OBSTACLE-FREE TRAJECTORY PLANNING OF A THREE-AXIS ARTICULATED ROBOT FOR PICK-AND-PLACE OPERATION

BY

M. N. H. SIDDIQUE

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN ENGINEERING (COMPUTER SCIENCE AND ENGINEERING)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY
DHAKA, BANGLADESH

NOVEMBER, 1995

OBSTACLE-FREE TRAJECTORY PLANNING OF A THREE-AXIS ARTICULATED ROBOT FOR PICK-AND-PLACE OPERATION

BY
M. N. H. SIDDIQUE

Accepted as satisfactory for partial fulfillment of the requirements for the degree of Master of Science in Engineering (Computer Science and Engineering), Bangladesh University of Engineering and Technology, Dhaka.

EXAMINERS

[illegible] 5/12/95

1. Dr. M. Kaykobad
Associate Professor,
Department of Computer
Science & Engineering,
BUET, Dhaka.

Chairman and Supervisor

[illegible]

2. Dr. A. K. M. Mostafa Kamal
Assistant Professor,
Department of Industrial &
Production Engineering,
BUET, Dhaka.

Co-supervisor

M. S. Alam. 5-12-95

3. Dr. Md. Shamsul Alam
Professor and Head,
Department of Computer
Science & Engineering,
BUET, Dhaka.

Member

[illegible]

4. Md. Abdus Sattar
Assistant Professor,
Department of Computer
Science & Engineering,
BUET, Dhaka.

Member

[illegible]

5. Dr.-Ing. M. Anwarul Azim
Professor and Head,
Department of Industrial &
Production Engineering,
BUET, Dhaka.

Member

CERTIFICATE

This is to certify that this thesis work has been done by me under the guidance of Dr. M. Kaykobad and Dr. A. K. M. Mostafa Kamal and it has not been submitted elsewhere for the award of any degree or diploma.

Countersigned	Countersigned	Signature of the Candidate
[illegible]	[illegible]	Siddique
(DR. M. KAYKOBAD)	(DR. A.K.M. MOSTAFA KAMAL)	(M.N.H.SIDDIQUE)
Supervisor	Co-supervisor	Roll no. 921803
		BUET, Bangladesh

ABSTRACT

Most industrial applications require a robot to move materials, parts etc. from one place to another within the workspace of the robot. Again there may be multiple objects within the work space. In such case an obstacle free trajectory planning system must be developed with the robotic system. The subject of obstacle-free trajectory planning is relatively new. Within the past few years only a handful people have been actively working on this topic. This thesis describes trajectory planning system for a three-axis articulated robot. Here a linear interpolation technique is used to generate the trajectory from pick point to place point. For solution of the arm equation a simple geometric approach is used. As a practical demonstration, the developed trajectory planning system has been applied on a Fischertechnik robot to grasp a specified chessman and move it from one square to another on the chessboard.

ACKNOWLEDGEMENT

It is a matter of pleasure for the author to acknowlwdgement his profound gratitude to his supervisor, Dr. M. Kaykobad, Associate Professor, Department of Computer Science and Engineering, BUET, for his advice, valuable guidance and constant encouragement throughout the endeavour.

The author would like to express his gratitude to his co-supervisor Dr. A. K. M. Mostafa Kamal, Assistant Professor, Department of Industrial and Production Engineering, BUET, for his cincere guidance, constant supervision, and valuable suggestins at all stages in conducting the work without whose help this work could have not been successful.

The author is indebted to Dr. Md. Shamsul Alam, Professor and Head, Department of Computer Science and Engineering, BUET, for his cooperation to provide the Fischertechnik Robot, Computer and other facilities to complete the work.

Thanks are due to the Department of Industrial and Production Engineering, BUET, to use their Robots which enabled the author gathering relevant experiences and enriched the research work.

The author is thankful to Prof. Dr.-Ing. M. Anwarul Azim, Professor and Head, Department of Industrial and Production Engineering, BUET, and Abdus Sattar, Assistant Professor, Department of Computer Science and Engineering, BUET for serving as members of the board of examiners.

Dedication

To my parents

TABLE OF CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol/ Abbreviation	Meaning
TCP	- Tool Center Point
WCS	- World Coordinate System
JCS	- Joint Coordinate System

LIST OF TABLES

LIST OF FIGURES

Chapter One

Introduction

1. Introduction

The twentieth century has witnessed the emergence of a number of revolutionary technologies. One of the most versatile amongst them has been robotics. The productivity, quality, and related economics offered by a modern robot is so persuasive that a recent trend to compete in the global market is to getting the factories robotized [34]. In Bangladesh Robotics technlogy has not received any attention at all in proportion to its importance. Although, in this age the use of robots in manufacturing has been too pervasive to avoid for any purpose. Under these circumstances there has been a feeling of extreme urgency even in many developing countries to invert themselves in this industrial revolutions. Therefore, the survival and growth of its industries would depend on its ability to harness the technological innovations like robotics and automation that would increase the productivity, quality, and cost-effectiveness. The department of computer science and engineering, BUET has got a three axis articulated robot which has been assembled and installed very recently. Hence, this field of technology deserves intensive work.

Installation of heavy robots in small scale productions is sometimes not cost effective. Robots of higher-degree-of-freedom are also very expensive to install in the industries. But robots with single, or two-degree-of freedom can be implemented to perform light work like pick and place operations in the industry will not be expensive moreover will increase the rate of production with consistent quality. This research work is directed to the task planning (i.e. motion planning and control) of a robot manipulator.

1.1 Objective of the Thesis

Nowadays robots are in wide use in industries. Industrial

application require the manipulator to move either from fixed position to a destination position along a preplanned path.
The robot cannot pick an object from an unspecified location and place it elsewhere. Also the time required to perform such operations depends on the trajectory followed by the robot's manipulator. Moreover, in this sort of application there could be a single object or multiple objects in the robot's workspace. If there are multiple objects, the system has to generate an obstacle free path.

This research work will develop a motion planning system which can guide and control the robot manipulator to pick an object from some predefined positions in the robot's workspace containing multiple objects and then to place the object in a desired position.

As a practical demonstration, the developed system will be used to grasp a specified chessman and move it from one square to another on the chessboard.

A detail study of the coordinate system will be made for the choice of the coordinate system. Through which position of chessman will be expressed, a number of possible trajectories between two given end points can exist. The manipulator may move along a smooth, polynomial trajectory that satisfies the position and orientation of the two end-points. A detail study of the formalism for joint-interpolated and straight-line path trajectory will be made.

For the position and orientation of the end-effector an inverse kinematic solution is required. In general , the inverse kinematics problem can be solved by various methods, such as inverse transform, screw algebra, dual matrices, dual quaternion, iterative and geometric approaches. A detail study will be made for ascertaining a suitable method.

1.3 Organization of the Thesis

The thesis is organized in seven chapters. The chapters are organized as follows:

Chapter I describes an introduction to the work and objectives of the thesis.

Chapter II discusses the basic terminology of robotics, the components and structure of different robots and its workspace. It also focuses the accuracy and repeatability of different manipulators.

Chapter III deals with the analytical study of the geometry of motion of a robot arm with respect to fixed reference coordinate system as a function of time without regard to the forces and moments that cause the motion. The direct kinematic analysis provides the basis for solving the inverse kinematics problem. The solution of the inverse kinematics problem becomes the basis of the interpolator algorithm which guides the end effector along the desired trajectory in space.

Chapter IV focuses attention on various trajectory planning schemes for obstacle free motion. It also deals with the formalism of describing the desired manipulator motion as sequence of points in space (Position and orientation of the manipulator) through which the manipulator must pass, as well as the space curve that it traverses.

Chapter V describes planning of the obstacle-free trajectory for pick and place operation. It deals with the most fundamental operation of robotic manipulator. Determination of the four discrete points, pick, lift-off, set-down and place points, of the pick-and-place trajectory is discussed. The algorithm of the pick-

and-place trajectory is developed in this chapter.

In chapter VI we discussed the development of the chess application in which mapping of the chessboard, calculation of the joint angles are given in detail. The control system of the Fischertechnik, hardware and software are also discussed in brief.

Chapter VII describes some features of the developed system that can be carried out in future. Also some conclusions are drawn in this chapter.

Chapter Two

Robotics: An Introduction

2. Robotics : An Introduction

The popular concept of a robot is a human like machine that can perform tasks with the apparent intelligence of human beings. According to the Robot Institute of America, " a robot is a reprogramable multifunctional manipulator designed to move material, parts, tools, or specialized devices through variable programmed motions for the performance of a variety of tasks".

Even this restricted version of a robot has several features that make it attractive in an industrial environment. Among the advantage often cited in favor of the introduction of robots are decreased labor costs, increased precision and productivity, increased flexibility compared with specialized machines, and more human working conditions as dull, repetitive, or hazardous jobs are performed by robots.

2.1 Components

An industrial robot consists of five major components:

i) A manipulator consisting of links and joints. The number of degree of freedom depends on the number of joints which can be either revolute (rotary) or prismatic (sliding). A revolute joint is like a hinge and allows relative rotation between two links. A prismatic joint allows a linear motion between two links. We use the convention (R) for representing revolute joints and (P) for prismatic joints.

Each joint represents the connection between two links. The joint variables, denoted by θ_i for a revolute joint and d_i for a prismatic joint, and represent the relative displacement between adjacent links.

ii) At the end of the manipulator there is an end-effector. Typical end-effectors are grippers, claws, welding guns, vacuum pickups, and similar devices depending upon the specific application of the robot.

iii) Drive systems to provide motive power to manipulator. Typical drive systems are electric motors, hydraulic and pneumatic motors or actuators.

iv) A controller to direct and sequence the movement of joints and end-effector. Invariably a digital computer or a microprocessor is used as a controller.

v) Sensors to measure position, velocity, force, torque, proximity, temperature and such other factors.

The number of joints determine the degree-of-freedom (DOF) of the manipulator. Typically, a manipulator should have at least six degree-of-freedom, three for position and three for orientation to reach and grasp an object from every point in its working environment. A manipulator having more than six degree of freedom is referred to as redundant manipulator which is able to reach around or behind obstacles in a working environment. But, the difficulty of controlling a manipulator increases rapidly with the number of links.

The workspace of a manipulator is the total volume swept out by the end-effector as the manipulator execute all possible motions. The workspace is constrained by the geometry of the manipulator as well as mechanical constraints on the joints. For example, a revolute joint be limited to less than a full 360^0 of motion. The workspace is often divided into reachable workspace and dexterous workspace. The reachable workspace is the entire set of points that is reachable by the manipulator. The dexterous workspace consists of the set of points that the manipulator can reach with an arbitrary orientation of the end-effector. Indeed, the dexterous workspace is a subset of the reachable workspace.

2.2 Accuracy and Repeatability

The accuracy of a manipulator is a measure of how close the manipulator can come to a given point within its workspace. Accuracy is closely related to spatial resolution, since the robot's ability to reach a particular point in space depends on its ability to divide its joint movement into small increments. There is typically no direct measurement of the end-effector position and orientation. One must rely on the assumed geometry of the manipulator and its rigidity to infer i.e. to calculate the end-effector position from measured joint angles. The primary method of sensing position errors in most cases is with position encoders located at the joints, either on the shaft of the motor that actuates the joint or on the joint itself. The final accuracy of robotic system depends on computational errors, mechanical inaccuracy in the construction of the manipulator, the controller resolution and flexibility effects such as the bending of links under gravitational and other loads, gear backlash, and a host of other static and dynamic effects. It is primarily for this reason that robots are designed with extremely high rigidity.

Repeatability refers to the robot's ability to position its wrist end back to a point in space that was previously taught. In other words, if a robot joint is instructed to move by the same angle from certain point a number of times, all with equal environmental conditions, it will be found that the resultant motions lead to differing displacements. As the robot is instructed to return to the same position in subsequent work cycles, it will not always return to that point, but instead will form a cluster of positions about that point. In general, repeatability will be better than accuracy. Mechanical inaccuracies in robot's arm and wrist components are principal sources of repeatability errors. Most robot manufacturers provide a numerical value for repeatability rather than the accuracy of their robots.

Rotational axes usually result in a large amount of kinematic and dynamic coupling among the links with a resultant accumulation of errors and a more difficult control problem.

One may think then what the advantages of revolute joint are in manipulator design. The answer lies primarily in the increased dexterity and compactness of revolute joint designs. The revolute joint manipulators are better abe to maneuver around obstacles and have wider range of possible applications.

2.3 Kinematic Arrangements

Robot manipulators can be classified by several criteria, such as their geometry, or kinematic structure, the type of application for which they are designed, the manner in which they are controlled etc. Most industrial manipulators at present have six or fewer degrees of freedom. These manipulators are usually classified kinematically on the basis of the arm or first three joints. The majority of these manipulators fall into one of the following geometric types:

- Articulated
- Spherical
- SCARA
- Cartesian
- Cylindrical

Articulated Configuration

The articulated manipulator is also called a revolute manipulator. Two common revolute designs are the elbow type and parallelogram linkage. In these arrangements joint axis z_2 is parallel to z_1 and both z_2 and z_1 are perpendicular to z_0. The structure and terminology associated with the elbow manipulator are shown in figure 2.1. Its workspace is shown in the figure 2.2. This configuration provides for relatively large freedom of movement in a compact space.

Spherical Configuration

By replacing the third or elbow joint in the revolute configuration by a prismatic joint one obtains the spherical configuration. The term spherical configuration derives from the fact that the spherical coordinate defining the position of the end-effector with respect to a frame whose origin lies at the intersection of the axes z_1 and z_2 are the same as the first three joint variables. A common manipulator with this configuration is the Stanford manipulator.

SCARA Configuration

The so-called SCARA is an abbreviation of Selective Compliant Articulated Robot for Assembly. The SCARA configuration is a recent and increasingly popular configuration. Unlike the spherical design which has z_0, z_1, z_2 mutually perpendicular, the SCARA has z_0, z_1, z_2 parallel.

