THE KINEMATICS OF MANIPULATORS UNDER COMPUTER CONTROL

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING

AND THE COMMITTEE ON THE GRADUATE DIVISION

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

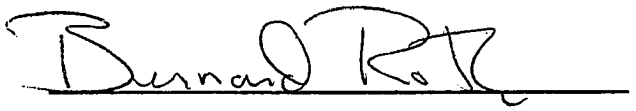
DOCTOR OF PHILOSOPHY

By

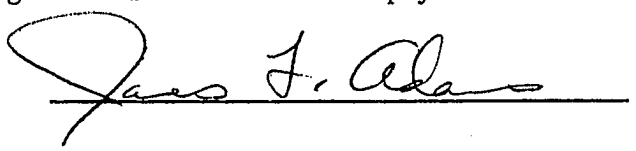
Donald Lee Pieper

October 1968

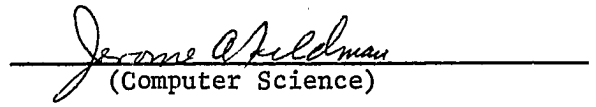
I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

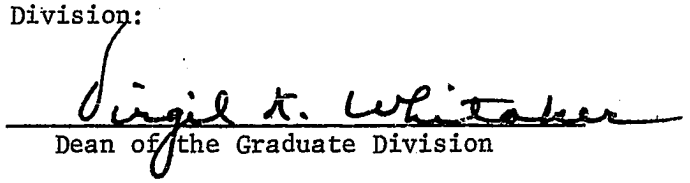


I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



(Computer Science)

Approved for the University Committee
on the Graduate Division:



Dean of the Graduate Division

THE KINEMATICS OF MANIPULATORS UNDER COMPUTER CONTROL

ABSTRACT

This dissertation is concerned with the kinematic analysis of computer controlled manipulators. Existing industrial and experimental manipulators are cataloged according to a new model which allows for the systematic description of both existing and new manipulators.

This work deals mainly with manipulators consisting of six degree-of-freedom open chains of articulated links with either turning (revolute) or sliding (prismatic) joints. The last link called the "hand" is the free end of the manipulator and has additional motion capabilities which make it possible to grasp objects.

The following problem is discussed: given the desired hand position and orientation along with the various link parameters defining the structure, what are the values of the manipulator variables that place the hand at the desired position with the desired orientation? Solutions to this problem are presented for any six degree-of-freedom manipulator with three revolute joints whose axes intersect at a point, provided the remaining three joints are revolute or prismatic pairs. These results can be expressed as a fourth degree polynomial in one unknown, and closed form expressions for the remaining unknowns.

It is shown that this is equivalent to the kinematic analysis of all single loop five-bar mechanisms with one spherical joint and four joints which are revolute or prismatic pairs. The extension to the case where only one pair of axes intersect is discussed. A similar solution for any manipulator with three prismatic joints is also given.

A numerical procedure based on velocity methods is developed to analyze manipulators which cannot be "solved" explicitly. This procedure is found to be superior to the widely used Newton-Raphson technique.

The problem of positioning a "digital arm" (i.e., a multi-link manipulator where each joint is only capable of several digital steps) is discussed. A simple searching algorithm using a look-ahead scheme is developed. A two-dimensional model and three-dimensional model are studied.

Given the solution to the position problem, a set of heuristics is developed for moving a six degree-of-freedom manipulator from an initial position to a final position through a space containing obstacles. A mathematical model of objects is developed so that possible conflict between objects and any link of the manipulator can be detected and avoided.

Some considerations in choosing a manipulator for use with a computer are discussed. A set of computer programs - in FORTRAN IV - are developed to perform the position analysis and trajectory generations for any six degree-of-freedom manipulator with turning joints.

ACKNOWLEDGEMENT

I would like to thank Professor B. Roth for suggesting this topic and advising me throughout the course of the research. I would also like to thank Professors J. Adams and J. Feldman for their reading and suggestions.

I am especially grateful to my wife Peggy for her help and encouragement.

Appreciation is also due Mrs. Judith Muller and Miss Vikki Locey for their excellent typing of the final report.

Financial support for this work was provided by the Advanced Research Projects Agency of the Office of the Secretary of Defense (SD-183).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

CHAPTER I

INTRODUCTION

Remote manipulation involves having a machine perform tasks requiring human dexterity. Originally, the purpose of a manipulator was to protect man from the hazards of performing the work himself. With the advance of technology, the variety of tasks performed in hostile environments has increased. In addition the scope of the tasks performed by machines has broadened, so that it is desirable for machines to extend the capabilities of men and to replace men at tedious as well as dangerous jobs. Although, today, many processes and machines are automatically controlled, the problems of remote manipulation have yet to be fully solved.

One approach to this problem is to use a digital computer to control a manipulator. Then with information obtained from visual as well as other sensory feedback, the computer would hopefully be able to direct the manipulator to perform tasks requiring some intelligence as well as dexterity.

This dissertation is concerned with the kinematic problems that arise when a manipulator is subjected to computer control. These include the problems of position analysis and trajectory generation.

In Chapter II, we discuss the classification and the description of manipulators, including a catalog of most of the existing commercial and special purpose manipulators.

The position problem is discussed in Chapter III. There we present methods to find values for the manipulator variables that will place the terminal device at a given position.

In Chapter IV, we present numerical methods that may be used to analyze manipulators too complex for analytic solution as described in Chapter III.

The problems of positioning a digital manipulator are discussed in Chapter V.

Trajectory generation - the problem of moving a manipulator from a given initial position to a specified final position - is studied in Chapter VI.

In Chapter VII we briefly discuss some considerations in choosing a manipulator for control by computer.

Chapter VIII presents the conclusions and some suggestions for future work.

In the next section we present a brief history of remote manipulation. This is followed by a summary of related work on intelligent automata. Since much of the research related to the position problem has occurred outside these fields, we discuss that work in Chapter III. In the last section of this chapter, the contribution of this dissertation to current research is presented.

1.1 History of Remote Manipulation

The development of remote manipulators followed closely the development of atomic energy. As the radiation level of atomic energy increased, so did the hazard to the operator. Thus, shielded environments and equipment to handle the material were needed. Early

experiments were carried out using tongs in shielded caves. For more complex experiments it was deemed necessary to develop remote controlled manipulators. It was felt that general purpose manipulators could be used to replace much special purpose equipment. Thus in 1947, the Argonne National Laboratory began research into remote manipulators and related equipment. The first manipulators built at Argonne had six degrees-of-freedom controlled by mechanical drives plus a hydraulically operated grip. Later versions were driven by electric motors. They worked well for simple tasks. However, there was no force feedback, making it difficult to perform experiments where articles came into contact with one another [1].*

In 1948 the people at Argonne decided to develop manipulators having force feedback with motion capability analogous to that of the human hand. This led to master-slave manipulators in which the motion of the master was mechanically coupled to the slave so that the forces in the slave would be approximately reflected in the master. Several versions of these were built at Argonne. One of these, the Model 8, has been produced by several companies and is commercially available [1, 2, 3, 4].

Although these mechanically coupled manipulators perform quite well, they have several drawbacks. The main disadvantage is the mechanical connection which requires the master and slave to be physically close together. This also means that the shielding enclosure must be designed

*Numbers in brackets designate references in the Bibliography (P.).

for the linkage. In addition the strength of the slave is limited by the strength of the operator's hand. These disadvantages are offset in part by the fact that the manipulators are fairly inexpensive and are able to perform intricate operations [1, 2, 3, 4, 5].

Externally powered master-slave manipulators using force reflecting servos have been developed by both Argonne and the General Electric Company. The Argonne machine is controlled with electromechanical servos while General Electric's ("Handyman") is hydro-mechanically controlled. These manipulators have proved as effective as the mechanically connected master-slaves. They have the advantage that the only connection between master and slave is an electrical cable. In addition, they have a variable force feedback ratio. However, their use is not as widespread as the mechanical type. Perhaps this is due to their high cost and the complexity of the force reflecting servo system [1, 6].

Powered manipulators, not of the master-slave type have also been successfully developed by General Mills, Inc., Programmed and Remote Systems Corporation, AMF, General Electric, Westinghouse Electric Company, FMC, among others. They are often used in radiation experiments along with the more precise master-slaves. They are also used in an underwater environment on submarines [7, 8]. Electric and hydraulic-powered prosthetic arms have also been developed [9, 10]. All these are generally controlled by joy sticks, toggle switches, or similar devices.

All of the above mentioned manipulators require the presence of a human operator. In their design much effort is made to have an integrated man-machine system. This is reflected in the research of Mosher [6, 11], Goertz [12], and Bradley [13] whose emphasis is directed

towards developing systems in which the operator does not feel his remoteness but is made to feel as if he were performing the task himself. This is accomplished with force reflecting servo-systems giving kinesthetic feedback similar to what a human would feel. Such work will have application in materials-handling, underwater work, and perhaps earth-moving equipment. It also may be applicable to problems of remote master-slave manipulators with time delay. Farrell [14] has indicated the feasibility of such schemes.

There are some problems that the master-slave system does not adequately solve. Since the master-slave system by definition requires a master, it does not remove the tedium that is basic to most manipulative tasks. In addition, for exploration of space, the time delay will become excessive for anything further distant than the moon. Thus we have motivation to develop manipulator systems with intelligence.

1.2 Intelligent Automata

Computer-manipulator systems such as AMF's Versatran and Unimation, Inc.'s Unimate [16] are presently in use in industrial materials-handling situations. These machines are programmed to move through a pre-determined series of positions. They are used on assembly lines to unload punch presses, conveyor belts and similar fixed cycle type operations. Working three shifts a day, they can economically compete with human operators [15]. However, they do not have any decision making ability, so that, if the parts are not in the right position or if the cycle time varies, these machines will not operate successfully. In addition they must be reprogrammed for slight changes in the process. It is thus desirable for such systems to incorporate decision making capabilities.

Ernst [18], using a manipulator equipped with sensory feedback, developed a hand-computer system capable of stacking blocks. His system was able to learn about its environment with information gained from touch sensors. The work at MIT's Project MAC [19] has recently extended the work of Ernst to include visual inputs and to develop a hand-eye system capable of manipulating objects. The aim of Project MAC is to develop an autonomous system with vision capable of performing manipulative tasks requiring increasing levels of decision making ability.

Rosen, Nilsson, Raphael, [20, 21, 22], and others at Stanford Research Institute have developed a mobile vehicle under computer control that performs tasks in a real environment. The primary goal is to develop a system receiving visual and other sensory information from the vehicle, and then use this information to direct the vehicle towards the completion of tasks requiring the abilities to plan ahead and learn from previous experience.

Other research in manipulator-computer systems has been in using small digital computers to assist rather than replace operators in manipulative tasks. Beckett [23] at Case Institute, has developed such a system in which a typical use of the computer is to find minimum transit time paths and direct the manipulator around predefined obstacles. In obstacle avoidance his routines keep the hand outside of effective boundaries placed around obstacles.

The Supervisory Controlled Manipulator, is again a system with limited intelligence intended to assist rather than replace an operator. For this system Whitney [24] developed a state-space model of manipulative tasks. He shows that tasks, such as pushing blocks on a table, or

deciding how many and in what order blocks should be moved or pushed aside in order to position a new block, may be expressed in terms of discrete state spaces. A state is defined to be the configuration of the task site.

The Hand-Eye Project, of the Stanford Artificial Intelligence Project [25] , is oriented toward solution of computer supervised hand-eye problems of increasing complexity. Current work is on basic problems involving manipulation of simple objects and analysis of visual data. Eventually it is hoped that the system will be developed to the point of being able to assemble machines.

1.3 Contributions of this Dissertation

In Chapter II the description of manipulators is put on a systematic basis. We present conditions leading to degeneracy in six degree-of-freedom manipulator and conditions in which combinations of one degree-of-freedom joints are kinematically equivalent to more complex joints. Finally, a catalog of existing manipulators is presented.

The main analytical work is presented in Chapter III. Here solutions to the position problem are discussed. Methods are given to solve any six degree-of-freedom manipulator containing three revolute joints, whose axes intersect at a point, provided the remaining three joints are revolutes or sliders. The extension of the method to more difficult arrangements is dealt with in the case where only one pair of revolute axes intersect. A method of solution for a six degree-of-freedom manipulator with three prismatic joints is also presented.

In Chapter IV a numerical procedure based on velocity methods is developed to analyze manipulators whose solutions cannot be expressed

as in Chapter III. This procedure, along with the more conventional Newton-Raphson method are programmed for a digital computer and the results compared.

In Chapter V methods are developed to place the end of a new type of digital manipulator at a specified position. A simple searching algorithm is made more powerful by the addition of look-ahead. The three dimensional problem is attacked with insight gained from studying a planar model.

The trajectory generation problem is discussed in Chapter VI. A set of heuristics is given for moving the manipulator from an initial position to a final position through a space containing obstacles. Possible conflict between all links of the manipulator and nearby obstacles is detected, and hopefully avoided.

In Chapter VII some considerations in choosing a manipulator for use with a digital computer are discussed. The desirability of being able to arbitrarily locate the hand throughout the workspace brings up the problem of zones. Some insight into this problem is presented.

Much of the above has been programmed and tested on a digital computer. In particular the numerical solutions and the heuristics for trajectory generation have been programmed to result in a fairly general kinematic package. With only small modification these routines could be used with any six degree-of-freedom manipulator.

CHAPTER II

CLASSIFICATION OF MANIPULATORS

2.1 The Basic Model

In order to analyze and compare manipulator configurations, it is desirable to develop a mathematical model that can be used to describe all manipulators. A manipulator is considered to be a group of rigid bodies or links. These links are connected and powered in such a way that they are forced to move relative to one another in order to position a hand or other type of terminal device. The first link is assumed connected to ground by the first joint while the last link is free and contains the hand. In addition, each link is connected to at most two others so that closed loops are not formed. For the purpose of this work, the assumption is made that the connection between links (the joints) have only one degree-of-freedom. With this restriction, two types of joints are practical — revolute and prismatic.* A revolute joint only permits rotation about an axis, while the prismatic joint allows sliding along an axis with no rotation. A schematic representation of these joints is shown in Fig. 2.1. If a manipulator is considered to be a combination of links and joints, with the first link connected to ground and the last link containing the terminal device, it may be classified by the type of joints and their order. For example, a manipulator comprised of three revolute joints, a prismatic joint,

*Although others might wish to include screw joints, we feel that the difficulties encountered in building screw joints make them impractical.

and two revolute joints, in that order, would be designated 3R-P-2R, where R is used for a revolute and P for a prismatic joint.

Given a broad classification according to the joints, a sub-grouping is made by looking at the links. Now, each joint has an axis associated with it, and two adjacent axes are connected by a link. Thus a link description is just the description of the relation between two adjacent axes. A link model, shown in Fig. 2.2, has the following parameters:

a_i: The common normal between the axis of the $i^{\underline{th}}$ joint and the axis of the $(i+1)^{\underline{th}}$ joint.

s_i: The distance between the lines a_i and a_{i-1} measured along the positive direction of the axis of the $i^{\underline{th}}$ joint.

θ_i: The rotation of the line a_i relative to the line a_{i-1} about the axis of the $i^{\underline{th}}$ joint.

α_i: The angle between the $(i+1)^{\underline{th}}$ axis and the $i^{\underline{th}}$ axis. The positive sense is determined according to the right-hand screw rule with the screw taken along a_i pointing from the $(i+1)^{\underline{th}}$ to the $i^{\underline{th}}$ axis.

If the joint i is a revolute, then a_i, s_i and α_i are constants while θ_i is the variable associated with that joint. If joint i is a prismatic joint, then a_i, α_i and θ_i are constants while s_i is the variable. The sub-classification is then made according to the non-zero a_i and s_i. For example, if all the a_i and s_i of a 4R manipulator were non-zero, it would have the sub-classification $s_1a_1s_2a_2s_3a_3s_4a_4$ or if $a_1 = s_2 = s_3 = 0$ it would be of the type $s_1a_2a_3s_4a_4$. It may be noted that for the last link, $i = n$, a_n, α_n and s_n are not well defined as axis $n+1$ is non-existent. For this reason,

if the last joint is a revolute, the parameters of the last link will not be included in the description. If, however, the last joint is a prismatic then s_n will be included. For the first link s_1 has an arbitrary reference that will be considered zero so that s_1 will be included only if the joint is prismatic. An example of a $\overline{4R}$, $s_2a_2s_3$ is shown in Fig. 2.3.

2.2 Special Cases: Degeneracy and Kinematic Equivalence

The most general manipulator has all non-zero link parameters. However, in practical manipulators there are many zero parameters which lead to special cases of interest. The first is degeneracy. This exists when the number of degrees-of-freedom of the last link is less than the number of joints. A manipulator with more than six joints would be classified in this category, as a rigid body can have a maximum of six degrees-of-freedom. The existence of four or more prismatic joints leads to degeneracy, since motion from one joint can in general be obtained as a linear combination of the motion of the remaining three. Also, if four or more revolute axes always intersect at a point, then rotation about one axis can be expressed as a combination of rotations about the other three. Special values of the parameters α can also lead to degeneracy. An example is given by those values of α for which four revolute axes are always parallel, and hence normal to the same plane.

In addition to degeneracy, non-zero parameters may make combinations of revolute and prismatic joints kinematically equivalent to more complex joints. Thus if three revolute axes intersect at a point

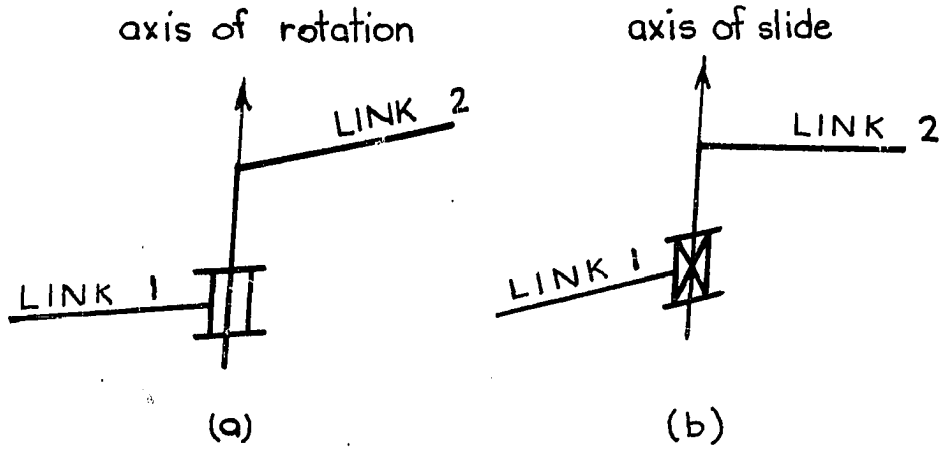


Figure 2.1. Schematic Representation of Joints.
(a) Revolute (b) Prismatic

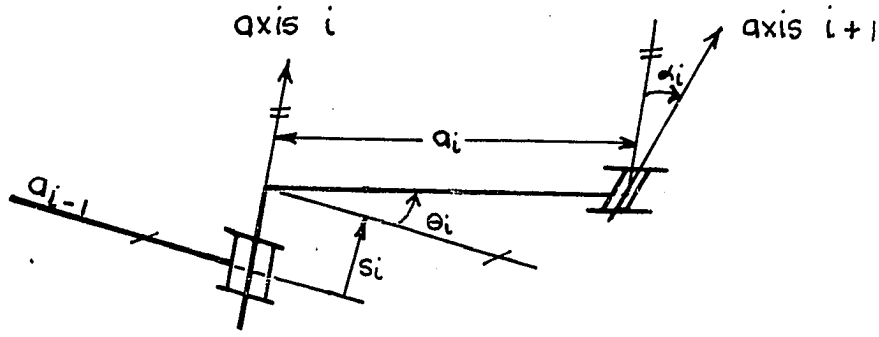


Figure 2.2. The Link Model.

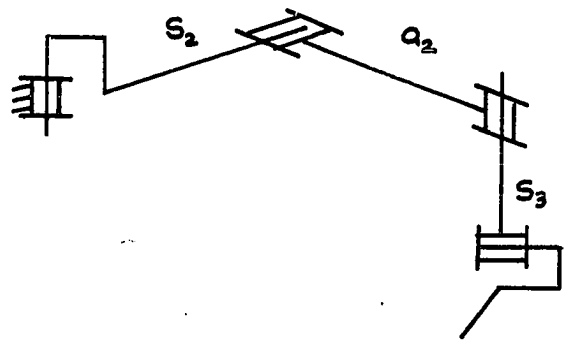


Figure 2.3. Schematic of a 4R, $s_2a_2s_3$ manipulator.

they are equivalent to a spherical joint which we denote by the symbol S. Also, if the axes of a revolute and a prismatic joint coincide, they are equivalent to a cylindrical joint denoted by the symbol C.

A 4R manipulator may be used to illustrate these special cases. The most general case is shown schematically in Fig. 2.4a. A sufficient condition for two axes to intersect is that their common normal be zero. For example if a_2 is zero, then axes 2 and 3 intersect (Fig. 2.4b). For three axes to intersect at a point, the two common normals, as well as the displacement along the intermediate axis must be zero. For example, if $a_2 = a_3 = s_3 = 0$, the result is equivalent to a spherical joint and the 4R manipulator is kinematically equivalent to an S-R manipulator (Fig. 2.4c). For four axes to intersect at a point (resulting in degeneracy), three adjacent common normals, and the displacements along the two intermediate axes must be zero (Fig. 2.4d). Degeneracy also occurs if the equivalent of two spherical joints exist. In this case, it is possible for the link connecting the two sphere centers to rotate about itself.

A cylindric joint results when the common normal and the angle between a revolute and adjacent prismatic joint are both zero. An example of an R-P-R being equivalent to an R-C manipulator is shown in Fig. 2.5.

2.3 A Catalog of Manipulators

With the above scheme we may classify most of the manipulators that have been built in the last several years. Some manipulators since they contain a very large number of links are omitted from the table.

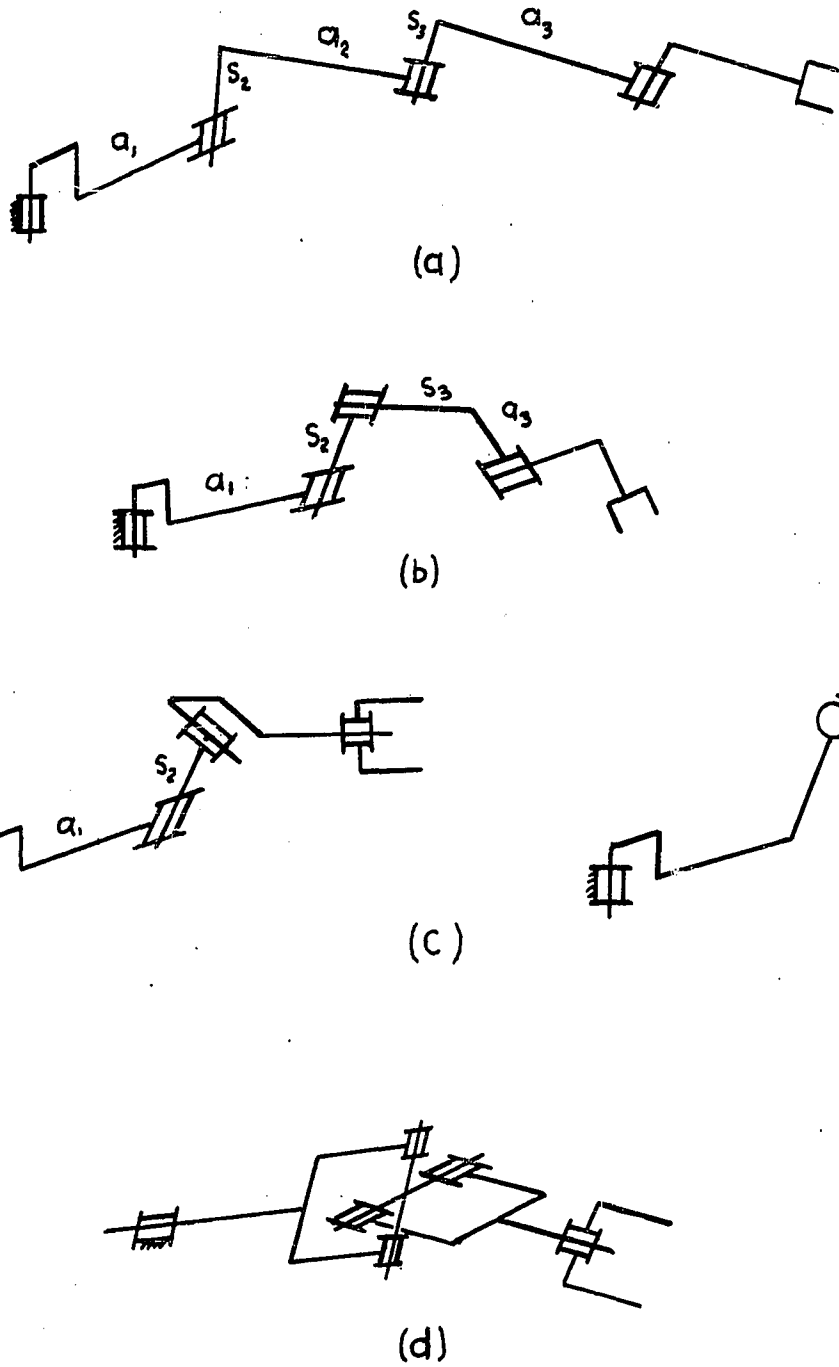
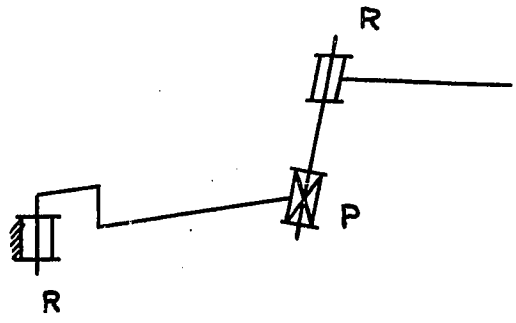
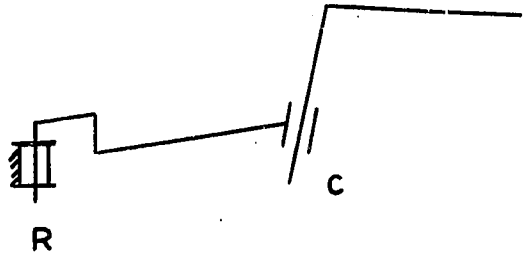


Figure 2.4. (a) A general $4R,a_1s_2a_2s_3a_3$ manipulator.
(b) A $4R,a_1s_2s_3$ with one pair of intersecting axes.
(c) A $4R,a_1s_2$ manipulator and spherical equivalent.
(d) A degenerate 4R manipulator.



(a)



(b)

Figure 2.5. (a) An R-P-R manipulator (b) The equivalent R-C.

These generally have a snake-like structure, and even though these manipulators may fit into the basic model they contain many joints usually with limited freedom in each joint and similar link parameters for all links. We call such manipulators "ORMS"* and consider them separately in Chapter 5.

Table 2.1 contains a catalog of some recently built manipulators.

*ORM is the Norwegian word for snake.

TABLE 2.1 CLASSIFICATION OF MANIPULATORS

	Manipulator	Mfg.	Use	Class	References
1	Handyman - slave -master	General Electric	master-slave remote handling	6R, $s_3a_3s_5$ 6R, $a_2s_3a_3a_4s_5$	[26, 27]
2	Model 8 (master-slave identical)	Argonne, C.R.L. AMF Mitsubishi	master-slave remote handling	2R-P-3R, s_3	[4, 5, 28, 30, 31]
3	Versatran	AMF	Industrial Robot remote handling	R-2P-R, s_2s_3	[15, 17, 29]
4	Unimate	Unimation Inc.	Industrial Robot remote handling	2R-P-2R, a_2s_3	[15, 16, 29]
5	Fleximan	AMF	Industrial Robot remote handling	ORM	[15]
6	Model 3500	PAR	Remote handling, undersea	5R, a_2a_3	[32]
7	Model 162	GMI - Litton Systems	Remote handling, undersea	5R, a_2a_3	[7]
8	Model 150	GMI	Remote handling undersea (Trieste II)	5R, $a_1a_2a_3$	[7]
9	Used on Trieste II	ACF Industries	Undersea	5R, a_2a_3	[7]
10	Used on Beaver	Autonetics	Undersea	7R-P, $s_2a_3a_4+a_5s_8$	[7]
11	MOBOT (manipulator only)	Hughes	Vehicle mounted remote handling	3R, a_2	[35, 36]
12	MRMU (manipulator only)	FMC	Vehicle mounted remote handling	6R-P, $a_2a_3a_4s_7$	[33, 34]
13	Tensor Arm	Marine Physical Lab., San Diego	Remote underwater	ORM	[37]

TABLE 2.1 (cont)

	Manipulator	Mfg.	Use	Class	References
14		Northern Electric Co. Ltd.	Prosthetic	4R, a_2a_3	[9]
15	Electric Arm - Stanford Artificial Intelligence Project	Rancho Los Amigos Hospital	Prosthetic, blockstacking	6R, $s_3a_4s_5a_5$	[10, 38]
16	Hydraulic Arm - Stanford Artificial Intelligence Project	Roth Associates	Smashing things	6R, $a_1a_2s_9$	
17	Aluminaut	General Electric	Underwater	5R, $a_2s_3a_3$	[39]

CHAPTER III

SOLUTIONS

3.1 Statement of the Problem

In remote manipulation it is desirable to place a rigid body (the hand) at a specified position in space with a specified orientation. Thus, a manipulator needs to have at least six degrees-of-freedom. More joints than six lead to a problem that is not deterministic with the specification of hand position and orientation. We therefore limit this work to manipulators with six degrees-of-freedom.

The problem we wish to solve may be stated as follows: given the desired hand position and orientation, along with the various link parameters, find the values of the manipulator variables that place the hand at the desired position with the desired orientation. This problem is related to the displacement analysis problem in three dimensional kinematics.

The result of the displacement analysis of a mechanism is the relationships between input and output. That is, if one link is driven in a prescribed manner, we wish to find the resulting position of the rest of the mechanism.

The most general one degree-of-freedom, single loop mechanism is the so-called "seven-bar chain". This mechanism is composed of seven one degree-of-freedom joints connected to one another in a general manner to form a single closed loop. Mechanisms comprised of spherical and cyclindric joints may be derived from this seven bar by an appropriate

choice of link parameters leading to kinematic equivalence, as discussed in Chapter II.

If one considers a seven bar mechanism where one link is considered fixed, while an adjacent link is driven relative to it by motion in the connecting joint, then the position and orientation of the driven link are known. The problem of displacement analysis is to find the resultant configuration of the mechanism, or equivalently the motion in each of the remaining six joints. We then observe that the manipulator problem resulting from specifying hand position and orientation is analogous to the displacement analysis problem resulting from driving one of the links.

3.2 Survey of Existing Solutions

Although displacement analysis of mechanisms has been of interest to kinematicians for many years, no method has been developed that can be applied to all cases. Dimentberg [40, 41] obtained solutions for several four-link mechanisms using screw algebra and Dual numbers. He also reduced the five-link RCRCR mechanism to the solution of a single polynomial of degree eight. Yang [42] using dual number matrices, was able to express the input-output relation of this mechanism as a single polynomial of degree four. Others have used (2x2) dual matrices, dual quaternians, and vector methods to obtain solutions of four link mechanisms [43, 44, 45]. The (4x4) matrix method developed by Denavit and Hartenberg [46] has also been used to analyze four-link mechanisms [47, 48]. For more than four links, this method has been applied using iterative numerical techniques [49]. Urquardt [50] showed that solutions were possible where the mechanisms had three or more prismatic pairs.

Earnest [51] has found geometric solutions to several special manipulator configurations. We present his solution to the manipulator shown in Figure 3.1:

> Referring to Figure 3.1, it can be seen that the point Q lies on a line formed by the intersection of a plane perpendicular to axis 1 containing line ℓ_1, and the plane perpendicular to axis 6 containing ℓ_2. In addition Q must lie on a sphere with P as a diameter. The intersection of the line and the sphere thus fix Q.

Sharpe [52] studies the problem of placing the end of a snake-like chain (which could be used as a manipulator) at a specified target. An "n-link snake" is composed of n links connected with revolute joints to form a planar chain. The joints in general have continuously variable angles. However, he does discuss the case where angles may take on only two values. He presents an adaptive approach using a simple searching procedure to handle this case.

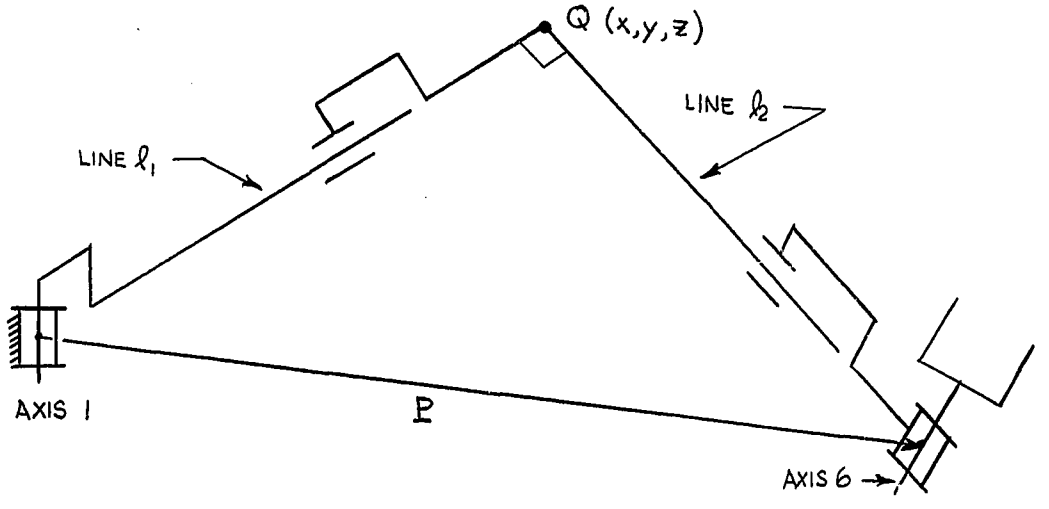


Figure 3.1. Example manipulator used to demonstrate geometric solution.

3.3 Method of Solution

In this work, we use (4x4) matrices to attack the manipulator problem. Solutions for manipulators containing three intersecting revolute axes are presented. The most complex of these requires the solution of a single polynomial of degree four. This is equivalent to the solutions of all single loop five-bar mechanisms containing one spherical joint and the rest either revolute or prismatic. Solutions for manipulators with any three joints prismatic are also presented. The extension to more difficult problems is discussed with a 6R, a_2a_4 manipulator having adjacent axes orthogonal used as an example.

3.3.1 Notation

Throughout the text we use scalar, vector, and matrix quantities. Matrices are denoted by capital letters and may have subscripts (e.g., A_2). Vectors are denoted by underlined letters and may have subscripts and one or more superscripts in front of the letter. Vectors are generally used to locate points relative to a coordinate system. The subscripts are used to differentiate between points, while the superscript indicates the coordinate system to which the point is referenced (e.g., $^{i+1}\underline{X}n$, would represent a vector from the origin of coordinate system $i+1$ to a point n). If no superscript appears it is assumed to be 1, or else no origin is implied. At times we wish to express a vector in a coordinate system which differs from the one in which the vector is formed (the so-called "reference system"). If the system used to express these coordinates is different from the reference system, we enclose the vector in brackets and use another superscript to denote the system in which the components are expressed (e.g., $^{i}\left[\,^{i+1}\underline{X}_n\right]$). If the outer superscript is not used, it is assumed

to be the same as the inner superscript. Scalar quantities are written as lower case letters, with or without subscripts (e.g., $a_1 s_1$). If they represent coordinates of points, then a superscript is sometimes used to designate the coordinate system to which they refer. Where no superscript is used, the number 1 is implied. Angles are denoted by lower case Greek letters with or without subscripts (e.g., θ_1 α). Points are occasionally given a name (e.g., "the point X_2 ") and referred to by name.

The trigonometric functions sin, cos, and tan are abbreviated s, c, and t respectively (e.g., sin θ_1 is written $s\theta_1$, cos α_1 as $c\alpha_1$, etc).

