1.399
43	0.000	89.455	**300.854**	90.550	1.278
44	0.000	89.455	300.705	**90.550**	1.183
45	0.000	**89.455**	300.705	90.405	1.086
46	0.000	89.586	**300.705**	90.405	1.000
47	0.000	**89.586**	300.580	90.405	0.916
48	0.000	89.694	**300.580**	90.405	0.847
49	0.000	89.694	300.478	**90.405**	0.781
50	0.000	89.694	**300.478**	90.306	0.714
51	0.000	**89.694**	300.395	90.306	0.661
52	0.000	89.775	300.395	**90.306**	0.607
53	0.000	89.775	**300.395**	90.232	0.559
54	0.000	89.775	300.255	**90.232**	0.471
55	0.000	**89.775**	300.255	90.109	0.390
56	0.000	89.874	**300.255**	90.109	0.326
57	0.000	**89.874**	300.162	90.109	0.260
58	0.000	89.938	**300.162**	90.109	0.221
59	0.000	89.938	300.104	**90.109**	0.182
60	0.000	89.938	**300.104**	90.058	0.147

Table C.1 (continued)

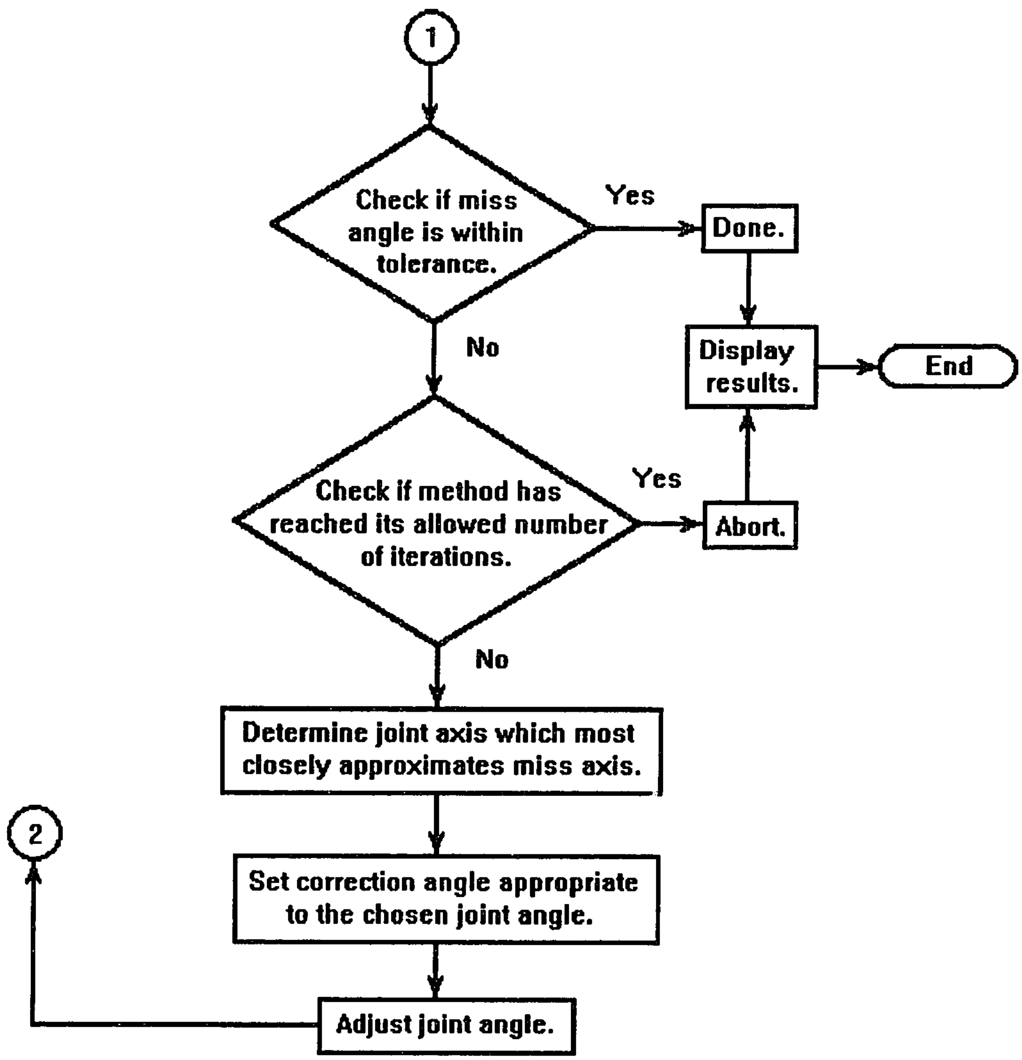
it. #	θ_1	θ_2	θ_3	θ_4	σ
61	0.000	**89.938**	300.067	90.058	0.124
62	0.000	89.970	300.067	**90.058**	0.102
63	0.000	89.970	**300.067**	90.032	0.086
64	0.000	89.970	300.043	**90.032**	0.068
65	0.000	89.970	**300.043**	90.016	0.058
66	0.000	**89.970**	300.027	90.016	0.048
67	0.000	89.984	**300.027**	90.016	0.039
68	0.000	**89.984**	300.018	90.016	0.033
69	0.000	89.993	300.018	**90.016**	0.027
70	0.000	89.993	**300.018**	90.009	0.023
71	0.000	89.993	300.011	**90.009**	0.018
72	0.000	89.993	**300.011**	90.004	0.015
73	0.000	**89.993**	300.007	90.004	0.013
74	0.000	89.996	**300.007**	90.004	0.010
75	0.000	89.996	300.005	90.004	0.009