Cartesian Configuration

A manipulator whose first three joints are prismatic is known as Cartesian manipulator. For the Cartesian manipulator the joint variables are the Cartesian coordinates of the end-effector with

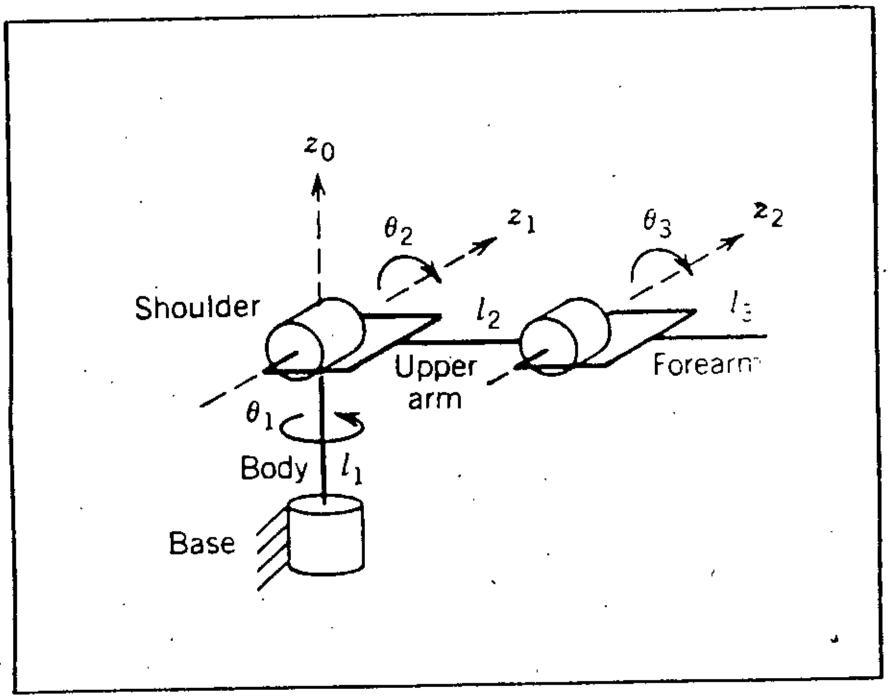

Figure 2.1 Structure of an articulated robot

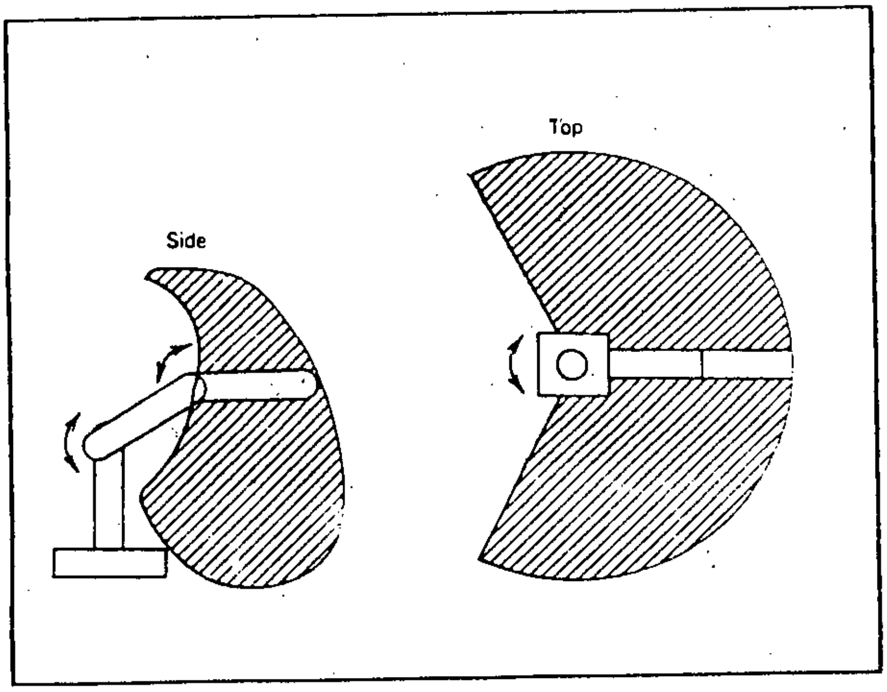

Figure 2.2 Work space of an articulated robot

respect to the base. As might the kinematic description of this manipulator is the simplest of all configuration. Cartesian configurations are useful for table-top assembly applications.

Cylindrical Configuration

If the first joint of a Cartesian-coordinate robot is replaced with a revolute joint, this produces a cylindrical-coordinate robot. The revolute joint swings the arm back and forth about a vertical base axis. As the name suggests, the joint variables are the cylindrical coordinates of the end-effector with respect to the base.

There are other ways of classifying robots. They are by power source, application area, and method of control.

Chapter Three

Kinematics

3. Kinematics

Robot arm kinematics deals with the analytical study of the geometry of motion of a robot arm with respect to a fixed reference coordinate system as a function of time without regard to the force and moment that cause motion. This chapter deals with two fundamental problems of both theoretical and practical interest in robot kinematics: direct kinematics and inverse kinematics.

3.1 The Direct Kinematics

A robotic arm consists of several mechanical joints. To define a position and orientation of the tool center point of the end-effector in three dimensional space an expression is to be determined as a function of joint angles. Since the links of the robot arm rotates and/or translates with respect to a reference coordinate, a relative coordinate or body attached frame is established along the joint axis for each link. A transformation is needed for setting up the correspondence between the elements of two coordinate systems that are rotated and/or translated relative to each other. The position and orientation can be expressed in each of these coordinate systems. When vector transformation is considered, the projections of the vector are transferred [3],[5].

There are several transformation techniques.

- Rotation matrix technique
- Quaternion and Rotation vectors
- Homogeneous transformation matrices technique

3.1.1 Rotation Matrix

In order to specify the position and orientation of the end-effector in terms of a coordinate system attached to the fixed base, coordinate transformations involving both rotations and translations are required.

Let us consider two coordinate systems, shown in figure 3.1, namely, the OXYZ coordinate system with OX, OY, OZ as its coordinate axes and the OUVW coordinate system with OU, OV, OW as its coordinate axes. Both coordinate systems have their origin at point O. OXYZ coordinate system is fixed in the three-dimensional space and considered to be the referenced coordinate. OUVW coordinate system is rotating with respect to the reference coordinate system.

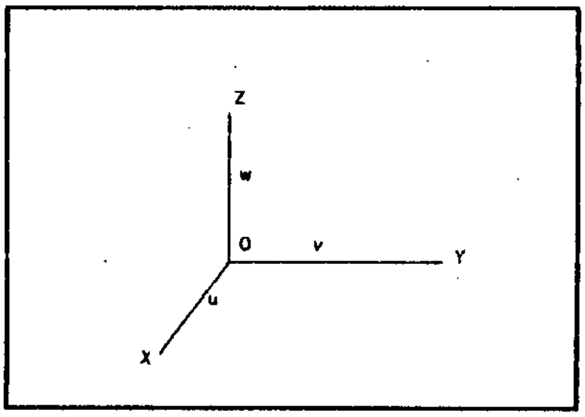

Figure 3.1 Two coordinate systems

A point P can be represented by its coordinates with respect to the either coordinates OXYZ and OUVW as

$$\begin{aligned} P_{uvw} &= (P_u, P_v, P_w)^T \\ P_{xyz} &= (P_x, P_y, P_z)^T \end{aligned} \qquad (3.1)$$

where P_{xyz} and P_{uvw} represent the same point in the space.

We are interested in finding a rotation matrix R that will transform the coordinates of P_{uvw} to the coordinates expressed with respect to the OXYZ coordinate system, after OUVW coordinate system has been rotated. That is,

$$P_{xyz} = R\, P_{uvw} \tag{3.2}$$

Thus using the definition of vector scaler product, we obtain

$$\begin{aligned} P_x &= I_x.I_u P_u + I_x.J_v P_v + I_x.K_w P_w \\ P_y &= J_y.I_u P_u + J_y.J_v P_v + J_y.K_w P_w \\ P_z &= K_z.I_u P_u + K_z.J_v P_v + K_z.K_w P_w \end{aligned} \tag{3.3}$$

It can be expressed in matrix form

$$\begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} I_x.I_u & I_x.J_v & I_x.K_w \\ J_y.I_u & J_y.J_v & J_y.K_w \\ K_z.I_u & K_z.J_v & K_z.K_w \end{bmatrix} \begin{bmatrix} P_u \\ P_v \\ P_w \end{bmatrix} \tag{3.4}$$

Using this notation, the matrix R will have the following representation

$$R = \begin{bmatrix} I_x.I_u & I_x.J_v & I_x.K_w \\ J_y.I_u & J_y.J_v & J_y.K_w \\ K_z.I_u & K_z.J_v & K_z.K_w \end{bmatrix} \tag{3.5}$$

Similarly, we can obtain the coordinates of Puvw from the coordinates of Pxyz

$$P_{uvw} = R'\, P_{xyz} \tag{3.6}$$

where R' can be expressed in the same way as R is represented

$$R = \begin{bmatrix} I_u . I_x & I_u . I_y & I_u . I_z \\ J_v . I_x & J_v . I_y & J_v . I_z \\ K_w . I_x & K_w . I_y & K_w . I_z \end{bmatrix} \tag{3.7}$$

Our goal is to find the rotation matrix that represents rotation of the moving coordinate system about the principal axes of the fixed reference coordinate system. Let OUVW coordinate system be rotated α angle about OX axis to arrive at a new location transformation matrix $R_{x,\alpha}$ can be derived from the above concept as follows

$$P_{xyz} = R_{x,\alpha} \, P_{uvw} \tag{3.8}$$

with Ix=Iu and where

$$R_{x,\alpha} = \begin{bmatrix} I_x . I_u & I_x . J_v & I_x . K_w \\ J_y . I_u & J_y . J_v & J_y . K_w \\ K_z . I_u & K_z . J_v & K_z . K_w \end{bmatrix} \tag{3.9}$$

Thus using the vector scaler product, we get the first basic rotation matrix as follows:

$$R_{x,\alpha} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{3.10}$$

A similar analysis can be used to derive expression for the second and third basic rotation matrix about OY axis with ß angle and about OZ axis with Θ angle.

$$R_{y,\beta} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \qquad (3.11)$$

$$R_{z,\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (3.12)$$

The matrices Rx,α, Ry,ß and Rz,Θ are basic rotation matrices from which other finite rotation matrices can be derived.

3.1.2. Composite Rotation Matrix

To establish an arbitrary orientation for the end effector, we need multiple rotations of basic form which is termed as composite rotation matrix, and it is obtained by multiplying a number of basic rotation matrices together. The rotation can be about the principal axis of the reference coordinate system as well as about its own principal axes.

Since, the operation of matrix multiplication is not commutative the order in which the basic rotation matrices are performed makes difference in resulting composite rotation. Therefore, distinct rules must be given as to how a composite rotation matrix should be constructed, and the following rules are used [4], [5].

1. Initially both coordinate systems are coincident , hence the rotation matrix is a 3x3 identity matrix. Initialize the rotation matrix to $R = I_3$.

2. If the rotating coordinate system is to be rotated about one of the principal axes of the reference coordinate system, then premultiply the previous rotation matrix with an appropriate basic rotation matrix.

3. If the rotating coordinate system is rotating about its own principal axes, then postmultiply the previous rotation matrix with an appropriate basic rotation matrix.

Using the above rules we can find the composite rotation matrix representing a rotation of α angle about OX axis followed by rotation of β angle about OY axis followed by a rotation of Θ angle about OZ axis.

$$R = R_{z,\theta}\, R_{y,\beta}\, R_{x,\alpha} \tag{3.13a}$$

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{3.13b}$$

$$= \begin{bmatrix} C\theta C\beta & C\theta S\beta S\alpha - S\theta C\alpha & C\theta S\beta C\alpha + S\theta S\alpha \\ S\theta C\beta & S\theta S\beta S\alpha + C\theta C\alpha & S\theta S\beta C\alpha - C\theta S\alpha \\ -S\beta & C\beta S\alpha & C\beta C\alpha \end{bmatrix}$$

where $C\Theta = \cos\Theta$, $C\beta = \cos\beta$, $C\alpha = \cos\alpha$

$S\Theta = \sin\Theta$, $S\beta = \sin\beta$, $S\alpha = \sin\alpha$

3.1.3. Quaternion and Rotation Vectors

The advantages of these methods are that the definition of the relative orientation consists of three or four parameters instead of nine (3x3 matrix) as in rotation matrix method, and the number of arithmetic operations required for kinematic solution of a robot manipulator can be decreased using the quaternion rotation vector methods.

The Quaternion is a four element vector. Three of which are components of a special vector and define the direction of the common rotation axis, and the fourth component provides information about the rotation angle.

Thus the quaternion Q has four elements which completely identify the direction of the axis of rotation and the amount of rotation. It consists of scaler element Q0 and a vector Q and can be written as

$$Q = q_0 + q = q_0 + q_1\hat{i} + q_2\hat{j} + q_3\hat{k} \qquad (3.14)$$

while i^, j^ and k^ are unit vectors along the X, Y, and Z axes respectively, and the scaler element Q0 equals to zero.

The quaternion which is used to identify relative rotation by an angle Q about an axis, whose direction is defined by the unit vector e^, was defined by Euler in [3] and [4].

$$\begin{aligned} Q &= \cos\frac{\theta}{2} + \hat{e}\sin\frac{\theta}{2} \\ Q &= \cos\frac{\theta}{2} + (e_1\hat{i} + e_2\hat{j} + e_3\hat{k})\sin\frac{\theta}{2} \end{aligned} \qquad (3.15)$$

where e_1, e_2 and e_3 are direction cosines of e^. We can write the quaternion as

$$Q = \cos\frac{\theta}{2} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \sin\frac{\theta}{2} \tag{3.16}$$

Vector Transformation Using the Quaternion

Let us assume that a vector V in frame O undergoes a rotation which is described by the quaternion Q in frame O. The resulting vector in frame O is V'. The relation between V and V' is given by the following equation [3].

$$v' = v + 2q_0(q \times v) + 2q \times (q \times v) \tag{3.17}$$

The above equation is also interpreted as a coordinate transformation, which means a description of the vector components in two frames with different orientation.

Let us consider a vector V in frame O in figure 3.2 shown below.