3.3.2 Mathematical Preliminaries

In order to analyze the kinematics of a manipulator, we first establish the relation between two Cartesian coordinate systems as shown in Figure 3.2. We define the following:

a_i: the length of the common normal between ${}^{i}z$-axis and ${}^{i+1}z$-axis .

α_i: the angle between ${}^{i+1}z$ and ${}^{i}z$ measured in the right-handed sense from ${}^{i}z$ along a line from ${}^{i}z$ to ${}^{i+1}z$.

s_i: distance from O_i to the common normal a_i .

θ_i: angle the common normal makes with ${}^{i}x$-axis.

Then there exists the transformation [46] to express the coordinates of a point in one system given its coordinates in the other. If we denote the coordinates in system i by $({}^{i}x, {}^{i}y, {}^{i}z)$ and in system i+1 by $({}^{i+1}x, {}^{i+1}y, {}^{i+1}z)$, we define the vectors ${}^{i}\underline{X}$ and ${}^{i+1}\underline{X}$ such that:

$$ {}^{i}\underline{X} = \begin{bmatrix} {}^{i}x \\ {}^{i}y \\ {}^{i}z \\ 1 \end{bmatrix} $$

and

$$ {}^{i+1}\underline{X} = \begin{bmatrix} {}^{i+1}x \\ {}^{i+1}y \\ {}^{i+1}z \\ 1 \end{bmatrix} $$

so that the transformation is:

$$ {}^{i}\underline{X} = A_i \, {}^{i+1}X $$

where

$$ A_i = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & s_i \\ 0 & 0 & 0 & 1 \end{bmatrix} . \qquad (3.1) $$

The inverse also exists and is defined by:

$$ {}^{i+1}\underline{X} = A_i^{-1} \, {}^{i}\underline{X} $$

where

$$ A_i^{-1} = \begin{bmatrix} c\theta_i & s\theta_i & 0 & -a_i \\ -s\theta_i c\alpha_i & c\theta_i s\alpha_i & s\alpha_i & -s_i c\alpha_i \\ s\theta_i s\alpha_i & -c\theta_i s\alpha_i & c\alpha_i & -s_i c\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.2) $$

For n+1 coordinate systems there are n transformations between neighboring systems. These may be multiplied, in the following order, to give the coordinates in the 1 system of any point fixed in the n+1 system:

$$^{1}\underline{X} = A_1 A_2 \cdots A_n \; ^{n+1}\underline{X}$$

Now to appropriately fix these coordinate systems in a manipulator, we make ^{i}z correspond to axis i , ^{i}x to common normal a_{i-1} and define ^{i}y in a right-handed sense. This is shown applied to a sample manipulator in Figure 3.3. For a six degree-of-freedom manipulator we write:

$$^{1}\underline{X} = A_1 A_2 A_3 A_4 A_5 A_6 \; ^{7}\underline{X} \tag{3.3}$$

where $^{1}\underline{X}$ is a vector to any point, expressed in the ground system and $^{7}\underline{X}$ is a vector to the same point expressed in a system fixed in the terminal device. We define

$$Aeq = A_1 \cdots A_6 \; . \tag{3.4}$$

With this definition (3.3) becomes:

$$^{1}\underline{X} = Aeq \; ^{7}\underline{X} \tag{3.5}$$

and the inverse yields:

$$^{7}\underline{X} = Aeq^{-1} \; ^{1}\underline{X} \tag{3.6}$$

Now, if we let $\underline{P}$ be a vector from the origin of system 1 to the origin of system 7, and $\underline{\ell}$, $\underline{m}$, and $\underline{n}$, be three unit vectors aligned with the ^{7}x, ^{7}y, ^{7}z axes respectively, then when $\underline{\ell}$, $\underline{m}$, $\underline{n}$, and $\underline{P}$ are expressed in system 1, they may be used with equation (3.5) to find Aeq . That is, using (3.5) we may write

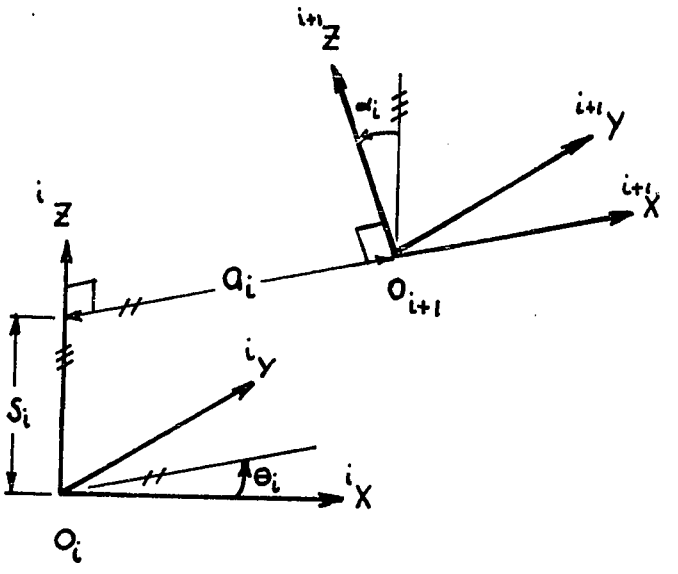


Figure 3.2. Relation between two coordinate systems.

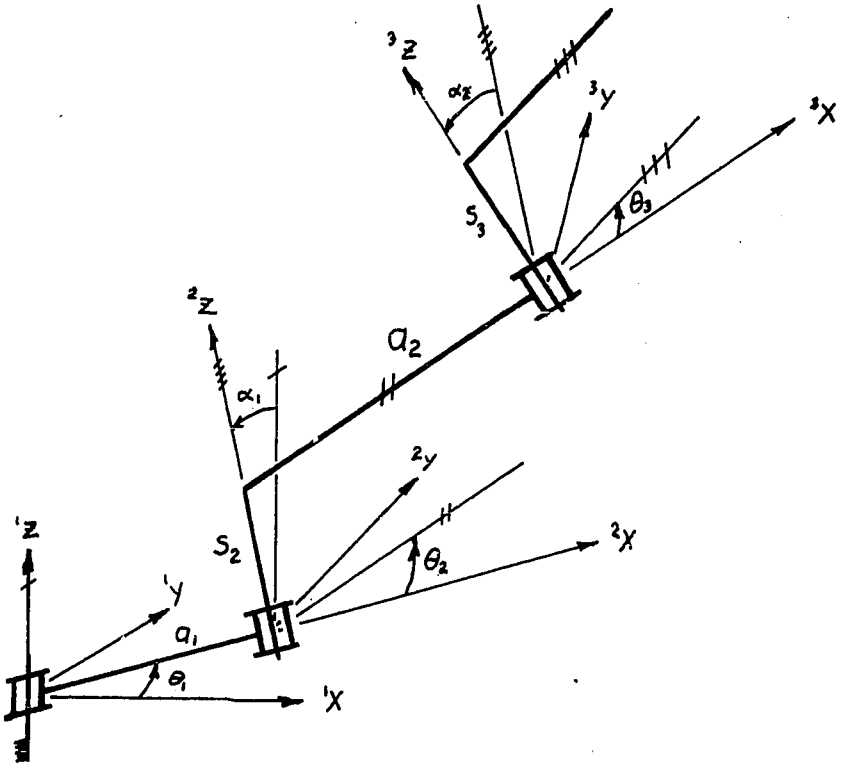


Figure 3.3. The relationship between coordinate systems fixed in the manipulator.

$$\begin{bmatrix} \ell_1 \\ \ell_2 \\ \ell_3 \\ 0 \end{bmatrix} = Aeq \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ 0 \end{bmatrix} = Aeq \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ 0 \end{bmatrix} = Aeq \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix} = Aeq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

from which we may solve for the elements of Aeq to obtain:

$$Aeq = \begin{bmatrix} \ell_1 & m_1 & n_1 & p_1 \\ \ell_2 & m_2 & n_2 & p_2 \\ \ell_3 & m_3 & n_3 & p_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{3.7}$$

It is thus seen that position and orientation of the terminal device can easily be found, knowing the manipulator variables, θ_i or s_i, i=1,.., 6 , by the matrix product equation (3.4).

However, for computer control of manipulators, the problem is to find the manipulator variables, given the terminal position and orientation (Aeq) .

We shall first consider a six-revolute arm and the problem of finding $\theta_i,\ldots, \theta_6$ given Aeq. Equation (3.4) represents twelve scalar equations, nine dealing with orientation and three with position. However, only three of the orientation equations are independent so that there are six equations in $\theta_i,\ldots, \theta_6$. These equations have terms of the form:

$$\begin{aligned} &c\theta_1 \quad c\theta_2 \quad c\theta_3 \quad c\theta_4 \quad c\theta_5 \quad c\theta_6 , \\ &s\theta_1 \quad c\theta_2 \quad c\theta_3 \quad s\theta_4 \quad s\theta_5 \quad s\theta_6 , \ \ldots \end{aligned} \tag{3.8}$$

These terms contain both sines and cosines, which we may define in terms of the tangent of the half-angle.

$$c\theta_i = \frac{1-t^2\frac{\theta_i}{2}}{1+t^2\frac{\theta_i}{2}} , \qquad s\theta_i = \frac{2t\,\frac{\theta i}{2}}{1+t^2\frac{\theta_i}{2}} \tag{3.9}$$

Then if we substitute (3.9) into the six equations, the typical term, as shown in (3.8) becomes (letting $t_i = \tan \frac{\theta_i}{2}$, i=1,..., 6 , and removing the denominators which are common):

$$t_1^2 t_2^2 t_3^2 t_4^2 t_5^2 t_6^2 + \ldots$$

Thus we see that these equations are quadratic in each of the unknowns and the degree of the highest degree term is 12.

However, not all the equations contain all of the unknowns and by judiciously choosing the three orientation equations, the unknowns θ_1 and θ_6 can be eliminated from some of the equations. We use the six equations:

$$F_1 (t_1, \ldots, t_5) = 0 \tag{3.10}$$

$$F_2 (t_1, \ldots, t_5) = 0 \tag{3.11}$$

$$F_3 (t_1, \ldots, t_5) = 0 \tag{3.12}$$

$$F_4 (t_2, \ldots, t_5) = 0 \tag{3.13}$$

$$F_5 (t_2, \ldots, t_5) = 0 \tag{3.14}$$

$$F_6 (t_2, \ldots, t_6) = 0 \tag{3.15}$$

which are obtained respectively from the '14', '24', '13', '33', '34', and '32', elements of the matrix of (3.4). We note that (3.10) - (3.14) do not contain t_6 , and (3.13) – (3.15) do not contain t_1 . Of the five equations in which the variables $t_1, \ldots, t_5$ appear at most quadratically, three equations are of degree 10, while, two are of degree eight. If we eliminate t_1 between (3.10), (3.11), and (3.12), the result is two equations of at most degree eight in the unknowns $t_2, \ldots, t_5$ whose total degree is 32. These together with (3.14) and (3.15) give us four equations for $t_2, \ldots, t_5$. Proceeding in this

manner eliminating one variable at a time, we would finally obtain a single polynomial of degree 524,288. Even though this method of elimination introduces extraneous roots, we would still expect, according to Bezouts' theorem*, $(10)^3 \times (8)^2$ or 64,000 common roots, a number much too large to cope with. The general problem, attacked in this manner, is insoluble. At this point we shall define a "soluble case" to be one in which the degree of the final eliminant is low enough to find all roots. In practice all the roots of an eighth degree polynomial can be found within a few seconds using a digital computer and the roots of a fourth degree within one-half second. A solution is said to be "closed-form" if the unknowns can be solved for symbolically.

Even though the general problem is beyond reach, many practical manipulator configurations are soluble. The existence of three revolute axes intersecting at a point leads to a soluble class. In the next sections we explore the possible combinations of three intersecting axes.

3.3.3. Last Three Axes Intersecting

If the last three joints are revolutes and their axes intersect as in Figure 3.4, then their point of intersection, as designated by the vector $\underline{P}_3$ is only a function of motion in the first three joints and the constant link parameters. $\underline{P}_3$ is known by specifying the hand position and orientation. We want to solve the three scalar equations represented by:

$$\underline{P}_3 = A_1 A_2 A_3 \begin{bmatrix} 0 \\ 0 \\ s_4 \\ 1 \end{bmatrix} \tag{3.16}$$

*Bezouts' theorem gives an upper bound to the number of common solutions for a set of equations. The upper bound is the product of the total degrees of all the equations.

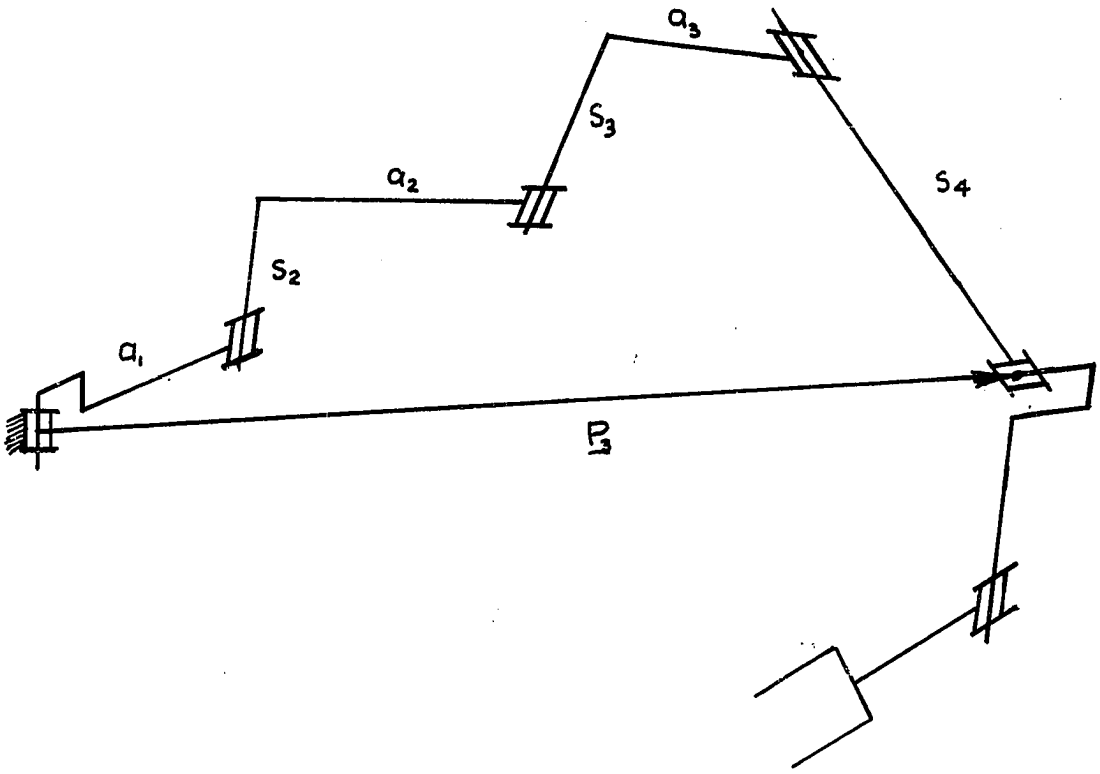


Figure 3.4. The most general manipulator having the last three revolute axes intersecting.

for the variables associated with the first three joints. We now derive an important result used in the solution of this problem. We define the vector $\underline{P}_j$

$$\underline{P}_j = A_1 \cdots A_j \begin{bmatrix} 0 \\ 0 \\ s_{j+1} \\ 1 \end{bmatrix} \tag{3.17}$$

where A_i $(i = 1, \ldots, j)$ is defined in equation (3.1). It is seen that $\underline{P}_j$ is a vector specifying the position of a point $(0, 0, s_{j+1})$ which is fixed in coordinate system $j+1$.

We may write (3.17) as

$$\underline{P}_j = (A_1A_2)\, A_3 \cdots A_j \begin{bmatrix} 0 \\ 0 \\ s_{j+1} \\ 1 \end{bmatrix}$$

$$= A_1A_2 \begin{bmatrix} f_1(\theta_3, \ldots, \theta_j) \\ f_2(\theta_3, \ldots, \theta_j) \\ f_2(\theta_3, \ldots, \theta_j) \\ 1 \end{bmatrix} \tag{3.18}$$

where

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ 1 \end{bmatrix} = A_3 \cdots A_j \begin{bmatrix} 0 \\ 0 \\ s_{j+1} \\ 1 \end{bmatrix} \tag{3.19}$$

Then using (3.1) for A_1 and A_2 (3.18) becomes

$$\underline{P}_j = \begin{bmatrix} c\theta_1 g_1 + s\theta_1 g_2 \\ s\theta_1 g_1 - c\theta_1 g_2 \\ \left(s\alpha_1 [s\theta_2 (a_2 + f_1) - c\theta_2 (-c\alpha_2 f_2 + s\alpha_2 f_3)] \right. \\ \left. + c\alpha_1 (s\alpha_2 f_2 + c\alpha_2 f_3 + s_2) + s_1 \right) \\ 1 \end{bmatrix} \tag{3.20}$$

where

$$g_1 = c\theta_2(a_2+f_1) + s\theta_2(-c\alpha_2 f_2 + s\alpha_2 f_3) + a_1 \tag{3.21}$$

$$g_2 = -s\theta_2 c\alpha_1(a_2+f_1) + c\theta_2 c\alpha_1(-c\alpha_2 f_2 + s\alpha_2 f_3) + s\alpha_1(s\alpha_2 f_2 + c\alpha_2 f_3 + s_2) \quad . \tag{3.22}$$

Denoting the components of $\underline{P}_j$ by x_j , y_j , z_j , we define

$$R_j = x_j^2 + y_j^2 + (z_j - s_1)^2 \quad . \tag{3.23}$$

With (3.20) for the components of $(\underline{P}_j)$, (3.23) becomes

$$R_j = f_1^2 + f_2^2 + f_3^2 + a_1^2 + a_2^2 + s_2^2 + 2a_2 f_1 + 2s_2(s\alpha_2 f_2 + c\alpha_2 f_3) + 2a_1[c\theta_2(a_2+f_1) + s\theta_2(-c\alpha_2 f_2 + s\alpha_2 f_3)] \tag{3.24}$$

We note from (3.20) and (3.24), that we may write:

$$R_j = (F_1 c\theta_2 + F_2 s\theta_2)2a_1 + F_3 \tag{3.25}$$

$$z_j = (F_1 s\theta_2 - F_2 c\theta_2)s\alpha_1 + F_4 \tag{3.26}$$

where,

$$F_1 = a_2 + f_1 \tag{3.27}$$

$$F_2 = -c\alpha_2 f_2 + s\alpha_2 f_3 \tag{3.28}$$

$$F_3 = f_1^2 + f_2^2 + f_3^2 + a_1^2 + s_2^2 + 2a_2 f_1 + a_2^2 + 2s_2(s\alpha_2 f_2 + c\alpha_2 f_3) \tag{3.29}$$

$$F_4 = c\alpha_1(s\alpha_2 f_2 + c\alpha_2 f_3 + s_2) \tag{3.30}$$

Equations (3.25) and (3.26) prove to be very useful as θ_1 has been eliminated, and θ_2 appears in a very simple form.

Returning to the manipulator problem, the above equations apply with $j = 3$. In which case by using (3.1) for A_3 (3.19) becomes:

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} s_4 s\theta_3 s\alpha_3 + a_3 c\theta_3 \\ -s_4 c\theta_3 s\alpha_3 + a_3 s\theta_3 \\ s_4 c\alpha_3 + s_3 \end{bmatrix} \tag{3.31}$$

so that with (3.21), (3.22), and (3.31) equation (3.20) represents three equations in three unknowns. If the first three joints are prismatic, then (3.20) represents three linear equations and is easily solved. The other possibilities are somewhat more difficult, but may be solved as follows:

3 Revolute - $\theta_1, \theta_2, \theta_3$ all variable

Substituting (3.31) in (3.27) - (3.30) yields respectively

$$F_1 = a_2 + s_4 s\theta_3 s\alpha_3 + a_3 c\theta_3 \tag{3.32}$$

$$F_2 = -c\alpha_2(-s_4 c\theta_3 s\alpha_3 + a_3 s\theta_3) + s\alpha_2(s_3 + s_4 c\alpha_3) \tag{3.33}$$

$$F_3 = a_1^2 + s_2^2 + a_2^2 + s_3^2 + a_3^2 + s_4^2 + 2s_2 s_3 c\alpha_2 + 2s_2 s_4 c\alpha_2 c\alpha_3 + 2s_3 s_4 c\alpha_3 + c\theta_3(2a_2 a_3 - 2s_2 s_4 s\alpha_2 s\alpha_3) + s\theta_3(2a_3 s_2 s\alpha_2 + 2a_2 s_4 s\alpha_3) \tag{3.34}$$

$$F_4 = c\alpha_1[a_3 s\theta_3 s\alpha_2 + s_3 c\alpha_2 + s_2 + s_4(-c\theta_3 s\alpha_2 s\alpha_3 + c\alpha_2 c\alpha_3)] \tag{3.35}$$

Now we note that the left hand side of (3.25) and (3.26) are known and that if $a_1 = 0$, (3.25) reduces to

$$R_3 = F_3 \tag{3.36}$$

When (3.34) is used in (3.36) it is simply a function of θ_3. Then making the additional substitution

$$c\theta_3 = \frac{1 - \tan^2 \frac{\theta_3}{2}}{1 + \tan^2 \frac{\theta_3}{2}} \tag{3.37}$$

$$s\theta_3 = \frac{2 \tan \frac{\theta_3}{2}}{1 + \tan \frac{\theta_3}{2}} \tag{3.38}$$

into (3.36), yields a quadratic in $\tan \frac{\theta_3}{2}$. Similar simplification results if $s\alpha_1=0$, as (3.26) reduces to a quadratic. If however $s\alpha_1$ and a_1 are non-zero, we eliminate $s\theta_2$ and $c\theta_2$ from (3.25) and (3.26) to obtain the polynomial

$$\frac{(R_3 - F_3)^2}{2a_1} + \frac{(z - F_4)^2}{s\alpha_1} = F_1^2 + F_2^2 \tag{3.39}$$

Upon making the $\tan \frac{\theta_3}{2}$ substitution and using (3.27) - (3.30) equation (3.39) is of degree four in $\tan \frac{\theta_3}{2}$. After getting θ_3 , θ_2 may be obtained from (3.25) or (3.26) and θ_1 from (3.20).

$\underline{S_1, \theta_2, \theta_3 \text{ variable}}$

Here we take the x and y components of $\underline{P}_3$ as defined in (3.20)

$$x = c\theta_1 g_1 + s\theta_1 g_2 \tag{3.40}$$

$$y = s\theta_1 g_1 - c\theta_1 g_2 \tag{3.41}$$

Solving for g_1 and g_2 we find

$$g_1 = xc\theta_1 + ys\theta_1 \tag{3.42}$$

$$g_2 = -yc\theta_1 + xs\theta_1 \tag{3.43}$$

so that g_1 and g_2 can be computed from (3.42) and (3.43). Then examining (3.21) and (3.22) using (3.31) we note

$$g_1 = c\theta_2 h_1(\theta_3) + s\theta_2 h_2(\theta_3) + a_1 \tag{3.44}$$

$$g_2 = c\alpha_1[c\theta_2 h_2(\theta_3) - s\theta_2 h_1(\theta_3)] + s\alpha_1 h_3(\theta_3) \tag{3.45}$$

where

$$h_1 = s_4 s\theta_3 s\alpha_3 + a_3 c\theta_3 \tag{3.46}$$

$$h_2 = s_4(c\theta_3 c\alpha_2 s\alpha_3 + s\alpha_2 s\alpha_3) - a_3 s\theta_3 c\alpha_2 + s_3 s\alpha_2 \tag{3.47}$$

$$h_3 = s_4(-c\theta_3 s\alpha_2 s\alpha_3 + c\alpha_2 c\alpha_3) + a_3 s\theta_3 s\alpha_2 + s_3 c\alpha_2 + s_2 \tag{3.48}$$

If $c\alpha_1 = 0$ then (3.45) is easily solved for θ_3. If $c\alpha_1 \neq 0$ we eliminate θ_2 from (3.44) and (3.45) to get the polynomial

$$h_1^2 + h_2^2 - (g_1 - a_1)^2 - \left[\frac{g_2 - s\alpha_1 h_3}{c\alpha_1}\right]^2 = 0 \tag{3.49}$$

Expressing $s\theta_3$ and $c\theta_3$ in terms of $\tan\frac{\theta_3}{2}$ leads to a polynomial of degree four. Upon obtaining the four roots of (3.49) we substitute into (3.44) and (3.45) to get θ_2 and finally (3.20) for s_1.

$\underline{\theta_1, s_2, \theta_3}$ variable

Solve (3.26) for s_2, using this in (3.25) results in a fourth degree polynomial in $\tan\frac{\theta_3}{2}$. Then proceed as in all revolute case.

$\underline{\theta_1, \theta_2, s_3}$ variable

Similar to $\theta_1\theta_2\theta_3$ variable with the exception of s_3 being the variable in the final polynomial which is of degree four.

$\underline{s_1 s_2 \theta_3}$ variable

The left-hand side of (3.44) may be computed from (3.42), then (3.44) which is quadratic may be solved for θ_3. Finally s_1 and s_2 may be found from (3.20).

$s_1\theta_2 s_3$ variable

It is possible to eliminate θ_2 as in the case of $s_1\theta_2\theta_3$ variable, resulting in a quadratic in s_3 .

$\theta_1 s_2 s_3$ variable

Equation (3.25) is solved for s_2 and used in (3.26) resulting in a quadratic in s_3 , θ_1 is found as in the all revolute case.

Methods have been presented to find the first three variables. At this time we leave the problem of finding the last three angles to be dealt with later in this work (see Section 3.3.6).

3.3.4 First Three Axes Intersecting

Next consider the three intersecting axes to be the first three, as in Figure 3.5. The solution of these is analogous to the previous example. We define a vector ${}^7\underline{P}$ from the hand to the point of intersection of the three axes, as shown in Figure 3.5. We note that when ${}^7\underline{P}$ is expressed in a coordinate system fixed in the hand, that it is just a function of the last three joints. That is:

$$ {}^7\underline{P} = A_6^{-1} A_5^{-1} A_4^{-1} A_3^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.50} $$

Using (3.2) for A_3^{-1} and forming $A_3^{-1}\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ we get

$$ {}^7\underline{P} = A_6^{-1} A_5^{-1} A_4^{-1} \begin{bmatrix} -a_3 \\ -s_3 c\alpha_3 \\ -s_3 c\alpha_3 \\ 1 \end{bmatrix} \tag{3.51} $$

If we use (3.2) to express A_6^{-1} , A_5^{-1} , and A_4^{-1} then the right-hand side of (3.51) just contains the three variables associated

with the last three joints. In addition, we compute the components of ${}^{7}\underline{P}$ from

$$ {}^{7}\underline{P} = Aeq^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} $$

where Aeq is the known matrix (3.7). We note that the rotation portion of Aeq^{-1} is just the transpose of the rotation portion of Aeq . In fact, if

$$ Aeq = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.52} $$

then

$$ Aeq^{-1} = \begin{bmatrix} a_{11} & a_{21} & a_{31} & a_{14}^{-1} \\ a_{12} & a_{22} & a_{32} & a_{24}^{-1} \\ a_{13} & a_{23} & a_{33} & a_{34}^{-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.53} $$

The elements denoted as a_{14}^{-1} , a_{24}^{-1} , a_{34}^{-1} are determined by simply applying

$$ Aeq^{-1}\, Aeq = I $$

thus

$$ a_{i_4}^{-1} = -(a_{1i}a_{14} + a_{2i}a_{24} + a_{3i}a_{34}) \qquad i=1,2,3 \tag{3.54} $$

From this point on the method of solution follows the same steps given in Section 3.3.3 for the case of the last three axes intersecting.

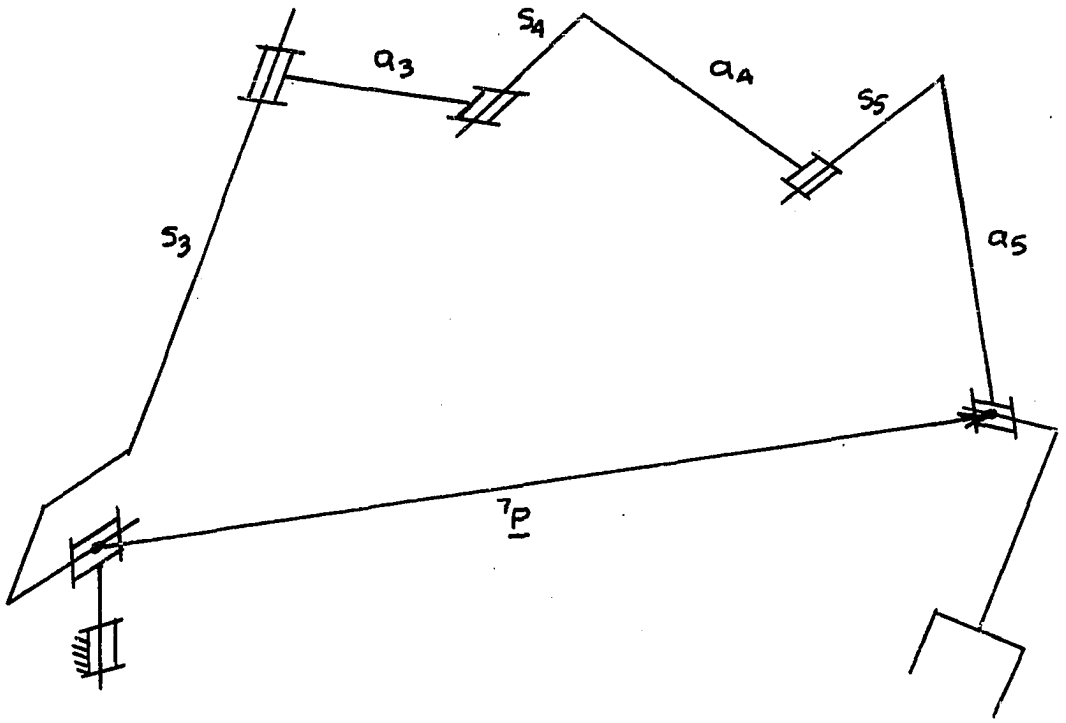


Figure 3.5. General Manipulator in which the First Three Revolute Axes Intersect at a Point.

3.3.5 Three Intermediate Axes Intersecting

Another possibility is for the three intersecting axes to be located as in Figure 3.6, where there are two other joints toward the base end and one on the hand end. We denote the position of the point of intersection by X_2 with the coordinate (x_2, y_2, z_2), and define the vector $\underline{X}_2$ from the base of the arm to X_2 and the vector ${}^7\underline{Y}_2$ from the origin of system 7 to X_2 as in Figure 3.6.

Consider the case where all joints are revolutes, then in system 7, the hand system, the point X_2 has a fixed z coordinate, and is a constant radius from the origin. We write the coordinates of X_2 in system 7, using equation (3.5) and A_{eq}^{-1} as defined in (3.53)

$$ {}^7x_2 = a_{11}x_2 + a_{21}y_2 + a_{31}z_2 + a_{14}^{-1} \tag{3.55} $$

$$ {}^7y_2 = a_{12}y_2 + a_{22}y_2 + a_{32}z_2 + a_{24}^{-1} \tag{3.56} $$

$$ {}^7z_2 = a_{13}y_2 + a_{23}y_2 + a_{33}z_2 + a_{34}^{-1} \tag{3.57} $$

Since ${}^7z_2$ is a constant, say C_1, (3.57) may be written

$$ C_1 = a_{13}x_2 + a_{23}y_2 + a_{33}z_2 + a_{34}^{-1} \tag{3.58} $$

We define the constant, C_2, to be the square of the radius

$$ C_2 = ({}^7x_2)^2 + ({}^7y_2)^2 + ({}^7z_2)^2 \tag{3.59} $$

Then using (3.55), (3.56), and (3.57) for ${}^7x$, ${}^7y$, and ${}^7z$ (3.59) becomes

$$ C_2 = x_2^2 + y_2^2 + z_2^2 - 2x_2a_{14} - 2y_2a_{24} - 2z_2a_{34} + a_{14}^2 + a_{24}^2 + a_{34}^2 \tag{3.60} $$

where (3.54) has been used for $a_{i_4}^{-1}$ $i = 1,2,3$

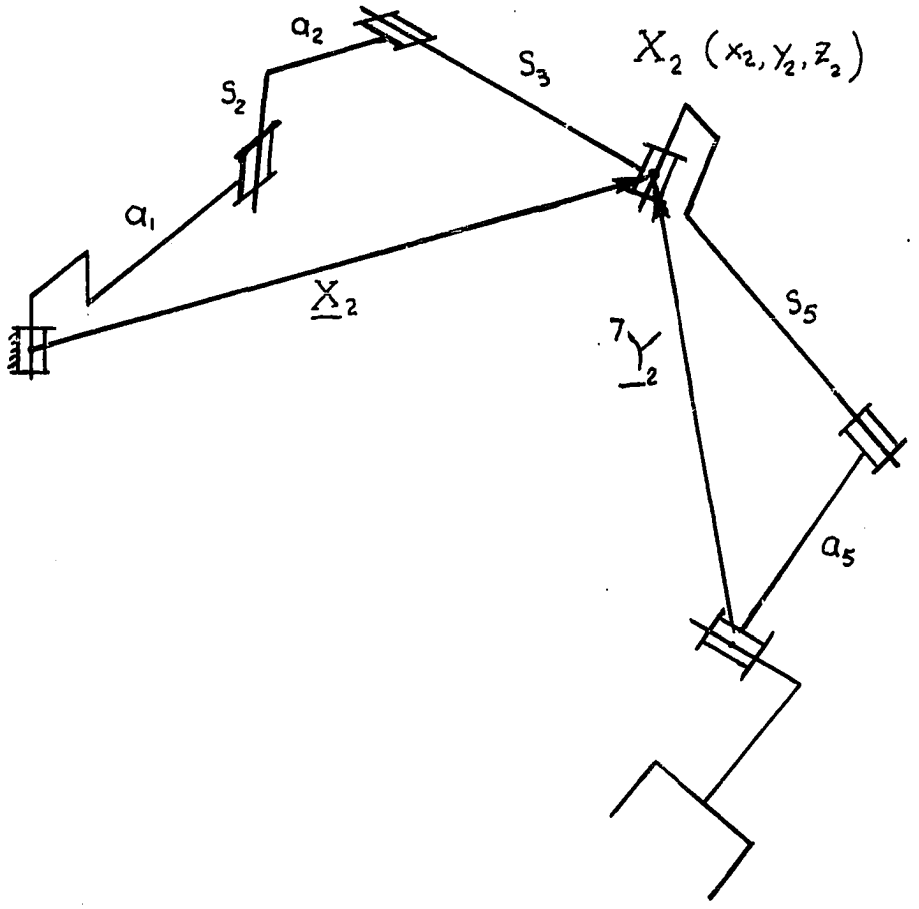


Figure 3.6. Manipulator with three intermediate revolute axes intersecting (i.e. $a_3=s_4=a_4=0$).