APPENDIX D

OPEN-LOOP METHOD FLOWCHART



OPEN-LOOP METHOD FLOWCHART
(Continued)



APPENDIX E

OPEN-LOOP METHOD PROGRAM CODE (FORTRAN)

```
* Miss Angle Iteration Method (MAIM)
* A Spherical Mechanism Orientation Program
* Open-loop version

* Method developed by John D. Kliminski,
* incorporating concepts proposed by Dr. Ian S. Fischer
* Program written by John D. Kliminski
* Via NJIT 1994              Updated: 10/19/94


*   Variable List:
* n          - number of links in the mechanism.
* th(n)      - joint angles, theta.
* alph(n)    - twist angles of the links, alpha.
* U(3,3)     - transformation matrix for a given set of coordinate
*            - frames.
* V(i,k)     - array of joint axes; the i index represents the x,y,z
*              vector coordinates of the axis and k is the joint index.
*              V is therefore a matrix of column vectors of dimension
*              (3,n+1).
* M(3)       - miss axis in x,y,z vector coordinates.
* sig        - miss angle, sigma.
* stol       - the acceptable tolerance of the magnitude of the miss
*            - angle.
* phi(n)     - element phi(i) is the angle between joint axis i and the
*              miss axis.
* s          - index of the joint axis which is nearest to collinear with
*              the miss axis.
* sigcor     - the amount of correction to a joint angle on a given
*            - iteration.
* it         - counter for the number of iterations performed.
* itmax      - prescribed maximum number of iterations allowed.
* R(3,3)     - general rotational transformation matrix.
* E(3,3)     - Euler rotation matrix expressing the hand frame in terms
*              of the base frame.
* ET(3,3)    - transpose of E (expressing the base frame in terms of the
*              hand frame).
* ap,bt,gm   - ZYX Euler angles (alpha, beta, gamma) expressing the
*              orientation of the hand frame.
* d2r,r2d    - conversion factors for degrees to/from radians.
* Y(i),Z(i)  - temporary storage arrays for vectors.


*                         --- Main Program ---

* Declare variables.
      integer n, it, itmax, s
      double precision alph(12), th(12), V(3,0:13),
     & M(3), sig, stol, sigcor, phi(12), p, pi, d2r, r2d,
     & ap, bt, gm
```

```
* Declare fundamental trigonometric and mechanism data as common.
      common /tm/ n, pi
      pi=4.d0*datan(1.d0)
      d2r=pi/180.d0
      r2d=180.d0/pi

      write(6,*)
      write(6,*) '  Miss Angle Iteration Method (MAIM) '
     & 'for Open-Loop Mechanisms'
      write(6,*)
      write(6,*)
      write(6,*) '(Enter angles on one line separated by commas '
     & 'or on separate lines.)'
      write(6,*)

* Input the specific data for the mechanism and convert all angles
* from degrees to radians.
      write(6,*) 'Enter the number of links in the manipulator: '
      read(5,*) n
      write(6,*) 'Enter the twist of each link (degrees): '
      read(5,*) (alph(i), i=1,n)
      do 12 i=1,n
12      alph(i)=alph(i)*d2r
      write(6,*) 'Enter the initial guesses for the joint',
     & ' angles (degrees): '
      read(5,*) (th(i), i=1,n)
      do 14 i=1,n
14      th(i)=th(i)*d2r
      write(6,*) 'Enter the tolerance for the miss angle (degrees): '