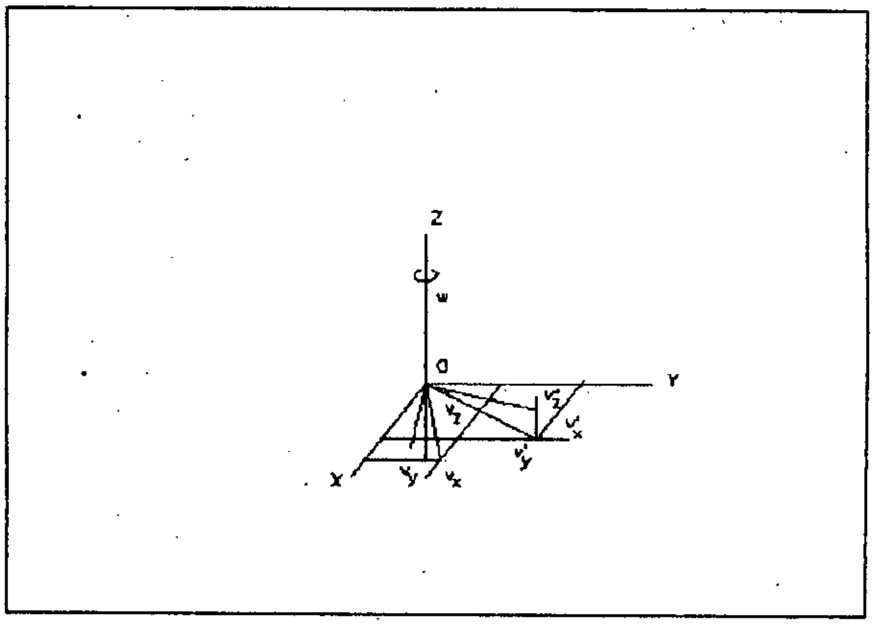

Figure 3.2 Rotation of a vector about the Z-axis

The frame undergoes a rotation defined by Q. The axis of rotation coincides with the Z axis, thus the quaternion definition of this rotation is as follows:

$$Q = \cos\frac{\theta}{2} + \begin{bmatrix}0\\0\\1\end{bmatrix}\sin\frac{\theta}{2} \tag{3.18}$$

where Q is the angle of rotation around the Z axis. The vector rotation results in a vector V' in frame O and is given by using quaternion multiplication as follows [4]:

$$\begin{bmatrix}v_x'\\v_y'\\v_z'\end{bmatrix} = \begin{bmatrix}v_x\\v_y\\v_z\end{bmatrix} + 2\cos\frac{\theta}{2}\begin{bmatrix}0\\0\\1\end{bmatrix} \times \begin{bmatrix}v_x\\v_y\\v_z\end{bmatrix}\sin\frac{\theta}{2} + 2\begin{bmatrix}0\\0\\1\end{bmatrix} \times \left(\begin{bmatrix}0\\0\\1\end{bmatrix} \times \begin{bmatrix}v_x\\v_y\\v_z\end{bmatrix}\right)\sin^2\frac{\theta}{2} \tag{3.19a}$$

$$\begin{bmatrix}v_x'\\v_y'\\v_z'\end{bmatrix} = \begin{bmatrix}v_x\\v_y\\v_z\end{bmatrix} + 2\cos\frac{\theta}{2}\sin\frac{\theta}{2}\begin{bmatrix}-v_x\\v_y\\0\end{bmatrix} + 2\sin^2\frac{\theta}{2}\begin{bmatrix}-v_x\\-v_y\\0\end{bmatrix} \tag{3.19b}$$

$$\begin{bmatrix}v_x'\\v_y'\\v_z'\end{bmatrix} = \begin{bmatrix}v_x\cos\theta - v_y\sin\theta\\v_y\cos\theta + v_x\sin\theta\\v_z\end{bmatrix} \tag{3.19c}$$

3.1.4. Homogeneous Transformation Matrix

The most general relationship between the two coordinate systems $O_0X_0Y_0Z_0$ and $O_1X_1Y_1Z_1$ as shown in figure 3.3 can be expressed as the

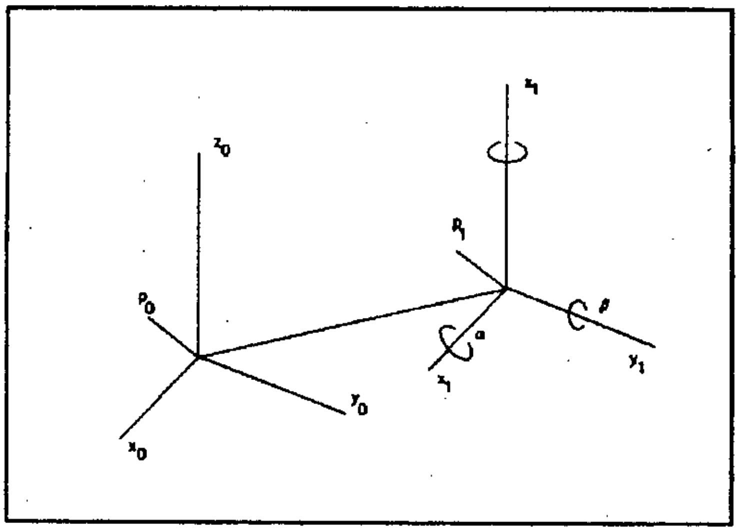

Figure 3.3 Translated frame

combination of a pure rotation and a pure translation. Pure rotations are sufficient to characterize the orientation of a robotic manipulator but to characterize the position of the manipulator relative to a coordinate system attached to robot base, translation is used. It is not possible to represent the translation with a 3x3 matrix. In robotics, we normally use a scale factor of $\sigma = 1$ for convenience. Thus four dimensional homogeneous coordinates are obtained from three dimensional physical coordinates by simply augmenting the vector with a unit fourth component [3], [4], [5],[8]. A homogeneous transformation matrix can be considered to consists of the following four submatrices

$$T \triangleq \begin{bmatrix} R_{3\times3} & P_{3\times1} \\ \eta^{T}_{1\times3} & \sigma_{1\times1} \end{bmatrix} \tag{3.20}$$

$$T = \begin{bmatrix} \text{Rotation matrix} & \text{Position vector} \\ \text{Perspective transformation} & \text{Scaling} \end{bmatrix}$$

Here the 3x3 submatrix R in the upper left corner of T is a rotation matrix. H represents the orientation of the mobile coordinate frame relative to the fixed reference frame. The upper right 3x1 submatrix P is the position vector which represents the translation. It specifies the position of the origin of the mobile coordinate frame relative to the fixed reference frame. The lower left 1x3 submatrix n^T is a perspective vector and represents perspective transformation which is used for computer vision and camera calibration. Here, the elements of this matrix are set to zero to indicate null perspective transformation. The scaler σ in the lower right corner of T is a nonzero scale factor which is typically set to unity.

If a physical point P in a three dimensional space is expressed in terms of homogeneous coordinates and we want to change from one coordinate to frame we extend the 3x3 rotation matrix to a 4x4 homogeneous transformation matrix. For pure rotation equations (3.10), (3.11), (3.12) expressed as rotation matrices, become

$$T_{x,\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.21)$$

$$T_{y,\beta} = \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.22}$$

$$Y_{z,\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.23}$$

The equations (3.21), (3.22), and (3.23) are called the basic homogeneous rotation matrices.

The 3x1 position vector of the homogeneous transformation matrix has the effect of translating the $O_1X_1Y_1Z_1$ coordinate system which has axes parallel to the reference coordinate system $O_0X_0Y_0Z_0$. The basic homogeneous translation matrices are as follows

$$T_{x,dx} = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.24}$$

$$T_{y,dy} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.25}$$

$$T_{z,dz} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.26}$$

A 4x4 homogeneous transformation matrix maps a vector expressed in homogeneous coordinates with respect to $O_1X_1Y_1Z_1$ coordinate system to the reference coordinate system $O_0X_0Y_0Z_0$. That is

$$P_1 = T_0^1 P_0 \tag{3.27}$$

The most general homogeneous transformation that we will consider may be written as [4], [8].

$$T = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n & s & a & p \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.28}$$

In equation (3.28) $n = (n_x \; n_y \; n_z)^T$ is a vector representing the direction of the O_1X_1 axis in the $O_0X_0Y_0Z_0$ system, $s = (s_x \; s_y \; s_z)^T$ represents the direction of the O_1Y_1 axis, and $a=(a_x \; a_y \; a_z)^T$ represents the direction of O_1Z_1 axis. The vector $P = (P_x \; P_y \; P_z)^T$ represents the vector from origin O_0 to the origin O_1 expressed in the $O_0X_0Y_0Z_0$ frame.

3.1.5 Composite Homogeneous Transformation Matrix

A homogeneous transformation represents both a rotation and translation of a mobile coordinate system with respect to a fixed reference coordinate system. A sequence of individual rotation and translation matrices can be multiplied together to obtain a composite homogeneous transformation matrix. Since matrix multiplications are not commutative operations, the order in which the rotations and translations are to be performed is important. Furthermore, the mobile coordinate system can rotate or translate about the unit vectors. Therefore, the effects of the different operations on the composite homogeneous transformation matrix are summarized as following rules [4], [5], [9].

1. Initially both coordinate system are coincident, hence the transformation matrix is a 4x4 identity matrix, therefore, initialize it to T = I.

2. Represent rotations and transitions using separate homogeneous transformation matrices.

3. Represent composite rotations as separate fundamental homogeneous rotation matrices.

4. If the mobile coordinate system is to be rotated about or translated along a unit vector of the fixed reference coordinate, then premultiply the homogeneous transformation matrix T by the appropriate basic homogeneous rotation or translation matrix.

5. If the mobile coordinate system is to be rotated about or translated along one of its own coordinate system, then postmultiply the homogeneous transformation matrix T by the appropriate basic homogeneous rotation or translation matrix.

Thus composite homogeneous transformation matrices are built up starting with the identity matrix and a rotation of α angle about OX axis, followed by rotation of β angle about OY axis followed by a rotation of Θ angle about OZ axis.

$$T_R = T_{z,\theta}\, T_{y,\beta}\, T_{x,\alpha} \tag{3.29a}$$

$$T_R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.29b}$$

$$T_R = \begin{bmatrix} \cos\theta\cos\beta & \cos\theta\sin\beta\sin\alpha - \sin\theta\cos\alpha & \cos\theta\sin\beta\cos\alpha + \sin\theta\sin\alpha & 0 \\ \sin\theta\cos\beta & \sin\theta\sin\beta\sin\alpha + \cos\theta\cos\alpha & \sin\theta\sin\beta\cos\alpha - \cos\theta\sin\alpha & 0 \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The homogeneous translation matrix for the three-dimensional space is

$$T_T = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.30}$$

Now the composite homogeneous transformation matrix can be obtained by premultiplying the composite relation matrix by the composite translation matrix.

$$T = T_T\, T_R \tag{3.31a}$$

$$T = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C\theta C\beta & C\theta S\beta S\alpha - S\theta C\alpha & C\theta S\beta C\alpha + S\theta S\alpha & 0 \\ S\theta C\beta & S\theta S\beta S\alpha + C\theta C\alpha & S\theta S\beta C\alpha - C\theta S\alpha & 0 \\ -S\beta & C\beta S\alpha & C\beta C\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.31b)$$

$$T_R = \begin{bmatrix} \cos\theta\cos\beta & \cos\theta\sin\beta\sin\alpha - \sin\theta\cos\alpha & \cos\theta\sin\beta\cos\alpha + \sin\theta\sin\alpha & dx \\ \sin\theta\cos\beta & \sin\theta\sin\beta\sin\alpha + \cos\theta\cos\alpha & \sin\theta\sin\beta\cos\alpha - \cos\theta\sin\alpha & dy \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha & dz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The homogeneous transformation matrix can be considered to consists of the four submatrices as in equation (3.20).

$$T = \begin{bmatrix} & R & & P \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.32)$$

where P = T_T and R = T_R with

$$R = T_R = \begin{bmatrix} C\theta C\beta & C\theta S\beta S\alpha - S\theta C\alpha & C\theta S\beta C\alpha + S\theta S\alpha \\ S\theta C\beta & S\theta S\beta S\alpha + C\theta C\alpha & S\theta S\beta C\alpha - C\theta S\alpha \\ -S\beta & C\beta S\alpha & C\beta C\alpha \\ 0 & 0 & 0 \end{bmatrix}$$

$$P = T_T = \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix}$$

The geometric interpretation of the homogeneous transformation matrix T is that it provides the orientation and position of a mobile coordinate system with respect to a fixed reference coordinate system. In robotic manipulator analysis of these two coordinate systems can be assigned to two consecutive links of a robot arm, say link i-1 and link i respectively. The link i-1 coordinate system is the fixed reference coordinate and the link i coordinate is the mobile coordinate system. Using the homogeneous transformation matrix we can specify a point P_i in the mobile coordinate i in terms of the fixed reference coordinate i-1.

$$P_{i-1} = T_{i-1}^{i} P_i \tag{3.33}$$

where T_{i-1}^{i} = 4x4 homogeneous transformation matrix relating two coordinate i and i-1.

P_i = 4x1 augmented position vector representing a point in the link i coordinate system.

P_{i-1} = 4x1 augmented position vector representing a point in the link i-1 coordinate system.

Sometimes, we also want to express the position and orientation of a point in the fixed reference coordinate system in terms of the mobile coordinate system.

$$P_i = \left(T_{i-1}^{i}\right)^{-1} P_{i-1} \tag{3.34}$$

Then, we need to find the inverse of the homogeneous transformation matrix T. Since the inverse of a rotation submatrix is equivalent to its transpose [4],[5] and hence the inverse of the homogeneous transformation matrix is as in equation (3.35).

$$\left(T_{i-1}^{i}\right)^{-1} = \begin{bmatrix} R^T & -R^T P \\ 0\ 0\ 0 & 1 \end{bmatrix} \tag{3.35}$$

3.1.6 Joint and Link Parameters

A robotic arm can be modeled as sequence of rigid links interconnected by the revolute of prismatic joints. Each joint link pair constitutes one degree of freedom. For a n-degree of freedom or n-axis robotic arm, there are n joint-link pairs. The link Θ is attached to a base where a reference coordinate frame is assigned and the remaining links are assigned to one coordinate frame, and the last with the tool or end-effector. The joint and links are numbered outwardly from the base. The relative position and orientation of two successive links can be specified by two joint parameters as shown in figure 3.4.