With $j = 2$ (3.20 becomes

$$\underline{P}_2 = \begin{bmatrix} [a_2(c\theta_1c\theta_2-s\theta_1s\theta_2c\alpha_1) + s_2s\theta_1s\alpha_1+a_1c\theta_1 + s_3(c\theta_1s\theta_2s\alpha_2 + \\ \qquad s\theta_1c\theta_2c\alpha_1s\alpha_2+s\theta_1s\alpha_1c\alpha_2)] \\ [a_2(s\theta_1c\theta_2+c\theta_1s\theta_2c\alpha_1) - s_2c\theta_1s\alpha_1 + a_1s\theta_1 \\ \qquad + s_3(s\theta_1s\theta_2s\alpha_2-c\theta_1c\theta_2c\alpha_1s\alpha_2-c\theta_1s\alpha_1c\alpha_2)] \\ [a_2s\theta_2s\alpha_1 + s_2c\alpha_1 + s_1 + s_3(-c\theta_2s\alpha_1s\alpha_2+c\alpha_1c\alpha_2)] \\ 1 \end{bmatrix} \tag{3.61}$$

and (3.27) - (3.30) become

$$F_1 = a_2 \tag{3.62}$$

$$F_2 = s_3s\alpha_2 \tag{3.63}$$

$$F_3 = a_1^2+s_2^2+a_2^2+s_3^2+2s_2s_3c\alpha_2 \tag{3.64}$$

$$F_4 = s_3c\alpha_1c\alpha_2+s_2 \tag{3.65}$$

So that using (3.62) - (3.65) and (3.23), equation (3.39) becomes

$$\left[\frac{(x_2^2+y_2^2+z_2^2-a_1^2-s_2^2-a_2^2-s_3^2-2s_2s_3c\alpha_2)}{2a_1}\right]^2 + \left[\frac{z_2-s_3c\alpha_1c\alpha_2-s_2}{s\alpha_1}\right]^2 = a_2^2+s_3^2s\alpha_2^2 \tag{3.66}$$

Then (3.58), (3.60) and (3.66) are three equations for the unknowns $(x_2y_2z_2)$. Ordinarily the system would result in an eighth degree eliminant but since (3.60) and (3.66) may be combined to form

$$\left[\frac{(C_2+2x_2a_{14}+2y_2a_{24}+2z_2a_{34}-a_{14}^2-a_{24}^2-a_{34}^2)-a_1^2-s_2^2-a_2^2-s_3^2-2s_2s_3c\alpha_2}{2a_1}\right]^2$$

$$+\left[\frac{z_2-s_3c\alpha_1c\alpha_2-s_2}{s\alpha_1}\right]^2 = a_2^2 + s_3^2s\alpha_2^2 \tag{3.67}$$

The equations (3.58) (3.60) and (3.67) may be combined to yield a single fourth degree polynomial in one variable, say z. After the values of z are determined it is possible to back substitute and obtain corresponding values for x and y.

Once the coordinates (x_2, y_2, z_2) of the point X_2 are found, θ_2 and θ_1 may readily be found from equation (3.61). θ_6 is easily evolved by noting:

$$\begin{bmatrix} {}^7x_2 \\ {}^7y_2 \\ {}^7z_2 \\ 1 \end{bmatrix} = {}^7\underline{Y}_2 = A_6^{-1} \begin{bmatrix} -a_5 \\ -s_5 s\alpha_5 \\ -s_5 c\alpha_5 \\ 1 \end{bmatrix} \tag{3.68}$$

Using (3.2) for A_6^{-1}, with $a_6 = s_6 = 0$, (3.68) becomes:

$${}^7\underline{Y}_2 = \begin{bmatrix} -a_5 c\theta_6 - s_5 s\theta_6 \\ a_5 s\theta_6 c\alpha_6 + s_5(-c\theta_6 s\alpha_5 c\alpha_6 - c\alpha_5 s\alpha_6) \\ -a_5 s\theta_6 s\alpha_6 + s_5(c\theta_6 s\alpha_5 s\alpha_6 - c\alpha_5 c\alpha_6) \\ 1 \end{bmatrix} \tag{3.69}$$

Since x_2, y_2 and z_2 are known (3.55), (3.56) and (3.57) may be used to calculate ${}^7\underline{Y}_2$. Then (3.69) may be solved for θ_6. The problem of solving for θ_3 θ_4 θ_5 will again be deferred (see Section 3.3.6).

The preceding solution was for all revolute joints. We now consider the cases in which joints 1, 2, and 6 may be prismatic.

$\underline{s_1 \theta_2 \theta_6}$ variable

Eliminating $s\theta_2$ and $c\theta_2$ between the x- and y- components of (3.61) results in a quadratic in x_2 and y_2. Then this equation along

with (3.58) and (3.60) can be reduced to a single fourth degree polynomial in either x_2 or y_2 .

$\underline{\theta_1 s_2 \theta_6}$ variable

Forming (3.25) and (3.26) with $j = 2$ and then eliminating s_2 between them, a fourth degree equation results in a manner similar to the all revolute case.

$\underline{s_1 s_2 \theta_6}$ variable

First s_2 may be eliminated between the x- and y- components of (3.61). The resultant is a linear equation which along with (3.58) and (3.60) can be combined to form a single quadratic.

If s_6 is variable instead of θ_6 , equations (3.58) and (3.60) no longer apply. However, the point X_2 must lie on a known line. This line, in the direction of axis 5 may easily be found, and may be written in terms of two known constant vectors $\underline{c}$ and $\underline{b}$ and the parameter t as:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \underline{c} + \underline{b}\, t \,, \tag{3.70}$$

where $\underline{b}$ is a unit vector parallel to this line and $\underline{c}$ is any fixed point on the line. Eliminating t , yields two equations between x_2 , y_2 , and z_2 . Then with these in place of (3.58) and (3.60), the procedure is the same as previously indicated.

The second possibility for three intermediate axes to intersect is as shown in Figure 3.7. This is just an inversion of the case treated in this section and may be solved in a similar manner.

3.3.6 Completing the Solution

It can be seen from the foregoing that if three adjacent revolute axes intersect at a point, then the solution to the problem can be reduced to a single equation of degree four. If, in addition, two of the remaining three joints are prismatic, the problem reduces to a quadratic.

Simplification will also result, if special geometry exist in addition to the three intersecting revolute axes. Consider the all revolute case, with only a_1 , a_2 , and s_4 non-zero and $\alpha_1 = 90^o$, $\alpha_2 = 0$, $\alpha_3 = 90^o$, $\alpha_4 = 90^o$, $\alpha_5 = 90^o$, as shown in Figure 3.8. This is the configuration used for the hydraulic arm at the Stanford Artificial Intelligence Project. With the above values, equation (3.17) becomes

$$\underline{P}_3 = \begin{bmatrix} a_1c\theta_1 + a_2c\theta_1c\theta_2 + s_4(c\theta_1c\theta_2s\theta_3+c\theta_1s\theta_2c\theta_3) \\ a_1s\theta_1 + a_2s\theta_1c\theta_2 + s_4(s\theta_1c\theta_2s\theta_3+s\theta_1s\theta_2c\theta_3) \\ a_2s\theta_2 + s_4(s\theta_2s\theta_3-c\theta_2c\theta_3) \\ 1 \end{bmatrix} \tag{3.71}$$

and (3.27), (3.28), (3.29), and (3.30) become

$$F_1 = a_2+s_4s\theta_3 \tag{3.72}$$

$$F_2 = s_4c\theta_3 \tag{3.73}$$

$$F_3 = 2a_2s_4s\theta_3 + s_4^2 + a_2^2 + a_1^2 \tag{3.74}$$

$$F_4 = 0 \tag{3.75}$$

So that equation (3.25) becomes:

$$R_3 = s_4^2 + a_2^2 + a_1^2 + 2a_2s_4s\theta_3 + 2a_1a_2c\theta_2 + 2a_1s_4(c\theta_2s\theta_3 + s\theta_2c\theta_3) \tag{3.76}$$

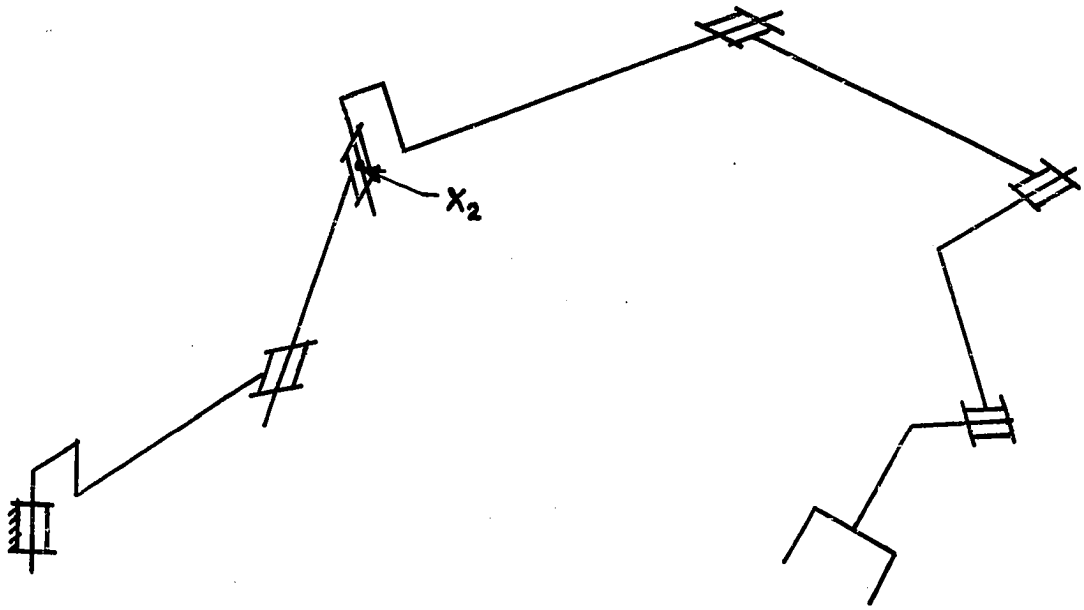


Figure 3.7. Second possibility for the case of the three intermediate revolute axes, shown intersecting at the point x_2.

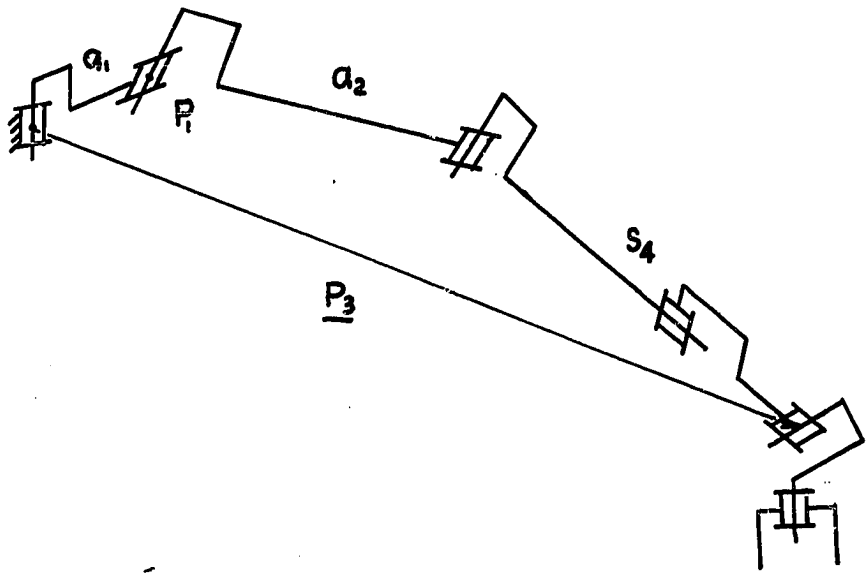


Figure 3.8. Schematic of the 6R, $a_1a_2s_4$ manipulator used at the Stanford Artificial Intelligence Project, with $\alpha_1=90^\circ$, $\alpha_2=0$, $\alpha_3=90^\circ$, $\alpha_4=90^\circ$, $\alpha_5=90^\circ$.

and (3.39) becomes:

$$\left[\frac{R_3-(2a_2s_4s\theta_3+s_4{}^2+a_2{}^2+a_1{}^2)}{2a_1}\right]^2 + z^2 = (a_2+s_4s\theta_3)^2 + s_4{}^2c\theta_3{}^2 \qquad (3.77)$$

which is quadratic in $s\theta_3$ when $c\theta_3{}^2$ is replaced by $1-s\theta_3{}^2$. After finding θ_3 we compute θ_2 from (3.71) and (3.76) and θ_1 from (3.71).

Since the above arm is used in the Stanford Artificial Intelligence Project, we shall use it to illustrate the method of finding the angles associated with the three intersecting axes. Designating the direction of the i^{th} axis by the unit vector $\underline{w}_i$, we write

$$\underline{w}_4 = A_1A_2A_3A_4 \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad (3.78)$$

Using (3.1) for $A_1,\ldots, A_4$ and the above values of α the result is

$$\underline{w}_4 = \begin{bmatrix} c\theta_1c\theta_2s\theta_3 + c\theta_1s\theta_2c\theta_3 \\ s\theta_1c\theta_2s\theta_3 + s\theta_1s\theta_2c\theta_3 \\ s\theta_2s\theta_3 - c\theta_2c\theta_3 \\ 0 \end{bmatrix} \qquad (3.79)$$

so that $\underline{w}_4$ may be computed from (3.79) as we have solved for θ_1, θ_2, and θ_3. $\underline{w}_6$ is known since the hand orientation is specified. In addition,

$$\underline{w}_4 \cdot \underline{w}_5 = \cos\alpha_4 \qquad (3.80)$$

$$\underline{w}_5 \cdot \underline{w}_6 = \cos\alpha_5 \qquad (3.81)$$

$$\underline{w}_5 \cdot \underline{w}_5 = 1 \qquad (3.82)$$

where α_4 and α_5 are link parameters of the arm. In fact $\alpha_4 = -90^o$ and $\alpha_5 = 90^o$. We can find the components of $\underline{\omega}_5$ by simultaneously solving (3.80), (3.81) and (3.82). We observe

$$^7[\underline{\omega}_5] = A_6^{-1} \; ^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{3.83}$$

using (3.53) for A_6^{-1} and A_5^{-1} with $\alpha_4 = \alpha_5 = 90^o$ and $\alpha_6 = 0$ (3.83) becomes:

$$^7[\underline{\omega}_5] = \begin{bmatrix} s\theta_6 \\ c\theta_6 \\ 0 \\ 0 \end{bmatrix} \tag{3.84}$$

and

$$^7[\underline{\omega}_5] = Aeq^{-1} \; \underline{\omega}_5 \tag{3.85}$$

where Aeq is the known matrix specifying hand position and orientation equation (3.7). Its inverse is found as in Section 3.3.4. So that we easily derive θ_6 by equating the right-hand sides (3.84) and (3.85). We also write

$$^7[\underline{\omega}_4] = A_6^{-1} A_5^{-1} A_4^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} s\theta_5 c\theta_6 \\ -s\theta_5 s\theta_6 \\ -c\theta_5 \\ 0 \end{bmatrix} \tag{3.86}$$

and

$$^7[\underline{\omega}_4] = Aeq^{-1} \; \underline{\omega}_4 \tag{3.87}$$

which yield θ_5 . To obtain θ_4 we proceed similarly

$$ {}^7[\underline{\omega}_3] = A_6^{-1} A_5^{-1} A_4^{-1} A_3^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} s\theta_4 c\theta_5 c\theta_6 - c\theta_4 s\theta_6 \\ -s\theta_4 c\theta_5 s\theta_6 - c\theta_4 c\theta_6 \\ s\theta_4 s\theta_5 \\ 0 \end{bmatrix} \qquad (3.88) $$

and

$$ {}^7[\underline{\omega}_3] = Aeq^{-1}\, \underline{\omega}_3 \qquad (3.89) $$

which yields θ_4.

We have indicated a procedure to find the rotation about three intersecting revolute axes when these are located at the hand. The method is applicable when any three axes intersect. However, the equations must then be rewritten in terms of the $\underline{\omega}_i$ and θ_i associated with these axes.

3.3.7 Solution for Any Three Joints Prismatic

A six degree-of-freedom manipulator with any combination of three revolute and three prismatic joints is soluble. This arises from the fact that, the orientation of the hand is independent of the displacement in the prismatic joints, and is only a function of rotation in the three revolute joints. In addition the orientation is independent of the position in space of the revolute axes. Consider the manipulator shown schematically in Figure 3.9. The direction of the first revolute axis is always fixed. With the hand orientation specified, the direction of the third revolute axis becomes fixed. In addition we know the angles which the axis of the second makes with the axes of the first and third revolutes. If we designate the direction of these revolute axes by the

unit vectors, $\underline{\omega}_2$, $\underline{\omega}_3$, and $\underline{\omega}_5$ then we may write

$$\underline{\omega}_2 \cdot \underline{\omega}_3 = \cos\beta_1 \tag{3.90}$$

$$\underline{\omega}_3 \cdot \underline{\omega}_5 = \cos\beta_2 \tag{3.91}$$

$$\underline{\omega}_3 \cdot \underline{\omega}_3 = 1 \tag{3.92}$$

where β_1 and β_2 are the known angles. The equations (3.90) (3.91), (3.92) are then solved for the components of $\underline{\omega}_3$. The joint angles may be found in a manner analogous to that used in the previous example, as the now known direction $\underline{\omega}_3$, can be expressed only as a function of θ_2 which leads to a simple equation for θ_2 . $\underline{\omega}_3$ can also be written in terms of θ_5 alone, yielding θ_5 . Once θ_2 and θ_5 are known, θ_3 is easily found by rewriting (3.4) as

$$A_3 = A_2^{-1} A_1^{-1} Aeq\, A_6^{-1} A_5^{-1} A_4^{-1}$$

Using the values we found for θ_2 and θ_5 plus the constant angles, we compute the rotation portion of the right-hand side of the above equation. Then writing A_3 as in (3.1), we may solve for $c\theta_3$ and $s\theta_3$, thus finding θ_3 . The displacements in the prismatic joints may be found from (3.4). Since all the angles are now known and the s's only appear linearly, the displacement portion of (3.4) easily yield these three unknowns s_1 , s_4 , and s_i .

3.3.8 More Difficult Arrangements

In the previous examples, the existence of three intersecting revolute axes enabled us to separate the problem into two parts - one dealing with position and the other with orientation. The two problems were then solved separately. That is we solved a three

degree-of-freedom position problem and then a three degree-of-freedom orientation problem. A more difficult problem is one in which position and orientation do not separate. An example is the case where just two revolute axes intersect. Consider the $6R$, $a_1 s_2 a_2 s_3 s_4 a_4 s_5 a_5$ manipulator shown in Figure 3.10. Here axes 3 and 4 intersect. The vectors ${}^7\underline{P}$, $\underline{Q}$, and $\underline{R}$ are as shown in Figure 3.10. We make the following observations:

$$\underline{Q} = A_1 A_2 \begin{bmatrix} 0 \\ 0 \\ s_3 \\ 1 \end{bmatrix} \tag{3.93}$$

$${}^7\underline{P} = A_6^{-1} A_5^{-1} \begin{bmatrix} -a_4 \\ -s_3 s\alpha_3 \\ -s_3 c\alpha_3 \\ 1 \end{bmatrix} \tag{3.94}$$

$$\underline{\omega}_3 = A_1 A_2 \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{3.95}$$

$${}^7[\omega_4] = A_6^{-1} A_5^{-1} A_4^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{3.96}$$

$${}^1[{}^7\underline{P}] = \underline{Q} - \underline{R} \tag{3.97}$$

$${}^1[{}^7\underline{P}] \cdot {}^1[{}^7\underline{P}] = \underline{Q}^2 + \underline{R}^2 - 2\underline{Q} \cdot \underline{R} \tag{3.98}$$

Then using (3.1) for the A's (3.93) - (3.96) become (taking $s_1 = s_6 = \alpha_6 = 0$) :

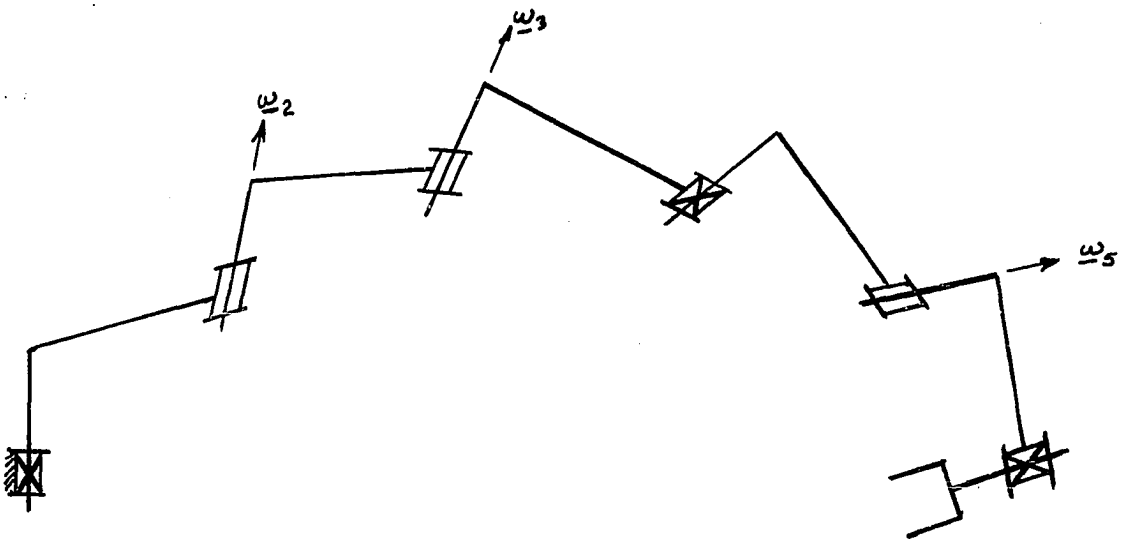


Figure 3.9. A general P-2R-P-R-P manipulator.

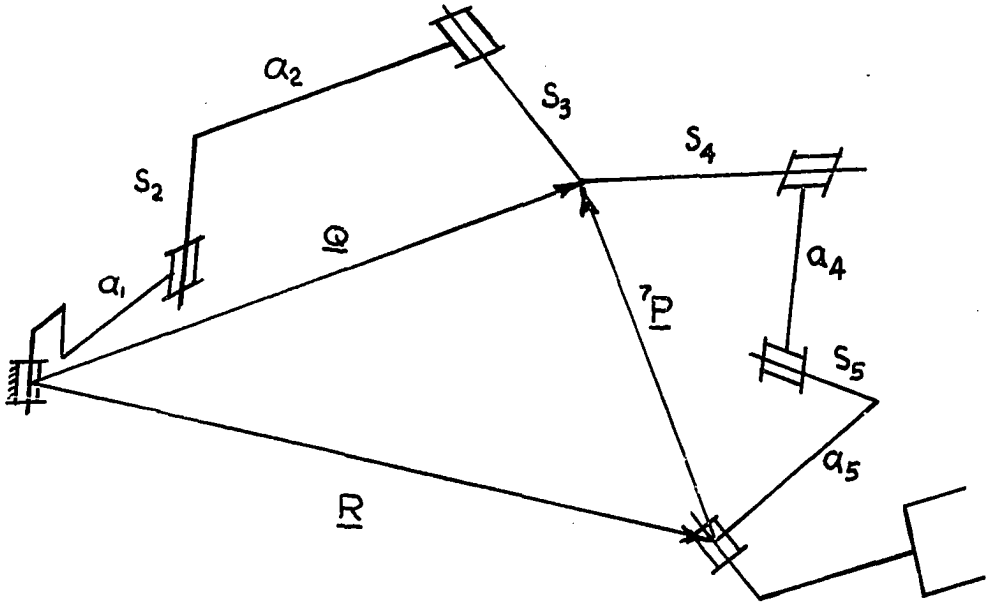


Figure 3.10. A $6R, a_1 s_2 a_2 s_3 s_4 a_4 s_5 a_5$ manipulator. Axes of joints 3 and 4 intersect.

$$\underline{Q} = \begin{bmatrix} s_3(c\theta_1 s\theta_2 s\alpha_2 + s\theta_1 c\theta_2 c\alpha_1 s\alpha_2 + s\theta_1 s\alpha_1 c\alpha_2) \\ +a_2(c\theta_1 c\theta_2 - s\theta_1 s\theta_2 c\alpha_1) + s_2 s\theta_1 s\alpha_1 + a_1 c\theta_1 \\ \\ s_3(s\theta_1 s\theta_2 s\alpha_2 - c\theta_1 c\theta_2 c\alpha_1 s\alpha_2 - c\theta_1 s\alpha_1 c\alpha_2) \\ +a_2(s\theta_1 c\theta_2 + c\theta_1 s\theta_2 c\alpha_1) - s_2 c\theta_1 s\alpha_1 + a_1 s\theta_1 \\ \\ s_3(-c\theta_2 s\alpha_1 s\alpha_2 + c\alpha_1 c\alpha_2) + a_2 s\theta_2 s\alpha_1 + s_2 c\alpha_1 \\ 1 \end{bmatrix} \tag{3.99}$$

$$^7\underline{P} = \begin{bmatrix} a_4(-c\theta_5 c\theta_6 + s\theta_5 s\theta_6 c\alpha_5) - a_5 c\theta_6 - s_5 s\theta_6 s\alpha_5 \\ +s_4(-s\theta_5 c\theta_6 s\alpha_4 - c\theta_5 s\theta_6 s\alpha_4 c\alpha_5 - s\theta_6 c\alpha_4 s\alpha_5) \\ \\ a_4(c\theta_5 s\theta_6 + s\theta_5 c\theta_6 c\alpha_5) + a_5 s\theta_6 - s_5 c\theta_6 s\alpha_5 \\ +s_4(s\theta_5 s\theta_6 s\alpha_4 - c\theta_5 c\theta_6 s\alpha_4 c\alpha_5 - c\theta_6 c\alpha_4 s\alpha_5) \\ \\ -a_4 s\theta_5 s\alpha_5 + s_4(c\theta_5 s\alpha_4 s\alpha_5 - c\alpha_4 c\alpha_5) - s_5 c\alpha_5 \\ \\ 1 \end{bmatrix} \tag{3.100}$$

$$\underline{\omega}_3 = \begin{bmatrix} c\theta_1 s\theta_2 s\alpha_2 + s\theta_1 c\theta_2 c\alpha_1 s\alpha_2 + s\theta_1 s\alpha_1 c\alpha_2 \\ s\theta_1 s\theta_2 s\alpha_2 - c\theta_1 c\theta_2 c\alpha_1 s\alpha_2 - c\theta_1 s\alpha_1 c\alpha_2 \\ -c\theta_2 s\alpha_1 s\alpha_2 + c\alpha_1 c\alpha_2 \\ 0 \end{bmatrix} \tag{3.101}$$

$$^7[\underline{\omega}_4] = \begin{bmatrix} s\theta_5 c\theta_6 s\alpha_4 + c\theta_5 s\theta_6 s\alpha_4 c\alpha_5 + s\theta_6 c\alpha_4 s\alpha_5 \\ -s\theta_5 s\theta_6 s\alpha_4 + c\theta_5 c\theta_6 s\alpha_4 c\alpha_5 + c\theta_6 c\alpha_4 s\alpha_5 \\ -c\theta_5 s\alpha_4 s\alpha_5 + c\alpha_4 c\alpha_5 \\ 0 \end{bmatrix} \tag{3.102}$$

In addition (3.99) and (3.100) respectively yield

$$\underline{Q}^2 = s_3^2 + a_2^2 + s_2^2 + a_1^2 + 2a_1a_2c\theta_2 + 2a_1s_3s\alpha_2s\theta_2 + 2s_2s_3c\alpha_2 \qquad (3.103)$$

$$^7\underline{P}^2 = a_4^2 + s_5^2 + a_5^2 + s_4^2 + 2a_4a_5c\theta_5 + 2s_4a_5s\theta_5s\alpha_4 + 2s_4s_5c\alpha_4 \qquad (3.104)$$

Our approach to this problem is to solve for the coordinates (x, y, z) of the point of intersection of axes 3 and 4. With this in mind we eliminate θ_2 between (3.99) and (3.103) which yields the polynomial:

$$\left[\frac{\underline{Q}^2 - (s_3^2+a_2^2+s_2^2+a_1^2+2s_2s_3c\alpha_2)}{2a_1}\right]^2 + \left[\frac{z - s_2c\alpha_1 - s_3c\alpha_1c\alpha_2}{s\alpha_1}\right]^2 = a_2^2 + s_3^2s\alpha_2^2 \qquad (3.105)$$

where we have defined $\underline{Q}$ in terms of its components

$$\underline{Q} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad (3.106)$$

and

$$\underline{Q}^2 = x^2 + y^2 + z^2 \qquad (3.107)$$

Similarly eliminating θ_5 between the z-component of (3.99) and (3.103) leads to

$$\left[\frac{^7\underline{P}^2 - (a_4^2+s_5^2+a_5^2+s_4^2+2s_4s_5c\alpha_4)}{2a_5}\right]^2 + \left[\frac{^7z+s_4c\alpha_4c\alpha_5+s_5c\alpha_5}{s\alpha_5}\right]^2 = s_4^2s\alpha_4^2 + a_4^2 \qquad (3.108)$$

We note $^1[^7\underline{P}] \cdot ^1[^7\underline{P}] = ^7\underline{P} \cdot ^7\underline{P}$ and using (3.98) for $^1[^7\underline{P}]^2$, we form

$$^7\underline{P}^2 = \underline{Q}^2 + \underline{R}^2 - 2\underline{Q} \cdot \underline{R} \qquad (3.109)$$

also

$$\begin{bmatrix} {}^7x \\ {}^7y \\ {}^7z \\ 1 \end{bmatrix} = Aeq^{-1} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.110}$$

where Aeq^{-1} is defined as in equation (3.53). Thus using (3.100) for ${}^7z$, (3.109) for ${}^7\underline{P}^2$, (3.108) becomes

$$\left[\frac{\underline{Q}^2 + \underline{R}^2 - 2\underline{Q} \cdot \underline{R} - (a_4^2+s_5^2+a_5^2+s_4^2+2s_4s_5c\alpha_4}{2a_5}\right]^2$$

$$+\left[\frac{a_{13}x + a_{23}y + a_{33}z + a_{14}^{-1} + s_4c\alpha_4c\alpha_5+s_5c\alpha_5}{s\alpha_5}\right]^2 = s_4^2s\alpha_4^2 + a_4^2 \tag{3.111}$$

We next want to express $\underline{\omega}_3$ and $\underline{\omega}_4$ in terms of x, y, and z and use the relation

$$\underline{\omega}_3 \cdot \underline{\omega}_4 = \cos\alpha_3 \tag{3.112}$$

For this we need $c\theta_1$, $s\theta_1$, $c\theta_2$, $s\theta_2$, $c\theta_5$, $s\theta_5$, $c\theta_6$, $s\theta_6$ explicitly expressed in terms of x, y, z. We note $c\theta_2$ and $s\theta_2$ are simply obtained from the z-components of (3.99), and from (3.103) and are

$$c\theta_2 = \frac{\dfrac{a_2\left[\underline{Q}^2 - (s_3^2+a_2^2+s_2^2+a_1^2+2s_2s_3c\alpha_2)\right]}{2a_1} - \dfrac{s_3s\alpha_2\left[z - (s_2c\alpha_1+s_3c\alpha_1c\alpha_2)\right]}{s\alpha_1}}{a_2^2 + s_3^2s\alpha_2^2} \tag{3.113}$$

$$s\theta_2 = \frac{\frac{a_2[z-(s_2c\alpha_1+s_3c\alpha_1c\alpha_2)]}{s\alpha_1} + \frac{s_3s\alpha_2}{2a_1}\left[Q^2-(s_3^2+a_2^2+s_2^2+a_1^2+2s_2s_3c\alpha_2)\right]}{a_2^2+s_3^2s\alpha_2^2} \tag{3.114}$$

$c\theta_1$ and $s\theta_1$ from (3.99) are, after simplification

$$c\theta_1 = \frac{x(s_3s\theta_2s\alpha_2+a_2c\theta_2+a_1) - y\left[s_3(c\theta_2c\alpha_1s\alpha_2+s\alpha_1c\alpha_2) - a_2s\theta_2c\alpha_1 + s_2s\alpha_1\right]}{x^2+y^2} \tag{3.115}$$

$$s\theta_1 = \frac{y(s_3s\theta_2s\alpha_2+a_2c\theta_2+a_1)+ x\left[s_3(c\theta_2c\alpha_1s\alpha_2+s\alpha_1c\alpha_2) - a_2s\theta_2c\alpha_1 + s_2s\alpha_1\right]}{x^2+y^2} \tag{3.116}$$

Where we may use (3.113) and (3.114) for $c\theta_2$ and $s\theta_2$. When (3.113) (3.114), (3.115) and (3.116) are used in (3.101) to express ω_3 in terms of x, y, and z, the result is a third degree expression in x, y, and z. If we do similar things with θ_5 and θ_6 for ω_4 then (3.112) becomes a polynomial of degree six in x, y, z. This along with (3.103) and (3.111) are three equations for x, y, and z. However, they are of such large degree that finding all the roots is not feasible. Though there are some special cases of interest.

For a 6R; a_2a_4 manipulator, with $\alpha_1 = \alpha_3 = \alpha_5 = 90^\circ$ and $\alpha_2 = \alpha_4 = -90^\circ$ the equations reduce to a degree which is workable. This configuration is shown in Figure 3.11. Equation (3.105) reduces to

$$x^2 + y^2 + z^2 = a_2^2 \tag{3.117}$$

and (3.111) reduces to

$$x^2 + y^2 + z^2 + x_4^2 + y_4^2 + z_4^2 - 2xx_4 - 2yy_4 - 2z_4z = a_4^2 \tag{3.118}$$

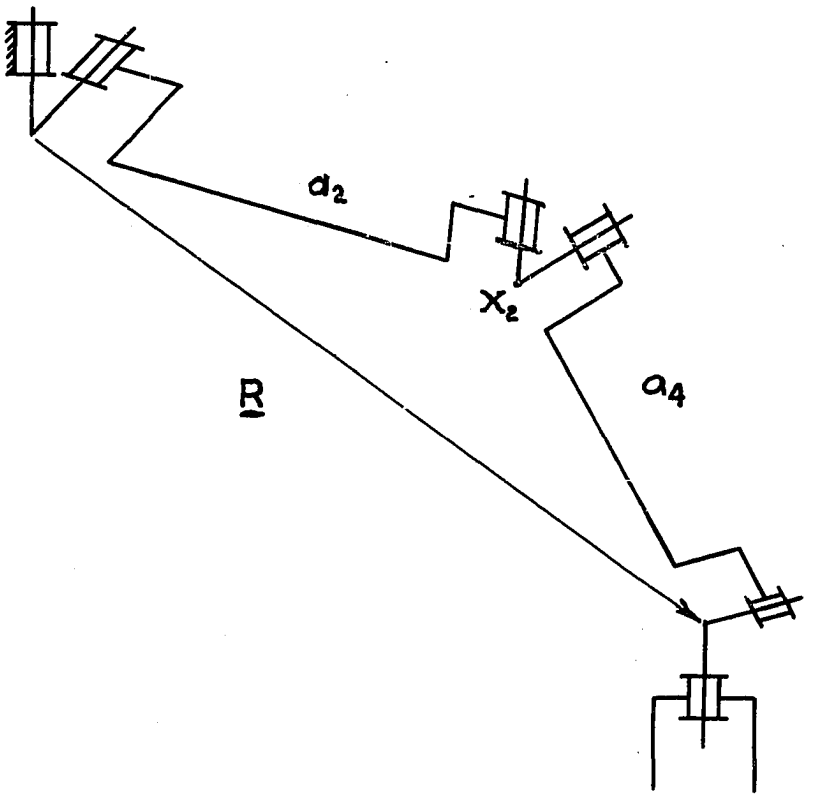


Figure 3.11. A $6R, a_2 a_4$ manipulator with adjacent axes orthogonal. $X_2(x,y,z)$ is the point of intersection of axes 3 and 4.

where $\underline{R}$ in (3.111) has been replaced by its components, x_4 , y_4 , z_4 , and the indicated dot product performed.