      read(5,*) stol
      stol=stol*d2r
      write(6,*) 'Enter the maximum number of iterations: '
      read(5,*) itmax
      write(6,*) 'Enter the ZYX Euler angles for the hand frame ',
     & 'coordinates - alpha, beta, gamma (degrees): '
      read(5,*) ap, bt, gm
      ap=ap*d2r
      bt=bt*d2r
      gm=gm*d2r
      write(6,*)
      write(6,*) 'Ok.'
      write(6,*)
      write(6,*) 'Working...'

* Initialize iteration counter.
      it=0
* Begin iterative procedure.
111   continue

* Develop joint axes.
      call axes (th, alph, ap, bt, gm, V)

* Calculate the miss angle and the miss axis vector.
      call mangle (th, alph, ap, bt, gm, M, sig,V)

* If current miss angle is within tolerance, end program.
      if (sig.le.stol) then
        goto 999
      endif
```

```
* If limit of iterations is exceeded, end program.
      if (it.ge.itmax) then
        goto 999
      endif

* Determine which joint angle to adjust to improve the miss angle.
* Calculate the angle between each joint axis, V, and the miss axis, M.
      do 4 i=1,n
        call anglevec (M, V, i, p)
4       phi(i)=p
* Select the joint axis closest to the miss axis.
      call angcomp (phi, s)

* Determine the magnitude and direction of the correction angle.
      call setcorr (phi, s, sig, sigcor)

* Adjust the appropriate joint angle by the correction angle.
      th(s)=th(s)+sigcor

* Reset all joint variables to range from 0 to 2*pi.
      call corrth (th)

* Increment iteration counter.
      it=it+1

* Repeat method for next iteration.
      goto 111

999   continue
* Display final results.
      write(6,*)
      if (it.ge.itmax) then
        write(6,*) '        *** Maximum iterations reached --- ',
     &   'method aborted ***'
        write(6,*)
      endif
      write(6,*)
      if (sig.le.stol) then
        write(6,*) '            Method successfully completed.'
        write(6,*)
      endif
      write(6,*) 'The Miss Angle was reduced to ',sig*r2d,' (degrees)'
      write(6,*) 'after',it,' iterations.'
      if (sig.le.stol) then
        write(6,*)
        write(6,*) 'This Miss Angle is within the specified',
     & ' tolerance of ',stol*r2d,' (degrees).'
        write(6,*) 'For the end effector coordinates of ',
     & ap*r2d,',',bt*r2d,',',gm*r2d,' (degrees) in ZYX Euler angles.'
      endif
      write(6,*)
      write(6,*) 'The final results (in degrees) are: '
      do 91 i=1,n
91    write(6,95) i,th(i)*r2d
95    format(' theta ',i2,' = ',f10.6)
      write(6,*)

      end
```

```
*                              --- Subroutines ---
      Subroutine axes (th, alph, ap, bt, gm, V)
      double precision th(n), alph(n), ap, bt, gm, V(3,0:n+1), pi,
     & E(3,3), ET(3,3), U(3,3), Z(3), Y(3)
      common /tm/ n, pi
* Develop joint axes, V0 through Vn+1, by matrix transformations.
* Define initial axis in hand frame, V(i,0) (of unit length).
      V(1,0)=0.d0
      V(2,0)=0.d0
      V(3,0)=1.d0
* Transform from hand axis, V(i,0), to first joint axis, V(i,1).
      call makeE (ap, bt, gm, E)
      call transp (E, ET)
* Set temporary storage array Z as V(i,0).
      do 26 i=1,3
26      Z(i)=V(i,0)
      call mat31mult (ET, Z, Y)
      do 27 i=1,3