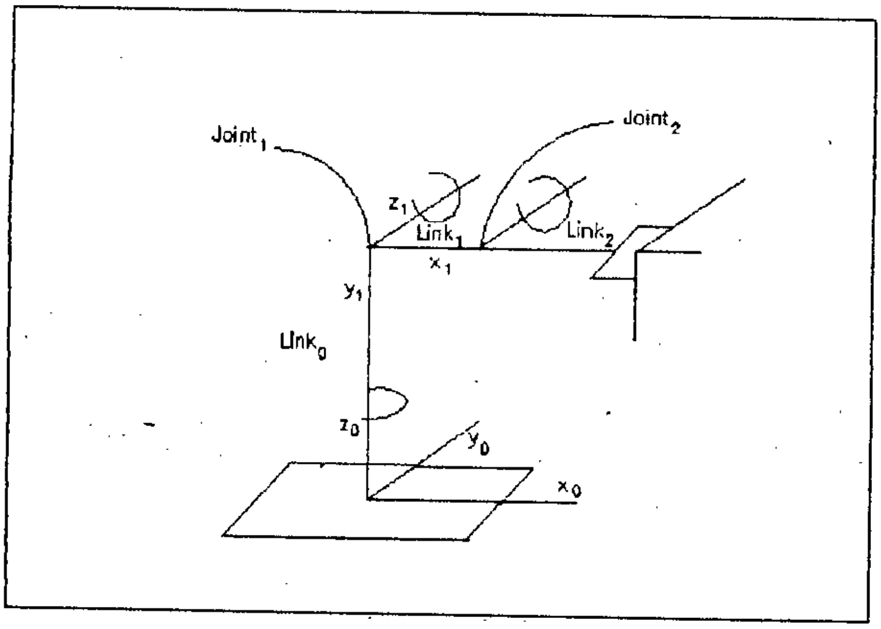

Figure 3.4 Successive links and joint parameters

Only six different joints are possible: revolute, prismatic, cylindrical, spherical, screw and planar. Of these, only revolute and prismatic joints are common in robotic manipulators. These types of joints and their parameters are described in [4] briefly.

A joint axis is established at the connection of two links. The relative position of two such connected links is given by d_i which is the distance measured along Θ_i between the normals. The joint angle θ_i between the normals is measured in a plane normal to the joint axis. Hence, the joint parameter d_i, called the joint distance, represents the translation along the axis of joint i and Θ_i, called the joint angle, represents the rotation about the axes of joint i.

A link is connected to two other links. Thus two joint axes are established at both ends of the connection. The relative position

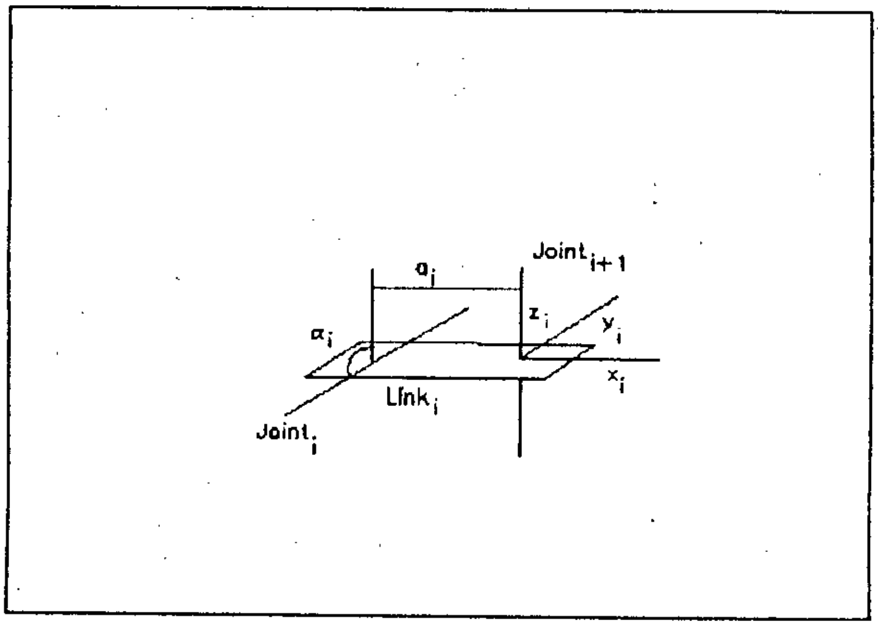

Figure 3.5 Link length a and link twist angle α

and orientation of the axes of the successive joints can be specified by two link parameters which is shown in figure 3.5. The structure of the link can be determined by a_i, called the link length, is the shortest distance measured along the common normal between the axes of joints i and i+1 and by α_i, called link twist angle, is the angle between the joint axes measured in a plane perpendicular to a_i. The link parameters are always constant and are specified as part of the mechanical design. Thus, these four parameters come in pairs: link parameters (a_i, α_i) determine the structure of the link and the joint parameters (d_i, Θ_i) determine the relative position of the two successive links.

For an N-axis robotic arm, there are 4N kinematic parameters that constitute minimal set to specify the configuration of a robot. For each axis, three of these parameters are fixed and depend on the mechanical design and the fourth parameter is the joint variable. The different values of the parameter for different mechanical designs are briefly discussed in [2]. The variable joint parameter depends on the type of the joint. For a revolute joint, the joint angle Θ_i is variable and the joint distance d_i is fixed. The table 3.1 shows the parameters with different type of joints.

Parameters	Symbol	Revolute joint	Prismatic joint
Joint angle	Θ_i	variable	fixed
Joint distance	d_i	fixed	variable
Link length	a_i	fixed	fixed
Link twist angle	α_i	fixed	fixed

Table 3.1 Kinematic parameters

3.1.7 Denavit - Hartenberg Representation

A systematic technique to establish the translation and rotational relationship for each adjacent links, was proposed by Denavit and Hartenberg (1955). Once these link-attached coordinate systems are assigned, transformation between adjacent coordinate systems can be represented by a 4x4 homogeneous transformation matrix.

An orthonormal Cartesian coordinate system (X^i, Y^i, Z^i) can be assigned to each link i where i=1,2,3,...,n. Each (X^i, Y^i, Z^i) coordinate of a robotic arm corresponding to joint i+1 and is fixed in link i. The base coordinate are defined as the 0th coordinate (X^0, Y^0, Z^0) and the nth coordinate (X^n, Y^n, Z^n) as the tool tip or the end-effector. The coordinate systems are assigned to the links by using the following D-H representation [3], [4], [5], [8].

1. Number the links and joints starting with the base and ending with end-effector. The starting base is denoted as link 0 and the end-effector is link n. Link i moves in respect to link i-1 around (for revolute) or along (for prismatic) joint i. Assign a right handed coordinate system (X^0, Y^0, Z^0) to the robot base and align Z^0 with the axis of joint 1. X^0 and Y^0 axes can be normal to Z^0 axis.

2. Establish link's coordinate system for each of the joints:

a) The Z^{i-1} axis is chosen along the axis of motion of the ith joint.

b) The X^i axis is chosen orthonormal to both Z^{i-1} and Z^i. If Z^i and Z^{i-1} are parallel, point X^i away from z^{i-1}.

c) The Y^i axis is chosen to from a right-handed coordinate system.

3. Define the joint parameters which are the four geometric quantities Θ_i, d_i, a_i and α_i.

a) Compute Θ_i as the angle of rotation from X^{i-1} to X^i measured about Z^{i-1} axis.

b) Compute d_i as the distance from origin of the (i-1)th coordinate system to the intersection of the Z^{i-1} axis and the X^i axis along the Z^{i-1} axis. For a prismatic joint d_i is variable and for a revolute joint d_i is fixed.

c) Compute a_i as the distance from intersection of the Z^{i-1} axis and X^i axis to the origin of the ith coordinate system along the X^i axis.

d) Compute α_i as the angle of rotation from Z^{i-1} measured about X^i axis.

4. Assign the nth coordinate (X^n, Y^n, Z^n) to the end effector. Set K_n = a along the direction Z^{n-1}. Establish the origin conveniently along Z^n, preferably at the center of the gripper or at tool tip. Set J_n = s in the direction of the gripper closure and set i_n = n as n = s x a.

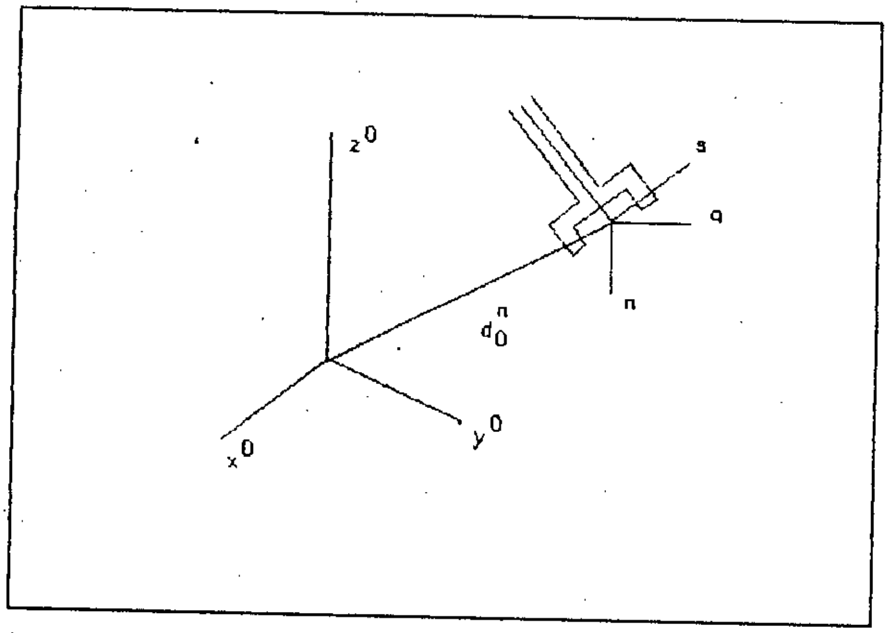

Figure 3.6 Hand coordinate system and [n, s, a]

Once the D-H coordinate system has been established for each link, a homogeneous transformation matrix can easily be developed relating to ith coordinate to (i-1)th coordinate. A point in the coordinate system can be expressed in (i-1)th coordinate system by performing the following transformations [4], [8]:

1. Rotate about the Z^{i-1} axis an angle of Θ_i to align the X^{i-1} axis with the X^i axis.

2. Translate along the Z^{i-1} axis a distance of d_i to bring the X^{i-1} and the X^i axes into coincidence.

3. Translate along the X_i axis a distance of a_i to bring the two origins as well as the x-axis into coincidence.

4. Rotate about the X^i axis an angle of α_i to bring the two coordinate system into coincidence.

Each of these four operations can be expressed by a homogeneous transformation matrix $A_{i-1}{}^{i}$ as a product of four basic transformation matrices. $A_{i-1}{}^{i}$ is known as the D-H transformation matrix for adjacent coordinate system, i and i-1. Thus we obtain

$$A_{i-1}^{i} = T_{R(z,\theta)}\, T_{T(x,d)}\, T_{T(x,a)}\, T_{R(x,\alpha)} \qquad (3.36a)$$

$$A_{i-1}^{i} = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.36b)$$

$$A_{i-1}^{i} = \begin{bmatrix} C\theta_i & -S\theta_i C\alpha_i & S\theta_i S\alpha_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\alpha_i & -C\theta_i S\alpha_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.36c)$$

Using the equations (3.35), we obtain the inverse of this transformation to be

$$\left[A_{i-1}^{i}\right]^{-1} = \begin{bmatrix} \cos\theta_i & \sin\theta_i & 0 & -a_i \\ -\cos\alpha_i\sin\theta_i & \cos\alpha_i\cos\theta_i & \sin\alpha_i & -d_i\sin\alpha_i \\ \sin\alpha_i\sin\theta_i & -\sin\alpha_i\cos\theta_i & \cos\alpha_i & -d_i\cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.37)$$

Where αi, a_i, d_i are fixed and Θ_i is the joint variable for a revolute joint.

3.1.8 The Arm Matrix

Once a D-H transformation matrix is obtained, we can arrive at a composite transformation matrix which transforms end-effector coordinates into robot base coordinates by multiplying successive transformation matrices starting at tool tip and ending of base. This composite homogeneous coordinate transformation matrix is termed as arm matrix. In particular, if T_{base}^{tool} represent a transformation from tool tip coordinate of link n to base coordinate at link 0, then

$$T_{base}^{tool} = A_0^1 A_1^2 A_2^3 \ldots A_{n-1}^{n} = \prod_{j-1}^{1} A_{j-1}^{j} \qquad (3.38)$$

$$\text{with} \quad j=1,2,3,\ \ldots\ n$$

$$T_{base}^{tool} = \begin{bmatrix} R_0^n & & P_0^n \\ & & \\ 0 \; 0 \; 0 & & 1 \end{bmatrix} \qquad (3.39)$$

$$T_{base}^{tool} = \begin{bmatrix} n & s & a & p \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.40)$$

Where n = normal vector of the end-effector,

s = sliding vector of the end-effector,

a = approach vector of the end-effector,

p = position vector of the end-effector.

Once an expression for the arm matrix is available, we can then substitute it into the equation (3.39), called the arm equation.

3.1.9. The Arm Equation of a Three-axis Articulated Robot

A three axis articulated arm with three revolute joints is shown in figure 3.7.

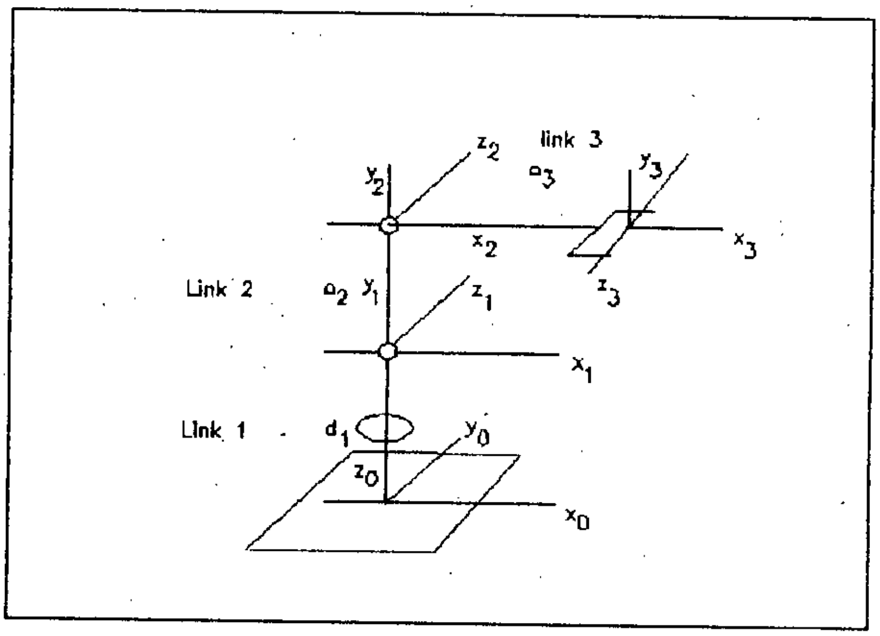

Figure 3.7 Three-axis articulated robot

The corresponding transformation matrices are derived according to the D-H presentation described in section 3.1.7

1. The links are numbered which can be seen from the figure depicted in figure 3.7.

2. Coordinate system are assigned to various links. The Z_i axes point to the joint's axes of rotation. The X_i axes are chosen perpendicular to both Z_{i-1} and Z_i axes.