(3.99), (3.100), (3.101), (3.102) reduce to

$$\underline{Q} = \begin{bmatrix} a_2 c\theta_1 c\theta_2 \\ a_2 s\theta_1 c\theta_2 \\ a_2 s\theta_2 \\ 1 \end{bmatrix} \tag{3.119}$$

$$^7\underline{P} = \begin{bmatrix} -a_4 c\theta_5 c\theta_6 \\ a_4 c\theta_5 s\theta_6 \\ -a_4 s\theta_5 \\ 1 \end{bmatrix} \tag{3.120}$$

$$\underline{\omega}_3 = \begin{bmatrix} -c\theta_1 s\theta_2 \\ -s\theta_1 s\theta_2 \\ c\theta_2 \\ 0 \end{bmatrix} \tag{3.121}$$

$$^7[\underline{\omega}_4] = \begin{bmatrix} -s\theta_5 c\theta_6 \\ s\theta_5 s\theta_6 \\ c\theta_5 \\ 0 \end{bmatrix} \tag{3.122}$$

from (3.119) we obtain

$$c\theta_1 = \frac{x}{a_2 c\theta_2} \tag{3.123}$$

$$s\theta_1 = \frac{y}{a_2 c\theta_2} \tag{3.124}$$

$$s\theta_2 = \frac{z}{a_2} \tag{3.125}$$

using (3.123), (3.124), (3.125) in (3.121)

$$\underline{w}_3 = \frac{1}{a_2^2 c\theta_2} \begin{bmatrix} -xy \\ -yz \\ a_2^2 c\theta_2^2 \\ 0 \end{bmatrix} \tag{3.126}$$

replacing $c\theta_2^2$ by $1 - s\theta_2^2$ and using $s\theta_2$ from (3.125), (3.126) becomes

$$\underline{w}_3 = \frac{1}{a_2^2 \left[1 - \frac{z^2}{a_2^2}\right]^{\frac{1}{2}}} \begin{bmatrix} -xz \\ -yz \\ -z^2 + a_2^2 \\ 0 \end{bmatrix} \tag{3.127}$$

from (3.120) we obtain

$$c\theta_6 = \frac{-r_x}{a_4 c\theta_5} \tag{3.128}$$

$$s\theta_6 = \frac{r_y}{a_4 c\theta_5} \tag{3.129}$$

$$s\theta_5 = \frac{-r_z}{a_4} \tag{3.130}$$

substituting (3.128), (3.129), and (3.130) in (3.122) and simplifying gives us

$$r[\underline{w}_4] = \frac{1}{a_4^2 \left[1 - \frac{r_z^2}{a_2^2}\right]^{\frac{1}{2}}} \begin{bmatrix} -r_y r_z \\ -r_y r_z \\ -r_z^2 + a_4^2 \\ 0 \end{bmatrix} \tag{3.131}$$

We now rotate $^7[\underline{\omega}_4]$ to express it in terms of system 1 by

$$\underline{\omega}_4 = \text{Aeq}\,^7[\underline{\omega}_4] \tag{3.132}$$

with Aeq as in (3.53) and $^7[\underline{\omega}_4]$ as in (3.131) we get

$$\underline{\omega}_4 = \frac{1}{a_4{}^2\left[1 - \frac{^7z^2}{a_4{}^2}\right]^{\frac{1}{2}}} \begin{bmatrix} -a_{11}{}^7x\,{}^7z - a_{12}{}^7y\,{}^7z - a_{13}{}^7z^2 + a_{13}a_4{}^2 \\ -a_{21}{}^7x\,{}^7z - a_{22}{}^7y\,{}^7z - a_{23}{}^7z^2 + a_{23}a_4{}^2 \\ -a_{31}{}^7x\,{}^7z - a_{32}{}^7y\,{}^7z - a_{33}{}^7z^2 + a_{33}a_4{}^2 \\ 0 \end{bmatrix} \tag{3.133}$$

To eliminate ^7x , ^7y , ^7x from (3.133) we use (3.110) with Aeq from (3.53) which after simplification yield:

$$\underline{\omega}_4 = \frac{1}{a_4{}^2\left[1 - \frac{^7z^2}{a_4{}^2}\right]^{\frac{1}{2}}} \begin{bmatrix} -(a_{13}x+a_{23}y+a_{33}z+a_{34}{}^{-1})(x-x_4) + a_{13}a_4{}^2 \\ -(a_{13}x+a_{23}y+a_{33}z+a_{34}{}^{-1})(y-y_4) + a_{23}a_4{}^2 \\ -(a_{13}x+a_{23}y+a_{33}z+a_{34}{}^{-1})(z-z_4) + a_{33}a_4{}^2 \\ 0 \end{bmatrix} \tag{3.134}$$

Then using (3.127) for $\underline{\omega}_3$ and (3.134) for $\underline{\omega}_4$ the equation $\underline{\omega}_3 \cdot \underline{\omega}_4 = 0$ results in the polynomial:

$$\begin{aligned} &z(a_{13}x+a_{23}y+a_{33}z+a_{34}{}^{-1})(x^2+y^2+z^2-xx_4-yy_4-zz_4-a_2{}^2-a_4{}^2) \\ &+ za_{34}{}^{-1}a_4{}^2 + z_4a_2{}^2(a_{13}x+a_{23}y+a_{33}z+a_{34}{}^{-1}) + a_2{}^2a_4{}^2a_{33} = 0 \end{aligned} \tag{3.135}$$

We note that linear combinations of the equations (3.117), (3.118) and (3.135) can be formed to reduce the degree of the equations. Equation (3.117) we leave as is. Combining (3.117) with (3.118) leads to

$$2x_4x + 2y_4y + 2z_4z + a_4{}^2 - a_2{}^2 - (x_4{}^2+y_4{}^2+z_4{}^2) = 0 \tag{3.136}$$

and using (3.136) and (3.117) in (3.135) yields

$$
\begin{aligned}
0 = {} & xz\left[a_{13}\left(-\frac{a_4^2}{2} - \frac{x_4^2 + y_4^2 + z_4^2}{2} - \frac{a_4^2}{2}\right)\right] \\
& + yz\left[a_{23}\left(-\frac{a_2^2}{2} - \frac{x_4^2 + y_4^2 + z_4^2}{2} - \frac{a_4^2}{2}\right)\right] \\
& + z^2\left[a_{33}\left(-\frac{a_2^2}{2} - \frac{x_4^2 + y_4^2 + z_4^2}{2} - \frac{a_4^2}{2}\right)\right] \\
& + x\,(a_{13} z_4 a_2^2) \\
& + y\,(a_{23} z_4 a_2^2) \\
& + z\left[(a_{33} z_4 a_2^2) + a_{34}^{-1} a_4^2 + a_{34}\left(-\frac{a_2^2}{2} - \frac{x_4^2 + y_4^2 + z_4^2}{2} - \frac{a_4^2}{2}\right)\right] \\
& + z_4 a_2^2 a_{34}^{-1} + a_2^2 a_4^2 a_{33}
\end{aligned}
\qquad (3.137)
$$

The equations (3.117), (3.136) and (3.137) are three equations for x, y, and z. The linear equation (3.117) can be used to eliminate one variable easily. Another variable can be eliminated between (3.136) and (3.137) leading to a polynomial of degree four. This procedure has been carried out and programmed on the PDP-6. An analysis program was used to generate inputs with known angles to check the results. A typical example was generated by the arbitrary input angles $\theta_1 = 34^\circ$, $\theta_2 = 21^\circ$, $\theta_3 = 78^\circ$, $\theta_4 = -56^\circ$, $\theta_5 = 23^\circ$, $\theta_6 = 1^\circ$ and link parameters $a_2 = a_4 = 15''$, which gave:

$$
Aeq = \begin{bmatrix}
-0.322 & -0.481 & 0.816 & 12.066 \\
0.555 & 0.641 & 0.577 & 18.035 \\
-0.801 & 0.598 & 0.037 & -5.609 \\
0 & 0 & 0 & 1
\end{bmatrix}
$$

For the above, the four sets of common roots were real and lead to four sets of angles for each root. They are shown in Table 3.1.

If $a_1 \neq 0$ then the solution may first be expressed as three quadratic equations in three unknowns (x , y , and z) and finally as an eighth degree polynomial. With $a_1 \neq 0$, $a_2 \neq 0$, $a_4 \neq 0$, and α's as before, and with x , y , z defined as before, (3.105) becomes

$$(x^2 + y^2 + z^2 - a_1^2 - a_2^2)^2 + 4a_1^2 (z^2 - a_2^2) = 0 \qquad (3.138)$$

and equation (3.111) reduces to (3.118) as previously noted. To form $\underline{w}_3$ we use $c\theta_1$ and $s\theta_1$ from (3.115) and (3.116). Next we use $c\theta_2$ and $s\theta_2$ from (3.113) and (3.114) and substituting these quantities into (3.101), we obtain after simplification:

$$\underline{w}_3 = \frac{1}{2a_2\left[\left(a_1 + \frac{1}{2a_1}(x^2+y^2+z^2-a_2^2-a_1^2)\right)\right]} \begin{bmatrix} -2xz \\ -2yz \\ -2z^2 + (x^2+y^2+z^2-a_2^2) + 2a_2^2 - a_1^2 \\ 0 \end{bmatrix} \qquad (3.139)$$

$\underline{w}_4$ is as before and given in (3.134). By using (3.139) and (3.138) in (3.112), by replacing $(x^2+y^2+z^2)$ with its equivalent from (3.118), and by simplifying, (3.112) becomes:

$$\begin{aligned} 0 = \; & x^2(2a_{13}z_4x_4) \\ & + y^2(2a_{23}z_4y_4) \\ & + z^2[a_{33}(-\underline{R}^2 - a_4^2 + a_1^2 - a_2^2) + 2a_{33}z_4^2] \\ & + xy[2a_{13}z_4y_4 + 2a_{23}z_4x_4] \\ & + yz[2a_{23}z_4^2 + a_{23}(-\underline{R}^2 - a_4^2 + a_1^2 - a_2^2) + 2a_{33}z_4y_4] \end{aligned}$$

X	Y	Z	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
11.601	7.831	5.376	34.00	21.00	78.00	-56.00	23.00	1.00
			34.00	21.00	258.00	236.00	157.00	-179.00
			-146.00	159.00	102.00	124.00	23.00	1.00
			-146.00	159.00	-78.00	56.00	157.00	-179.00
10.657	8.670	6.023	39.13	23.67	74.70	-63.85	24.12	6.81
			39.13	23.67	254.70	243.85	155.88	-173.19
			-140.87	156.33	105.30	116.15	24.12	6.81
			-140.87	156.33	-74.70	63.85	155.88	-173.19
-1.505	12.308	-8.441	96.97	-34.24	97.02	162.09	74.88	162.82
			96.97	-34.24	-82.98	17.91	105.12	-17.18
			-83.03	214.24	82.98	-17.91	74.88	168.82
			-83.03	214.24	262.98	197.91	105.12	-17.18
5.257	6.542	-12.432	51.22	-55.98	47.32	80.12	56.01	-102.72
			51.22	-55.98	227.32	99.88	123.99	77.28
			-128.78	235.98	132.68	260.12	56.01	-102.72
			-128.78	235.98	-47.32	-80.12	123.99	77.28

Table 3.1

Numerical Results Obtained from Example Solution of 6R, a_2a_4 Manipulator

$$+ zx[2a_{33}z_4x_4 + 2a_{13}z_4^2 + a_{13}(-\underline{R}^2 - a_4^2 + a_1^2 - a_1^2)]$$
$$+ x[a_{13}z_4(-\underline{R}^2 + a_4^2 - a_1^2 + a_2^2) + 2a_{33}a_4^2x_4 + 2a_{34}^{-1}z_4x_4]$$
$$+ y[a_{23}z_4(-\underline{R}^2 + a_4^2 - a_1^2 + a_2^2) + 2a_{33}a_4^2y_4 + 2a_{34}^{-1}z_4y_4]$$
$$+ z[a_{33}z_4(-\underline{R}^2 + a_4^2 - a_1^2 + a_2^2) + 2a_{33}a_4^2z_4 + a_{34}^{-1}(2z_4^2 - \underline{R}^2 + a_4^2 + a_1^2 - a_2^2)]$$
$$+ [a_{34}^{-1}z_4(-\underline{R}^2 + a_4^2 - a_1^2 + a_2^2) + a_{33}a_4^2(-\underline{R}^2 + a_4^2 - a_1^2 + a_2^2)] . \quad (3.140)$$

When (3.118) is used for $x^2 + y^2 + z^2$ in (3.138), that equation becomes quadratic. This together with (3.118), and (3.140) are the three quadratics for x, y, and z. Eliminating two variables produces a single polynomial of degree eight. The preceding was programmed on the PDP-6 to yield a final polynomial in z. For the link parameters $a_1=13$, $a_2=15$, $a_4=15$, several examples were run. Examples were found in which eight sets of values did indeed satisfy the three quadratics. One of these, generated in the input angles $\theta_1=9^o$, $\theta_2=175^o$, $\theta_3=188^o$, $\theta_4=173^o$, $\theta_5=174^o$, $\theta_6=169^o$, led to the following set of elbow positions:

	x	y	z
1	-25.342	11.820	1.048
2	-24.457	-13.534	1.200
3	-1.914	-0.569	0.294
4	-1.919	-0.304	1.307
5	-1.960	0.399	-0.019
6	-1.979	-0.168	-0.641
7	-12.297	0.735	14.985
8	-18.119	1.073	-14.088

Now, in order to extend the above problem to include $a_5 \neq 0$, we must define a new variable

$$W = x^2 + y^2 + z^2 \quad (3.141)$$

We then replace the terms $(x^2+y^2+z^2)$ with W in (3.105) (3.111), and (3.112) and appropriately rewrite w_4. Equations (3.105), (3.111), (3.112), and (3.141) become quadratic in w, x, y, and z. The details of this may be found in Appendix V. According to Bezout's theorem this system has at most 16 sets of common roots. However, no method is known by which three of the variables may be eliminated to attain one polynomial of only degree 16.

To summarize the above we have found that:

1. A 6R, a_2a_4 may have as many as four different configurations leading to the same hand position and orientation.
2. A 6R, $a_1a_2a_4$ may have as many as eight different elbow positions (the elbow is considered to be the point of intersection of axes 3 and 4) leading to the same hand position and orientation.
3. A 6R, $a_1a_2a_4a_5$ will have at most 16 different positions that the elbow can assume for each fixed hand position and orientation.

CHAPTER IV

NUMERICAL SOLUTIONS

Our solutions so far have been made possible by the existence of special geometry. To analyze more general cases, iterative procedures must be used.. Two procedures are presented to handle these cases. The first employs the well-known Newton-Raphson technique* and the second applies velocity methods.

4.1 Newton-Raphson

The Newton-Raphson method assumes the existence of an approximate solution. Then the equations are linearized and an increment to this approximation is computed hopefully leading to a more accurate approximation. We write

$$\theta_i = \theta_{io} + \delta\theta_i \qquad i = 1,\ldots,6 \tag{4.1}$$

where θ_{io} is the first approximation, and $\delta\theta_i$ is the increment, and θ_i is the more accurate approximation. We may then write (3.4) as

$$A_i = \begin{bmatrix} c(\theta_{io}+\delta\theta_i) & -s(\theta_{io}+\delta\theta_i)c\alpha_i & s(\theta_{io}+\delta\theta_i)s\alpha_i & a_i c(\theta_{io}+\delta\theta_i) \\ s(\theta_{io}+\delta\theta_i) & c(\theta_{io}+\delta\theta_i)c\alpha_i & -c(\theta_{io}+\delta\theta_i)s\alpha_i & a_i s(\theta_{io}+\delta\theta_i) \\ 0 & s\alpha_i & c\alpha_i & s_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

*This method was applied to seven link mechanisms with revolute pairs by Uicker, Denavit, and Hartenberg [49]. The approach presented here is similar to theirs. They assumed the motion of one link to be prescribed as an input and found the displacement of the rest of the mechanism as they incremented the input.

Then expanding $c(\theta_{io} + \delta\theta_i)$ and $s(\theta_{io} + \delta\theta_i)$, using trigonometric identities for the sum of two angles and letting $c(\delta\theta_i)=1$, $s(\delta\theta_i)=\delta\theta_i$ (4.2) becomes

$$A_i = \begin{bmatrix} c\theta_{io} & -s\theta_{io}c\alpha_i & s\theta_{io}s\alpha_i & a_i c\theta_{io} \\ s\theta_{io} & c\theta_{io}c\alpha_i & -c\theta_{io}s\alpha_i & a_i s\theta_{io} \\ 0 & s\alpha_i & c\alpha_i & s_i \\ 0 & 0 & 0 & 1 \end{bmatrix} + \delta\theta_i \begin{bmatrix} -s\theta_{io} & -c\theta_{io}c\alpha_i & c\theta_{io}s\alpha_i & -a_i s\theta_{io} \\ c\theta_{io} & s\theta_{io}c\alpha_i & s\theta_{io}s\alpha_i & a_i c\theta_{io} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

which we write as

$$A_i = A_{io} + \delta\theta_i B_{io} \tag{4.4}$$

where A_{io} and B_{io} are defined from (4.3). Using (4.4) in the basic eqn. (3.4) and retaining only terms of degree one or less in $\delta\theta_i$ we have

$$\begin{aligned} &\delta\theta_1 (B_{10}A_{20}A_{30}A_{40}A_{50}A_{60}) \\ &+ \delta\theta_2 (A_{10}B_{20}A_{30}A_{40}A_{50}A_{60}) \\ &+ \delta\theta_3 (A_{10}A_{20}B_{30}A_{40}A_{50}A_{60}) \\ &+ \delta\theta_4 (A_{10}A_{20}A_{30}B_{40}A_{50}A_{60}) \\ &+ \delta\theta_5 (A_{10}A_{20}A_{30}A_{40}B_{50}A_{60}) \\ &+ \delta\theta_6 (A_{10}A_{20}A_{30}A_{40}A_{50}B_{60}) = Aeq - A_{10}A_{20}A_{30}A_{40}A_{50}A_{60} \end{aligned} \tag{4.5}$$

The matrix equation (4.5) contains six independent linear equations that may be used to compute $\delta\theta_i, (i=1,\ldots,6)$. It is noted that the preceding was developed for revolute joints, but the method is also applicable to manipulators with prismatic joints provided appropriate changes are made in the B_{io}.

This method lends itself to computation on a digital computer. A program has been written implementing this scheme. The inputs to the

program are the joint angles of the manipulator in its initial state, and the desired final hand position and orientation. The output is a set of angles leading to the final state. If the final state is a great deal different from the initial state, then solutions of (4.5) will not yield "small" corrections and the method will not converge. In order that (4.5) be valid it is necessary to generate intermediate targets. The right-hand side of (4.5) represents a translation and a rotation. Intermediate goals are specified by taking a fraction of the total rotation and translation. The program begins with the initial angles as the first approximation. Then it computes the next approximation based on an intermediate goal. A new intermediate goal is computed and the process continues until a satisfactory set of angles is found or the method fails to converge after a fixed number of iterations. See Appendix I for details.

4.2 Iterative Velocity Method

The iterative velocity method is based on the fact that a change in position and orientation of a rigid body (in this case the hand) can be expressed as a screw - a rotation about and a translation along a single fixed axis. In addition, for small motion, it can be shown that the screw is related to the angular velocity.

We write $\underline{W}$ and $\underline{V}$ as approximations respectively to the angular velocity of the hand and the linear velocity of a point in the hand at

*On the right-hand side of (4.5), Aeq represents the desired position and the product of the six matrices represents the present position. Hence, the difference gives the displacement which may be represented as a rotation and a translation.

the origin:

$$\underline{W} = \frac{\Delta\varphi}{\Delta t}\ \underline{n} \tag{4.6}$$

$$\underline{V} = h\ \frac{\Delta\varphi}{\Delta t}\ \underline{n} - \underline{n} \times \underline{r}\ \frac{\Delta\varphi}{\Delta t}\quad , \tag{4.7}$$

where quantities on the left-hand side of the above are found from the screw; $\Delta\varphi$ is the amount of rotation, $\underline{n}$ is a unit vector parallel to the screw axis, H is the pitch of the screw, and $\underline{r}$ is a vector from the origin to the screw axis. The details are shown in Appendix II. In addition we may express the angular and linear velocity as functions of the rotations in the arm joints. That is

$$\underline{W} = \sum_{i=1}^{6} \underline{W}_i \tag{4.8}$$

$$\underline{V} = \sum_{i=1}^{6} \underline{W}_i \times \underline{r}_i \tag{4.9}$$

where $\underline{W}_i$ is the angular velocity of the hand due to the rotation about axis i and $\underline{r}_i$ is a vector from the origin of system 1 to axis i. We make the approximation

$$\underline{W}_i = \frac{\Delta\theta_i}{\Delta t}\ \underline{n}_i \quad (i=1,\ldots,6) \tag{4.10}$$

where $\underline{n}_i$ is a unit vector parallel to axis i and we assume that the motion of the hand from initial to final position is small, so that (4.8) and (4.9) may be written using (4.10) as

$$\underline{W} = \sum_{i=1}^{6} \frac{\Delta\theta_i}{\Delta t}\ \underline{n}_i \tag{4.11}$$

$$\underline{V} = \sum_{i=1}^{6} \frac{\Delta\theta_i}{\Delta t}\ \underline{n}_i \times \underline{r}_i \tag{4.12}$$

Then equating the right hand sides of (4.11) and (4.12) to the right-hand sides of (4.6) and (4.7), and we obtain two vector equations representing six scalar linear equations in $\Delta\theta_i$, i=1,...,6. Equating and dividing by Δt yields:

$$\sum_{i=1}^{6} (\Delta\theta_i \, \underline{n}_i) = \Delta\varphi \underline{n} \tag{4.13}$$

$$-\sum_{i=1}^{6} (\Delta\theta_i \, \underline{n}_i \times \underline{r}_i) = H \, \Delta\varphi \, \underline{n} - \underline{n} \times \underline{r} \, \Delta\varphi \tag{4.14}$$

The right-hand sides of (4.1) and (4.14) are computed from the known changes in position and orientation of the hand. Since the initial configuration of the manipulator is known, we have values for the $\underline{n}_i$ and $\underline{r}_i$. As long as the changes in position and orientation, as represented by the screw, are small, then the solution of this set of equations gives small changes in the joint angles. Thus the $\underline{r}_i$ and the $\underline{n}_i$ do not change very much and we are justified in using their values in the initial state. To apply this method we must insure that the right-hand sides of these equations are small. Therefore, for large motions, we take only a portion of the screw to compute the incremental change in the angles. We also limit the change that is made at each iteration.

If any of the revolutes are replaced by prismatic joints, this method may still be applied with appropriate changes in (4.11) and (4.12).

A computer program has been written utilizing the above scheme, details of which can be found in Appendix II.

4.3 Comparison of the Methods

It was desirable to test and compare these methods to determine their practicability in finding solutions for complex manipulators. In addition it was hoped that the velocity method would be faster as it does not require the matrix multiplication that the more conventional Newton-Raphson method does. A $6R_{,}s_2a_2s_3a_4$ manipulator was used as a trial for the two methods. For the purpose of testing, the target hand positions were generated by sets of known angles. Programs were written in FORTRAN IV for the PDP-6. With this machine, an iteration using Newton-Raphson took 0.140 seconds while the velocity method took 0.097 seconds. A typical example is:

With the parameters of the arm fixed at

$$\alpha_1 = \alpha_3 = \alpha_5 = 90^o$$

$$\alpha_2 = \alpha_4 = -90^o$$

$$a_2 = 0.375$$

$$s_3 = 12.2$$

$$a_3 = 0.375$$

$$s_5 = 9.5$$

and arbitrarily

$$a_6 = 5.9$$

$$\alpha_6 = 0 \quad ,$$

the target was generated by the angles

$$\theta_1 = 100^o,\ \theta_2 = 100^o,\ \theta_3 = 30^o,\ \theta_4 = 30^o,\ \theta_5 = 50^o,\ \theta_6 = 0\ .$$

This leads to the hand position specified by position vector

$$\underline{P}_2 = \begin{bmatrix} 27.915 \\ -0.869 \\ 0.314 \end{bmatrix}$$

and the orientation (specified by two unit vectors fixed in the hand, $\underline{L}_2$ pointing in the direction of the hand and $\underline{N}_2$ in the direction of the sixth revolute axis);

$$L_2 = \begin{bmatrix} -0.869 \\ 0.492 \\ -0.042 \end{bmatrix} , \qquad \underline{N}_2 = \begin{bmatrix} 0.314 \\ 0.486 \\ -0.816 \end{bmatrix} .$$

The initial configuration of the arm was

$$\theta_1 = 70^o,\ \theta_2 = 80^o,\ \theta_3 = 40^o,\ \theta_4 = 0,\ \theta_5 = 60^o,\ \theta_6 = 30^o$$

with

$$\underline{P}_1 = \begin{bmatrix} 14.229 \\ 20.295 \\ 14.141 \end{bmatrix} , \quad \underline{L}_1 = \begin{bmatrix} -0.979 \\ 0.196 \\ 0.061 \end{bmatrix} \quad \underline{N}_1 = \begin{bmatrix} -0.105 \\ -0.220 \\ -0.970 \end{bmatrix} .$$

The velocity method resulted in:

$$\theta_1 = 100.00,\ \theta_2 = 100.00,\ \theta_3 = 30.00,\ \theta_4 = 30.00,\ \theta_5 = 50.00,\ \theta_6 = 0.00$$

$$\underline{P}_2 = \begin{bmatrix} 27.914 \\ 21.001 \\ 24.863 \end{bmatrix} , \quad \underline{L}_2 = \begin{bmatrix} -0.870 \\ 0.492 \\ -0.042 \end{bmatrix} \quad \underline{N}_2 = \begin{bmatrix} 0.313 \\ 0.486 \\ -0.816 \end{bmatrix}$$

Number of iterations = 10

Run time = 0.97 seconds

The Newton-Raphson method resulted in:

$$\theta_1 = 100.00 \quad \theta_2 = 100.00 \quad \theta_3 = 30.02 \quad \theta_4 = 30.00 \quad \theta_5 = 49.98 \quad \theta_6 = -0.01$$

$$\underline{P}_2 = \begin{bmatrix} 27.914 \\ 21.002 \\ 24.862 \end{bmatrix} \quad \underline{L}_2 = \begin{bmatrix} -0.870 \\ 0.492 \\ -0.042 \end{bmatrix} \quad \underline{N}_2 = \begin{bmatrix} 0.314 \\ 0.486 \\ -0.816 \end{bmatrix}$$

Number of iterations = 13

Run time = 1.82 seconds

From the results of many tests similar to the above the following was observed:

1. For small motions (rotations of about 10^{p} in each joint) both methods converged to solutions but the velocity method generally had fewer iterations.
2. For larger motions (rotations of 10^{o} - 90^{o} in each joint), Newton-Raphson did not always converge within the upper limit of 400 iterations. The velocity method did in all cases tested.
3. For even larger motions, the velocity method did not always converge within 400 iterations but did converge in all examples when allowed more than 400 iterations. However, such was not the case with Newton-Raphson. In some examples even after 4000 iterations, it still did not converge.

Both methods become very time consuming whenever the course of the solution takes the arm to the equivalent of a "stretched out" position. That is whenever the hand is in a position from which it cannot move in an arbitrary direction and rotate about an arbitrary axis, the system of equations formed in both the above methods degenerates. Generally, these methods work their way out of such predicaments by taking very small steps, and by benefiting from round-off error inherent in the computations. The further the distance between initial and final states, the more degeneracies that are likely to be encountered enroute to the final state.

Even though convergence is occasionally slow, the velocity method reached a solution in all cases tested, thereby proving it to be useful for complex manipulators. In particular for a short range of motion it was very efficient. Thus it might be used most effectively to find the final set of angles, when a first approximation has been obtained using a rather simplified model of the manipulator.

CHAPTER V

A DIGITAL MANIPULATOR

5.1 Description of the Manipulator

One type of manipulator whose solution does not fall into the class previously discussed is one containing more than six degrees-of-freedom but having a limited motion in each joint. An articulated arm of this type having many degrees-of-freedom was described by Anderson and Horn [37]. They found that such a design was practical for use in an underwater laboratory. In fact, they claim that this design optimized many desirable criteria such as slenderness, cost, microdexterity and range of operation.

If, in addition to restricting the range of freedom at each joint, we allow only a finite number of states to exist at each joint, then the arm becomes digital in nature. This makes it easy to be interfaced with a digital computer. The concept of such an arm was suggested by L. Leifer who together with V. Scheinman developed working models for the Stanford Artificial Intelligence Project (see Figure 5.1). Since the arm is snake-like in form, they named it the "ORM" (the Norwegian word for snake).

We shall examine the problem of finding a solution for a digital arm. It can be seen that knowledge of the state of each joint together with knowledge of the link geometry is sufficient to specify the position of the hand. The orientation freedom of this device is limited. In practice it would need to have a wrist capable of putting the hand in

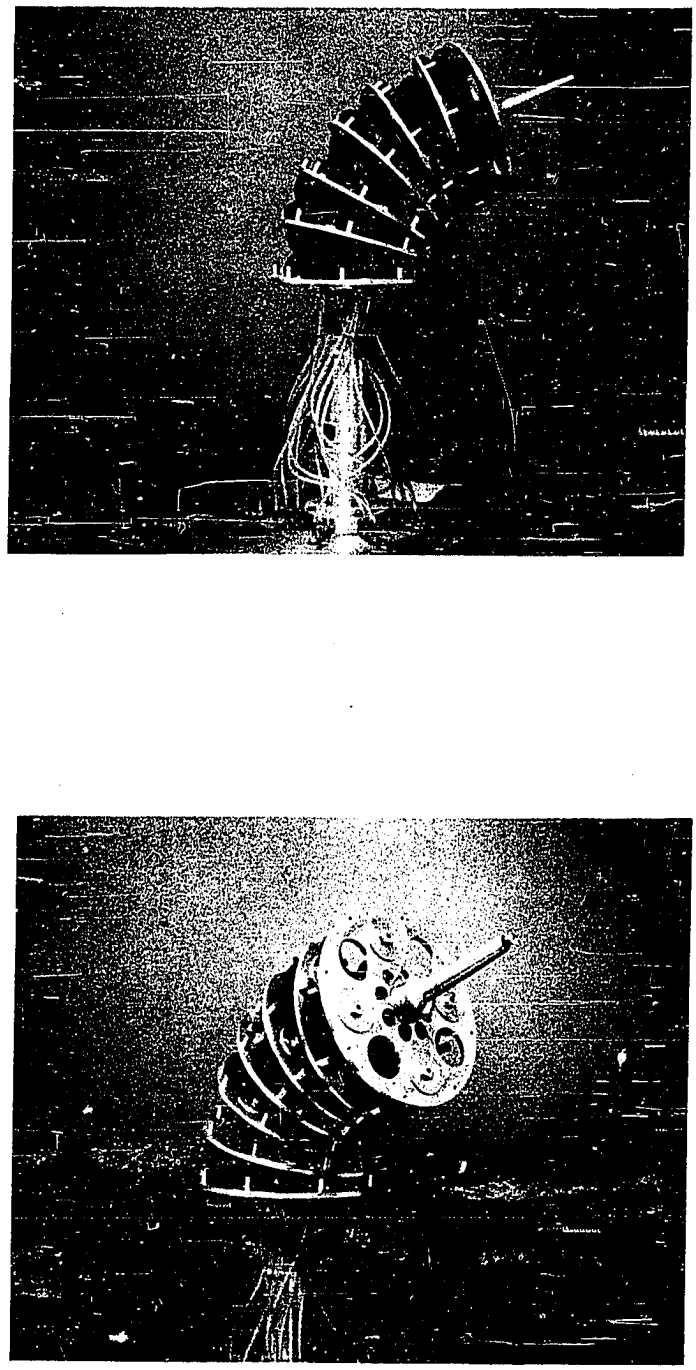


Figure 5.1. Working model of the ORM developed at Stanford.

the proper orientation. We, therefore, examine the problem: given the desired position of a point in the hand, to find the state of each joint leading to this position.

The problem is divided in two parts. The first is to consider a two-dimensional arm and develop a technique to solve it. The second is to develop a method for three dimensions with insight gained from looking at the two-dimensional problem.

5.2 Two-Dimensional Model

The two-dimensional or planar arm to be considered is binary in nature. In other words, there are only two states for each joint. If the arm is made up of n links, there are 2^n possible configurations. A model of this arm is shown in Figure 5.2, where the angle between two adjacent links can be either $+\theta_o$ or $-\theta_o$ where θ_o is a constant.

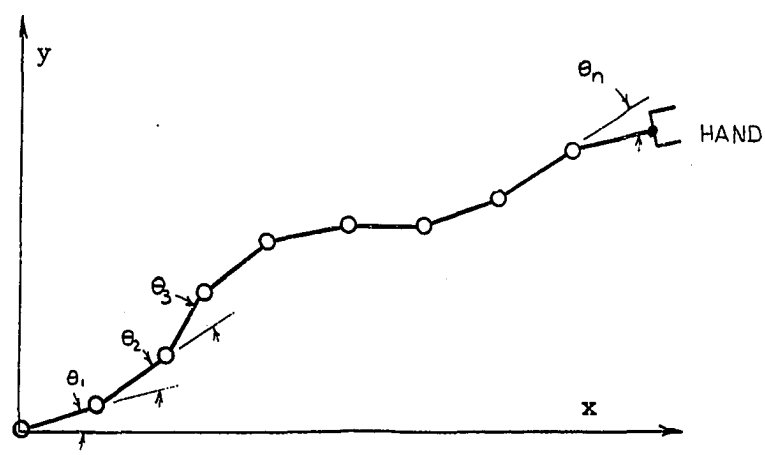


Figure 5.2. Binary Arm.

If we number the joints 1,....n, and denote the rotation in the i^{th} joint by θ_i, i=1,....n, our problem is to find the θ_i such that the end of the final link is "close" to the desired target. Since there are exactly 2^n possible configurations, there are at most 2^n points that the end of the arm (the hand) can

reach. Thus, in general, the hand cannot be placed at an arbitrary point. Hopefully, however, with sufficient links the hand could be placed close to any arbitrary point within its workspace.

There exists a well-defined transformation (see Appendix III) to find the position of the hand, given the θ_i . However, the inverse problem (i.e., given the hand position, find the associated θ_i) has yet to be solved. Theoretically, we could exhaustively examine the 2^n possibilities, construct a table and then choose the one that places the hand closest to the target, but in practice this would be too time consuming. We therefore need a systematic method to help in dealing with such a large solution space. If we define the error as the Euclidian distance of the hand from the target, the scheme outlined in Figure 5.3 suggests itself.

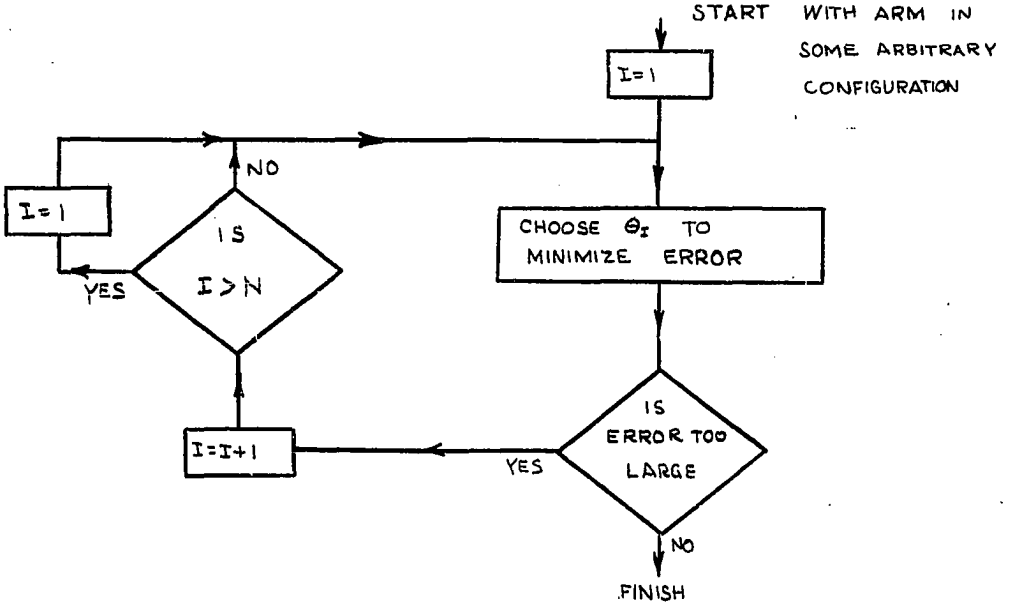


Figure 5.3. Sequential Search Procedure.

In this method the arm is put initially into some arbitrary configuration and the position of the hand computed. Starting at the

origin each joint is examined sequentially to see if the other state at that joint would reduce the error. If it does reduce the error, the change is made. That is, the rotation at that joint is considered to be reversed and the position of the hand computed while the rest of the arm remains rigid. The state of that joint is then changed if necessary to reduce the error. However, the existence of local minima prevent convergence of this method in many cases. It is possible to get improvement by using look-ahead. Instead of considering changing each joint singly, the results of changing that joint along with changes in the next k joints are considered. This may be called k-stage look-ahead, and would involve computing the hand position 2^{k+1} times for each joint. There are now many strategies possible using combinations of 0,1,2.... - stage look-ahead. There is, of course, a trade-off between the amount of look-ahead and computation time.

For instance, one strategy was to use no look-ahead until the error could not be reduced, then try 1-stage until no improvement resulted then 2-stage etc. The process was halted if the error was sufficiently small or the look-ahead became too large (usually 3-stage was as much as was allowed).