27      V(i,1)=Y(i)
* Obtain other axes, V(i,2) through V(i,n+1), by successive
* post-multiplication of V(i,1) by the U transformation matrix.
      do 2 j=1,n
* Initialize temporary storage array Z as V(i,1).
        do 22 i=1,3
22        Z(i)=V(i,1)
* Post-multiply by the appropriate transformation matrices.
        do 24 i=j,1,-1
          call makeU (th, i, alph, U)
          call mat31mult (U, Z, Y)
          do 23 ii=1,3
23          Z(ii)=Y(ii)
24        continue
* Return resulting matrix product Z to V(i,j+1), rounding off elements
* to eliminate multiplication precision errors.
        do 21 i=1,3
          Z(i)=(dint(Z(i)*1.d12))/1.d12
21        V(i,j+1)=Z(i)
2       continue
      return
      end

      Subroutine makeE (ap, bt, gm, E)
* Create matrix to transform from base frame to hand frame using
* ZYX Euler angles.
      double precision ap, bt, gm, E(3,3), pi
      common /tm/ n, pi
      E(1,1)=dcos(ap)*dcos(bt)
      E(1,2)=dcos(ap)*dsin(bt)*dsin(gm)-dsin(ap)*dcos(gm)
      E(1,3)=dcos(ap)*dsin(bt)*dcos(gm)+dsin(ap)*dsin(gm)
      E(2,1)=dsin(ap)*dcos(bt)
      E(2,2)=dsin(ap)*dsin(bt)*dsin(gm)+dcos(ap)*dcos(gm)
      E(2,3)=dsin(ap)*dsin(bt)*dcos(gm)-dcos(ap)*dsin(gm)
      E(3,1)=-dsin(bt)
      E(3,2)=dcos(bt)*dsin(gm)
      E(3,3)=dcos(bt)*dcos(gm)
      return
      end

      Subroutine makeU (th, i, alph, U)
```

```
* Creates appropriate U transformation matrices.
      double precision th(n), alph(n), U(3,3), pi
      integer i
      common /tm/ n, pi
      U(1,1)=dcos(th(i))
      U(1,2)=-dcos(alph(i))*dsin(th(i))
      U(1,3)=dsin(alph(i))*dsin(th(i))
      U(2,1)=dsin(th(i))
      U(2,2)=dcos(alph(i))*dcos(th(i))
      U(2,3)=-dsin(alph(i))*dcos(th(i))
      U(3,1)=0.d0
      U(3,2)=dsin(alph(i))
      U(3,3)=dcos(alph(i))
      return
      end

      Subroutine magnvec (G, mG)
* Returns the magnitude of a vector.
      double precision G(3), mG
      mG=dsqrt(G(1)**2+G(2)**2+G(3)**2)
      return
      end

      Subroutine anglevec (A, B, k, ang)
* Computes the angle between two vectors from the dot-product of
* the vectors.
      double precision A(3), B(3,0:n+1), ang, D(3), mA, mD, dp,
     & pi, rdp
      integer k
      common /tm/ n, pi
      dp=A(1)*B(1,k)+A(2)*B(2,k)+A(3)*B(3,k)
      call magnvec (A, mA)
      do 41 j=1,3
41      D(j)=B(j,k)
      call magnvec (D, mD)
      rdp=(dint(dp/(mA*mD)*1.d12))/1.d12
      ang=dacos(rdp)
      return
      end

      Subroutine mangle (th, alph, ap, bt, gm, M, sig,V)
* Compute the miss angle and the miss axis based on the imaginary screw
* motion of vector V(i,n+1) rotating to coincide with vector V(i,1).
      double precision th(n), alph(n), M(3), sig, B(3,3), R(3,3),
     & E(3,3), ET(3,3), ap, bt, gm, U(3,3), sc, pi,
     & V(3,0:n+1)
      common /tm/ n, pi

* Develop transformation matrix B.
* Initialize B as I.
      do 30 i=1,3
      do 30 j=1,3
        if (i.eq.j) then
          B(i,j)=1.d0
          else
            B(i,j)=0.d0
        endif
30    continue
* Transform from hand to base.
      call makeE (ap, bt, gm, E)
```

```
      call transp (E, ET)
      call mult33mat (B, ET, B)
* Perform successive multiplications of U matrices to construct B.