3. Joint parameters are established. The distance between (X_0Y_0 Z_0) and (X_1 Y_1 Z_1) along Z_0 axis is d_1. The axes Z_0 and Z_1 intersect and therefore $a_1 = 0$. The orientation angle α_1, measured from the Z_0 axis to Z_1 around X_1 is $\alpha_1 = +90^0$. The angle Θ_1, measured from X_0 to X_1, is the variable parameter of joint 1. Similarly other parameters are found that are given in the table 3.2.

Joint	Θ_i	d_i	a_i	α_i
1	Θ_1	d_1	0	$+90^0$
2	Θ_2	0	a_2	0
3	Θ_3	0	a_3	0

Table 3.2 Joint parameters

4.The D-H transformation matrices are determined by substituting the joint parameters from table 3.2.

$$A_0^1 = \begin{bmatrix} C_1 & 0 & S_1 & 0 \\ S_1 & 0 & -C_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_1^2 = \begin{bmatrix} C_2 & -S_2 & 0 & a_2C_2 \\ S_2 & C_2 & 0 & a_2S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.41}$$

$$A_2^3 = \begin{bmatrix} C_3 & -S_3 & 0 & a_3C_3 \\ S_3 & C_3 & 0 & a_3S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $C_i = \cos\Theta_i$ and $S_i = \sin\Theta_i$.

The arm equation obtained by multiplying the successive transformation matrices together, which yields

$$T_0^3 = A_0^1\, A_1^2\, A_2^3 \tag{3.42a}$$

$$T_0^3 = \begin{bmatrix} C_1C_2C_3+C_1S_2S_3 & -C_1C_2S_3-C_1S_2C_3 & S_1 & a_3C_1C_2C_3-a_3C_1S_2S_3+a_2C_1C_2 \\ S_1C_2C_3-S_1S_2S_3 & -S_1C_2S_3-S_1S_2C_3 & -C_1 & a_3S_1C_2C_3-a_3S_1S_2S_3+a_2S_1C_2 \\ S_2C_3+C_2S_3 & C_2C_3-S_2S_3 & 0 & a_3S_2C_3+a_3C_2S_3+a_2S_2+d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.42b)$$

where $C_1 = \cos\Theta_1$, $S_1 = \sin\Theta_1$, $C_2 = \cos\Theta_2$, $S2 = \sin\Theta_2$, $C_3 = \cos\Theta_3$ and $S_3 = \sin\Theta_3$.

3.2 Inverse Kinematics

In the previous section we developed a procedure of the arm equation for given joint variables. The solution to the arm equation is useful, because it provides us with a relationship which explicitly shows the dependence of the tool configuration on the joint variables. But in certain types of tasks like assembly, welding and sealing operations the manipulator's end- effector requires to follow a straight-line path or at least approximated closely. A straight-line trajectory is, in general, formulated in terms of position and orientation of the tool. It is then necessary to determine appropriate values for the joint variables to properly configure the tool. This problem is the inverse kinematic problem. That is, finding the joint variables in terms of tool position and orientation is inverse kinematic problem. In general, the problem of inverse kinematics can be stated as follows:

Given a 4x4 homogeneous transformation matrix (3.43),

$$H = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} \qquad (3.43)$$

find the solution of the equation (3.44)

$$T_0^n(q_1, \ldots, q_n) = H \qquad (3.44)$$

Where T_0^n $(q_1, \ldots, q_n) = A_1 \ldots A_n$.

The equation (3.44) results in 12-nonlinear equations in n-unknown variables, which can be written as

$$T_{ij}(q_1, \ldots, q_n) = h_{ij} \qquad (3.45)$$

with i = 1,2,3 and j = 1,2,3,4.

Where T_{ij} and h_{ij} refer to 12 non-trivial entries of T_0^n and H respectively. Since the bottom row of T_0^n and H are (0 0 0 1), the 4 equations are trivial.

Let the position and orientation of the final frame of the Fischertechnik robot is given as

$$T_0^3 = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.46)$$

To find the corresponding joint variables Q_1, Q_2, and Q_3, we must solve the following simulteneous set of non-linear trigonometric equations.

$$\begin{aligned}
&C_1C_2C_3+C_1S_2S_3 = n_x\\
&-C_1C_2C_3-C_1S_2C_3 = s_x\\
&S_1 = a_x\\
&S_1C_2C_3-S_1S_2S_3 = n_y\\
&-S_1C_2S_3-S_1S_2C_3 = s_y\\
&-C_1 = a_y\\
&S_2C_3+C_2S_3 = n_z \qquad\qquad (3.47)\\
&C_2C_3-S_2S_3 = .s_z\\
&0 = a_z\\
&a_3C_1C_2C_3-a_3C_1S_2S_3+a_2C_1S_2 = p_x\\
&a_3S_1C_2C_3-a_3S_1S_2S_3+a_2S_1C_2 = p_y\\
&a_3S_2C_3+a_3C_2S_3+a_2S_2+d_1 = p_z
\end{aligned}$$

The above equations indicate that the arm matrix T_0^3 is a function of sine and cosine of Θ_1, Θ_2 and Θ_3. Thus we have 12- equations with 3 unknown joint angles. Since we have more equations than unknowns, one can immediately conclude that multiple solutions exist. Furthermore, these equations are difficult to solve directly in closed-form.

In solving the inverse kinematics problem, we are interested in finding a closed-form of the equations which means finding an explicit relationship. Close form solutions are preferable for two reasons:

- In certain applications where location is provided by a vision system where direct kinematic equations must be solved at a rapid rate and having closed form expressions is a practical need.

- The kinematic equations in general have multiple solutions. Having closed form solutions allow one to develope rules for choosing a particular among several.

The inverse kinematics may or may not have a solution. Therefore, it is necessary to analyse the conditions under which solutions to inverse kinematics exist.

3.2.1 Existence of Solutions

The practical question of the existence of solutions to inverse kinematics problem depends on engineering as well as mathmatical considerations. The details of an explicit solution to the inverse kinematics problem depend upon the robot or the class of robots being used. Furthermore, the motion of the revolute joints may be restricted to less than a full 360° degree of rotation so that not all mathematical solutions of kinematic equations will correspond to physically realizable configuration of the manipulator. Once a solution to the mathematical equations is identified, it must be further checked to see whether or not it satisfies all constraints on the range of possible joint variables.

Let us examine the conditions for which the inverse kinematic problem yields a solution. Firstly, if the intended location of the tool centre point is outside the robot's work space, then there exist no solution to the inverse kinematic problem in general. Secondly, if the intended location of the tool is within the workspace, then still there are some tool orientations, which are not atainable without violating the joint variable constraints. Indeed, if the robot has fewer than three degrees of freedom to orient the tool, then whole classes of orientations are unrealizble. This is analyzed in detail in [5]. If we have to find a general solution to the inverse kinematics problem, one for which the joint variables can be found which generates a physically realizable tool configuration, then the number of unknowns i.e. joint variables must at least match the number of independent constraints. That is $n \geq 6$ where n is the number of axes.

The lower bound n is a necessary condition but not a sufficient condition. The tool position must be within the robot's work space and the tool operation must be such that none of the joint variable limits are violated. Even when these additional constraints on the values of position and orientations are satisfied, there is no guarantee that a closed-form expression for a solution to the inverse kinematics problem can be obtained (Pieper,1968) [5].

Which solutions do exist typically they are not unique. Multiple solutions can arise in a number of ways. For robots where n > 6, n is the number of axes, there exists infinitely many solutions to the inverse kinnematics problem. These robots are reffered to as kinematically redundant robots. A redundant robot can reach around an obstacle and manipulate an otherwise inaccessible object. Some of the degree of freedom can be used to avoid the obstacle while the other remaining degrees of freedom are used to configure the tool.

Even when a robot is not redundant, there are situations in which the solution to the inverse kinematics problem is not unique. When the size of the joint-space is sufficiently large, several distinct solutions can arise as can be seen in the figure 3.8.

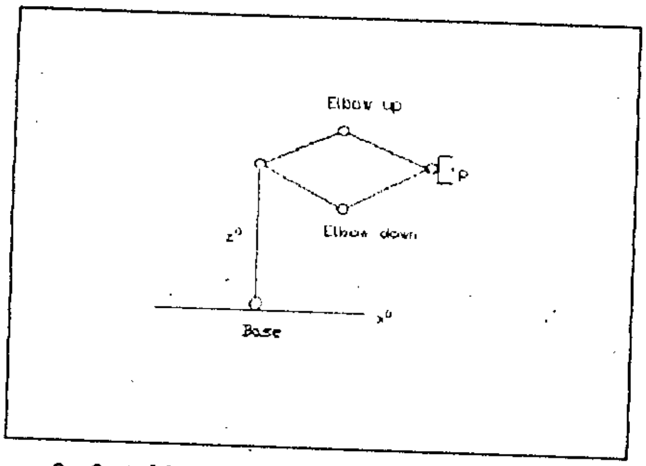

Figure 3.8 Elbow up and elbow down solution

These two solutions are identical in tool-configuration space since they provide the same position and orientation. In joint-space, the two solutions are distinct and which one is to be considered is of practical interest because of the chance of collision between the links of the arm and obstacles resting on the work surface.

3.2.2 Tool Configuration

The tool configuration can be represented by (P, R) where P represents the tool position and R represents tool orientation relative the base. Tool center point (TCP) position P is a translation vector and tool orientation R is a rotation matrix. For the solution to the inverse kinematics problem the tool configuration must be provided as input. Therefore, redundant information is desired to be eliminated from the rotation matrix [5].

3.2.3 Solution to the Arm Equation

In general, the inverse kinematics problem can be solved by various methods such as

- Inverse transform (Paul et al., 1981)
- Iterative (Uicker et al., 1964)
- Dual matrices (Denavit, 1956)
- Dual quaternion (Yang and Freudenstein, 1964)
- Screw algebra (Koli and Soni, 1975)
- Geometric approaches (Lee and Ziegler, 1984)

Paul et al. (1981) presented an inverse transformation technique using 4x4 homogeneous transformation matrices in solving the kinematics solution. The resulting solution is correct but it suffers from the fact that the solution does not give clear indication on how to select an appropriate solution from the several possible solutions for a prticular arm configuration. The solution requires user intervention for a particular picked up which is actually user's geomatric intention dependant or an expert system is required to change the right answer.

Uicker et al. (1964) and Milenkovic and Huang (1983) represented iterative solution for the most industrial robots. The iterative solution often require more calculation and it does not guarntee convergence to the correct solution, especially in the singular and degenerate case. Furthermore there is no clear indication on how to choose the right solution for a particular arm configuration.

Most present day manipulator designs are kinematically simple because firstly it is partly due to the difficulty of general solution to the inverse kinematics problem. Secondly there are few techniques that can provide a general solutiion to the inverse kinematics problem for arbitrary manipulator configurations. Therefore the added difficulty involved in treating the general solutions seem to be unjustified. A number of general solutions are provided by [11], [12], [13], and [14]. Furthermore the complexity of the inverse kinematics perobləm increases with the number of non-zero link parameters which depend on the type of the robot. In these cases a geometric approach is the most simplest and most convenient method.

Consider the three-axis articulated Fischertechnik robot shown in figure 3.9.

To find the first joint variable angle $\Theta 1$ for the manipulator shown in the figure, we project tool centre point (TCP) onto the X_0Y_0 plane as shown in the figure.

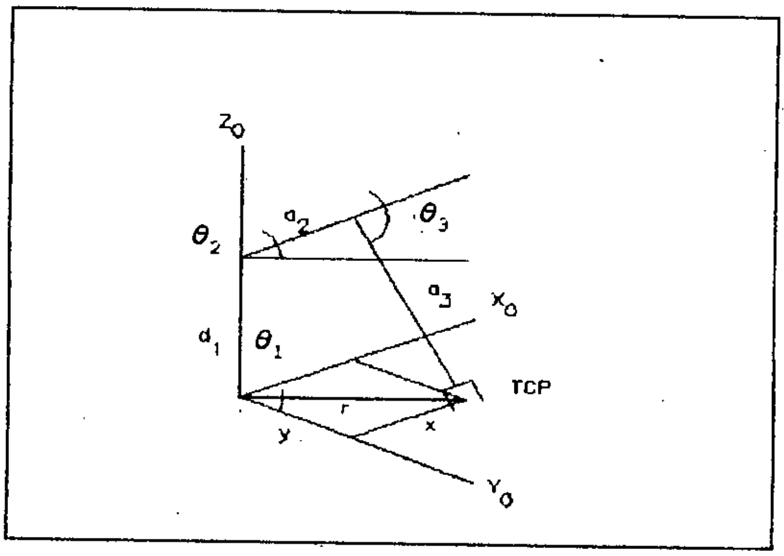

Figure 3.9 Projection of TCP on x_0y_0 plane

We find Θ_1 from this projection as follows.

$$\theta_1 = \tan^{-1}\left(\frac{x}{y}\right) \tag{3.48}$$

where $\tan^{-1}(x/y)$ denotes the two argument arctangent function, and $\tan^{-1}(x/y)$ is defined for all (x,y) not equal to 0 and the unique angle such that

$$Cos\theta_1 = \frac{x}{\sqrt{x^2+y^2}} \tag{3.49a}$$

or

$$Cos\theta_1 = \frac{x}{r} \tag{3.49b}$$

$$Sin\theta_1 = \frac{y}{\sqrt{(x^2+y^2)}} \tag{3.50a}$$

or

$$Sin\theta_1 = \frac{y}{r}$$
$$with \;\; r = \sqrt{(x^2+y^2)} \qquad (3.50b)$$

These solutions are valid unless $x_0 = y_0 = 0$. In this case equation is undefined and the manipulator is in a singular configuration. The singular configuration is shown in figure 3.10. In this position the TCP intersects the z_0 axis. Thus there are infinitely many solutions for θ_1 when TCP intersects z_0. This solution can only be avoided by an offset of d_1 from the z_0 axis. But in Fischertechnik robot this is beyond the work space.