For purposes of trying these strategies, an arm with twenty-four 1 inch links, with possible rotations of $\pm 15^{o}$ in each joint was modeled in the computer. Tests were started with the arm extended along the x-axis as shown in Figure 5.4. The results are presented in Table 5.1. Computation times are shown in Table 5.2.

Table 5.1. Results of various strategies applied to Binary Arm.

TARGET (in.)	NO LOOK-AHEAD CONTINUED UNTIL NO IMPROVEMENT RESULTED				1-STAGE LOOK-AHEAD CONTINUED UNTIL NO IMPROVEMENT RESULTED				2-STAGE LOOK-AHEAD CONTINUED UNTIL NO IMPROVEMENT RESULTED				NO LOOK-AHEAD UNTIL NO IMPROVEMENT FOLLOWED BY 1-STAGE LOOK-AHEAD UNTIL NO IMPROVEMENT, THEN 2-STAGE, THEN 3-STAGE						
	POSITION REACHED		ERROR	NO. OF	POSITION REACHED		ERROR	NO. OF	POSITION REACHED		ERROR	NO. OF	POSITION REACHED		ERROR	NO. OF LOOPS WITH EA. LOOK			
(x,y)	x	y	in^2	LOOPS	x	y	in^2	LOOPS	x	y	in^2	LOOPS	x	y	in^2	0	1	2	3
0,0	-.241	.314	0.157	2	0.00	0.00	0.000	2	0.00	0.00	0.000	2	-.241	.314	0.157	2	*	*	*
10,0	11.08	1.07	2.325	3	11.08	1.07	2.325	3	10.53	0.00	0.278	5	10.53	0.00	0.218	3	7	*	*
5,5	9.12	5.26	17.021	3	3.83	4.75	1.446	3	3.83	4.75	1.446	3	9.12	5.62	17.021	3	*	*	*
10,5	10.14	5.40	0.176	4	10.14	5.40	0.176	6	10.40	5.06	0.259	4	10.14	5.40	0.176	4	*	*	*
5,10	4.69	12.14	4.667	3	5.16	9.82	0.057	4	6.20	10.58	1.783	3	5.44	10.51	1.151	3	2	*	*
10,10	12.10	8.80	5.846	4	10.12	9.20	0.061	5	10.40	10.13	0.172	4	10.13	9.06	0.899	4	3	*	*
20,0	23.46	-.500	12.202	2	20.89	-.018	0.785	5	19.94	-.107	0.015	4	20.17	.027	0.030	2	4	*	*

*No further improvement resulted.

Table 5.2. Computation times for one loop of sequential search (Binary Arm).

	NO LOOK-AHEAD	1-STAGE	2-STAGE	3-STAGE
TIME PER LOOP (sec.)	1.0	1.5	2.0	3.0

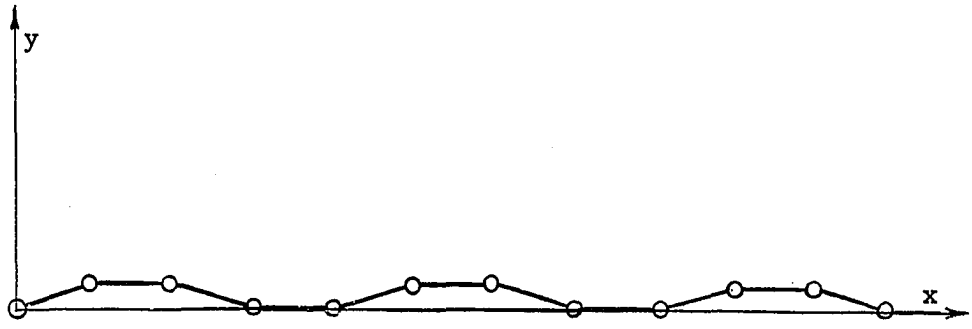


Figure 5.4. The arbitrarily chosen initial starting configuration for the arm.

It can be seen that no strategy tried was best in all cases. These methods had the additional disadvantage that the shape of the arm itself was not predictable. It was hoped that improvement would result if the searching was started after the hand was put at a point near the target by some simple procedure.

In order to place the hand near the target, a curve connecting the origin and the target was generated, whose arc length was equal to the length of the arm, and whose curvature did not exceed that which the arm could assume. Curves made up of segments of four circles having the above properties were used (details of the derivation of these circles are presented in Appendix III), and a rought attempt was made to match the arm to this curve. After the rough curve match, the previously described searching technique with 1-stage look-ahead was used. Various curve matching algorithms and different radius circles were used. Some of the results are shown in Figures 5.5 through 5.9.

Figure 5.5 shows the configuration resulting after four loops with no look-ahead, then two loops of 1-stage look-ahead. The procedure was started with the arm aligned along the x-axis as in Figure 5.4. Figure 5.6 is the result of first matching the arm to a curve composed

of segments of four circles, and then using two loops of 1-stage look-ahead.

Since the arm can tilt $\pm 15^{o}$ at each joint and the joints are a fixed distance apart the arm bends in a circle if all the tilts are in the same direction. The radius of this circle is the minimum that the arm can assume. In Figure 5.6, the radii of the circles used to generate the circle segment curve have this minimum radius. To match the arm to the curve, the state of joint i was chosen so that joint i+1 on the arm was as close as possible to point i+1 on the circle-segment curve. As can be seen, this procedure definitely influenced the shape of the final result. It may be noted from Figure 5.6, that in the attempt to match the arm to the circle-segment curve the arm lagged the curve. One attempt to remedy this, was to use larger radii circles to generate the curve. It can be seen from Figure 5.7 that this improved the match.

To reduce the lag even further, the state at joint i was chosen so that joint i+1 was as close as possible to point i+2 on the curve. This appeared successful as can be seen in Figure 5.8. An attempt to match the arm to the curve in both slope and position was made. Figure 5.9 shows the results of this scheme. The arm somewhat took on the shape of the curve, but not as much as in the other schemes.

The results presented were for the target (10,10). However, they are similar to those obtained for other targets. The best results were obtained when the radii of the circle segments were larger than the minimum. For radii of too great a magnitude, no curve existed that

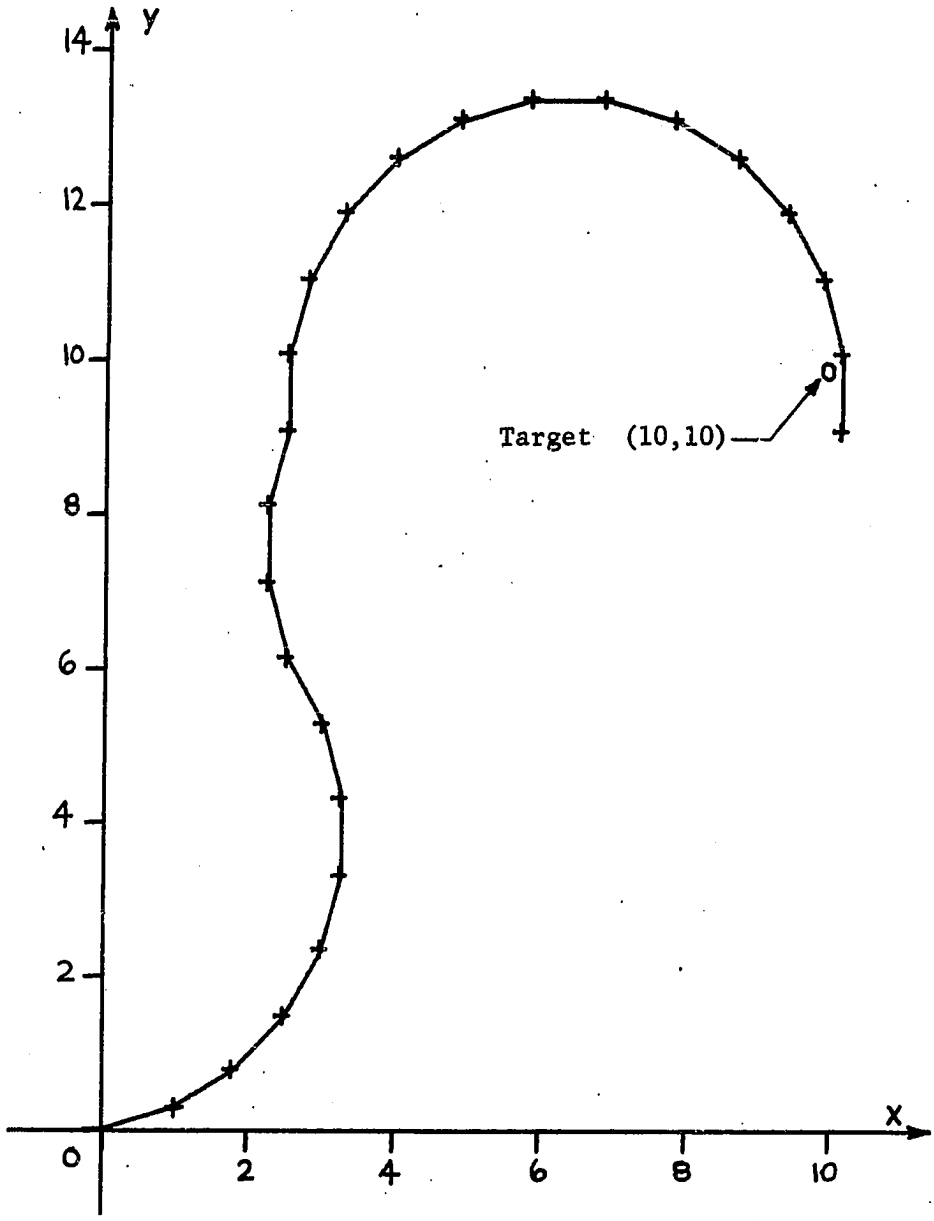


Figure 5.5. Result of arm after 4 loops of 0 look-ahead and two loops of 1-stage look-ahead. The starting configuration was along the x-axis, as in Fig. 5.4.

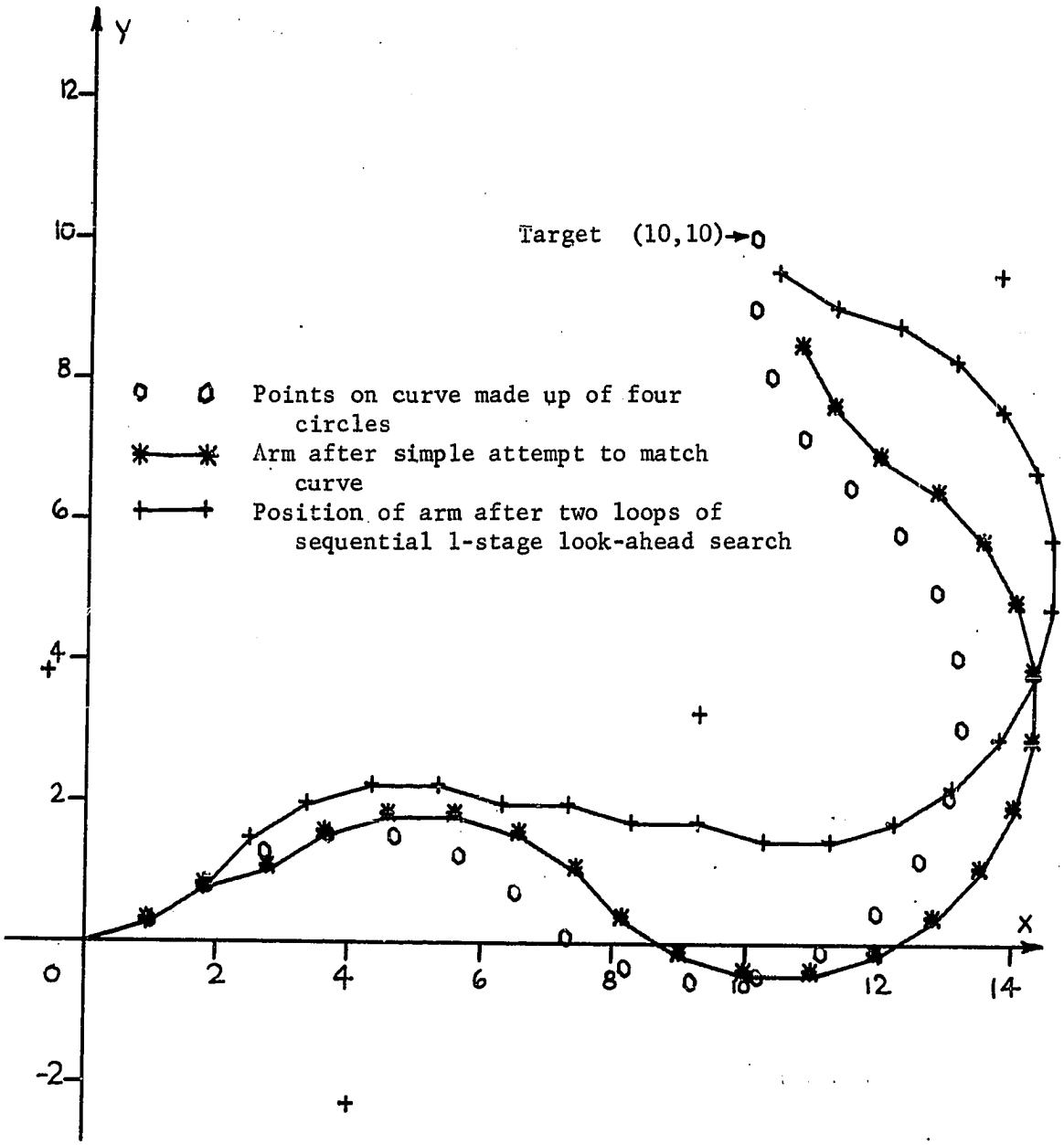


Figure 5.6. Result of trying to match arm to curve made up of segments of four circles, and the improvement after two loops of 1-stage look-ahead. Radii of circles is equal to minimum radius that the arm can assume. The curve matching technique was to choose Θ_i so that point i+1 on the arm was as close as possible to point i+1 on the curve.

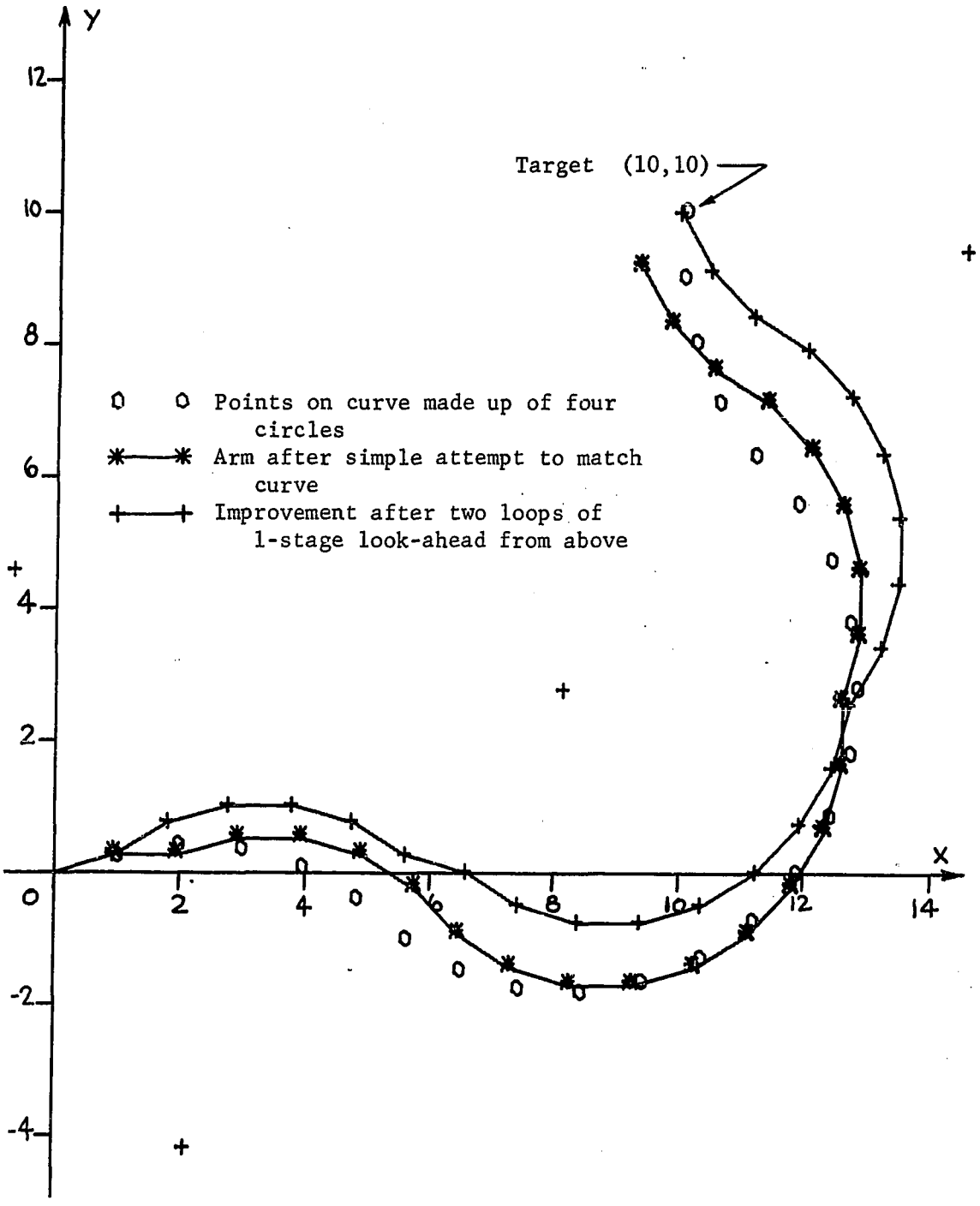


Figure 5.7. Similar strategy to that in Fig. 5.6. In this case the radii of the circles is 1.2 times the minimum radius that the arm could assume.

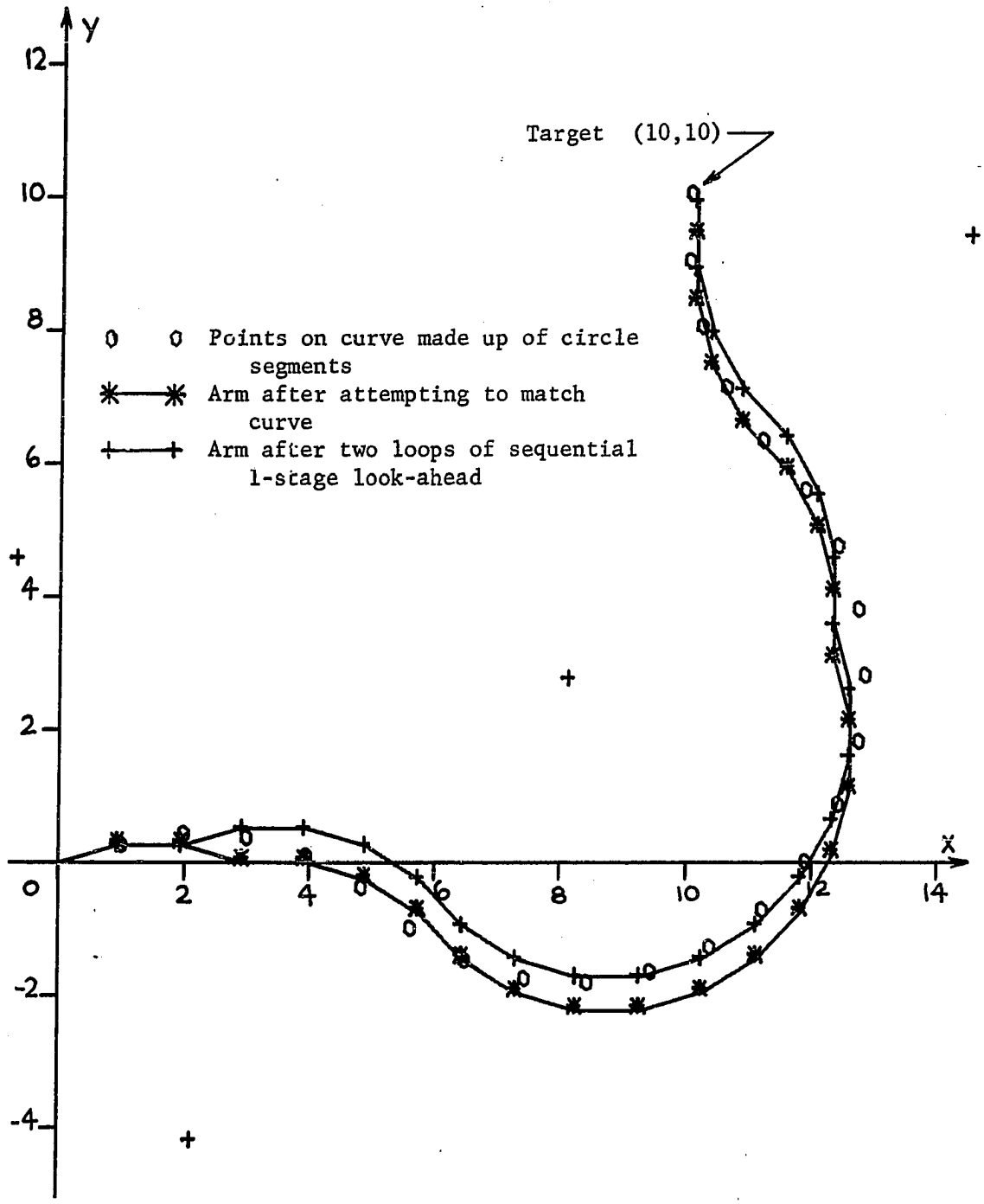


Figure 5.8. Result of trying to match arm to curve made up of segments of four circles, and improvement after two loops of 1-stage look-ahead. Radii of circles is 1.2 times minimum that arm can assume. The curve matching technique used was to sequentially choose θ_i so that point i+1 on the arm was as close as possible to point i+2 on the curve.

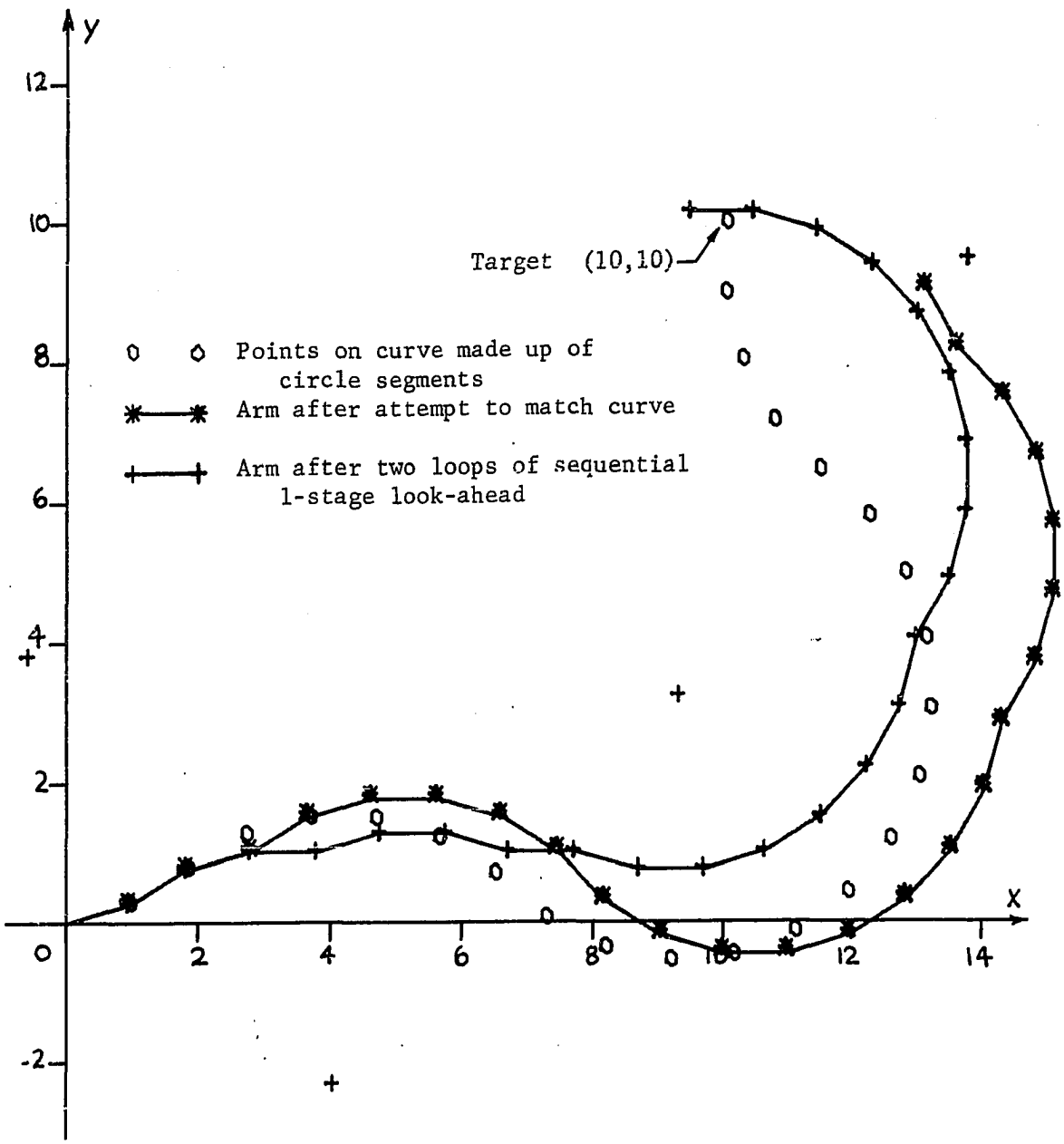


Figure 5.9. Result of trying to match arm to curve made up of segments of four circles, and improvement after two loops of 1-stage look-ahead. Radii of circles is equal to minimum radius arm can assume. The matching technique was to sequentially choose Θ_i so that $(x_{i+1}-\hat{x}_{i+1})^2+(y_{i+1}-\hat{y}_{i+1})^2+5\left[\left(\frac{y_{i+1}-y_i}{x_{i+1}-x_i}\right)-\left(\frac{\hat{y}_{i+1}-\hat{y}_i}{\hat{x}_{i+1}-\hat{x}_i}\right)\right]$ was a minimum, where (x_i, y_i) is the coordinate of joint i on the arm and $(\hat{x}_i, \hat{y}_i)$ is the coordinate of point i on the curve. This matching criteria puts a weight on the slope as well as the position of the links.

could connect the origin and the target point. A value of 1.2 times the minimum seems to be about optimum. It was found that this allows for a measure of control over the final shape of the arm as well as generally reducing the position error.

5.3 Three-Dimensional Model

This arm is similar to its 2-dimensional counterpart. The difference is that two axes of rotation exist at each joint. The axes intersect and are 90^{o} apart. (See Figure 5.10.) We assume our model to be constructed so that eight states are allowed at each

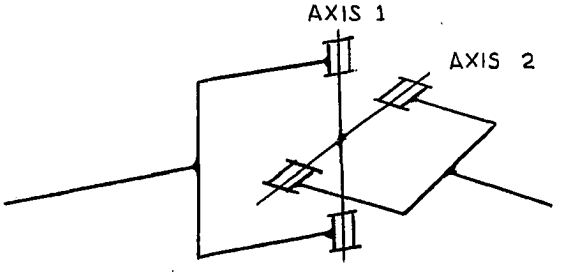


Figure 5.10. Typical joint in 3-dimensional digital arm.

joint. These are: either a rotation of $\pm\theta_o$ about axis 1 with none about axis 2, or a rotation of $\pm\theta_o$ about axis 2 with none about axis 1, or rotations about each such that the net result is a rotation $\pm\theta_o$ about axes midway between axes 1 and 2. Thus, two links can be tilted with respect to one another $\pm\theta_o$ about 4 different axes that are 45^{o} apart. If we denote the rotation about axis 1 as θ and the rotation about axis 2 as φ, then the possible states are:

State	θ	φ
1	$+\theta_o$	0
2	$-\theta_o$	0
3	0	$+\theta_o$
4	0	$-\theta_o$
5	$+\tan^{-1}(\frac{1}{\sqrt{2}}\tan\theta_o)$	$\sin^{-1}(\frac{1}{\sqrt{2}}\sin\theta_o)$
6	$+\tan^{-1}(\frac{1}{\sqrt{2}}\tan\theta_o)$	$-\sin^{-1}(\frac{1}{\sqrt{2}}\sin\theta_o)$
7	$-\tan^{-1}(\frac{1}{\sqrt{2}}\tan\theta_o)$	$-\sin^{-1}(\frac{1}{\sqrt{2}}\sin\theta_o)$
8	$-\tan^{-1}(\frac{1}{\sqrt{2}}\tan\theta_o)$	$+\sin^{-1}(\frac{1}{\sqrt{2}}\sin\theta_o)$

where the values of θ and φ in states 5-8 are the actual rotations about axes 1 and 2, leading to equivalent tilts about axes at 45° to 1 and 2 (see Appendix III). An n-link arm would then have 8^n possible configurations. Again there is a well-defined transformation to find the position of the hand given the angles (See Appendix III) but no such transformation exists to find the angles given the hand position.

The procedure presented in Figure 5.3 is still applicable except 8 states exist at each joint. Then each joint would be examined sequentially and the state at that joint is chosen which minimizes the Euclidian distance of the hand from the target. With k-stage look-ahead variations in the joint under consideration plus all possible combinations of the next k are considered. Then only the joint under consideration is moved. The position of the hand must thus be computed 8^{k+1} times for each joint and the time involved in these computations will limit the amount of possible look-ahead.

An arm with 24 1-inch links and possible tilt of $\pm 15^{o}$ between each link, was modeled in the computer. The arm was placed initially in the configuration of Figure 5.3 and strategies involving combinations of no look-ahead and 1-stage look-ahead were tried. Results are presented in Table 5.3. Computation time for one loop of sequential search is shown in Table 5.4. Details of the algorithms used are in Appendix III.

In general, the results are encouraging. It seems that this approach works better in three-dimension than in two as the errors are lower. The reason for the improved behavior can be attributed to the additional possible states at each joint. Computation time is longer in three dimensions, limiting look-ahead to one stage if real time problems are to be undertaken by the arm.

5.4 Discussion

Many variations of the aforementioned strategies are possible. For example, one may start sequential searching and making moves at the hand then work toward the origin. It is also possible to find the joint at which a change would reduce the error by the largest amount, make this change, and then continue the process of changing the joint that makes maximum reduction in error.

A reason for starting at the origin and working toward the hand is that in general a fixed rotation near the origin will cause the greatest deflection of the hand. Thus, in many cases making a change near the origin turns out to be the change that makes the maximum reduction in error.

Another approach is to take two joints at random and consider the result of simultaneous changes in each. The two-dimensional

Table 5.3. Results from various strategies applied to a 3-dimensional Digital Arm.

TARGET (in.)	NO LOOK-AHEAD. SEARCH CONTINUED UNTIL NOTHING CHANGED					1-STAGE LOOK-AHEAD. SEARCH CONTINUED UNTIL NOTHING CHANGED					1 LOOP OF NO LOOK-AHEAD FOLLOWED BY 1 LOOP OF 1-STAGE				1 LOOP OF 1-STAGE LOOK-AHEAD FOLLOWED BY 1 LOOP WITH NONE			
	POSITION REACHED			ERROR	NO. OF	POSITION REACHED			ERROR	NO. OF	POSITION REACHED			ERROR	POSITION REACHED			ERROR
X,Y,Z	X	Y	Z	in^2	LOOPS	X	Y	Z	in^2	LOOPS	X	Y	Z	in^2	X	Y	Z	in^2
2, 3, 18	1.63	3.42	18.25	0.374	3	2.00	3.06	17.92	0.110	2	2.00	3.06	17.92	0.11	2.50	3.77	18.30	0.925
10,10,0	9.84	10.08	-0.06	0.036	3	10.01	9.87	0.48	0.019	2	9.80	9.81	-0.02	0.73	9.91	9.21	-0.39	0.160
2,13,18	1.87	12.62	18.41	1.106	4	1.19	13.03	18.27	0.082	3	2.51	13.57	18.25	.651	1.90	13.15	18.84	0.731
2, 1, 20	1.77	1.72	20.30	0.667	3	1.97	0.81	20.57	0.180	3	1.89	1.29	20.40	.258	1.85	0.49	21.38	2.188

Table 5.4. Computation time for 1 loop of sequential search (3-dimensional arm).

	NO LOOK-AHEAD	1-STAGE	2-STAGE
TIME PER LOOP (sec.)	1.5	3.0	10.0

curve matching scheme could be extended to three-dimensions to give the arm a better starting configuration. It might be possible to break the spatial problems down into planar problems. Another idea is to divide space up into several regions, store a configuration that places the hand in each and then initially start the hand in the region closest to the target.

Many strategies are possible, none clearly better than others. Perhaps further study would show that certain ones work better in certain areas of space. It then might be possible for the computer to learn which was best for a particular region.

Different error criteria might be better. Cartesian coordinates, with the base of the arm aligned along the x-axis were used. This might mean that error in the x-direction should be weighted differently than error in the y- or z-direction. Perhaps the arm can better reduce angular error than radial error and this should be taken into account. Again learning might be applicable in selecting one error criteria for a given region or for optimizing weights placed on different quantities in an error function which is to be minimized.

In order that this arm be useful, the points in the reachable space must be close together. With a 24 link arm, there are 8^{24} (approximately 10^{21}) possible configurations. There will be fewer reachable points than configurations, but reducing 8^{24} by a factor of 10 or 100 or 1000 still leaves a large number of points. Near the boundary of reachable space, the points will be further apart, but in the interior, the density whould be very high. Assuming that a 24 link arm has a working volume of $5 \cdot 10^4$ cubic inches and 10^{18} reachable points exist, then

the average density is $2x10^{13}$ points per cubic inch, or if the points were equally spaced, they could be 0.00004 inches apart. This leads one to believe reachable points should be close enough to any arbitrary point in the space. The problem, of course, is to find the configuration that leads to a position near a desired point.

The results indicate that it is possible to find a solution for this digital manipulator. The solutions obtained are far from optimal but close enough to be useful. The dimensionality of the problem is staggering at times, but it is in fact the large number of solutions that give hope for any sub-optimal technique.

Further improvement is possible. By streamlining the subroutines used for basic computations, computer time could be reduced. The incorporation of different strategies for different zones would be useful.

Although the problem of finding a set of angles to place the hand at a given target appears soluble, the arm itself has serious limitations. The primary drawback is the inability to control its motion. Since there are discrete states at each joint, a wild motion is likely as each change is made. That is, the position is undefined when motion occurs. In addition, positions close in space may be very different in arm configuration. In conclusion, the arm is interesting but in its present state has no immediate usefulness.

CHAPTER VI

TRAJECTORY GENERATION

6.1 Problem Statement

In remote manipulation a typical problem is to move from an initial configuration to some new position and then to grasp an object. In order to carry out this task, the position problem must be solved. This results in a set of values specifying how much to rotate each joint in order to move the manipulator from its current or initial configuration to the desired final state. However, in such a case no explicit information exists describing the intermediate states between the initial and final position. It should be noted that the initial and the final configurations may be physically far apart, and the space through which the manipulator must move to attain the final state will in general contain obstacles. It is therefore necessary to find a "path" along which the manipulator can move and not collide with any of the obstacles. This problem will be referred to as trajectory generation. We attempt to solve this by defining sets of intermediate values for the joint angles which lead the manipulator to the final state in a manner which avoids collisions.

A person performing manipulative tasks avoids obstacles very simply. His eyes observe a possible conflict and he knows intuitively to raise his elbow or change his direction slightly. He sees "the world" in which he is working. He knows immediately which objects he is likely to encounter and which he will not come near. For a computer controlled

manipulator the problem is not so simple. The problems of "world" modeling, conflict detection, and collision avoidance must all be faced in order to generate a trajectory between initial and final manipulator configurations.

As a first step in dealing with this very difficult problem, a set of routines have been developed that provide a mathematical description of the world. Other routines simulate proposed trajectories through the space and sequentially examine points along the trajectory for obstacle conflict. If conflict is detected these routines suitably modify the trajectory. Several basic strategies to get from the initial to the final position are programmed so that if one fails, another can be explored. A block diagram of this system is shown in Figure 6.1.

In the development of these routines, an attempt has been made to be as general as possible in order that the programs be applicable to any manipulator, performing a wide variety of tasks. In the next sections, we present a description of these routines.

6.2 World Model, Obstacle Description, and Conflict Detection

For this system a simple model of the "world" is used. The basic elements of the world are assumed to be: planes, spheres and cylinders. It is assumed that all objects of interest can be modeled with these elements.