      do 31 i=1,n
        call makeU (th, i, alph, U)
        call mult33mat (B, U, B)
31    continue

* Obtain R by transposing B.
      call transp (B, R)

* Compute angle of rotation.
      sc=(R(1,1)+R(2,2)+R(3,3)-1.d0)/2.d0
* Correct for possible propagation of errors.
      if ((dabs(sc)).gt.1d0) then
        sc=dint(sc)
      endif
      sig=dacos(sc)

      if (sig.ne.0.d0) then
* Develop miss axis from rotation matrix and angle of rotation.
        M(1)=(R(3,2)-R(2,3))/(2.d0*dsin(sig))
        M(2)=(R(1,3)-R(3,1))/(2.d0*dsin(sig))
        M(3)=(R(2,1)-R(1,2))/(2.d0*dsin(sig))
      endif

      return
      end

      Subroutine transp (B, R)
* Transposes a 3x3 matrix B to make R.
      double precision B(3,3), R(3,3), pi
      common /tm/ n, pi
      do 38 i=1,3
      do 38 j=1,3
38      R(i,j)=B(j,i)
      return
      end

      Subroutine mult33mat (A, B, C)
* Multiplies 3x3 matrices A and B to produce C.
      double precision A(3,3), B(3,3), C(3,3), pi
      common /tm/ n, pi
      do 35 ir=1,3
      do 35 ic=1,3
        C(ir,ic)=0.d0
35      C(ir,ic)=A(ir,1)*B(1,ic)+A(ir,2)*B(2,ic)+A(ir,3)*B(3,ic)
      return
      end

      Subroutine angcomp (phi, s)
* Determine the joint axis which is closest to being collinear with
* the miss axis (represented by the phi angle furthest from pi/2.)
      double precision phi(n), phi0, phi1, pi
      integer s
      common /tm/ n, pi
      phi0=0.d0
      s=0
      do 5 i=1,n
        phi1=dabs((pi/2.d0)-phi(i))
```

```
        if (phi1.ge.phi0) then
          s=i
          phi0=phi1
        endif
5       continue
      return
      end

      Subroutine setcorr (phi, s, sig, sigcor)
* Determine the desired magnitude and direction of the correction angle.
      double precision phi(n), sig, sigcor, pi
      integer s
      common /tm/ n, pi
* Determine the magnitude based on a percentage of the miss angle.
      if ((phi(s).gt.(0.95*pi/2.d0)).and.
     & (phi(s).lt.(1.05*pi/2.d0))) then
* Establish minimum correction angle (in case phi(i) is very close
* to pi/2).
        sigcor=sig*0.1
          else
            sigcor=(dabs(pi/2.d0-phi(s))/(pi/2.d0))*sig/2.d0
      endif
* Accommodate step size for the case of a very small angle.
      if (sig.lt.1.d-1) then
        sigcor=sigcor*2.d0
      endif
* Determine the sign of the correction angle based on the relative
* orientation of the joint axis and the miss axis.
      if (phi(s).gt.(pi/2.d0)) then
        sigcor=-sigcor
      endif
      return
      end

      Subroutine mat31mult (A, B, C)
* Multiplies a 3x3 matrix A and a 3x1 matrix B to produce a 3x1
* matrix C.
      double precision A(3,3), B(3), C(3), pi
      common /tm/ n, pi
      do 2 ir=1,3
        C(ir)=0.d0
        C(ir)=A(ir,1)*B(1)+A(ir,2)*B(2)+A(ir,3)*B(3)
2       continue
      return
      end

      Subroutine corrth (th)
* Routine to correct all joint angles to be between 0 and 2*pi.
      double precision th(n), pi
      common /tm/ n, pi
      do 7 i=1,n
        if (th(i).lt.0.d0) then
          th(i)=th(i)+2.d0*pi
        endif
        if (th(i).gt.2.d0*pi) then
          th(i)=th(i)-2.d0*pi
        endif
7       continue
      return
      end
```