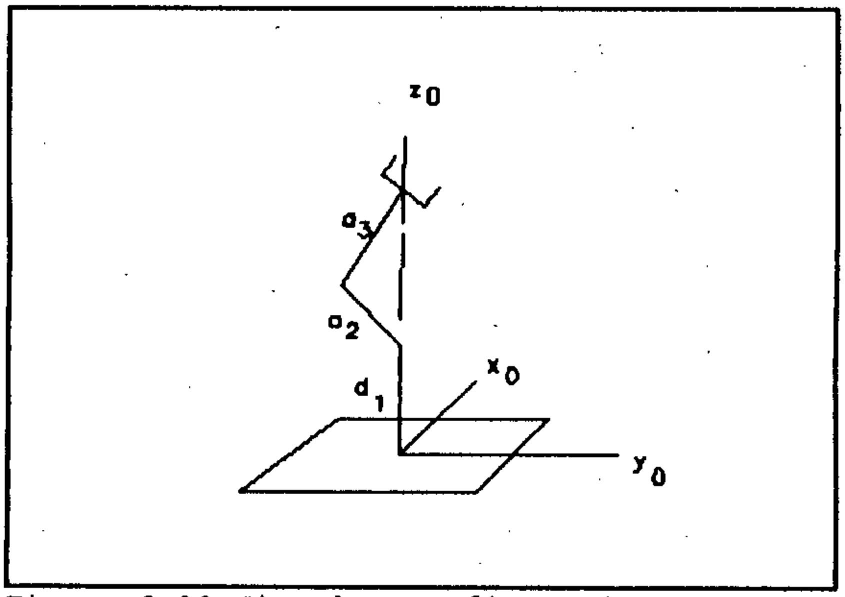

Figure 3.10 Singular configuration

To find the joint variable for the second and third joint, the angles θ_2 and θ_3, we consider the plane formed by the second and third link as shown in figure 3.11.

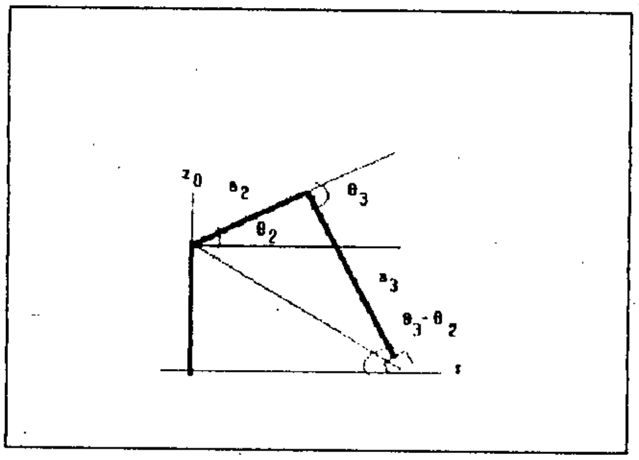

Figure 3.11 Projection on z_0r plane

Since the solution of joint two and three is planar, the solution is analogous to that of the two-link manipulator. We want to find the joint variables Θ_2 and Θ_3 in terms of r and z_0 which forms a plane along the z_0 axis. For a given z and r we can solve the joint angles. Since the direct kinematics equations are nonlinear, a solution may not be easy to find nor there is a unique solution in general.

If the (r,z) i.e given (x,y) in x-y plane are out of reach of the manipulator, there will be no solution at all. If (r,z) i.e given (x,y) in x-y plane is within the robot's work space, there may be two solutions as shown in figure 3.12.

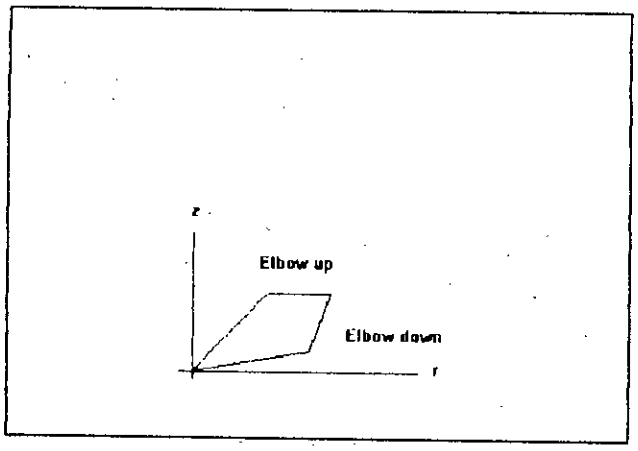

Figure 3.12 Two solutions: elbow up and elbow down

There may be exactly one solution if the manipulator arm is fully extended to reach the point.

Using law of cosines we obtain the angle Θ_3. Detail calculation is shown in appendix A.

$$Cos\theta_3 = \frac{r^2+z^2-a_2^2-a_3^2}{2a_2a_3} \tag{3.51}$$

We could now determine Θ_3 from the above equation

$$\theta_3 = Cos^{-1}\left(\frac{r^2+z^2-a_2^2-a_3^2}{2a_2a_3}\right) = Cos^{-1}(D)$$
$$where \quad \frac{r^2+z^2-a_2^2-a_3^2}{2a_2a_3} \tag{3.52}$$

From trigonometric identities we find

$$Sin\theta_3 = \pm\sqrt{1-\cos^2\theta_3} = \pm\sqrt{1-D^2} \tag{3.53}$$

Now Θ_3 can be found as

$$Tan\theta_3 = \frac{\sin\theta_3}{\cos\theta_3} = \frac{\pm\sqrt{1-D^2}}{D} \tag{3.54}$$

or

$$\theta_3 = Tan^{-1}\left(\frac{\pm\sqrt{1-D^2}}{D}\right) \tag{3.55}$$

The advantage of this approach is that both the elbowup and elbowdown solutions are recovered by choosing the possitive and negative singns in the equation.

Similarly Θ_2 can be found as

$$Tan\theta_2 = \frac{ra_3\sin\theta_3 - z(a_3\cos\theta_3 + a_2)}{r(a_3\cos\theta_3 + a_2) + za_3\sin\theta_3} \tag{3.56}$$

or

$$\theta_2 = \tan^{-1}\left(\frac{ra_3\sin\theta_3 - z(a_3\cos\theta_3 + a_2)}{r(a_3\cos\theta_3 + a_2) + za_3\sin\theta_3}\right) \tag{3.57}$$

Detailed calculation can be found in appendix A. It is to be noticed that the angle Θ_2 depends on Θ_3 which makes sense physically since we would expect to require a different value for Θ_2 depending on which solution is choosen for Θ_3 [8], [15].

Chapter Four

Trajectory Planning

4. Trajectory Planning

In the previous chapter we discussed the kinematics and inverse kinematics of a manipulator which is the background for a robotic arm to control along a preplanned path.

The space curve that the manipulator hand moves along from the initial position and orientation to a final position and orientation is called a path. The desired path is presented by a polynomial function which generates a sequence of time variant control set points in between through which the manipulator must pass. Trajectory planning generally interpolate or approximate these points to form a path. By trajectory, we mean the time history of desired joint positions, velocities and of joint accelerations. Path endpoints can be specified either in cartesian coordinates or in joint coordinates. Joint coordinates are not suitable for specifying path endpoints because joint axes of most manipulators are not orthogonal and usually specified in cartesian coordinates. There may exist several possible trajectories between two endpoints. Therefore a systematic approach is required. There are two common approaches used for trajectory planning. In the first approach, the user explicitly specify a set of constraints such as continuity and smoothness on position, velocity and acceleration of selected points in joint coordinates. A parameterized trajectory is then obtained from a class of polynomial functions of degree n within the time interval $[t_0, t_f]$ by interpolating the points that satisfies the constraints. In the second approach, the user explicitly specifies the path by an analytical function such as a straight-line in cartesian coordinates. A Trajectory is determined in cartesian or joint coordinates which approximates the desired path satisfying the path constraints.

We are interested in developing a suitable formalism for defining and describing the desired trajectory of the manipulator hand between the path endpoints.

4.1 Review of Works on Trajectory Planning

For a manipulator to perform a task, the main problem to be solved are: firstly to generate a trajectory in real time between the present position and the desired position of the end-effector task space coordinate. Secondly, to generate a joint space trajectory which makes the end-effector the above task space trajectory, the inverse kinematic problem. This may have to be done in the presence of additional constraints like avoiding obstacles and keeping within joint limits.

There are variety of robot control algorithms existing which are conventionally divided into two stages. Path or trajectory planning and path tracking or control. This division simplifies the control scheme of highly nonlinear and coupled manipulator dynamics. The path control attempts to make the robot's actual position and velocity match some desired values of position and velocity which are provided by trajectory planner. The trajectory planning usually determines the timing of manipulator position and velocity without considering it's dynamics.

The simplicity obtained from the division into trajectory planning and path tracking comes at the expense of efficiency. The source of the inefficiency is the trajectory planning. To drive the robot at the maximum efficiency, the trajectory planning must consider the robot's dynamic properties. The more accurate the dynamic model is, the better the robot's capacities can be used. But, most of the trajectory planning algorithms presented to date consider very little about dynamics [16].

There are several constraints of considerable interest that the control problem is being faced. There may be obstacles in the manipulators work space which imposes obstacle constraints. There may be a specified path which the manipulator must traverse that imposes path constraints on the manipulator hand. Obstacle constraints and path constraints both give rise to four possible control models as follows.

		Obstacle Constraints	
		Yes	No
Path Constraints	Yes	Off-line collision free path planning + On-line path tracking	Off-line path planning + On-line path tracking
	No	Positional control + On-line obstacle detection and collision avoidance	Positional control

Table 4.1 Four possible control models

Some approaches have been made for concurrent generation of trajectory and trajectory formation of arm movement using artificial neutral network concept [24], [25]. But there are three criticisms of current neutral network models for trajectory formation. Mostly, the back propagation algorithm has been used to train a multi-layer feed forward network. The implementations show major deficiencies such as

1) Non-uniform error over the work space.

2) Their spatial representation of time.
3) Failure of convergence even after several hundred thousand of iterations.

Research on time optimal trajectory of manipulators dates back to the work of Khan and Routh [26]. Practical applicable solutions along a prescribed path were derived by Shin and McKay [16] based on the technique of parameterizing the path with a single variable defining the position along the path. In their works, the torque constraints are transformed in the phase plane of scalar variable. Pfeiffer and Johanni proposed a method that differ from the above works in their search for the switching points. Slotine and Yang improved the efficiency of planning algorithm by constructing limit curves in contrast with maximum velocity curve.

Recently, Shiller and Lu extended the algorithms to handle the case where assumption of maximum acceleration and maximum deceleration along the solution curve is no more valid [20].

While collision free trajectory planning for a single robot with stationary obstacles has been handled in many works. In [17] an algorithm for determining the shortest distance collision free path is developed.

Because of physical constraints such as torque, force, velocity and acceleration, the optimum control of industrial robots is a difficult problem [18]. An alternative approach is to divide the problem into two parts: Optimum path planning for off-line processing followed by on-line path tracking. The path tracking can be achieved by adopting the existing approach. The path planning is done at joint level [19]. It is to plan a path along which an optimization can be achieved, and then to manipulate the robot to the path.

4.2 General Consideration

Trajectory planning can be conducted in either spaces

- In the Joint-variable space
- In the Cartesian space

For Cartesian space trajectory planning, the time history of the manipulator hand's position, velocity, and acceleration are planned. The corresponding joint position, velocity and accelerations are derived from the hand information. Generally, cartesian path planning can be realized in two coherent steps. In the first step a set of knot points or interpolating points in cartesian coordinates are selected according to some rules. In the second step a class of functions are specified to link these knot points according to some criteria. There are two major approaches in choosing the criteria.

- Cartesian space oriented method
- Joint space oriented method

In Cartesian space oriented method straight line segments are used to link the adjacent knot points. The velocity and acceleration of the hand between these segments are controlled by converting them into joint coordinates and smoothed by a quadratic interpolation routine. Paul's straight line trajectory scheme uses homogeneous transformation matrix to represent target position. But matrices are moderately expensive to store and computations on them require more operations. Furthermore, the matrix representation for rotation is highly redundant and this lead to numerical inconsistencies.

Taylor extended and redefined Paul's method by using quaternion representation to describe the location of the hand instead of homogeneous transformation matrix. Because of the properties of quaternion, transitions between the hand locations require less computation. Straight line trajectory planning using homogeneous transformation matrix and using quaternion can be found in [4].

In joint space oriented method, a low degree polynomial function in the joint variable space is used to approximate the segment bounded by adjacent knot points on the straight-line path. Physically, the actuator of each joint is subject to saturation and cannot furnish an unlimited amount of torque and force. Therefore torque and force constraints must be taken into
consideration while planning straight line trajectory. This physical constraints burdened the problem with difficulty which lead it to an alternative approach. The optimum control problem can be divided into two coherent phases : Off-line optimum trajectory planning followed by on-line path tracking. Lin et al. proposed cubic polynomial trajectory method that used low degree polynomials in the joint-variable space to approximate the straight-line path [4]. The original work can be found in [19].

Taylor proposed joint variable space motion strategy called bounded deviation joint path, which selects enough intermediate points during the preplanning phase to guarantee that the manipulator hand's deviation from the cartesian straight-line path on each motion segment stays within pre-specified error bounds. Details of the method can be found in [4],[5]. In general, cartesian space trajectory planning termed as straight-line trajectory planning, which will be discussed in detail in the next section.

In many instances, the path will not be completely specified. Instead, knot points along the path, such as initial and terminal points and perhaps intermediate via-points, will be specified. In these circumstances, the trajectory planning is done by interpolating the knot points to produce a smooth trajectory.

For joint variable space trajectory planning, the time history of all joint variables and their first two time derivatives are planned to describe the desired motion of the manipulator. The computation consists of a trajectory function which must be updated in every control interval. Four constraints are imposed on the planned trajectory. First, the trajectory set points must be readily calculable in a non-iterative manner. Second, intermediate positions must be determined and specified deterministically. Third, the continuity of the joint problem and its first two time derivatives must be guaranteed so that the planned joint trajectory is smooth. Finally, extraneous motions, such as wondering must be minimized.

The above four constraints on the planned trajectory will be satisfied if the time history of the joint variables can be specified by polynomial sequences of degree n or less such that the required joint position, velocity and acceleration at the initial and terminal knot points as well as intermediate knot points are satisfied and the joint position, velocity and acceleration are continuous on the entire time interval. One approach is to specify a seventh-degree polynomial for each joint i . It is different to determine the coefficients from the entire trajectory. An alternative approach is to split the entire trajectory into segments. There are different ways a joint trajectory can be split. The most common methods are

- Two quartic and one cubic (4-3-4) trajectory segments
- Two cubic and one quintic (3-5-3) trajectory segments
- Five cubic (3-3-3-3-3) trajectory segments

Each joint trajectory is split into either three-segment or a five-segment trajectory. The difficulty encountered here is that the equations for computing the polynomial coefficients for segments become coupled. An alternative to cubic polynomial interpolation is to use piece wise linear interpolation with parabolic-blends between knot points [5], [4].