The boundaries of the workspace, usually formed by table tops or walls, are modeled as infinite planes. These planes are represented by a unit vector, $\underline{b}$, and by scalar t. Vector $\underline{b}$ is normal to the plane and points inward toward the workspace. Scalar t, the distance of the plane from the origin, is measured in the $\underline{b}$ - direction.

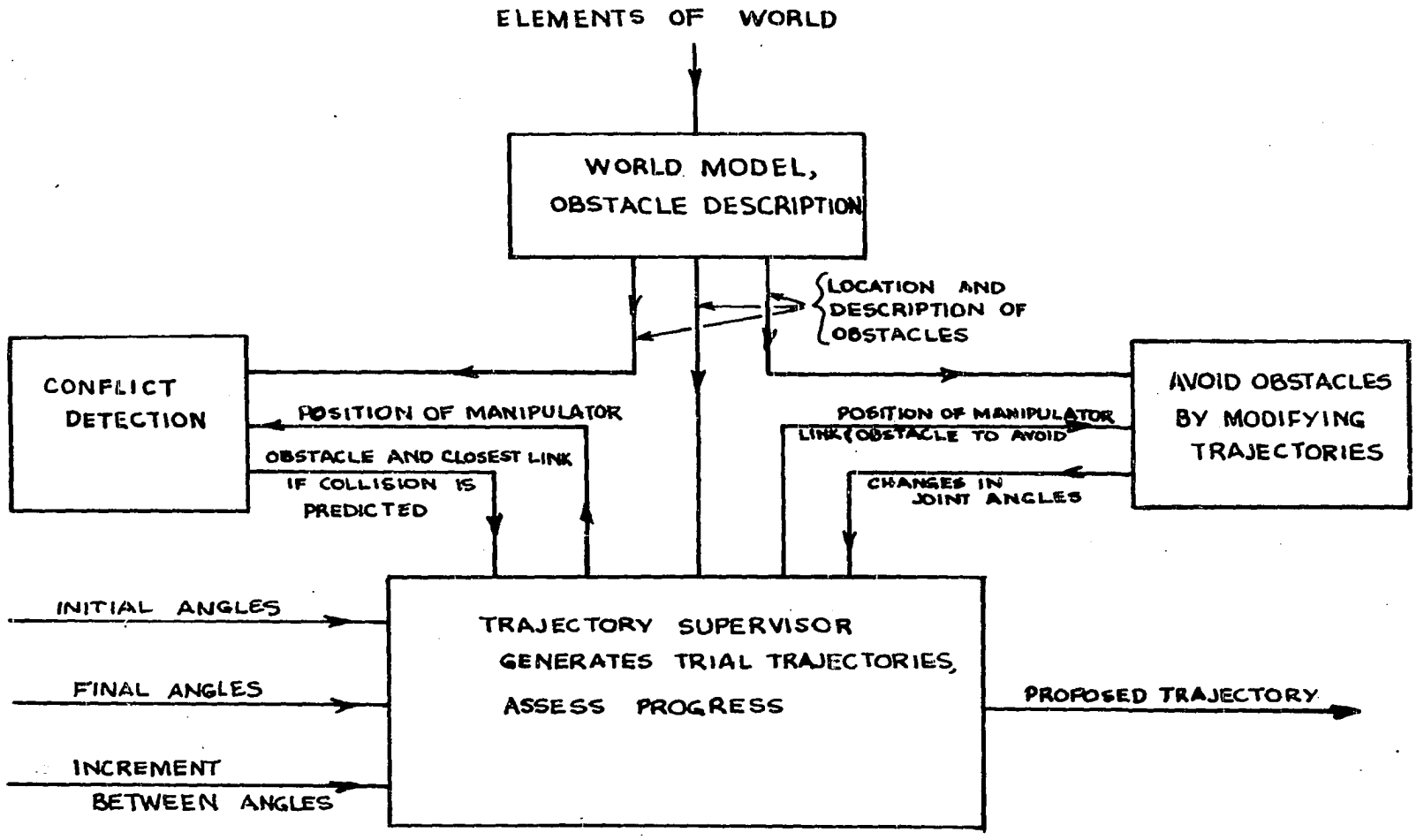


Figure 6.1. Block diagram of system to generate trajectories through a space containing obstacles.

Simple, somewhat regular objects which are not planar or cylindrical in shape are modeled as the smallest sphere that circumscribes the object. A typical object modeled in this manner might be a cube, a pencil sharpener or a coffee cup. The assumption is made that all such objects are supported by an infinite plane. Thus we represent a sphere by $\underline{a}$, a vector describing the location of its center, $\underline{b}$, a unit vector from the direction of support, and t, the radius of the sphere.

Cylinders are used to model objects containing a predominant axis such as a tower. In addition, cylinders are building blocks for more complex objects. For example, a manipulator is modeled as a group of cylinders each of which corresponds to one of the manipulator's structural members. The assumption is made that all cylinders are supported from an infinite plane or from another cylinder. We then represent a cylinder by a line segment corresponding to its axis and by the maximum distance of points in the object from this line, d. We have then:

b: a unit vector parallel to the axis pointing away from the direction of support

a: a vector describing the position of the base of the axis

t: the length of the axis

d: the radius of the cylinder

With this representation it is convenient to consider the cylinders to have a hemisphere capped on each end thereby assuring that all points on the surface have the same minimum distance, d from the line segment representing the axis.

Each obstacle, then, has a list of properties which include its type (plane, sphere, or cylinder), the composite object to which it belongs, and the aforementioned parameters which are required for its quantitative representation. The interpretation of a, b, t and d is varied according to the type of the object. Such models of all objects considered to be obstacles to the manipulator are stored in the computer.

The process of conflict detection consists of determining if the manipulator and the objects in its workspace will be brought to the same place at the same time. This is accomplished by computing the distance between the elements of the manipulator and the elements of the workspace at various positions along the proposed trajectory. A conflict is then predicted if this distance becomes too small.

It is undesirable to compute the distance between the manipulator and all of the objects in the workspace. To consider all the objects at each position along the trajectory would be time consuming. In addition, much of this computation would be wasted as for an arbitrary position, the manipulator would be so far from a large number of objects that a collision with these would be very unlikely. We would thus like to consider only objects near the manipulator. For this reason we divide the reachable space of the manipulator into small regions. In all the work to date, sixty-four subdivisions have been used. The workspace is considered to be a rectangular parallelapiped with edges parallel to the axes of a fixed Cartesian coordinate system. The small regions are defined as the volumes between three sets of equally spaced planes parallel to each of three mutually orthogonal faces of the workspace. Then a list of objects completely or partially inside each region is associated with

that region. For conflict detection, only the objects occupying the same region or regions as the manipulator are considered.

As a result of dividing the space into regions, we have the problem of finding in which region(s) various obstacles are located. In addition we will have to identify the region(s) the manipulator occupies at various positions along its trajectory. We wish to keep this analysis simple in order that the time saved in not having to deal with all obstacles in the workspace is not lost in trying to locate the manipulator in various regions. Since the faces of our subdivisions are made perpendicular to the coordinate axes, we can easily eliminate many regions by comparing the minimum and maximum coordinates (x, y, z) of an obstacle, with the coordinate boundaries of the regions. To find in which of the remaining regions an obstacle lies we compute the distance from the center of each of the regions to the obstacle (a fairly simple process in view of the simple world model). We then compare this distance with the radius of a sphere totally enclosing the region to determine if the object is in the sphere. If the object is in the sphere, we assume it to be in the region. This procedure may cause an object to be considered inside a region when in reality it is outside. However, this process is considerably simpler than trying to find whether an object cuts any part of the actual region.

Routines were developed which divide space into regions and appropriately enter or remove objects from lists associated with the regions. These routines also store the properties of each obstacle.

The conflict detection routine starts with the first link of the manipulator and finds which regions this link is in. It then finds the

distances between all the objects in these regions and the link. A collision is predicted if the distance between any object and the link is small enough so that, with the link continuing along its present course, a conflict would occur. If a collision is predicted, a flag is set and the routine specifies the obstacle and the link closest to the obstacle. If no collision is detected the procedure is carried out for the remaining links in the manipulator. A block diagram of this program is presented in Figure 6.2.

The method for determining distance between a manipulator link and an object depends upon the type of object. For spherical objects, the distance between the sphere center and the cylinder-axis of the link is computed. The actual distance is then found by decreasing this by the sum of the radii of the sphere and the cylinder (representing the link). For planes, the distance between the plane and the cylinder-axis of the link is computed. This distance is decreased by the radius of the cylinder to form the actual distance. For objects modeled as cylinders, we find the distance between the object axis and link axis, and decrease this by the combined radii of the cylinders. Details of these calculations are found in Appendix IV.

6.3 Trajectory Generation and Obstacle Avoidance

We have the problem of finding a series of closely spaced intermediate positions connecting initial and final states. These represent a trajectory that the manipulator can follow while avoiding all obstacles in the workspace. The approach used is to start by choosing a plausible trajectory, simulate the motion along the trajectory and then if conflict

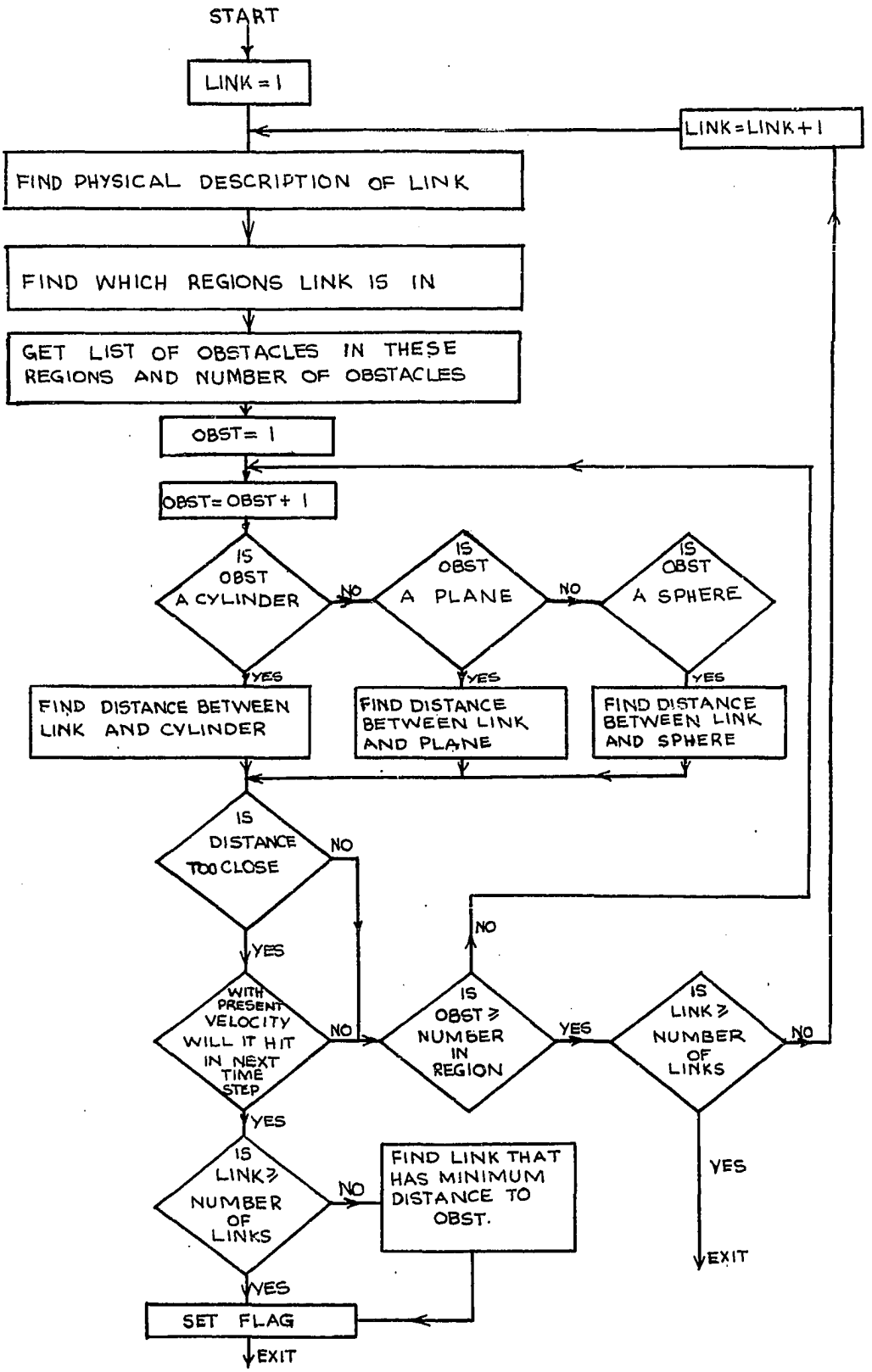


Figure 6.2. Block diagram for conflict detection routine.

occurs, to modify the trajectory. This modification is made on the basis of local geometric conditions in the area of conflict. (A program, called AVOID, accomplishing this will be discussed in more detail later in this section.)

Often if more than one obstacle is present, a move that appears good to avoid one obstacle is bad to avoid another. This may lead the manipulator to oscillate between objects. It is also possible for some joints to be at their physical limits so that the avoidance routine does not find a good move. Finally, the avoidance routine itself may come up with a non-productive move. It is therefore necessary to continually ascertain whether or not progress is being made toward the goal. If no progress is being made, it is then necessary to decide whether a slight change in strategy is sufficient or whether a whole new strategy is in order.

A program based on the above approach called TRLTRJ, has been written. The inputs to the program are two sets of joint angles, one set specifying the initial position and the other specifying the final position. In addition the desired increment between intermediate positions is specified. The output from the program is an array of angles specifying the intermediate positions.

Four basic strategies are built into the program. The first, and least complex, just increments each angle towards the final goal. The second strategy computes two intermediate positions to move the manipulator up and then over a concentration of obstacles. The third and fourth strategies both try to fold the manipulator to shorten it and then move it in front of any obstacles. These last two differ in that one

shortens the manipulator by moving joints in one direction toward the physical stops while the other folds the manipulator by moving the joints toward the stops in the other direction (in the program, we call these directions positive and negative, respectively).

The program starts by trying the first strategy. If any obstacle is encountered, this strategy is abandoned for the time being and the second strategy started. If while pursuing this second basic strategy, a conflict is predicted, an attempt is made to modify the trajectory using the program AVOID. If this strategy fails after using AVOID the program continues and tries the third and finally, if necessary, the four strategy in a similar manner. If the fourth strategy fails, the program returns to the first strategy and tries it using AVOID. If it does not produce a trajectory, it is assumed that all obstacle avoidance strategies have failed and the program halts.

If any of the following occur, the program considers that no progress is being made and hence a strategy has failed:

1. The avoidance routine (AVOID) is not able to generate a move due to joints being at their physical stops.
2. A collision is predicted with the manipulator at the same point on the trajectory where a conflict had previously been predicted with that same obstacle (hence it is assumed the program is in a loop).
3. A conflict is predicted with a plane for the second time, while trying to avoid the same obstacle. (Assumedly we cannot get around the obstacle.)

4. The manipulator oscillates between two obstacles, and no net progress toward a goal is being made.

5. More than 200 intermediate sets of angles have been selected without the manipulator reaching a goal.

6. More than 350 intermediate sets of angles have been explored.

The following conditions cause a slight change in strategy but do not cause the strategy to be abandoned.

1. A plane of infinite extent is encountered while trying to avoid an obstacle. At this point we assume that the manipulator is moving in the wrong direction to go around this obstacle. The strategy is to go back to the first point we encountered this obstacle and try to go around it by going in the opposite direction. (This notion of direction will become more clear with the description of AVOID.)

2. An oscillation of the manipulator between two obstacles is detected. The action that the program takes is to go back to the point where the second of the obstacles was encountered and try to go around it in the opposite direction. If this happens twice at the same position on the trajectory, it is assumed that no progress is being made and the strategy has failed.

A block diagram of TRLTRJ is presented in Figure 6.3.

The subroutine AVOID is used to generate small perterbations in a trajectory when conflict is predicted. The program attempts to define

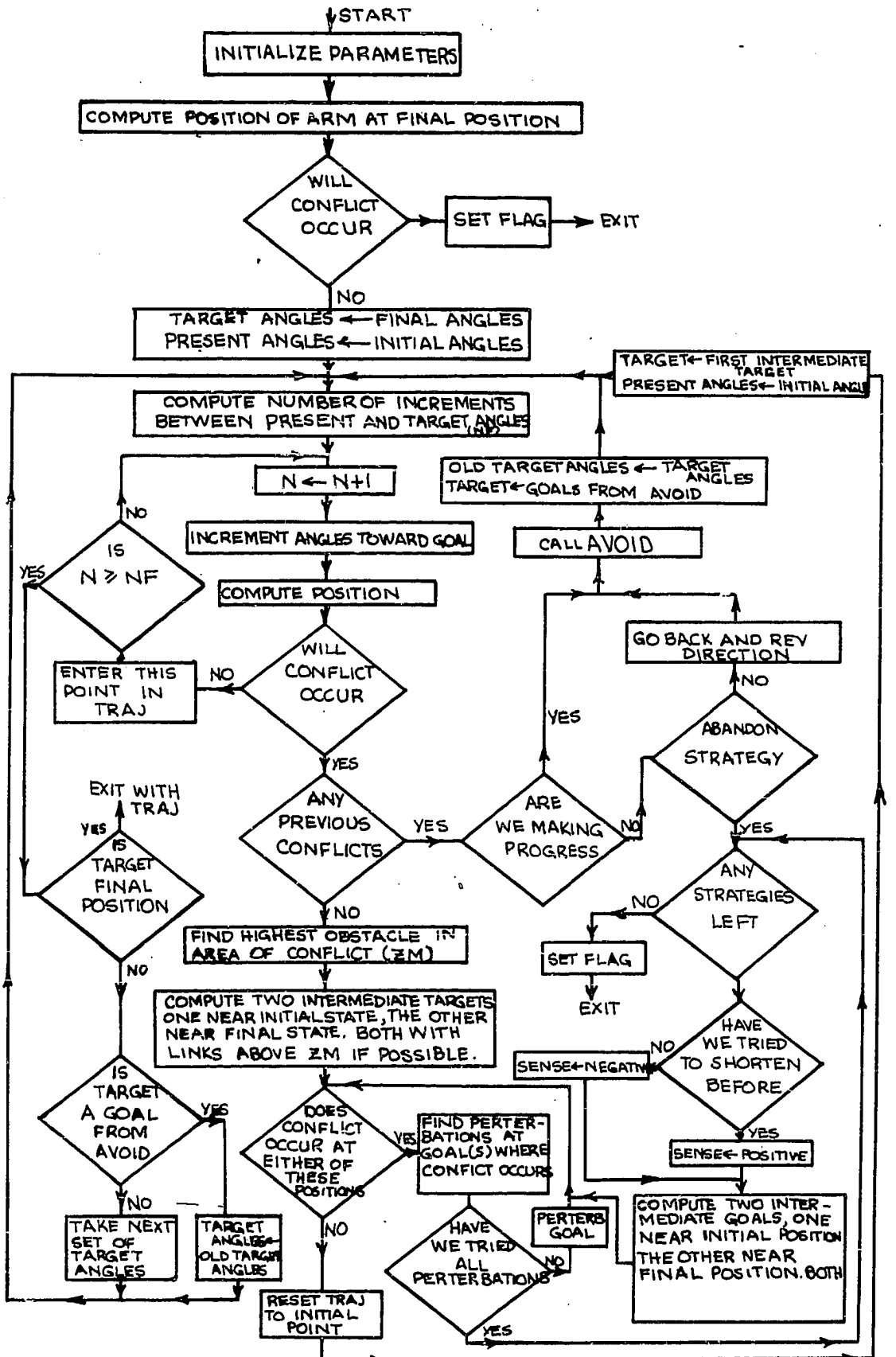


Figure 6.3. Block diagram of TRLTRJ.

a "good" direction and a "bad" direction. It then tries to move the link for which a collision has been predicted as much as possible in the good direction while not moving in the bad direction. This is accomplished by defining small changes in the joint angles of all the links between the base and the "colliding" link. Ideally these angle changes are chosen so that the link will have a large velocity component in the good direction and zero component in the bad direction. If the link does not have enough freedom (i.e., there are too few joints preceding the link or the joints are at their physical stops) to make a move in this manner, an attempt is made to move in the negative bad-direction. If this too is not possible then no move is made and a flag is set indicating that the strategy has failed.

The underlying idea used in choosing a good direction is that all obstacles are supported by either an infinite plane or another obstacle. Then if an obstacle lies between the manipulator and the target, one could eventually get around the obstacle by moving away from the direction of support. In addition an attempt is made to move in the general direction of the target. This target will normally be the final position but may be an intermediate goal generated in a strategy of TRLTRJ. If the predicted conflict has occurred in the process of avoiding a different obstacle, the target becomes the position generated by AVOID when the manipulator encountered the first obstacle. The good direction is chosen taking into account the type of obstacle and the relation between link, obstacle and target as follows:

If the obstacle is a plane or a sphere, the good direction is specified by a vector from the point of conflict on the link to the same

point on the link with the manipulator at the target position. (The assumption here is that the sphere is relatively small and lies on the plane of support.)

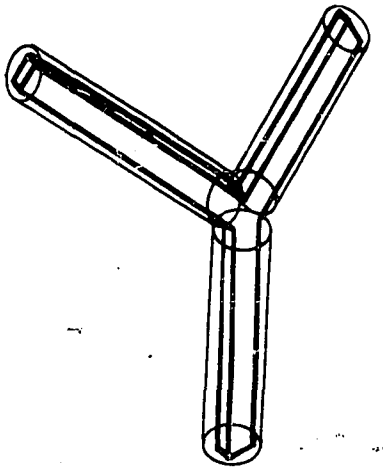
If the obstacle is a cylinder, then the process is a bit more complicated. Recall that a cylinder may be part of a more complex obstacle (for example the towers in Figure 6.4). If no other part of the possibly complex obstacle, of which the cylindrical obstacle is a member has been recently encountered, then the good direction is:

1. The direction of the axis of cylinder, if the obstacle appears* to be between the manipulator and the target.
2. The vector sum of unit vectors in the axis direction and the direction the link must move to get to the target, if the link is above the cylinder.
3. The direction the link must move to get to the target, if obstacle is not between the manipulator and the target.

When the cylindrical obstacle is part of a more complex obstacle and when an element of this complex obstacle has been previously encountered, then the good direction is similar to the above with the following exception: the positive axis direction is replaced by the negative axis direction whenever the point of conflict on the obstacle is nearer the far end of the obstacle (i.e., away from the point of

*We say "appears" because a cylindrical object may not itself be between the manipulator and the goal, but the complex obstacle which it belongs to, may indeed be between the manipulator and the goal.

(a) simple tower



(b) Y-shaped tower

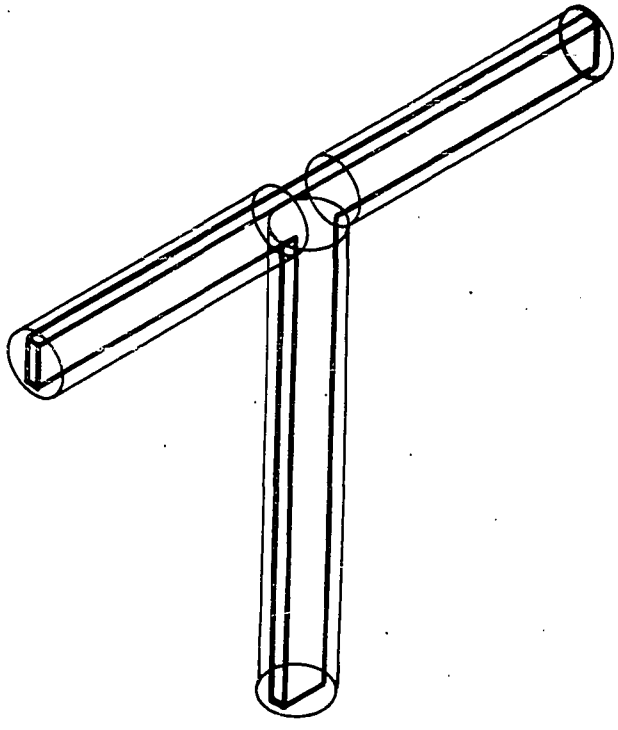


Figure 6.4. Towers used as obstacles. (a) model is a single cylinder. (b) and (c) are each modeled with three cylinders.

support). In this way we are able to follow the contour of a complex obstacle.

Once a good direction has been established for an obstacle it remains the same until:

1. The manipulator is clear of the obstacles, or
2. An oscillation has been detected by TRLTRJ, in which case wherever the positive axis direction was to be used, it is to be replaced by the negative axis direction and vice-versa.

The bad direction is always specified by a vector from the link to the obstacle, along the line defining the minimum distance between them.

A block diagram of AVOID is presented in Figure 6.5.

6.4 A Test of the Program

The trajectory generating routines were tested by incorporating them into the block stacking program developed at the Stanford Artificial Intelligence Project [25]. The block stacking program represents current research work in hand-eye systems. An electric motor driven manipulator of type 6R, $s_3\ a_3\ s_5\ a_5$ (see Figure 6.6) and a vidicon T.V., interfaced with the PDP-6 computer form the basic system. Programs written by Singer and Pingle [25] enable the manipulator to pick up blocks from a table and build block towers. The blocks are originally placed at random on the table but within view of the vidicon. A block is then located on the table by appropriate analysis of the T.V. picture. Next, the manipulator moves to grasp the block and then places it to build a tower. A new block is found and the process continues.

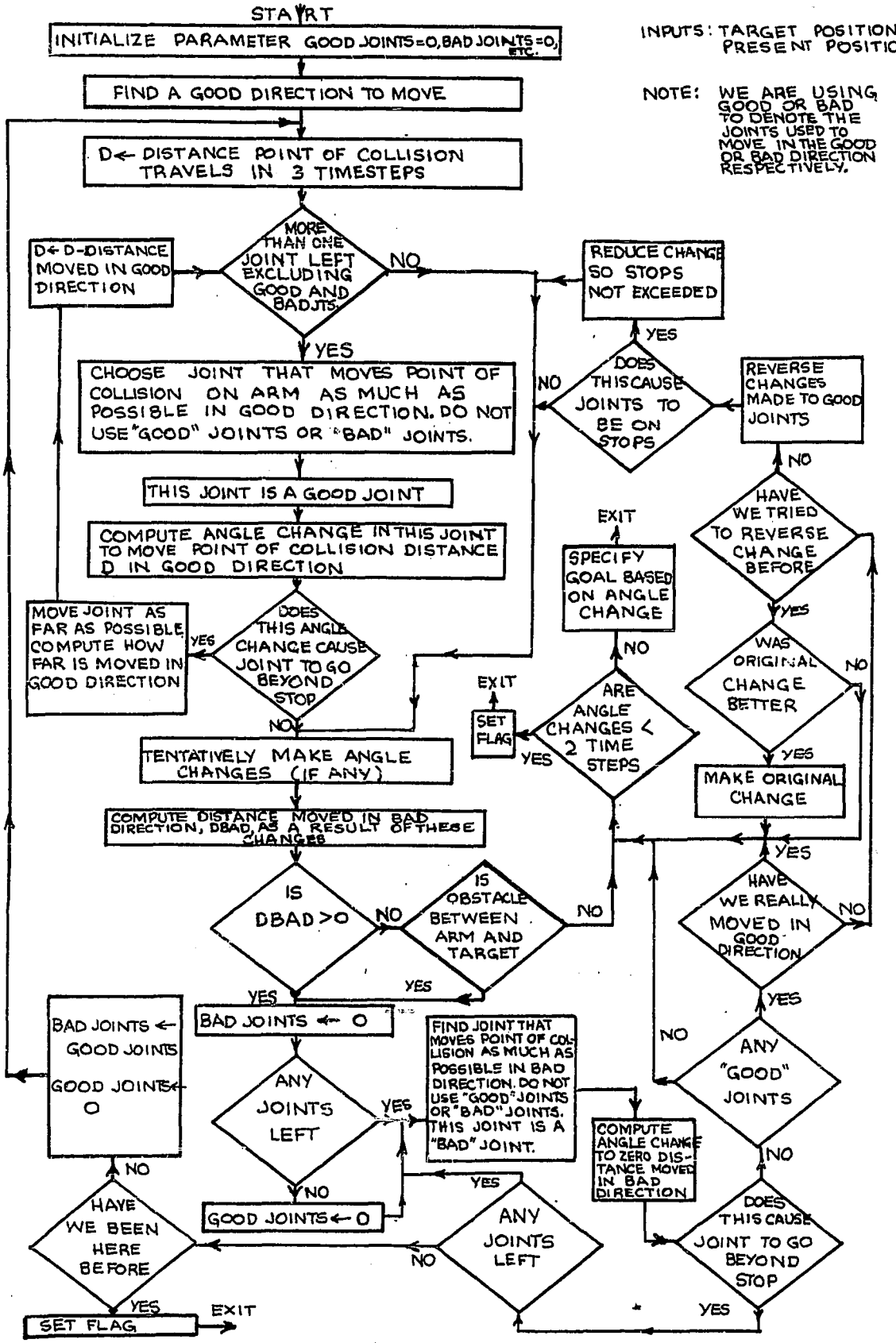
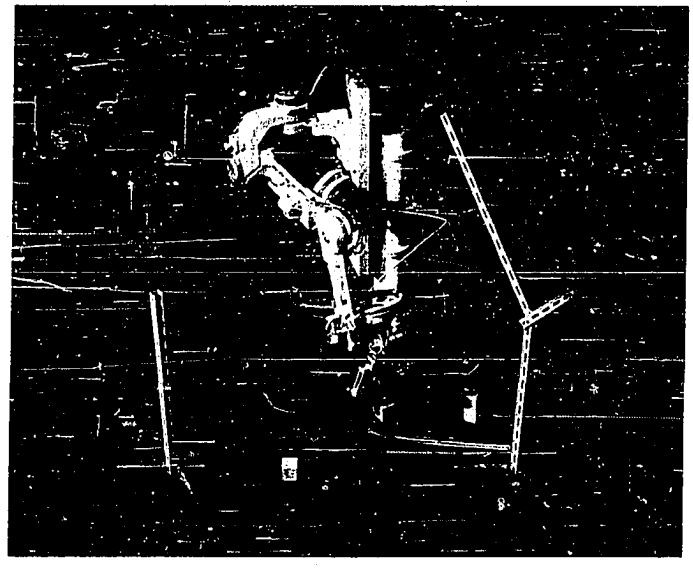


Figure 6.5. Block diagram of AVOID.



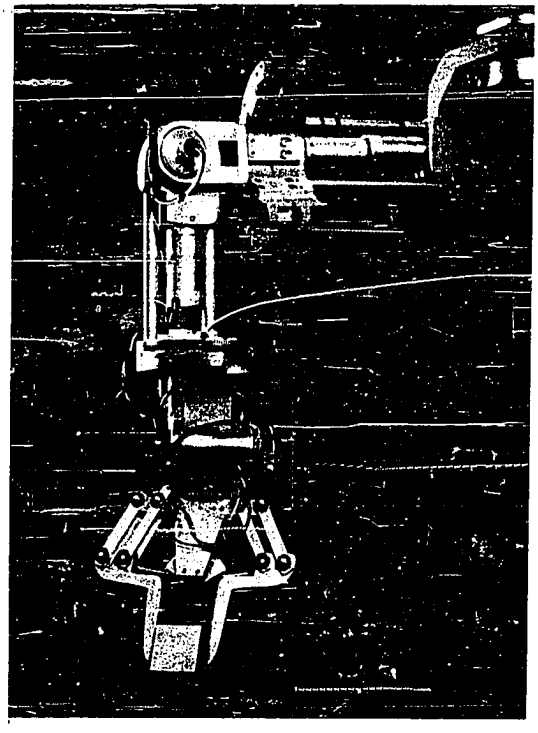


Figure 6.6. Electric Arm at Stanford Artificial Intelligence Project. A prothestic arm originally built at Rancho Los Amigos Hospital, this arm has been modified for use in hand-eye research.

The trajectory generating routine is used to find trajectories whenever the arm is moved except in the last stages of actually picking up or setting down a block. At these times, the manipulator control is transferred to a special routine whose function is to lower or raise the hand along a specified path with a specified orientation.

Objects considered to be obstacles are the table top, the support structure for the arm, and any block towers that have been built. In addition, to make life difficult for the program, several other obstacles (see Figure 6.4) were added. Since the range of vision of the T.V. camera is small, and its recognition powers to date is limited to cubes, a sub-program was written so that the external obstacles could be added to the data structure by commands from the teletype.

After allowing the program to run, with the different obstacles in varying locations, the trajectory generation program was seen to perform fairly well (see Figure 6.7). Where possible, it was generally able to go over or in front of the obstacles. However, the procedure occasionally failed when the manipulator was started in a configuration in which joints were near their physical stops. In these cases a successful maneuver might have been to move those joints well away from the stops and try again (a procedure not built into the program). In addition, if the objects were so placed that the arm could only get through by going between two objects, failure generally occurred.

Whenever more than one or two strategies were tried, the computation might run upwards of 20 seconds. However, most manipulative scenes are fairly static, so that once a trajectory had been found through a given set of obstacles, it could be used repeatedly. This process would save

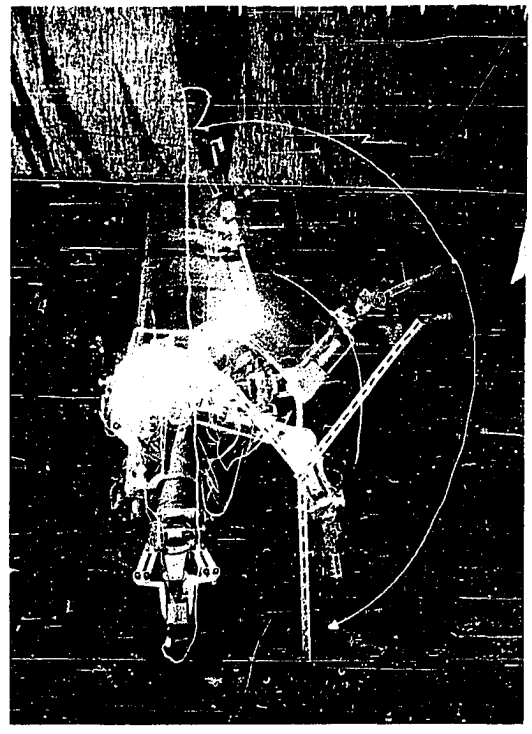


Figure 6.7. Example of trajectory enabling manipulator to go over obstacles.

having to re-analyze the trajectory for every move, thus conserving computer time. In addition we could move backward on the same trajectory to get back through the space again.

With this program we have attacked the problem of moving a multi-link manipulator through a space composed of three-dimensional objects. Had we been concerned with having just the hand avoid obstacles on a plane, the problem would have been much less complex, as the hand could be made to follow an arbitrary curve. Such is not the case, when considering a conflict with all links of a manipulator. We cannot independently specify the position of each link of a six degree-of-freedom manipulator. There are just not enough freedoms. However, the programs developed above do enable us to deal with the problem of conflict for a general multi-link manipulator. These programs perform the basic function of allowing a manipulator to perform tasks in the presence of obstacles.

CHAPTER VII

CRITERIA IN THE DESIGN OF A MANIPULATOR FOR COMPUTER CONTROL

7.1 Kinematic Criteria

As mentioned earlier, a manipulator needs to have six degrees-of-freedom to grasp a rigid body with a specified orientation at a specified position in space. In addition, the kinematic solution must be easily programmed and solved. This indicates the desirability of a closed-form solution rather than iterative techniques. The closed-form solutions are faster and find all configurations leading to the desired terminal position and orientation while iterative techniques find only one.* In fact, the iterative schemes may not find a solution even though several may exist. The question of the existence of a solution is important, as this existence indicates whether a given position and orientation is physically attainable. It is desirable to have solutions exist throughout the workspace or at least know where they do not exist. Thus a factor in the design of a manipulator is the zones in which the terminal device can be placed in an arbitrary manner.

The problem of zones is closely allied to that of solutions. The existence of a solution for a given position and orientation automatically guarantees that that point is within the zone of reachable points. One method of investigating zones would be to solve the position problem for

*The iterative technique may however be used to good advantage when the distance between positions is very small. Then the iterations converge quickly, and there is only one solution being sought.

many points and many manipulator configurations. This however is very lengthy and not at all general. Alternatively, we attempt to give a few general remarks about zones.