APPENDIX F

RESULTS FOR AN OPEN-LOOP EXAMPLE

The following is the complete list of intermediate data from the example of the MAIM method applied to an open-loop mechanism presented in Section 5.3. This table contains, for each iteration, the iteration number, "it. #", the current joint angles at that iteration, $\theta_1, \ldots, \theta_n$, and the miss angle, σ, for the mechanism in the configuration resulting from those joint angles. The joint angle shown in boldface indicates the joint axis which is closest to the miss axis in that configuration. This joint angle is the one to be adjusted. The adjustment step-size is not shown in the table but can easily be found by comparing two successive values of an adjusted joint angle.

Table F.1 Results for the MAIM method applied to the SSRMS arm example.

it. #	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	σ
0	10.000	0.000	11.000	0.000	**0.000**	0.000	2.000	153.090
1	10.000	0.000	11.000	0.000	**26.125**	0.000	2.000	142.775
2	10.000	0.000	11.000	0.000	50.730	**0.000**	2.000	134.640
3	10.000	0.000	11.000	0.000	**50.730**	336.157	2.000	134.099
4	10.000	0.000	11.000	0.000	82.379	336.157	**2.000**	120.420
5	10.000	**0.000**	11.000	0.000	82.379	336.157	332.612	123.217
6	10.000	32.411	11.000	0.000	**82.379**	336.157	332.612	92.157
7	10.000	**32.411**	11.000	0.000	105.935	336.157	332.612	88.264
8	10.000	**61.534**	11.000	0.000	105.935	336.157	332.612	62.211
9	10.000	81.330	11.000	0.000	105.935	**336.157**	332.612	46.339
10	10.000	**81.330**	11.000	0.000	105.935	352.735	332.612	44.882
11	10.000	97.309	11.000	0.000	105.935	**352.735**	332.612	31.706
12	10.000	**97.309**	11.000	0.000	105.935	8.399	332.612	34.731
13	10.000	**111.911**	11.000	0.000	105.935	8.399	332.612	24.221
14	10.000	123.172	11.000	0.000	**105.935**	8.399	332.612	19.929
15	10.000	123.172	11.000	0.000	**97.829**	8.399	332.612	12.509
16	10.000	123.172	**11.000**	0.000	92.112	8.399	332.612	8.103
17	10.000	123.172	**7.511**	0.000	92.112	8.399	332.612	6.505
18	**10.000**	123.172	5.498	0.000	92.112	8.399	332.612	6.318