Planning the trajectory in joint-variable space has three advantages over the cartesian space:

i) Trajectory is planned directly in terms of controlled variables during motion,

ii) Trajectory planning can be done in near real time, and

iii) Overall joint trajectory is easier to plan.

Before going to the trajectory planning for one robot, let us discuss different methods in brief in the next section.

4.3. Joint Interpolated Trajectories

Because of the various disadvantages in straight-line trajectory planning, the joint interpolated trajectory planning scheme will be adopted.

In planning a joint-interpolated trajectory for a robotic arm some rules are to be considered that are suggested by Paul [4].

1. When picking up an object, the motion of the hand must be directed away from an object to avoid collision with the supporting surface.

2. If we specify a lift-off point along the normal vector to the surface out from the initial position and if we require the hand to pass through this position, we then have an admissible position. If we further specify the time required to reach this position, we can then control the speed of the

hand at which the object is to be lifted.

3. The same set of lift-off requirements for the arm-motion is also true for the set-down point of the final position motion so that the correct approach direction can be obtained and controlled.

4. From rules 2 and 3, we have four positions for each arm: Initial, Lift-off, Set-down, and Final as can be seen in figure 4.1. Thus we have four position constraints.

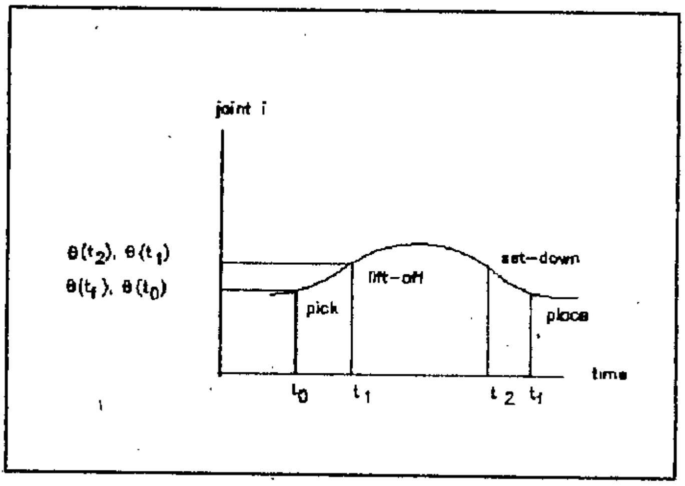

Figure 4.1 Position condition for a joint trajectory

In the initial and final positions, velocity and acce-leration are given which are normally zero. In the lift-off and set-down positions, velocity and acceleration are continuous for intermediate points. The constraints of a typical joint trajectory are shown in table 4.1.

5. The extrema of all joint trajectories must be within the physical and geometric limits of each joint.

6. Time considerations:Time is based on the rate of approach of the hand to and from the surface in initial and final

trajectory segment, and is some fixed constant based on the characteristics of the joint motors.

Time in intermediate trajectory segment is based on maximum velocity and acceleration of the joints, and the maximum of these times is used.

A class of polynomial functions of degree n or less is chosen to satisfy the joint position, velocity and acceleration at these knot points. There are several class of polynomials used for each joint i. One approach is to specify a seventh degree polynomial for the entire trajectory. An alternative approach is to split the entire trajectory into several segments so that different interpolating polynomials of a lower degree can be used to interpolate in each trajectory segment [4].

7th degree polynomial trajectory

The trajectory consists of the 7th degree polynomial for each joint i is as follows

$$h_i(t) = a_7t^7+a_6t^6+a_5t^5+a_4t^4+a_3t^3+a_2t^2+a_1t^1+a_0 \qquad (4.1)$$

The joint position, velocity and acceleration are continuous on the entire time interval $[to,t_f]$. The unknown coefficients $a_7, a_6, \ldots, a_0$ can be determined from known positions and continuity conditions. It is difficult to find the extrema from such a high degree polynomial and interpolation of the given knot points may not be satisfactory. Furthermore, it tends to have extraneous motion.

Segmented trajectory

Different methods are used to reduce extraneous motion by splitting the trajectory into several segments. The following methods are most common among them:

4-3-4 Trajectory: Each joint trajectory is divided into three segments. The first segment is a fourth-degree polynomial specifying the trajectory from initial position to the lift-off position.

$$h_1(t) = a_{14}t^4+a_{13}t^3+a_{12}t^2+a_{11}t+a_{10} \qquad (4.2)$$

The second segment is a third-degree polynomial specifying the trajectory from the lift-off position to the set-down position.

$$h_1(t) = a_{13}t^3+a_{12}t^2+a_{11}t+a_0 \qquad (4.3)$$

The last segment is a fourth-degree polynomial specifying the trajectory from the set-down position to the final position.

$$h_1(t) = a_{14}t^4+a_{13}t^3+a_{12}t^2+a_{11}t+a_{10} \qquad (4.4)$$

The number of polynomials for a 4-3-4 trajectory of an N-joint manipulator will have N-joints trajectories or Nx3 = 3N trajectory segments and 7N polynomial (3 for third degree polynomial function and 4 for fourth degree polynomial function) coefficient to evaluate and the extrema of 3N trajectory segments. Calculation of a 4-3-4 joint trajectory can be found in [4].

3-5-3 trajectory: Each joint trajectory is divided into three segments as in 4-3-4 trajectory. The first segment is a third-degree polynomial specifying from initial position to the lift-off position..

$$h_1(t) = a_{13}t^3+a_{12}t^2+a_{11}t+a_{10} \tag{4.5}$$

The second segment is fifth degree polynomial specifying the trajectory from the lift-off position to the set-down position.

$$h_1(t) = a_{15}t^5+a_{14}t^4+a_{13}t^3+a_{12}t2+a_{11}t+a_{10} \tag{4.6}$$

The last segment is again a third-degree polynomial specifying the trajectory from set-down position to final position.

$$h_1(t) = a_{13}t^3+a_{12}t^2+a_{11}t+a_{10} \tag{4.7}$$

The number of polynomials for a 3-5-3 trajectory of an N-joint manipulator will have N-joint trajectories i.e Nx3 trajectory segments and 8N polynomial coefficients (3 for third-degree polynomial and 5 for five-degree polynomial) to evaluate. Also the extema of the 3N trajectory segments are to find out. Similar calculation technique as for 4-3-4 trajectory can be applied for 3-5-3 joint trajectory.

5 - **Cubic trajectory** : Each joint trajectory is divided into five segments and a cubic spline function of third degree polynomial for five trajectory segments is used. The interpolation of a given function by a set of cubic polynomials, presenting continuity in the first and second derivatives at the interpolation points, is known as cubic spline function. The general equation of a five-cubic polynomial for each joint trajectory segment is given by the following equation.

$$h_1(t) = a_3t^3+a_2t^2+a_1t+a_0 \tag{4.8}$$

In five-cubic spline trajectory, we have six interpolation points. Two extra interpolating points must be selected in addition to four positions, namely initial, left-off, set-down and final points to provide enough boundary conditions for solving the unknown coefficients in the polynomial sequence. These two extra points can be selected in between the lift-off and set-down positions, where continuity of velocity and acceleration must be satisfied with the known time interval. A general solution to the five cubic spline trajectory is given in [4]. The five cubic spline has the advantage that firstly, it is the lowest degree polynomial function that allows continuity in velocity and acceleration, and secondly, it reduces computational effort due to low-degree polynomial functions.

89411

Linear Interpolation with parabolic Blends: Using cubic interpolating polynomials, the acceleration on each sample period is linear. In many applications there are a good reason for insisting on constant accelerations and within each sample period along a portion of path. Besides that, most of the industrial robot controllers are programmed to use constant accelerations on each sample period. A constant acceleration profile is shown in figure 4.2 . The associated velocity and position profiles are shown in figure 4.3 and 4.4 respectively.

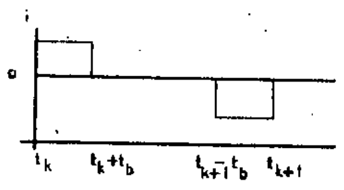

Figure 4.2 Constant acceleration profile

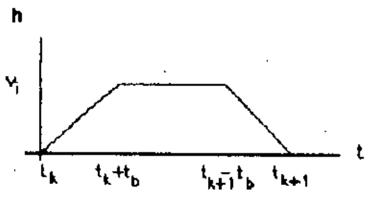

Figure 4.3 Velocity profile

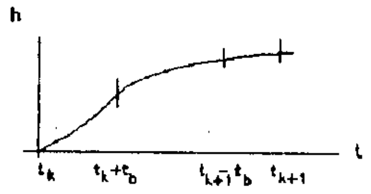

Figure 4.4 Position profile

In such applications, an alternative approach to generate a suitable joint space trajectory is to use piece-wise-linear interpolation between the knot-point as shown in figure 4.5.

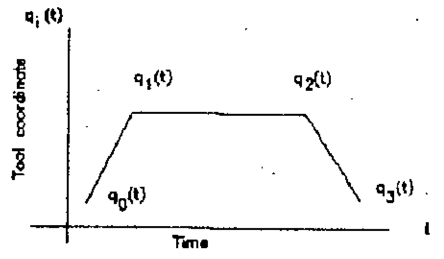

Figure 4.5 Piece-wise-linear interpolation

Although piece-wise-linear interpolating is computationally very efficient and it efficiently decouples the polynomial coefficients. But it suffers from one major drawback that the generated path is not smooth. By introducing parabolic blends which smoothly connects the adjacent linear segments, this drawback has been remedied as can be seen from figure 4.6.

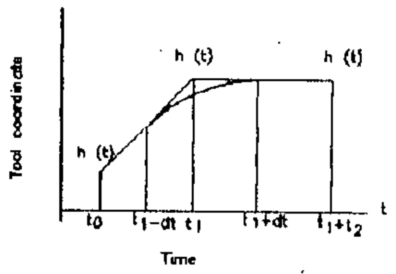

Figure 4.6 Linear interpolation with parabolic blend

The position trajectory has three paths: a quadratic or parabolic initial position, a linear midsection, and a parabolic final section. The linear segment with parabolic blends is such that the velocity is rampted up to its specific value initially and then rampted down at the goal position [5],[6],[8].

Consider the two segment trajectory shown in figure 4.6. Let the segments $\{h_0(t),\ h_1(t)\}$ and $\{h_1(t),\ h_2(t)\}$ are to be traversed in time T_1 and T_2 respectively with constant velocity v. Then we obtain

$$v_i = \frac{\Delta h_i(t)}{T_i} \tag{4.9}$$

$$\textit{where} \quad \Delta h_i(t) = h_i(t) - h_{i-1}(t) \quad ; 1 \le i \le 2$$

If $v_1 \neq v_2$ then an instantaneous infinite acceleration is required at time T_1 to achieve the abrupt change in velocity. To keep the acceleration finite, we must gradually change the velocity over some transition interval $[\ T_1 - \Delta T,\ T_1 + \Delta T\ ]$ where $\Delta T > 0$. Here we apply a constant acceleration starting at time $T_1 - \Delta T$. A smooth transition can be achieved from velocity v_1 t time $T_1 - \Delta T$ to velocity v_2 at time $T_1 + \Delta T$. The tool will be at the same point where it would have been if the original piece wise linear path had been followed. Since the acceleration is constant, the trajectory during the transition interval $[\ T_1 - \Delta T,\ T_1 + \Delta T\ ]$ will be a quadratic polynomial of the general from.

$$h(t) = a\,\frac{(t-T_1+\Delta T)^2}{2} + b\,(t-T_1+\Delta t) + C \tag{4.10}$$

If transition time ΔT is known, the required acceleration a can then be found in terms of transition time ΔT, traversal time T_1, T_2

and segment displacement Δh_1, Δh_2. To evaluate a, we apply the necessary conditions. The velocity at the start of transition is

$$\dot{h}(T_1-\Delta T) = \frac{\Delta h_1}{T_1}$$

Differentiating the equation (4.10) we obtain

$$\dot{h}(t) = a\ (t-T_1+\Delta T)+b \tag{4.11}$$

Evaluating the result in equation (4.11) at $T_1 - \Delta T$ yields

$$\begin{aligned}\dot{h}(t) &= a(T_1-\Delta T-T_1+\Delta T)+b\\ \frac{\Delta h_1}{T_1} &= b\end{aligned} \tag{4.12}$$

The velocity at the end of transition is

$$\dot{h}(T_1+\Delta t) = \frac{\Delta h_2}{T_2} \tag{4.13}$$

Evaluating the result in equation (4.11) at $T_1 + \Delta T$ yields

$$\frac{\Delta h_2}{T_2} = 2a\Delta T+b \tag{4.14}$$

From equation (4.12) and (4.14) we obtain

$$a = \frac{\Delta h_2 T_1 - \Delta h_1 T_2}{2\Delta T . T_1 . T_2} \qquad (4.15)$$

From equation (4.15) we find that $\Delta T > 0$ for a finite acceleration a. Transition time is also bounded from above, because transition time must not lie inside the overall interval
$[0, T_1 + T_2]$. Thus the sounds on the transition time ΔT are

$$0 < \Delta T < \min\{T_1, T_2\} \qquad (4.16)$$

If there are blend regions associated with the points adjacent to h_1, then the transition time ΔT will have to be still smaller so that successive blends do not overlap. The parabolic-blend trajectory do not go through the knot point $h_1(t)$ at time T_1, but it can make go close the knot point if the transition time ΔT is made small enough [5].

Chapter Five

Obstacle-free Trajectory Planning

5. Obstacle-free Trajectory

The subject of collision free path planning is relatively new. Within the past few years only a handful people have been actively working on this topic. Among them are Pieper [12] and Widdoes [27] who used planes, cylinders, and spheres to represent obstacles. Udupa [28] discretized the space into cells which were labelled free if there is no object in that cell and labelled as obstacled if there is any object in that cell. The list of free cells are joined together to form a collision free path. Lozano-Perez [29] described linked polyhedra using swept volumes. The rotation range is then divided into a finite number of slices. Brooks [30] adopts the idea of generalized cones which are equivalent to swept volumes. Free spaces are represented as overlapping generalized cones.

In the methods described above, some determined the free space inside which the point robot may move freely without collision with obstacles, while others determine the forbidden region so that a collision free path may be traced along the boundaries of the region.

Let the task to be performed be limited within the space shown in the figure 5.1.

The plane is divided into 64 squares of equal sizes. The squares are labelled as free if there is no object in that square. The manipulator can generate the trajectory by finding the free square and joining them together to form the path. There may be several possible trajectories between the two points shown in the figure 5.2.