When on the boundary of the zone of reachable space, the hand cannot be moved in an arbitrary direction or rotated about an arbitrary axis. Another way of saying this is that the hand cannot move along an arbitrary screw. Mathematically this happens whenever the determinant formed from the left-hand sides of equations (4.13) and (4.14) vanishes. The existence of a solution would enable us to express the $\underline{W}_i$ and $\underline{r}_i$, appearing in (4.13) and (4.14), in terms of the hand position and orientation. Then forming the determinant we would have a polynomial in terms of the hand position and orientation whose vanishing would correspond to the boundary of reachable space. We would then have a surface in six-space which bounds reachable space.

As this representation is highly non-linear, as well as dependent upon the existence of a solution, it is often more fruitful to examine the problem from a geometrical viewpoint. For example, consider the $6R,s_3s_5$ manipulator with all adjacent pairs of axes perpendicular, as is shown in Figure 7.1. We note that the wrist point, W, defined by the vector $\underline{P}$, can lie anywhere within a sphere of radius r about the shoulder point, 0, where:

$$(s_3-s_5)^2 \leq r^2 \leq (s_3+s_5)^2 \qquad (7.1)$$

Furthermore, if the wrist position is fixed, then the direction the hand points, defined by $\underline{a}_6$ in Figure 7.1, is arbitrary. Through appropriate rotations in joints 5 and 6, $\underline{a}_6$ can be made to point in any direction. However, the total orientation of the hand cannot be

arbitrarily specified for a fixed wrist point as the direction defined by ω_6 in Figure 7.1 is limited in range. We note that ω_6 must always be perpendicular to ω_5. Hence, ω_6 may lie anywhere in a plane perpendicular to ω_5. Now the specification of the wrist point does not fix the elbow point, A, and in fact the triangle OAW may be rotated about $\underline{P}$. We observe, then, that $\underline{\omega}_5$ must lie on a cone whose axis is $\underline{P}$, with apex at W and whose cone angle is fixed by triangle OAW. Then $\underline{\omega}_6$ will lie in planes through W, perpendicular to the elements of this cone. This defines a second cone, inside which $\underline{\omega}_6$ can never point. These cones are shown in Figure 7.2. Referring to Figure 7.2, the elements of cone 1 form the locus of $\underline{\omega}_5$ while $\underline{\omega}_6$ will always lie outside cone 2.

If it is desirable for the hand to have a full range of orientation freedom, then a manipulator whose last three joints are revolute and whose axes intersect is appropriate. Consider such a configuration, shown in Figure 3.8. Here the last three axes intersect and provide maximum orientation freedom for the hand. In addition this configuration has a wide range of positions that the wrist point, defined by $\underline{P}_3$, can assume. Referring to Figure 3.8 we note that the wrist point can be placed anywhere inside a circle normal to axis 2, about P_1, whose radius r obeys the constraint:

$$(a_2-s_4)^2 \leq r^2 \leq (a_2+s_4)^2 \qquad (7.2)$$

Rotation of the first joint then rotates this circular anulus to generate a torus which is the locus of points the wrist can reach.

It is possible to examine many manipulator configurations in this manner. Table 7.1 presents the results of such examination of 6R manipulators with two and three non-zero link parameters. Whether or not a solution exists is also included in Table 7.1.

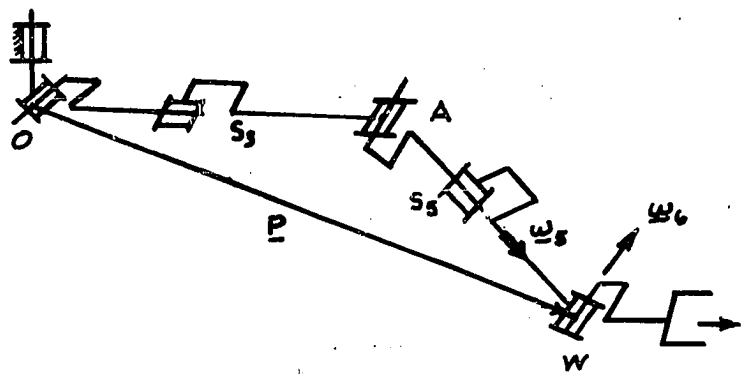


Figure 7.1. A $6R,s_3s_5$ manipulator

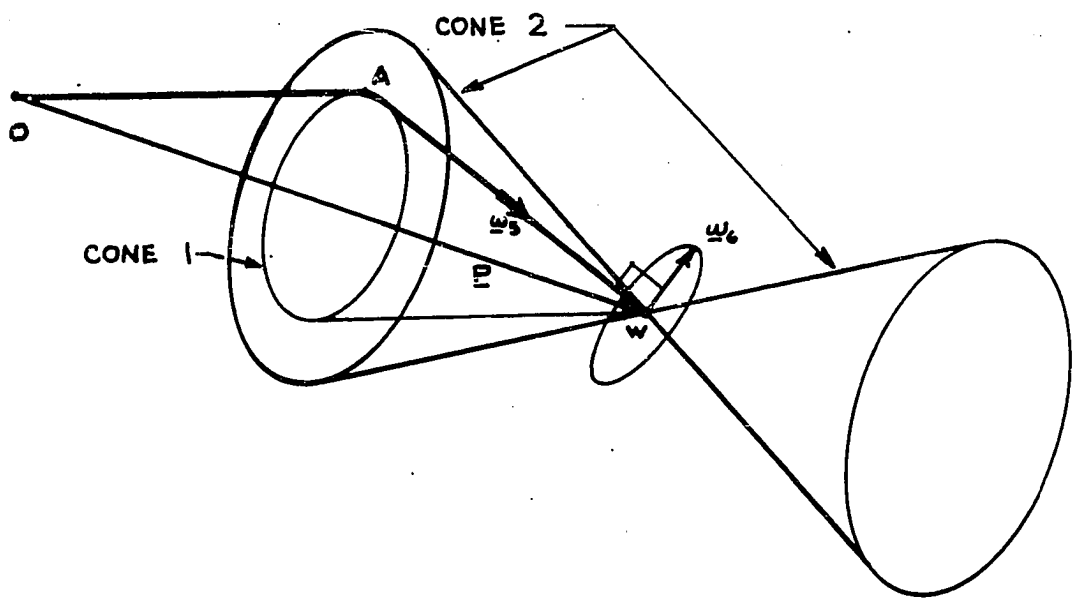


Figure 7.2. Cones showing possible loci for $\underline{\omega}_5$ and $\underline{\omega}_6$

TABLE 7.1

Solubility and Orientation Restrictions in 6R Manipulators

MANIPULATOR	REMARK	MANIPULATOR	REMARK	MANIPULATOR	REMARK
1 a_1s_2	D	16. a_2s_3	D	31. s_4a_4	D
2. a_1a_2	D	17. a_2a_3	S G	32. s_4s_5	D
3. a_1s_3	D	18. a_2s_4	S G	33. s_4a_5	D
4. a_1a_3	S G	19. a_2a_4	N R	34. a_4s_5	D
5. a_1s_4	S G	20. a_2s_5	S R	35. a_4a_5	D
6. a_1a_4	S G	21. a_2a_5	S R	36. s_5a_5	D
7. a_1s_5	D	22. s_3a_3	D	37. $a_1s_2a_2$	D
8. a_1a_5	D	23. s_3s_4	S G	38. $a_1s_2s_3$	D
9. s_2a_2	D	24. s_3a_4	S G	39. $a_1s_2a_3$	S G
10. s_2s_3	D	25. s_3s_5	S R	40. $a_1s_2s_4$	S G
11. s_2a_3	S G	26. s_3a_5	S R	41. $a_1s_2a_4$	S G
12. s_2s_4	S G	27. a_3s_4	D	42. $a_1s_2s_5$	D
13. s_2a_4	S G	28. a_3a_4	S G	43. $a_1s_2a_5$	D
14. s_2s_5	D	29. a_3s_5	S R	44. $a_1a_2s_3$	D
15. s_2a_5	D	30. a_3a_5	S R	45. $a_1a_2a_3$	S G

Table 7.1 (continued)

MANIPULATOR	REMARK	MANIPULATOR	REMARK	MANIPULATOR	REMARK
46. $a_1a_2s_4$	S G	63. $a_1a_4a_5$	S R	80. $s_2s_4a_4$	S G
47. $a_1a_2a_4$	N R	64. $a_1s_5a_5$	D	81. $s_2s_4s_5$	S G
48. $a_1a_2s_5$	S R	65. $s_2a_2s_3$	S G	82. $s_2s_4a_5$	S R
49. $a_1a_2a_5$	S G	66. $s_2a_2a_3$	S G	83. $s_2a_4s_5$	S G
50. $a_1s_3a_3$	S G	67. $s_2a_2s_4$	S G	84. $s_2a_4a_5$	S R
51. $a_1s_3s_4$	S G	68. $s_2a_2a_4$	N R	85. $s_2s_5a_5$	D
52. $a_1s_3a_4$	N G	69. $s_2a_2s_5$	S R	86. $a_2s_3a_3$	S G
53. $a_1s_3s_5$	S R	70. $s_2a_2a_5$	S R	87. $a_2s_3s_4$	S G
54. $a_1s_3a_5$	S R	71. $s_2s_3a_3$	S G	88. $a_2s_3a_4$	N G
55. $a_1a_3s_4$	S G	72. $s_2s_3s_4$	S G	89. $a_2s_3s_5$	S R
56. $a_1a_3a_4$	N R	73. $s_2s_3a_4$	N R	90. $a_2s_3a_5$	S R
57. $a_1a_3s_5$	N R	74. $s_2s_3s_5$	S R	91. $a_2a_3s_4$	S G
58. $a_1a_3a_5$	N R	75. $s_2s_3a_5$	S R	92. $a_2a_3a_4$	N R
59. $a_1s_4a_4$	S G	76. $s_2a_3s_4$	S G	93. $a_2a_3s_5$	N R
60. $a_1s_4s_5$	S R	77. $s_2a_3a_4$	N R	94. $a_2a_3a_5$	N R
61. $a_1s_4a_5$	S R	78. $s_2a_3s_5$	N R	95. $a_2s_4a_4$	N R
62. $a_1a_4s_5$	S R	79. $s_2a_3a_5$	N R	96. $a_2s_4s_5$	N R

Table 7.1 (continued)

MANIPULATOR	REMARK	MANIPULATOR	REMARK	MANIPULATOR	REMARK
97. $a_2s_4a_5$	N R	105. $s_3s_4a_4$	S G	113. $a_3s_4a_5$	S R
98. $a_2a_4s_5$	N R	106. $s_3s_4s_5$	S R	114. $a_3a_4s_5$	S R
99. $a_2a_4a_5$	N R	107. $s_3s_4a_5$	S R	115. $a_3a_4a_5$	S R
100. $a_2s_5a_5$	S R	108. $s_3a_4s_5$	S R	116. $a_3s_5a_5$	S R
101. $s_3a_3s_4$	S G	109. $s_3a_4a_5$	S R	117. $s_4a_4s_5$	D
102. $s_3a_3a_4$	S G	110. $s_3s_5a_5$	S R	118. $s_4a_4a_5$	D
103. $s_3a_3s_5$	S R	111. $a_3s_4a_5$	S G	119. $s_4s_5a_5$	D
104. $s_3a_3a_5$	S R	112. $a_3s_4s_5$	S R	120. $a_4s_5a_5$	D

Key to Remarks: D - degenerate
S - Soluble
N - Insoluble
R - Restricted orientation for reachable wrist positions
G - No orientation restriction for reachable wrist positions

7.2 Additional Considerations

Aside from kinematic considerations, there are many less objective criteria in choosing a manipulator for use with a computer system. For the sake of completeness we mention some of these additional considerations, and give a few remarks about several of the more important ones.

1. Ease of Interface with a Digital Computer.

   The actuators of a manipulator must be such that their control may be easily assumed by a digital computer. In addition position feedback must be available. This will generally be from potentiometer or shaft encoders.

2. Power Source.

   Manipulators are in general electrically, hydraulically, or pneumatically powered. Electricity is universally available and inexpensive. Hydraulic power provides the means for converting a large amount of energy to motion with a minimum of weight, thus an advantage where speed is required. Pneumatically powered manipulators, working off of air, are cleaner than hydraulic systems. However, for safety reasons, they must operate at a much lower pressure and therefore will have poorer dynamic response.

3. Structural Rigidity.

   The structural members must have a minimum deformation under load so that the position of the hand may be accurately computed from the rotations in the joints.

In addition the joints must contain a minimum of play for the same reason.

4. Range of Freedom

It is desirable that each joint of a manipulator possess a large range. Even though a position might be reachable from a kinematic point of view, the physical limits on actuators will greatly reduce the range of these. In fact, many of the problems encountered while using th obstacle avoidance programs were due to the very restricted range of motion on the electric arm (Figure 6.6).

5. The Outline of the Manipulator.

We would like the manipulator to have a slim outline so that it could work in tight places. In addition a smooth profile might be desirable so that it would be easily recognizable in a T.V. image.

6. Other Factors.

Additional factors to be considered are: precision, speed, cost controllability (i.e., the ability to follow a prescribed path), and safety.

When choosing a manipulator we cannot hope to maximize all of these considerations. Many of these are influenced by the type of task performed by the manipulator. For example, if a goal for the hand-eye system is to assemble a machine containing small electronic components, the manipulator must be capable of very delicate movement and position accuracy. For tasks involving throwing or catching objects, the arm

must be able to move rapidly, and be accurately controlled. Thus some applications require obvious tradeoffs (e.g., precision and speed), and in others certain considerations predominate.

From experience with the two manipulators used at the Stanford Artificial Intelligence Project we may make some comment on specific arms. The project presently has two arms. One is a modified electric prosthetic arm (Figure 6.6). The other is hydraulically powered (Figure 7.3).

The d.c. electric motor driven arm has proven acceptable for stacking blocks. After some experimentation, a rate modulated pulse dc system seems to be an excellent way to control the arm. With position feedback via potentiometers, and an external power supply, it is satisfactorily interfaced with the computer. However, it is somewhat lacking in the range of freedom and structural integrity - problems that could be overcome with a second generation arm of this type. It is not particularly fast nor particularly precise. The precision problem stems partly from the poor structure, and partly from the control problem caused by the inherent inertia in the motors. It is expected that with refinement of the control scheme, the precision and controllability could be considerably improved.

Although experience with the hydraulic arm is limited at this time, it shows promise of great speed. It also appears structurally sound, and has a wide range of freedom in its joints. It is somewhat massive due to its high speed and torque capabilities. At this time, the control problem using two-stage servo-valves appears soluble. The physical danger to personnel and equipment is obvious and this arm is housed in a room

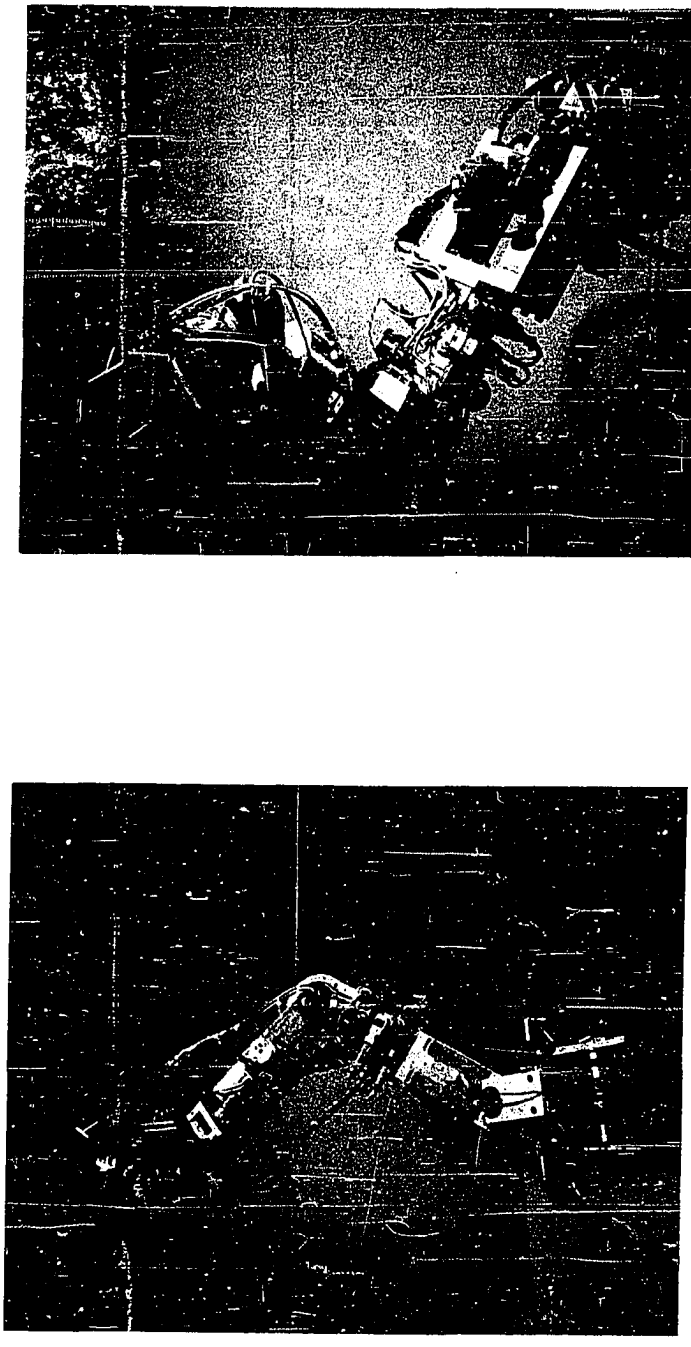


Figure 7.3. Hydraulic Arm at Stanford Artificial Intelligence Project.

isolated from the computer. This makes the interface with the computer T.V. system difficult, though soluble. In addition the forces involved require that the arm be firmly anchored to the floor.

At present these manipulators are used for fairly simple tasks. As the hand-eye program becomes more advanced the tasks will become more involved. At some future time, then, one might expect to be able to say more about choosing a manipulator for computer control in a more complex environment.

CHAPTER VIII

CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

In this dissertation, the kinematic problems associated with manipulators have been explored. It is hoped that the classification scheme and catalog of manipulators, presented in Chapter II, will lead to manipulators being compared on a scientific basis. Manipulators whose exteriors seem much different, are often kinematically equivalent. Thus solutions for one manipulator are applicable to another.

It is seen that the problem of positioning a manipulator is directly related to the displacement analysis of mechanisms. The solutions presented for cases with three revolute axes intersecting at a point seem to be previously unknown. These results therefore represent a contribution to spatial linkage analysis.

It is felt that these solutions, along with the extension to the special cases with only pairs of axes intersecting, give insight into the kinematic analysis problem for the most general six degree-of-freedom manipulator. That is, for the special case of three intersecting pairs of axes, four different configurations were found leading to the same hand position and orientation. For two pairs of intersecting axes, eight configurations were found, and for only one pair of axes intersecting, sixteen configurations were shown to be possible. In all of these special cases, adjacent axes were orthogonal, and the adjacent common normals intersected one another. Since no axes nor adjacent common normals intersect in the most general problem, it is almost

certain that the general problem has even more possible configurations leading to the same hand position and orientation.

The possibility of a very large number of configurations indicates that, even if a solution to the general problem could be expressed as a single polynomial in one unknown, this polynomial would be of such a high degree that it would be impossible to find all the roots. We conclude, then, that the complete solution to the most general problem is not at this time technically feasible: perhaps, someone, someday will solve the problem. Kinematicians have been trying for over 50 years.

The iterative technique, based on velocity, was found to be superior to the Newton-Raphson method both in the amount of time taken per iteration, and in the range of distance between positions where convergence occurred. The iterative technique may be used to good advantage when the distance between positions is small. Thus an approximate model having a closed form solution could be used to find starting points from which the numerical procedure could be used to find actual solutions.

The problem of placing the end of a digital manipulator at a target appears soluble. The results have shown that the hand can be placed close to an arbitrary point. Different strategies could be developed that might save computer time and improve performance. Matching the arm to a curve, as was done in the planar model, would undoubtably help shape the arm. However, if this manipulator is to be used, its inherent drawbacks must be remedied. That is, the motion between states must be made controllable.

The trajectory generation and obstacle avoidance routines were found to perform a basic function: they allow a manipulator to work

within a space composed of large obstacles. Previous attempts at obstacle avoidance dealt only with keeping the hand away from obstacles. In this work, possible conflict between obstacles and all links of the manipulator is considered. Future work should attempt to remedy the following shortcomings:

1. The description of obstacles needs to be improved. In this work, obstacle properties were defined using a data structure within the confines of FORTRAN. Many useful properties, such as the relation between obstacles were not stored. With the development of a more sophisticated world model, using a higher level programming language, the manipulator and obstacles could be modeled more precisely. This would lead to more accurate conflict detection and better information about which direction to move to get away from an obstacle.
2. The problem of moving between two closely spaced obstacles has not been adequately solved.
3. The computer time to generate trajectories may be excessive. This is in part due to the attempt to make the routines applicable to a variety of manipulators. For example, the analysis program used to compute hand position and orientation is applicable to the most general arm. Fast machine language subroutines to perform dot products would decrease machine time.

4. At present the routines do not benefit from past experience. Improvement might result if previously generated trajectories were stored and parts of them were used over again when similar situations arose.

The problem of zones has not been fully explored. Although a mathematical interpretation of zones is presented, it is not totally satisfactory as it depends upon the existence of a solution. Geometrical methods give insight into special cases, however they have the disadvantage of not being generally applicable.

In this work six degree-of-freedom manipulators were studied because it is necessary to have six degrees-of-freedom to grasp an object at an arbitrary position with an arbitrary orientation. However, since manipulators with more than six freedoms have not been studied, future work might involve investigating the use of additional freedoms. For example, extra degrees-of-freedom would be useful in avoiding obstacles.

It is felt that the theoretical results of this investigation, and the computer programs developed from them, yield a "universal" kinematic analysis and trajectory generator procedure. It is expected that the package of computer programs (which will be further documented in a project memo) can be applied to any six degree-of-freedom manipulator with turning joints.

APPENDIX I

DETAILS OF SOLUTION BY NEWTON-RAPHSON METHOD

The inputs to the program are:

NBM: the maximum number of iterations

THM: the maximum allowable correction in radians

XBM: the maximum allowable change in target

$\underline{L}_1\underline{L}_2\underline{N}_1\underline{N}_2$: Two vectors fixed in the hand in their initial and final positions. $\underline{L}$ is the direction the hand points (the direction of ^{7}x-axis) and $\underline{N}$ is the direction of the sixth revolute axis. The subscript 1 refers to initial, 2 to the final position .

$\underline{P}_1$ and $\underline{P}_2$: Vectors specifying the initial and final position respectively of a point in the hand.

Theta: A 1x6 vector giving the initial joint angles (i.e., θ_i , i=(1,...,6)

In addition the program uses the following subprograms:

ARMCON: Specifies the parameters of the arm

HANDPO: Analysis program that computes the position and orientation of all the links in the arm, using the present values of the joint angles.

MATINV: Routine to invert matrices and solves the linear equation $A\underline{x} = \underline{b}$.

The program basically solves the matrix equation (4.5). The A_{io} are

found from the analysis program and the coefficients of the $\delta\theta_i$ are generated by successive matrix multiplication. The matrix Aeq is obtained from the inputs. It is of the form

$$Aeq = \begin{bmatrix} (\underline{L}_2) & (\underline{M}_2) & (\underline{N}_2) & (\underline{P}_2) \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a_{11} a_{12} a_{13} a_{14} \\ a_{21} a_{22} a_{23} a_{24} \\ a_{31} a_{32} a_{33} a_{34} \\ 0 \quad 0 \quad 0 \quad 1 \end{bmatrix} \qquad \text{(A1-1)}$$

Since the rotation portion of the matrix is composed of nine elements and only three are independent, we select the equations formed from elements a_{11}, a_{13}, and a_{32} which together with a_{14}, a_{24}, a_{34} give us six independent equations.

Specification of Intermediate Goal

If the changes in position and orientation represented by the right-hand side of equation (4.5) is too large, (4.5) is not valid and an intermediate goal is necessary. The unit vector $\underline{L}$ and $\underline{N}$ respectively rotate through angles defined by the arccosines of $\underline{L}_1 \cdot \underline{L}_2$ and $\underline{N}_1 \cdot \underline{N}_2$, about axes defined by $\underline{L}_1 \times \underline{L}_2$ and $\underline{N}_1 \times \underline{N}_2$. For intermediate goals, $\underline{L}_1$ and $\underline{N}_1$ are rotated through fractions of their total rotation. In addition, the same fraction of $\underline{P}_2 - \underline{P}_1$ is added to $\underline{P}_1$. A block diagram of this is presented.

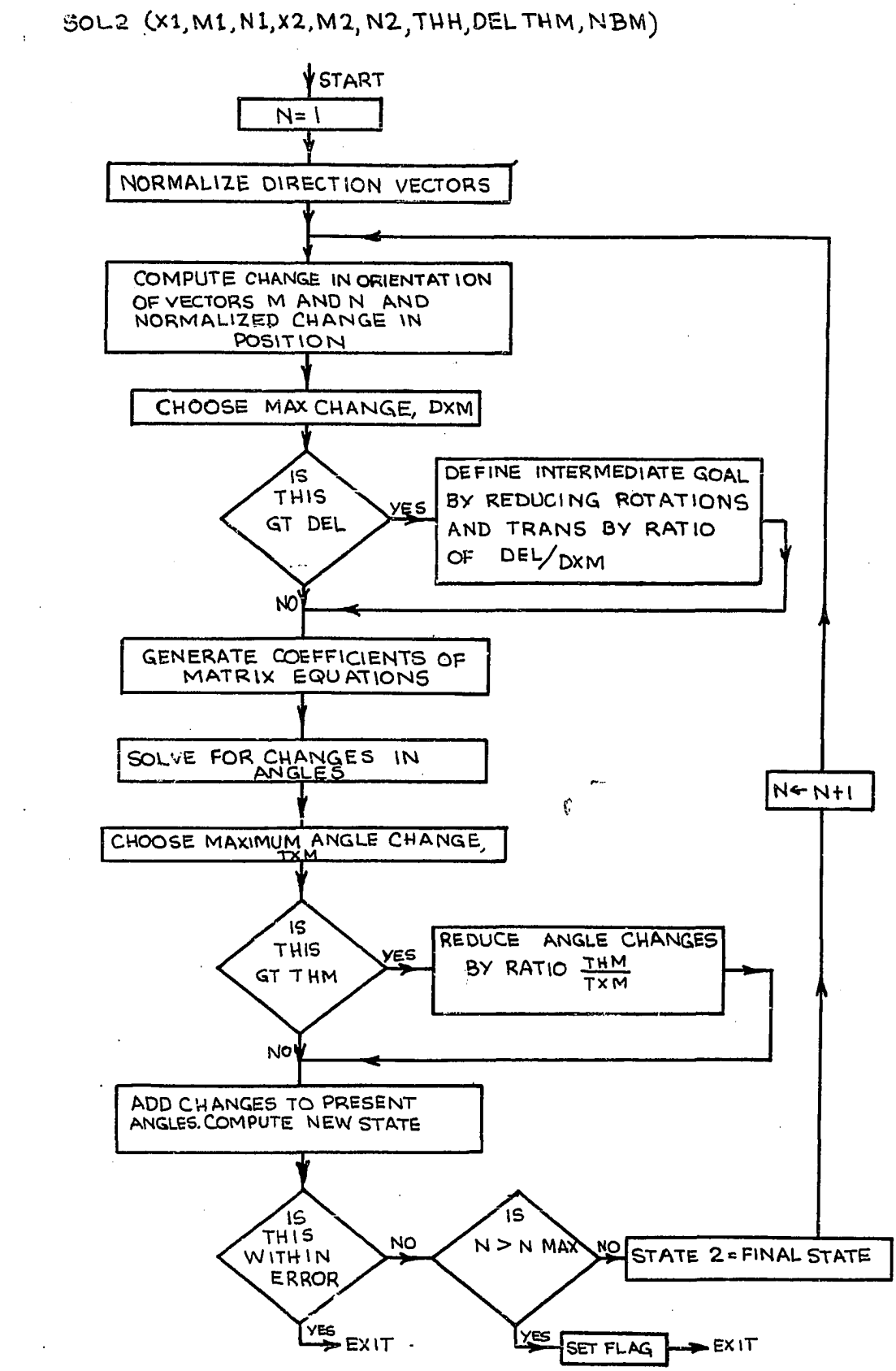


Figure A1.1. Block diagram of SOL2--solution using Newton-Raphson.

APPENDIX II

DETAILS OF ITERATIVE VELOCITY METHOD

We first present a method for finding the screw given:

$\underline{L}_1$ and $\underline{M}_1$: Two vectors fixed in the hand in its initial state.

$\underline{L}_2$ and $\underline{M}_2$: The same two vectors after a change in position and orientation.

$\underline{P}_1$ and $\underline{P}_2$: Vectors from the origin of the 1 system to the same point in the hand before and after the change in position and orientation.

The direction of the screw axis $\underline{n}$ and the magnitude of the rotation φ can be found from the following statement of Euler's theorem:

$$\underline{n} \tan \frac{\varphi}{2} = \frac{(\underline{L}_2-\underline{L}_1) \times (\underline{N}_2-\underline{N}_1)}{(\underline{L}_2-\underline{L}_1) \cdot (\underline{N}_2+\underline{N}_1)} \tag{A2-1}$$

if we define

$$\underline{W} = \frac{(\underline{L}_2-\underline{L}_1) \times (\underline{N}_2-\underline{N}_1)}{(\underline{L}_2-\underline{L}_1) \cdot (\underline{N}_2+\underline{N}_1)} \tag{A2-2}$$

then

$$\underline{n} = \frac{\underline{W}}{|\underline{W}|} \tag{A2-3}$$

$$\varphi = 2 \arctan |\underline{W}| \, . \tag{A2-4}$$

The normal from the origin to the screw axis, $\underline{r}$, is computed from

$$\underline{r} = \frac{1}{2}\left[\underline{P}_1+\underline{P}_2 + \underline{W} \times \frac{(\underline{P}_2-\underline{P}_1)}{\underline{W}^2} - \underline{W} \cdot \frac{(\underline{P}_2+\underline{P}_1)}{\underline{W}^2} \, \underline{W} \right] . \tag{A2-5}$$

The magnitude of the translation, S, is

$$S = \frac{\underline{W} \cdot (\underline{P}_2 - \underline{P}_1)}{|\underline{W}|} \tag{A2-6}$$

Lastly, defining the pitch of the screw H as

$$H = \frac{S}{\varphi} \tag{A2-7}$$

$$\varphi = \frac{\underline{W} \cdot (\underline{P}_2 - \underline{P}_1)}{2\,|\underline{W}|\,\arctan\,(|\underline{W}|)} \quad , \tag{A2-8}$$

we have all the necessary parameters of the screw.

We next show that for infinitesimal motion, $\underline{W}$ is related to the angular velocity. We write

$$\underline{L}_2 = \underline{L}_1 + \frac{d\underline{L}_1}{dt}\Delta t + \frac{d^2\underline{L}_1}{dt_2}\Delta t^2 + \ldots \tag{A2-9}$$

$$\underline{N}_2 = \underline{N}_1 + \frac{d\underline{N}_1}{dt}\Delta t + \frac{d^2\underline{N}_1}{dt^2}\Delta t^2 + \ldots \tag{A2-10}$$

$$\varphi = \cancel{\varphi_1}^{\,0} + \frac{d\varphi}{dt}\Delta t + \frac{d^2\varphi}{dt^2}\Delta t^2 + \ldots \tag{A2-11}$$

using the above in (A2-2)

$$\underline{W} = \frac{\frac{d\underline{L}^1}{dt}\Delta T \times \frac{d\underline{N}_1}{dt}\Delta t + \ldots}{\frac{d\underline{L}^1}{dt}\Delta T \cdot 2\underline{N}_1 + \ldots} = \tag{A2-12}$$

and in its equivalent from (A2-1)

$$\underline{W} = \tan\left(\frac{\varphi}{2}\Delta t + \ldots\right)\underline{n} \quad . \tag{A2-13}$$

Then equating the right-hand sides of (A2-13) and (A2-12) and taking the limit as $\Delta t \to 0$ we get

$$\dot{\varphi}\,\underline{n} = \frac{\frac{d\underline{L}_1}{dt} \times \frac{d\underline{N}}{dt}\,1}{\frac{d\underline{L}}{dt}1 \cdot \underline{N}\,1} \tag{A2-14}$$

which are equivalent expressions for the angular velocity of the hand.

If we define the approximate angular velocity, $\underline{\omega}$ to be

$$\underline{\omega} = \frac{\Delta\varphi}{\Delta t}\,\underline{n} \tag{A2-15}$$

and the rotation is small so that from (A2-3) we find $\underline{n}$ and from (A2-4)

$$\Delta\varphi = 2 \arctan |\underline{W}| \tag{A2-16}$$

Now the approximate velocity of a point in the hand at the origin is:

$$\underline{V} = H\underline{\omega} - \underline{\omega} \times \underline{r} \tag{A2-17}$$

$$= H\frac{\Delta\varphi}{\Delta t}\underline{n} - \frac{\Delta\varphi}{\Delta t}\underline{n} \times \underline{r} \tag{A2-18}$$

where H, $\Delta\varphi$, $\underline{n}$, and $\underline{r}$ are the screw parameters formed from the change in hand position and orientation.

Inputs to the program

This program has the same inputs as the Newton-Raphson program. In addition to the same subprograms, it requires:

SCREW which computes the screw defined by $\underline{P}_1\underline{P}_2\underline{L}_1\underline{L}_2\underline{N}_1\underline{N}_2$ using equations (A2-1) - (A2-7).

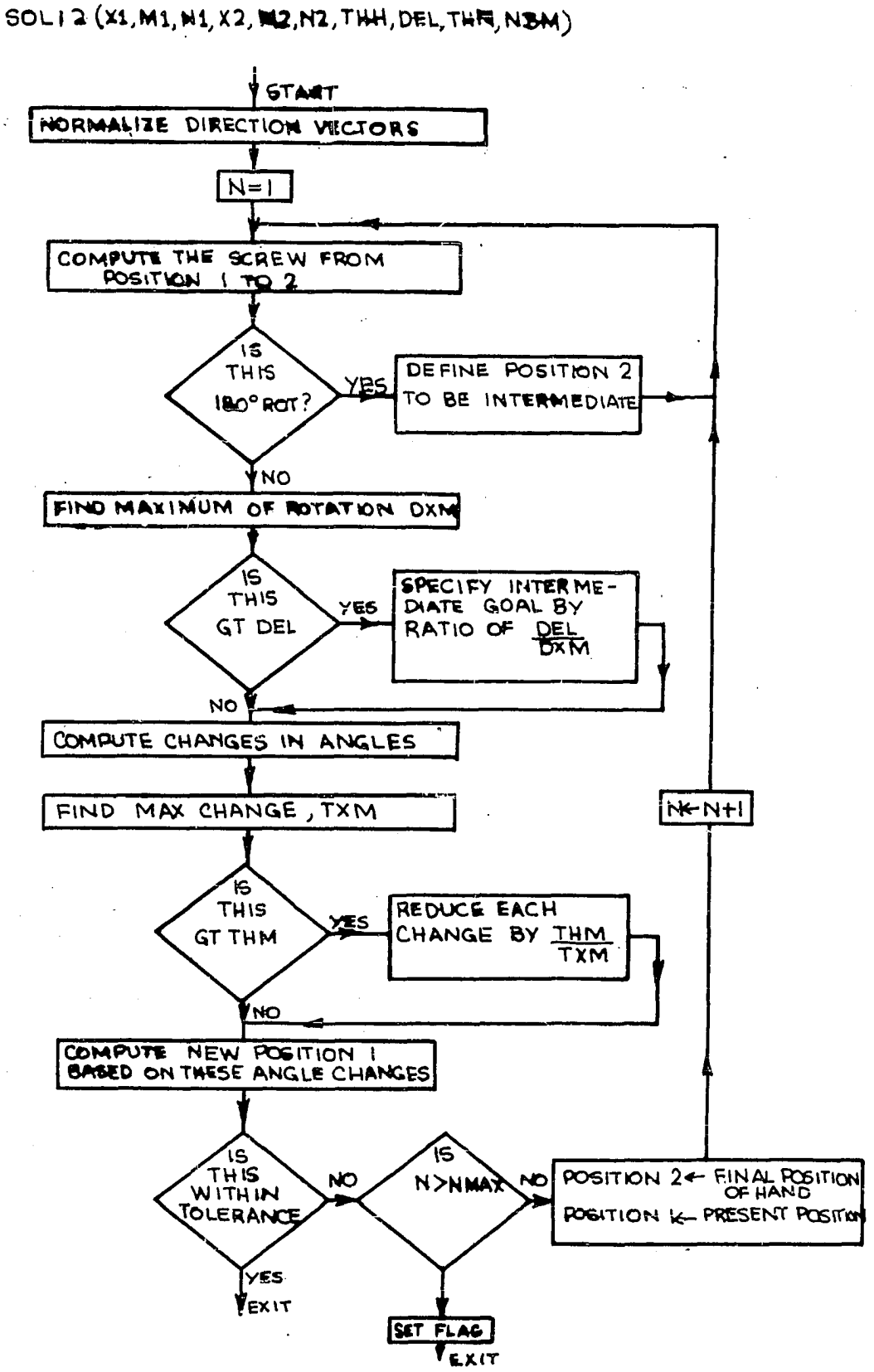


Figure A2.1. Block diagram of SOL12 - the iterative velocity method.