Table F.1 (continued)

it. #	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	σ
19	8.015	123.172	**5.498**	0.000	92.112	8.399	332.612	4.788
20	**8.015**	123.172	2.916	0.000	92.112	8.399	332.612	5.095
21	4.153	123.172	2.916	0.000	92.112	**8.399**	332.612	2.080
22	4.153	123.172	**2.916**	0.000	92.112	9.608	332.612	3.015
23	**4.153**	123.172	1.587	0.000	92.112	9.608	332.612	3.172
24	2.198	123.172	1.587	0.000	92.112	**9.608**	332.612	1.914
25	**2.198**	123.172	1.587	0.000	92.112	10.880	332.612	2.769
26	1.024	123.172	1.587	0.000	92.112	**10.880**	332.612	2.243
27	1.024	123.172	**1.587**	0.000	92.112	12.315	332.612	3.262
28	**1.024**	123.172	359.947	0.000	92.112	12.315	332.612	3.291
29	359.070	123.172	359.947	0.000	92.112	**12.315**	332.612	2.079
30	**359.070**	123.172	359.947	0.000	92.112	13.669	332.612	2.991
31	357.832	**123.172**	359.947	0.000	92.112	13.669	332.612	2.448
32	357.832	124.713	359.947	0.000	**92.112**	13.669	332.612	1.626
33	**357.832**	124.713	359.947	0.000	90.786	13.669	332.612	1.405
34	356.758	124.713	359.947	0.000	**90.786**	13.669	332.612	0.560
35	**356.758**	124.713	359.947	0.000	90.463	13.669	332.612	0.535
36	356.449	124.713	359.947	0.000	90.463	**13.669**	332.612	0.349
37	356.449	124.713	359.947	0.000	**90.463**	13.891	332.612	0.501
38	**356.449**	124.713	359.947	0.000	90.249	13.891	332.612	0.522
39	356.142	124.713	359.947	0.000	90.249	**13.891**	332.612	0.332
40	**356.142**	124.713	359.947	0.000	90.249	14.108	332.612	0.470
41	355.945	124.713	359.947	0.000	90.249	**14.108**	332.612	0.383
42	355.945	124.713	359.947	0.000	**90.249**	14.347	332.612	0.548
43	**355.945**	124.713	359.947	0.000	89.969	14.347	332.612	0.552
44	355.613	124.713	359.947	0.000	89.969	**14.347**	332.612	0.344
45	**355.613**	124.713	359.947	0.000	89.969	14.566	332.612	0.490
46	355.412	**124.713**	359.947	0.000	89.969	14.566	332.612	0.403
47	355.412	124.962	359.947	0.000	**89.969**	14.566	332.612	0.277
48	**355.412**	124.962	359.947	0.000	89.742	14.566	332.612	0.239
49	355.229	124.962	359.947	0.000	**89.742**	14.566	332.612	0.093
50	**355.229**	124.962	359.947	0.000	89.686	14.566	332.612	0.089
51	355.175	124.962	359.947	0.000	89.686	**14.566**	332.612	0.055
52	355.175	124.962	359.947	0.000	**89.686**	14.600	332.612	0.079
53	**355.175**	124.962	359.947	0.000	89.651	14.600	332.612	0.082
54	355.126	124.962	359.947	0.000	89.651	**14.600**	332.612	0.051
55	**355.126**	124.962	359.947	0.000	89.651	14.633	332.612	0.072
56	355.096	124.962	359.947	0.000	89.651	**14.633**	332.612	0.059
57	355.096	124.962	359.947	0.000	**89.651**	14.669	332.612	0.084
58	**355.096**	124.962	359.947	0.000	89.608	14.669	332.612	0.084
59	355.045	124.962	359.947	0.000	89.608	**14.669**	332.612	0.052
60	**355.045**	124.962	359.947	0.000	89.608	14.702	332.612	0.074

Table F.1 (continued)

it. #	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	σ
61	355.015	**124.962**	359.947	0.000	89.608	14.702	332.612	0.061
62	355.015	125.000	359.947	0.000	**89.608**	14.702	332.612	0.042
63	**355.015**	125.000	359.947	0.000	89.573	14.702	332.612	0.036
64	354.987	125.000	359.947	0.000	**89.573**	14.702	332.612	0.014
65	**354.987**	125.000	359.947	0.000	89.565	14.702	332.612	0.013
66	354.979	125.000	359.947	0.000	89.565	14.702	332.612	0.008

REFERENCES

Craig, John J., 1986, *Introduction to Robotics: Mechanics and Control*, Addison-Wesley Publishing Company, Reading, Massachusetts, ISBN 0-201-10326-5, pp. 42-44, 46-49.

Crane, Carl D., Carnahan, Tim, and Duffy, Joseph, 1991, "A Kinematic Analysis of the Space Station Remote Manipulator System (SSRMS)," *Journal of Robotic Systems*, Vol. 8, pp. 637-658.

Denavit, J., and Hartenberg, R. S., 1955, "A Kinematic Notation for Lower Pair Mechanisms Based on Matrices," *ASME Journal of Applied Mechanics*, June, pp. 215-221.

Doughty, Samuel, 1988, *Mechanics of Machines*, John Wiley & Sons, Inc., New York, New York, ISBN 0-471-84276-1, pp. 1-2.

Fischer, I. S., 1988, "Application of the Principle of Transference to the Evaluation of Translational Displacements in Spatial Mechanisms," *Trends and Developments in Mechanisms, Machines, and Robotics*, Vol. 2, pp. 7-11.

Fischer, I. S., 1989, "Displacement Errors in Cardan Joints Caused by Coupler-Link Joint-Axis Offset," *First National Applied Mechanisms and Robotics Conference*, Cincinnati, Ohio, pp. 921-926.

Fischer, I. S., Spring 1993, Lecture notes, NJIT graduate course ME637 "Spatial Mechanisms."

Fu, K. S., Gonzalez, R. C., and Lee, C. S. G., 1987, *Robotics: Control, Sensing, Vision, and Intelligence*, McGraw-Hill Book Company, New York, New York, ISBN 0-07-022625-3, pp. 6-7.

Lee, H. Y., Woernle, C., and Hiller, M., 1991, "A Complete Solution for the Inverse Kinematic Problem of the General 6R Robot Manipulator," *Journal of Mechanical Design*, Vol. 113, pp. 481-486.

Manseur, Rachid, 1992, "A Complete Kinematic Analysis of Four-Revolute-Axis Robot Manipulators," *Mechanism and Machine Theory*, Vol. 27, pp. 575-586.

Nikravesh, Parviz E., 1988, *Computer-Aided Analysis of Mechanical Systems*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, ISBN 0-13-164220-0, pp. 35-38.

Paul, Richard P., 1981, *Robot Manipulators: Mathematics, Programming, and Control*, The MIT Press, Cambridge, Massachusetts, ISBN 0-262-16082-X, pp. 29-34, 43-47.

REFERENCES
(Continued)

Pennock, G. R., and Yang, A. T., 1985, "Application of Dual-Number Matrices to the Inverse Kinematics Problem of Robot Manipulators," *ASME Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 107, pp. 201-208.

Ridley, P. R., 1994a, "Robot Kinematics - I. Graphical Solution of the Inverse Equations of Closure," *Mechanism and Machine Theory*, Vol. 27, pp. 1043-1052.

Ridley, P. R., 1994b, "Robot Kinematics - II. Analytical Solution of the Inverse Equations of Closure," *Mechanism and Machine Theory*, Vol. 27, pp. 1053-1063.

Sandor, George N., and Erdman, Arthur G., 1984, *Advanced Mechanism Design: Analysis and Synthesis, Volume 2*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, ISBN 0-13-011437-5, pp. 1-2, 16-24, 609-664.

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.

Yang, A. T., and Freudenstein, F., 1964, "Application of Dual-Number Quaternion Algebra to the Analysis of Spatial Mechanisms," *ASME Journal of Applied Mechanics*, Vol. 86E, pp. 300-308.