APPENDIX III

MATHEMATICAL DETAILS FOR THE DIGITAL MANIPULATOR

A3.1 Transformation to find hand position given the angles (Planar Case)

We use the basic link model described in Chapters II and III. For the planer case, the z-coordinates and the angles between adjacent links are all zero. In addition, we assume that all the common normals are the same length, d, so that we may rewrite (3.1) as

$$A_i = \begin{bmatrix} c\theta_i & -s\theta_i & dc\theta_i \\ s\theta_i & c\theta_i & ds\theta_i \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{(A3-1)}$$

and similiarly from (3.17) we may describe the position of the hand (x,y) by:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = A_1 \ldots A_n \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad \text{(A3-2)}$$

where n is the number of links in the arm.

A3.2 Transformation to Find Hand Position Given the Angles (3-Dimensional Case)

Consider the link element shown in Figure A3.1. The link model (Chapter II) and transformations (Chapter III) are applicable to

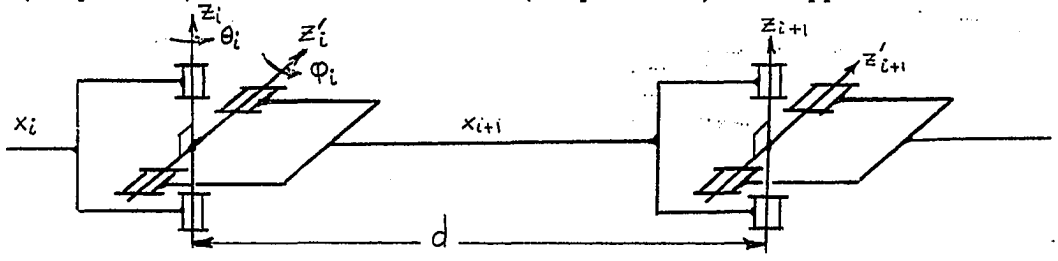


Figure A3.1. The Basic Element for a Three-Dimensional Digital Arm.

this manipulator. We note that each of the "joints" of Figure A3.1 actually contains two degrees of freedom. Using (3.1) we may write the transformation due to a rotation about each axis, so that the transformation between adjacent elements may be written

$$B_i = \begin{bmatrix} c\theta_i c\varphi_i & -s\theta_i & -c\theta_i\, s\varphi_i & d\, c\theta_i\, c\varphi_i \\ s\theta_i\, c\varphi_i & c\theta_i & -s\theta_i\, s\varphi_i & d\, s\theta_i\, c\varphi_i \\ s\varphi_i & 0 & c\varphi_i & d\, s\varphi_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(A3-3)}$$

and the coordinates, of the end point of the last link, (x,y,z) are

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = B_1 \cdots B_n \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad \text{(A3-4)}$$

where n is the number of elements in the arm.

To find rotations about axes z_i and z_i' (Figure A3.1) which lead to a tilt of θ_o about an axis midway between z_i and z_i', we note that this is equivalent to rotating z_i through 45^o about x_i and then rotating x_{i+1} through θ_o about the new z_i axis. Then using equation (3.1) to express these rotations, the resulting transformation matrix is

$$T = \begin{bmatrix} c\theta_o & -s\theta_o & 0 & dc\theta_o \\ \frac{s\theta_o}{\sqrt{2}} & \frac{c\theta_o}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & \frac{ds\theta_o}{\sqrt{2}} \\ \frac{s\theta_o}{\sqrt{2}} & \frac{c\theta_o}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{ds\theta_o}{\sqrt{2}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(A3-5)}$$

Then the direction of the x_{i+1} axis is represented by a vector composed of the '11', '21', and '31' elements of (A3-5). These elements

must be equal to the corresponding elements of (A3-3) which lead to:

$$s\varphi = \frac{s\theta_o}{\sqrt{2}}$$

and

$$t\theta = \frac{t\theta_o}{\sqrt{2}}$$

A3.3 Derivation of Curve Composed of Segments of Four Circles

We want to find a curve made up of segments of four circles connected in such a way that adjacent circles are tangent to one another. Thus a smooth transition between the elements of the curve exists. In addition we require that the total arc length be specified. We also specify the slope of the curve at each end and the radii of the circles. Consider such a curve shown in Figure A3.2. Given the radii of the circles and the position of the base of the arm, we easily locate center A.

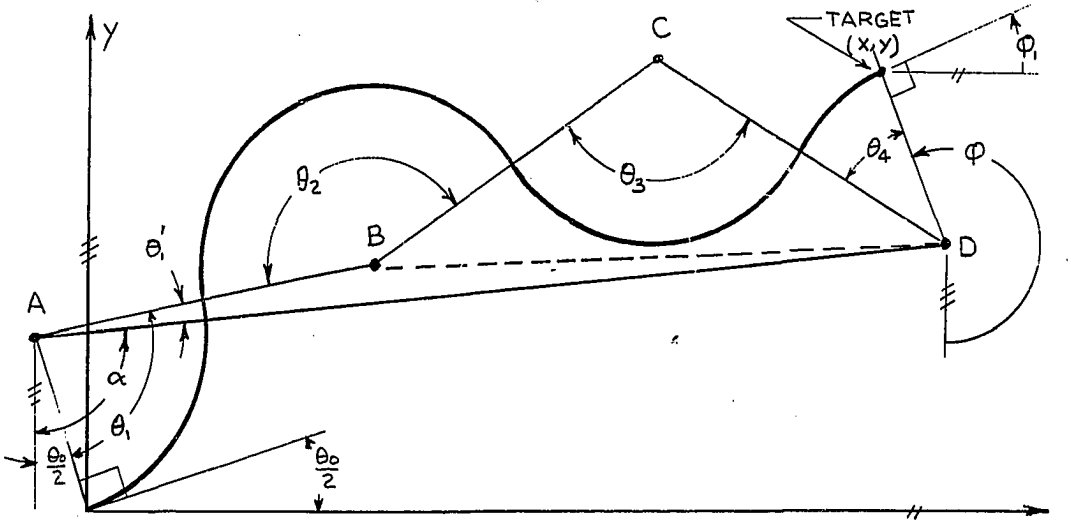


Figure A3.2. Curve composed of segments of four circles.

The known angle that the tangent to the curve at the base makes with the horizontal $\frac{\theta_o}{2}$ is one-half the allowable joint rotation. The end point on the curve is specified by its coordinates (x,y), and the end slope by the angle φ_1 . This then fixes center D , from which the line segment $\overline{AD}$, and the angles φ and α are defined. Next we must locate the centers B and C , and find the angles θ_1, θ_2, θ_3, and θ_4 which define each segment of the curve. For the purpose of derivation, we introduce the angle θ_1' between line $\overline{AD}$ and $\overline{AB}$.

If we denote the radii of the circles by r , then:

$$\left.\begin{aligned} \overline{OA} &= r \\ \overline{AB} = \overline{BC} &= \overline{CD} = 2r \end{aligned}\right\} \qquad \text{(A3-6)}$$

and define R such that

$$R = \overline{AD} \quad . \qquad \text{(A3-7)}$$

Then from Figure A3-2 we observe

$$\theta_1 + \frac{\theta_o}{2} = \alpha + \theta_1' \qquad \text{(A3-8)}$$

and

$$\varphi = \theta_1 + \frac{\theta_o}{2} + (\pi - \theta_2) - (\pi - \theta_3) + (\pi - \theta_4)$$

or

$$(\theta_1 + \theta_3) - (\theta_2 + \theta_4) = \varphi - \frac{\theta_o}{2} - \pi \qquad \text{(A3-9)}$$

Next impose the constraint that the total arc length is L :

$$r(\theta_1 + \theta_2 + \theta_3 + \theta_4) = L$$

or

$$(\theta_1 + \theta_3) + (\theta_2 + \theta_4) = \frac{L}{r} \quad . \qquad \text{(A3-10)}$$

Combining (A3-9) and (A3-10) by subtraction and addition yields

$$(\theta_1 + \theta_3) = \tfrac{1}{2}\left(\frac{L}{r} + \varphi - \frac{\theta_o}{2} - \pi\right) \tag{A3-11}$$

$$(\theta_2 + \theta_4) = \tfrac{1}{2}\left(\frac{L}{r} - \varphi + \frac{\theta_o}{2} + \pi\right) . \tag{A3-12}$$

Writing the law of cosines for ΔBCD yields

$$\overline{BD}^2 = 8r^2(1 - \cos\theta_3) . \tag{A3-13}$$

Then writing the law of cosines for ΔBAD yields

$$\overline{BD}^2 = 4r^2 + R^2 - 4rR\cos\theta_1' \tag{A3-14}$$

and we may easily eliminate $\overline{BD}^2$ from (A3-13) and (A3-14) to obtain

$$\cos\theta_1' = -\frac{r}{R} + \frac{1}{4\frac{r}{R}} + 2\frac{r}{R}\cos\theta_3 . \tag{A3-15}$$

Now we combine (A3-8) and (A3-11) to get

$$\theta_1' + \theta_3 = \beta \tag{A3-16}$$

where

$$\beta = \tfrac{1}{2}\left(\frac{L}{r} - \pi + \varphi - 2\alpha + \frac{\theta_o}{2}\right) \tag{A3-17}$$

We next eliminate θ_3 between (A3-15) and (A3-16), which after simplification results in

$$k_1\cos^2\theta_1' + k_2\cos\theta_1' + k_3 = 0 \tag{A3-18}$$

where

$$k_1 = 1 - \frac{4r}{R}\cos\beta + 4\frac{r^2}{R^2} \tag{A3-19}$$

$$k_2 = 2\left(1 - \frac{2r}{R}\cos\beta\right)\left(\frac{r}{R} - \frac{1}{4\frac{r}{R}}\right) \tag{A3-20}$$

$$k_3 = \left(\frac{r}{R} - \frac{1}{4\frac{r}{R}}\right)^2 - 4\frac{r^2}{R^2}(1 - \cos^2\beta) \tag{A3-21}$$

from which θ_1' may be obtained. Knowing θ_1' we compute θ_1' from (A3-8) which locates center B . Once B is known, center C is found by considering the intersection of two circles of radius 2r ,

one with center at B , the other at A . Once the circle centers are located, the angles θ_2 , θ_3 , θ_4 can be easily found.

A3.4 Description of Programs - 2-Dimensional Model

First the arm is put into an initial configuration either in an arbitrary manner or with the subroutine INITIAL that matches the arm to the curve composed of circle segments.

Recalling that the coordinate transformation exists:

$$^{1}\underline{X} = A_1 \cdots A_n \, {}^{n+1}\underline{X}$$

the multiplication of the transformation is broken into 3 parts

$$(A_1 \cdots A_{index\ -1})\ (A_{index} \cdots A_{index\ +\ look})\ (A_{index\ +\ look\ +\ 1} \cdots A_n)$$

where "index" is the number of the joint under consideration and "look" is an integer giving the number of stages of look-ahead. Then the first and third term are generated by a subroutine that transforms coordinates. These are temporarily stored. Then the middle term is generated for all possible θ_i , i = index , ... , index + look . The matrix multiplication is performed for each, and θ_{index} is chosen that leads to minimum error. Index is then incremented by 1 and the process repeated.

Description of INITIAL:

This subroutine generates a curve composed of segments of four circles, and then generates points on this curve corresponding to joints of the arm. The arm is then made to follow the curve by various techniques.

Techniques for curve matching.

1) θ_i is moved so as to put position of joint i+1 as close as possible to point i+1 on the curve.

2) position of joint i+1 and slope of link i are made as close as possible to curve at corresponding point on curve.

3) A procedure named "match-ahead" tries to match $(i+1)^{\underline{th}}$ joint with $(i+1+\Delta)^{\underline{th}}$ point on curve (Δ is the number of joints of match-ahead) by moving θ_i.

The radius of the four circles can be made greater than or equal to the smallest radius that the arm can turn.

A3.5 Description of Programs - Three-Dimensional Model

This program is similiar to the two-dimensional case with the exception of changes to make it more efficient.

After the starting configuration and amount of look-ahead are specified, three matrices are generated. They are

1) the identity matrix

2) $(B_1 \quad \cdots \quad B_{1+look})$

3) $(B_{2+look} \quad \cdots \quad B_n)$

where "look" is the amount of look-ahead and B_i is defined as in Equation A3-3. Call these M_1, M_2, and M_3,

Then $(M_1)(M_2)(M_3)$ is the total transformation matrix. This product is evaluated for the 8^{look+1} different M_2 matrices and the state at joint 1 chosen which minimizes the error.

Then M_1, M_2, and M_3 are redefined

$$M_1 = M_1 \times B_1$$

$$M_3 = B_{2+look}^{-1} \times M_3$$

$$M_2 = (B_2 \times, \ldots, \times B_{2+look})$$

and θ_2 is chosen in a manner similiar to that of θ_1. The process continues.

APPENDIX IV

DETAILS OF CONFLICT DETECTION

The methods presented here result directly from classical vector geometry.

A4.1 Objects as Spheres

Note that the physical links of the manipulator are modeled as cylinders. We compute the distance from the line segment that is the axis of this cylinder to the center of the sphere. The problem is then to find the distance between a point and a line segment. Consider the line

$$\underline{r} = \underline{a} + \underline{b}t \qquad \text{(A4-1)}$$

and the point P, described by the vector $\underline{p}$ as shown in Figure A4.1.

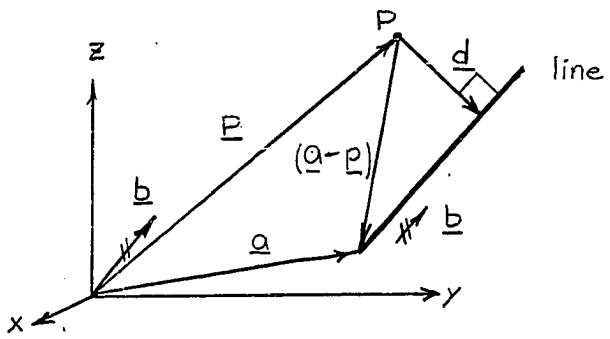


Figure A4.1. Distance Between Point and Line.

If $\underline{a}$ is a vector to the end of the line segment, $\underline{b}$ a unit vector parallel to the line, t_1 the length of the line segment and

$$0 \leq t \leq t_1$$

then $\underline{r}$ is the locus of points on the line segment. A vector, $\underline{d}$, from

the point to the line, normal to the line is given by

$$\underline{d} = \underline{b} \times [(\underline{a}-\underline{p}) \times \underline{b}] = (\underline{a}-\underline{p}) - \underline{b}\cdot(\underline{a}-\underline{p})\underline{b} \qquad \text{(A4-2)}$$

To see if this normal cuts the segment of interest, we note from Figure A4.1 and equation (A4-1)

$$\underline{r} = \underline{a} + \underline{b}t^* = \underline{p} + \underline{d} \qquad \text{(A4-3)}$$

where t^* is the value of t where the normal intersects the line, then from (A4-3)

$$t^* = \underline{b}\cdot[\underline{d} - (\underline{a}-\underline{p})] \qquad \text{(A4-4)}$$

Then if

$$0 \leq t^* \leq t_1 ,$$

the normal intersects the segment. If this is not the case, we find the distance between P and the end points of the line segment and choose the minimum. A routine called PTLINE performs these calculations.

A4.2 Objects as Infinite Planes

Here we find the distances between the end points of a line segment and a plane. Consider the plane described by $\underline{b}$, a unit vector normal to the plane and p, the distance of the plane from the origin measured in the $\underline{b}$ - direction. If $\underline{r}$ describes a point, then a vector from the plane to the point, normal to the plane is given by, $\underline{d}$:

$$\underline{d} = (\underline{r}\cdot\underline{b} - p)\underline{b} \qquad \text{(A4-5)}$$

A routine PLLINE performs the computations.

A4.3 Objects as Cylinders

The problem is to find the shortest distance between two line segments, which are the axes of the cylinders. Consider line 1 and

line 2 given by:

$$\underline{r}_1 = \underline{a}_1 + \underline{b}_1 t_1 \tag{A4-6}$$

$$\underline{r}_2 = \underline{a}_2 + \underline{b}_2 t_2 \tag{A4-7}$$

as shown in Figure A4.2.

If respectively: $\underline{a}_1$, $\underline{a}_2$ are vectors to the end of the line segments, $\underline{b}_1$, $\underline{b}_2$ unit vectors parallel to the lines, t_{10} , t_{20} the length of the line segments and

$$0 \leq t_1 \leq t_{10}$$

$$0 \leq t_2 \leq t_{20}$$

then $\underline{r}_1$, $\underline{r}_2$ are the loci of points on the line segments. A vector, $\underline{d}$, from line 2 to line 1,

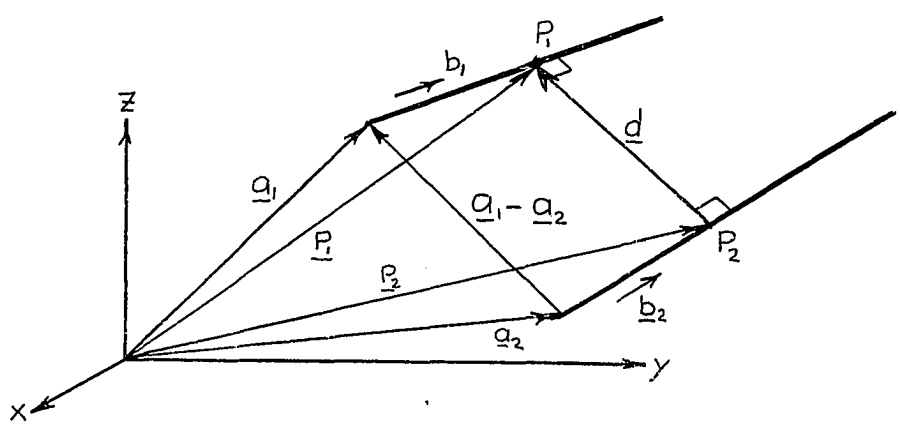


Figure A4.2. Distance Between Two Lines.

normal to both is given by

$$\underline{d} = \frac{(\underline{a}_1 - \underline{a}_2) \cdot (\underline{b}_1 \times \underline{b}_2)}{1 - (\underline{b}_1 \cdot \underline{b}_2)^2} (\underline{b}_1 \times \underline{b}_2) \tag{A4-8}$$

Now to find where the normal cuts the lines we find where line 2 pierces the plane containing line 1 and $\underline{d}$. The locus of points, $\underline{r}$, in this plane may be written

$$(\underline{r} - \underline{a}_1) \cdot \left[\underline{b}_1 \times (\underline{b}_1 \times \underline{b}_2)\right] = 0 \tag{A4-9}$$

This is of the form:

$$\underline{r} \cdot \underline{n} - p = 0 \ , \qquad \text{(A4-10)}$$

where $\underline{n}$ and p may be found from (A4-9). Then line 2 pierces this plane at the point p_2, defined by the vector $\underline{p}_2$:

$$\underline{p}_2 = \underline{a}_2 + \frac{\underline{b}_2(p-\underline{a}_2 \cdot \underline{n})}{\underline{b}_2 \cdot \underline{n}} \qquad \text{(A4-11)}$$

where $\underline{n}$ and p are as in (A4-10). Now the corresponding point on line 1 is found from:

$$\underline{p}_1 = \underline{p}_2 + \underline{d} \ . \qquad \text{(A4-12)}$$

We next determine if $\underline{p}_1$ and $\underline{p}_2$ lie on the segments of interest in a manner similar to that used in A4.1. If this is not the case then we check the distance between endpoints of one line segment to the other line segment and vice versa with the methods of A4.1. A routine LNLINE performs these computations. It uses PTLINE if necessary.

APPENDIX V

SOLUTION OF A $6R,a_1a_2a_4a_5$ MANIPULATOR AS FOUR QUADRATICS

In section 3.3.8 we showed how the solution of a $6R,a_1a_2a_4$ manipulator could be reduced to a single polynomial of degree eight. We now present the details of how a $6R,a_1a_2a_4a_5$ manipulator (with adjacent axes orthogonal) can be expressed as four quadratic equations in four unknowns. As in section 3.3.8, we write equations for the coordinate (x, y, z) of the point of intersection of axes 3 and 4 (see Figure 3.10).

To include $a_5 \neq 0$, we use equation (3.100) with $a_4 \neq 0$, $a_5 \neq 0$ $\alpha_4 = -90^\circ$, $\alpha_5 = 90^\circ$, $s_2 = s_3 = s_4 = s_5 = 0$, to obtain:

$$ {}^7\underline{P} = \begin{bmatrix} -a_4\, c\theta_5\, c\theta_6 - a_5\, c\theta_6 \\ a_4\, c\theta_5\, s\theta_6 + a_5 s\theta_6 \\ -a_4\, s\theta_5 \\ 1 \end{bmatrix} \qquad \text{(A5-1)} $$

Similarly (3.104), and (3.111) become respectively:

$$ {}^7\underline{P}^2 = a_4^2 + a_5^2 + 2a_4\, a_5\, c\theta_5 \qquad \text{(A5-2)} $$

and

$$ \left[\underline{Q}^2 + \underline{R}^2 - 2\underline{Q}\cdot\underline{R} - (a_4^2 + a_5^2)\right]^2 + 4a_5^2\left[a_{13}x + a_{23}y + a_{33}z + a_{14}^{-1}\right]^2 = 4a_4^2a_5^2 \qquad \text{(A5-3)} $$

Then from the components of ${}^7\underline{P}$ in (A5-1) we find:

$$ c\theta_6 = -\frac{{}^7x}{a_4 c\theta_5 + a_5} \qquad \text{(A5-4)} $$

$$ s\theta_6 = \frac{{}^7y}{a_4 c\theta_5 + a_5} \qquad \text{(A5-5)} $$

$$ s\theta_5 = -\frac{{}^7z}{a_4} \qquad \text{(A5-6)} $$

and from (3.104):

$$c\theta_5 = \frac{{}^7\underline{P}^2 - a_4^2 - a_5^2}{2a_4a_5} \tag{A5-7}$$

Then equation (3.122) may be written with $c\theta_5$, $s\theta_5$, $s\theta_6$, and $c\theta_6$ from (A5-4) - (A5-7) which results in:

$${}^7[\underline{\omega}_4] = \frac{1}{2a_4(a_4c\theta_5+a_5)}\begin{bmatrix} -2\,{}^7x\,{}^7z \\ -2\,{}^7y\,{}^7z \\ -2\,{}^7z^2 + {}^7\underline{P}^2 + a_4^2 - a_5^2 \\ 0 \end{bmatrix} \tag{A5-8}$$

To express $\underline{\omega}_4$ in system 1 we use (3.132) and (A5-8) to obtain:

$$\underline{\omega}_4 = \frac{1}{2a_4(a_4c\theta_5+a_5)}\begin{bmatrix} -2\,{}^7z\,(a_{11}\,{}^7x+a_{12}\,{}^7y+a_{13}\,{}^7z)+a_{13}({}^7\underline{P}^2+a_4^2-a_5^2) \\ -2\,{}^7z\,(a_{21}\,{}^7x+a_{22}\,{}^7y+a_{23}\,{}^7z)+a_{23}({}^7\underline{P}^2+a_4^2-a_5^2) \\ -2\,{}^7z\,(a_{31}\,{}^7x+a_{32}\,{}^7y+a_{33}\,{}^7z)+a_{33}({}^7\underline{P}^2+a_4^2-a_5^2) \\ 0 \end{bmatrix} \tag{A5-9}$$

Making use of (3.109) for ${}^7P^2$ and (3.110) for ${}^7x$, ${}^7y$, and ${}^7z$, (A5-9) becomes after simplification:

$$\underline{\omega}_4 = \frac{1}{2a_4(a_4c\theta_5+a_5)}\begin{bmatrix} \left(-2(x-x_4)(a_{13}x+a_{23}y+a_{33}z+a_{34}^{-1})\right. \\ \left.+a_{13}(W+\underline{R}^2-2xx_4-2yy_4-2zz_4+a_4^2-a_5^2)\right) \\ \left(-2(y-y_4)(a_{13}x+a_{23}y+a_{33}z+a_{34}^{-1})\right. \\ \left.+a_{23}(W+\underline{R}^2-2xx_4-2yy_4-2zz_4+a_4^2-a_5^2)\right) \\ \left(-2(z-z_4)(a_{13}x+a_{23}y+a_{33}z+a_{34}^{-1})\right. \\ \left.+a_{33}(W+\underline{R}^2-2xx_4-2yy_4-2zz_4+a_4^2-a_5^2)\right) \\ 0 \end{bmatrix} \tag{A5-10}$$

where the dot product in (3.109) has been evaluated, $\underline{R}$ has been expressed in terms of its components (x_4, y_4, z_4), and $(x^2 + y^2 + z^2)$ has

been replaced by the new variable W (equation (3.141). We now may form (3.112) with $\alpha_3 = 90^o$. We use (3.129) for ω_3 and (A5-10) for ω_4, and obtain after dividing by common factors and simplifying:

$$\begin{aligned} 0 = {} & 2z(a_{13}x+a_{23}y+a_{33}z+a_{34}^{-1})\left[-\underline{R}^2-a_4^2+a_5^2-a_2^2+a_1^2\right] \\ & +(W+a_2^2-a_1^2)\left[a_{33}(W+\underline{R}^2-2xx_4-2yy_4-2zz_4+a_4^2-a_5^2)\right. \\ & \left. +2z_4(a_{13}x+a_{23}y+a_{33}z+a_{34}^{-1})\right] \\ & +2za_{34}^{-1}(W+\underline{R}^2-2xx_4-2yy_4-2zz_4+a_4^2-a_5^2) \end{aligned} \qquad \text{(A5-11)}$$

Then writing (A5-3) with $\underline{Q}$ and $\underline{R}$ expressed in terms of their components (x, y, z) and (x_4, y_4, z_4) respectively, along with the definition (3.141), yields:

$$\begin{aligned} & \left[W+\underline{R}^2-(2xx_4+2yy_4+2zz_4)-(a_4^2+a_5^2)\right]^2 \\ & +4a_5^2\left[(a_{13}x+a_{23}y+a_{33}z+a_{14}^{-1})^2-a_4^2\right]=0 \end{aligned} \qquad \text{(A5-12)}$$

Similarly (3.138) becomes after introducing W from (3.141):

$$(W-a_1^2-a_2^2)^2+4a_1^2(z^2-a_2^2) = 0 \ . \qquad \text{(A5-13)}$$

The equations (A5-11), (A5-12), (A-13), and (3.141) are the four quadratics in the unknowns x, y, z, and W.

BIBLIOGRAPHY

1. Goertz, R. C., "Manipulators Developed at ANL", Proceedings of the 12th Conference on Remote Systems Technology, ANS, November, 1964, pp. 117-136.

2. Goertz, R. C., "Fundamentals of General-Purpose Remote Manipulators", Nucleonics, Vol. 10, No. 11, November, 1952, pp. 36-42.

3. Goertz, R. C., "Manipulators Used for Handling Radioactive Materials", Chap. 27 of Human Factors in Technology, New York: McGraw-Hill, 1963.

4. AMF Atomics, Greenwich, Conn., "Model 8 Manipulator" (Manufacturer's Brochure).

5. Central Research Laboratories, Inc., Red Wing, Minn., "CRL Master-Slave Manipulators", (Manufacturer's Brochure).

6. Mosher, R. S., "Handyman to Hardiman", Society of Automotive Engineers, paper no. 670088, Automotive Engineering Congress, Detroit, Mich., Jan., 1967.

7. Hunley, W. H., and W. G. Houck, "Existing Underwater Manipulators", American Society of Mechanical Engineers, paper no. 65-UNT-8.

8. Karinen, R. S., "Land-Based Remote Handling Background of Underwater Handling Equipment", American Society of Mechanical Engineers, paper no. 65-UNT-7.

9. "Hydraulics Tries its Hand", Machine Design, February 29, 1968, pp. 24-26.

10. Karchak, A., Jr. and J. R. Allen, "Investigation of Externally Powered Orthotic Devices", Final Project Report, V.R.A. Grant RD-1461-M-67, Attending Staff Association of the Rancho Los Amigos Hospital, Downey, California, February 1968.

11. Mosher, R.S., "Industrial Manipulators", Scientific American, October 1964, pp. 88-96.

12. Goertz, R. C., and F. Bevilacqua, "A Force-Reflecting Positional Servomechanism", Nucleonics, Vol. 10, No. 11, November 1952, pp. 43-45.

13. Bradley, W. E., "Telefactor Control of Space Operations", Astronautics and Aeronautics, May 1967, pp. 32-38.

14. Ferrell, W. R., "Remote Manipulation with Transmission Delay", IEEE Trans. Human Factors in Electronics, HFE -6 (1), September 1965, pp. 24-32.

15. Barnes, S., "The Coming Age of Robots", Machine Design, November 19, 1964, pp. 150-156.

16. Unimation Inc., Danbury, Conn., "Unimate", (Manufacturer's Brochure).

17. Simon, R. C., "Versatran Industrial Robot Applications", American Society of Tool and Manufacturing Engineers, paper no. AD67-162, Creative Manufacturing Seminars, 1967.

18. Ernst, H. A., "A Computer-Operated Mechanical Hand", Sc.D. Thesis, M.I.T., Department of Electrical Engineering, December 1961.

19. Project MAC, Artificial Intelligence Memo Series, M.I.T.

20. Rosen, C. A., and N. J. Nilsson (eds.), "Application of Intelligent Automata to Reconnaissance", Stanford Research Institute, Third Interium Report on Contract AF 30(602)-4147 (November 1966).

21. Rosen, C. A., and N. J. Nilsson, "An Intelligent Automaton", IEEE International Convention Record (1967).

22. Raphael, B., "Programming a Robot", Stanford Research Institute, November 1967 (preliminary draft version).

23. Beckett, J. T., "A Computer-Aided Control Technique for a Remote Manipulator", Digital Systems Laboratory, Case Institute of Technology, paper no. 1-67-44 (1967).

24. Whitney, D. E., "State Space Models of Remote Manipulation Tasks", Ph.D. Thesis, M.I.T., Department of Mechanical Engineering, January 1968.

25. Pingle, K.K., J.A. Singer, and W.M. Wichman, "Computer Control of a Mechanical Arm through Visual Input", paper presented at IFIP '68 Congress, Edinburgh, 1968.

26. Mosher, R. S., and W. B. Knowles, "Operator-Machine Relationships in the Manipulator", Aeronautical Systems Division, Wright-Patterson Air Force Base, Dayton, Ohio, Technical Report 61-430, pp. 173-186.

27. "The Handyman -- Slave Arms Getting Closer to Human Motions", Product Engineering, March 30, 1959, pp. 21-22.

28. Goertz, R. C., "Mechanical Master Slave Manipulator", Nucleonics, November 1964, Vol. 12, No. 11, pp. 45-46.

29. "What Robots Do Now for Industry", Business Week, May 20, 1967, pp. 114-115.

30. "The Automatic Machine Operator", Mechanical Engineering, June 1967, p. 57.

31. "Mitsubishi Master-Slave Manipulator", Mitsubishi International Corporation, San Francisco, Calif., (Manufacturer's Brochure No. GL-60401-E).

32. "Model 3500 Manipulator", Programmed and Remote Systems Corporation, St. Paul, Minn., (Manufacturer's Brochure).

33. "MRMU System", Ordinance Division, FMC Corporation, San Jose, Calif., (Manufacturer's Brochure No. ORD-1164-B1).

34. Jones, D. G., "A Test Bed for the Evaluation of Remote Salvage Concepts", AIAA Second Annual Meeting, July 1965, paper no. 65-525.

35. Clark, J. W., "The Mobot: A Fully Remote, Mobile Handling Device", Nuclear Electronics Laboratory, Hughes Aircraft Co., Culver City, Calif., Technical Memorandum 627.

36. Clark, J. W., "Unmanned Ground Support Equipment", Aeronautical Systems Division, Wright-Patterson Air Force Base, Dayton, Ohio, Technical Report 61-430, pp. 43-63.

37. Anderson, V. C., and R. C. Horn, "Tensor Arm Manipulator Design", American Society of Mechanical Engineers, paper 67-DE-57.

38. Wichman, W. M., "Use of Optical Feedback in the Computer Control of an Arm", Engineers Thesis, Electrical Engineering Department, Stanford University, August 1967.

39. "The Aluminaut Manipulator System", General Electric Co., Speciality Materials Handling Products Operation, Schenectady, New York, (Manufacturer's Brochure).

40. Dimentberg, F. M., "The Determination of Positions of Spacial Mechanisms", Izdat. Akad. Nauk SSSR, Moscow, USSR, 142 pgs.

41. Dimentberg, F. M., "A General Method for the Investigation of Finite Displacements of Spacial Mechanisms and Certain Cases of Passive Joints", Akad. Nauk. SSSR Trudi Sem. Teorii Mash. Mekh. 5, No. 17, 1948, pp. 5-39, (Purdue Translation No. 436, Purdue University, Lafayette, Indiana (1959)).

42. Yang, A. T., "Displacement Analysis of Spacial Five Link Mechanisms Using (3x3) Matrices", with Dual-Number Elements", ASME paper no. 68-MECH-6 (to be published in Trans. ASME).

43. Denavit, J., "Description and Displacement Analysis of Mechanisms Based on (2x2) Dual Matrices", Ph.D. Dissertation, Field of Mechanical Engineering, Northwestern University, Evanston, Ill., June 1956.

44. Yang, A. T. and F. Freudenstein, "Application of Dual Number Quarternian Algebra to the Analysis of Spacial Mechanisms", Journal of Applied Mechanics, Vol. 31, Trans. ASME, Vol. 86, Series E, 1964, pp. 300-308.

45. Chase, M. A., "Vector Analysis of Linkages", Journal of Engineering for Industry, Trans ASME, Series B, Vol. 85, 1963, pp. 289-297.

46. Denavit, J., and R. S. Hartenberg, "A Kinematic Notation for Lower Pair Mechanisms Based on Matrices", Journal of Applied Mechanics, Vol. 22, Trans. ASME, Vol. 77, 1955, pp. 215-221.

47. Hartenberg, R. S., and J. Denavit, Kinematic Synthesis of Linkages, McGraw-Hill, New York, 1964, Ch. 12.

48. Uicker, J. J., Jr., "Dynamic Force Analysis of Spacial Linkages", ASME paper no. 66-Mech-1 (published in Trans. ASME 1967).

49. Uicker, J. J., Jr., J. Denavit and R. S. Hartenberg, "An Iterative Method for the Displacement Analysis of Spacial Mechanisms", Journal of Applied Mechanics, Vol. 32, Trans. ASME, Vol. 86, Series E, 1964, pp. 309-314.

50. Urquhart, P., "Analysis of the Seven-Bar Chain", Journal of Mechanisms, Vol. 2 1967, pp. 77-83.

51. Earnest, L., Private Communication, 1968.

52. Sharpe, T. G., "Sequential Decision and Control Problems in Snake Forms", Engineer's Thesis, Electrical Engineering Department, Stanford University, Stanford, Calif., June 1966.