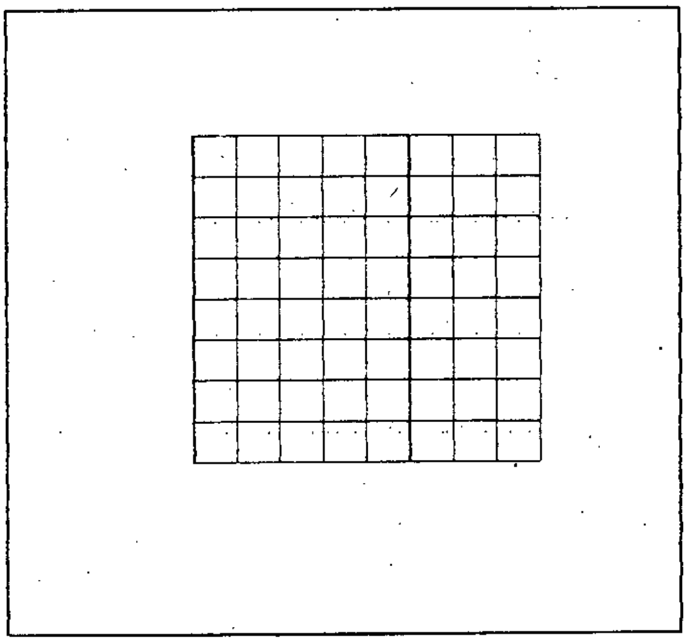
Figure 5.1 Task space of the robot

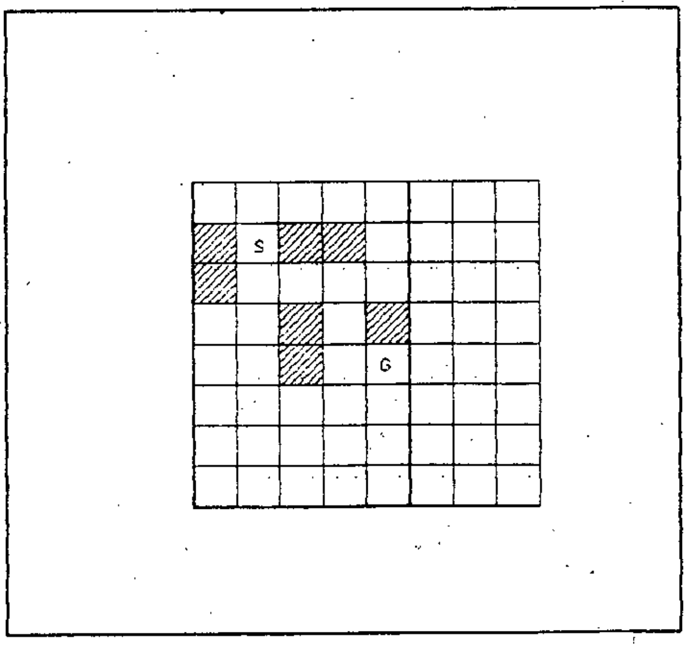

Figure 5.2 Free path to target

Finding the free squares requires an exhaustive search which is very time consuming. Furthermore, the trajectory found by such an exhaustive search may not ensure the shortest path. Also there may not exist any free path to the target position as with the case shown in figure 5.3.

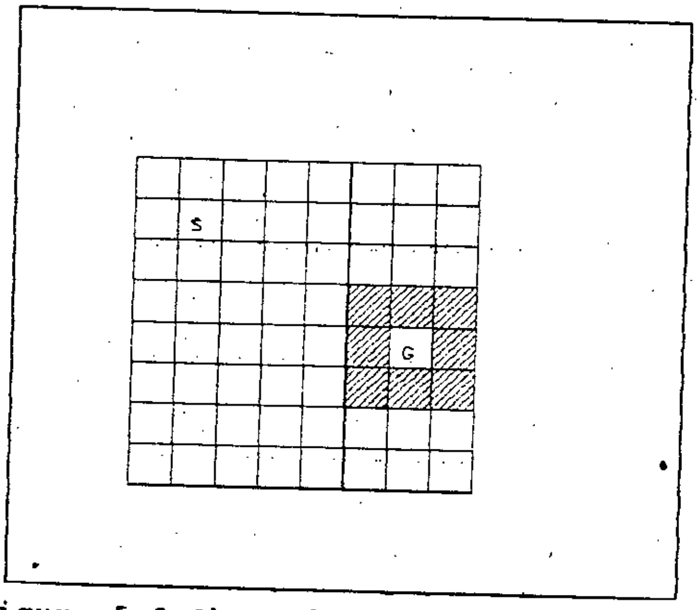

Figure 5.3 Obstacle around the target

Again, if the space is divided into cells of equal sizes as shown in figure 5.4, then it requires mapping of all cells containing obstacles and of all free cells. It also has to keep record of the height of the objects within the cells so that it can generate a trajectory avoiding collision with obstacles. The generated trajectory will not be smooth, and the manipulator has to stop at every knot point as shown in figure 5.4. Such a trajectory with short burst of high acceleration causes the manipulator to produce a jerky motion which can burn out the drive circuits and strip gears.

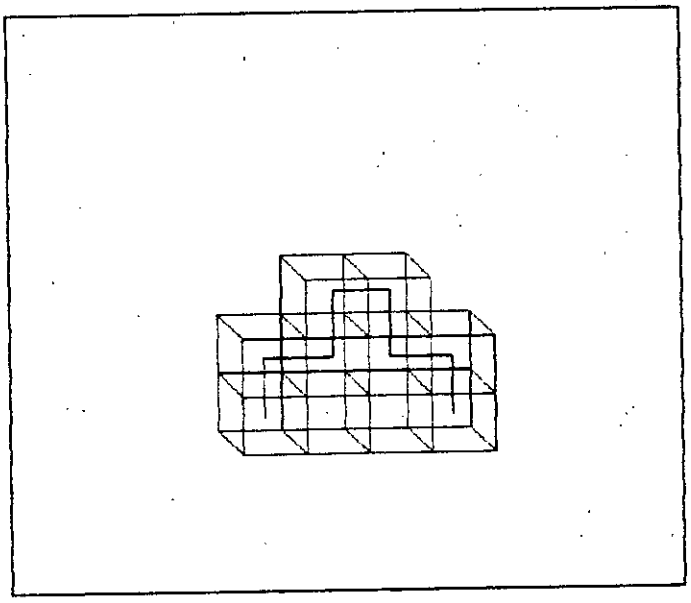

Figure 5.4 Trajectory formed by combining cells

Such a trajectory planning algorithm will not be helpful, and therefore, will be unjustified in this case. An alternative approach is to generate a pick-and-place trajectory consisting of four discrete points between the current position and the target position [31].

5.1 Pick-and-Place Operation

Perhaps the most fundamental robotic manipulator task is the pick-and-place operation. This type of operation is needed, e.g., in the automated loading and unloading of machines. More generally, pick-and-place operations are used to alter the distribution of parts within the workspace. The sequence of operations are as follows:

i) Move the manipulator from its current position to the starting position
ii) Grasp the object, and
iii) Move the object to the goal position

Since the speed of each joint of the robot cannot be controlled independently, simple point to point control can be used to execute pick and place operations with the minimum pick-and-place trajectory, which consists of the following four discrete points.

i) Pick point
ii) Lift-off point
iii) Set-down point
iv) Place point

Additional points can be chosen such as via points which lies between the lift-off point and the set-down point.

5.1.1 Pick and Lift-off Point

At one end of the pick-and-place trajectory is the pick point, whose coordinate frame is denoted as T^{pick}_{base} . This represents the initial position and orientation of the object being manipulated. It can be denoted as follows:

$$T^{pick}_{base} = \begin{bmatrix} R^{pick} & p^{pick} \\ 0\ 0\ 0 & 1 \end{bmatrix} \tag{5.1}$$

The pick position p^{pick} is taken to be the center of mass. The pick orientation R^{pick} must also be specified. The pick point is specified by the user, or is perhaps computed with a vision system whose orientation should be orthogonal to the plane where the object is currently resting.

The second point of pick-and-place trajectory is the lift-off point, whose coordinate frame is denoted as T^{lift}_{base}. The lift-off point is a point near the pick point where the manipulator attempts to reach down to pick up the object. The lift-off point is an intermediate point that is inferred from pick point i.e. once the pick point is determined, the associated lift-off point can then be computed from the following:

$$T^{lift}_{base} = \begin{bmatrix} R^{lift\text{-}off} & p^{lift\text{-}off} \\ 0\ 0\ 0 & 1 \end{bmatrix} \qquad (5.2)$$

$$p^{lift\text{-}off} = p^{pick} - vR^{pick} i^{3} \qquad (5.3)$$

Where v is the approach distance. The tool position at the lift-off point is obtained by starting of the pick position and moving backward a distance v along the approach vector. The tool orientation at lift-off point is identical to tool orientation at the pick point i.e. $R^{pick} = R^{lift\text{-}off}$. We can rewrite the equation (5.2) as follows:

$$T^{lift}_{base} = \begin{bmatrix} R^{pick} & p^{pick} - vR^{pick} \\ 0\ 0\ 0 & 1 \end{bmatrix} \qquad (5.4)$$

5.1.2 Place and Set-down Point

At the other end of the pick-and place trajectory is the place point, whose coordinate frame is denoted as T^{place}_{base}. This represents the final position and orientation of the object and whose orientation should be orthogonal to the plane. It is explicitly specified by the user and can be denoted as follows:

$$T_{base}^{place} = \begin{bmatrix} R^{place} & p^{place} \\ 0\ 0\ 0 & 1 \end{bmatrix} \qquad (5.5)$$

The last point of the four point pick-and-place trajectory is the set-down point, whose coordinate frame is denoted as T^{set}_{base}. The set-down point is analogous to the lift-off point , where the manipulator attempts to reach down the place point. Once the place point frame T^{set}_{base} has been determined, the associated set-down point frame can be calculated from the following equation.

$$T_{base}^{set} = \begin{bmatrix} R^{set} & p^{set} \\ 0\ 0\ 0 & 1 \end{bmatrix} \qquad (5.6)$$

$$p^{set} = R^{place} - VR^{place} i^{3} \qquad (5.7)$$

Where v is the approach distance. The tool position at set-down point is obtained by moving a distance v along the approach vector from the place point.

Again, the tool orientation at the place point is identical to the tool orientation at set-down point i.e. $R^{set} = R^{place}$. Thus we can rewrite the equation (5.6) as follows:

$$T_{base}^{set} = \begin{bmatrix} R^{place} & p^{place} - VR^{place} \\ 0\ 0\ 0 & 1 \end{bmatrix} \qquad (5.8)$$

5.1.3 Via Points

To avoid collision with the objects one or more via-points may have be to inserted between the lift-off and set-down points. A pick-and -place trajectory is shown in figure 5.5 with pick, lift-off, set-down, place, and via points.

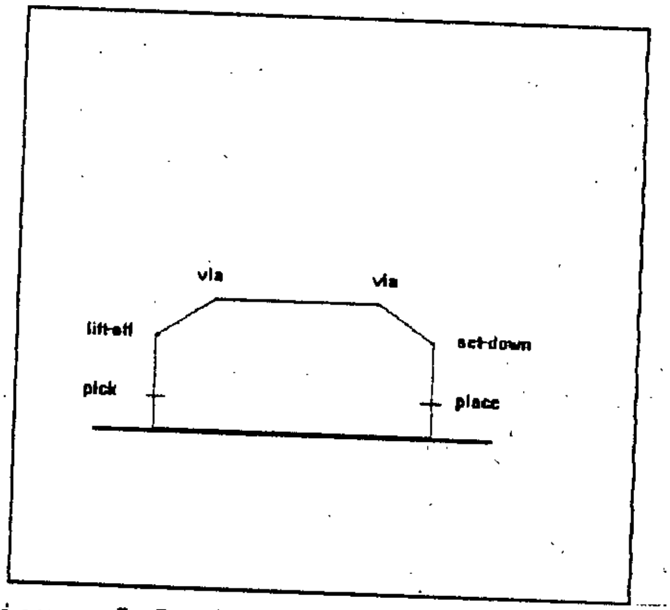

Figure 5.5 Pick-and-Place trajectory

5.2 Determination of Pick-and-Place Point

The pick and place point are specified by the user or perhaps computed with a vision system. The present system has not any vision system. Therefore, the pick and place points are to be specified by a user defined point in a coordinate system within the space. The plane has been divided into finite number of squares of equal size whose coordinate points are known.

The sliding vector should be selected so as to grasp the object along two parallel faces where the separation between the faces is smaller than the maximum opening of the tool. A tool configuration for pick and place point is shown in the figure 5.6.

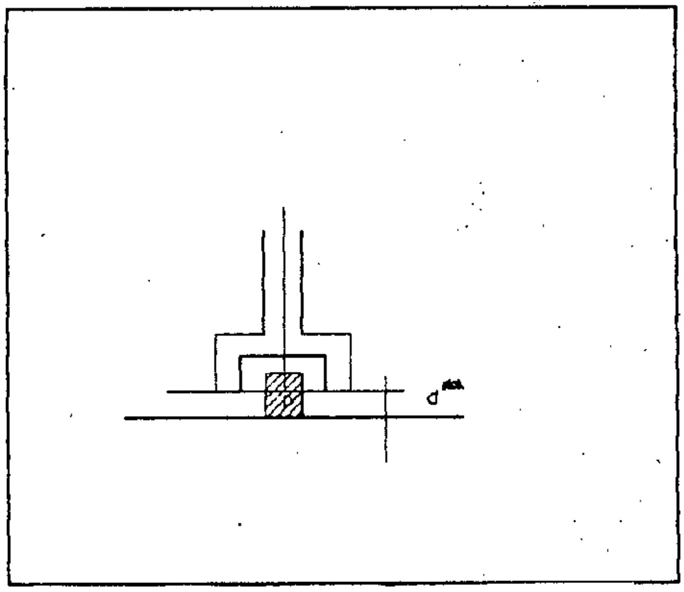

Figure 5.6 Tool configuration

Therefore, the two motions that are most important and critical are the movement from the lift-off pint to the pick point and the movement from set-down point to the place point, in both cases the robot is approaching the work surface, loading on the robotic arm due to gravity will often tend to increase the speed of the movements, particularly when the approach vector is $r^3 = - \underline{i}^3$. This increase in speed can sometimes cause an overshoot of the pick and place point. When the tool moves from the lift-off point to the pick point, an overshoot followed by damped oscillations about the pick point can cause the object to be knocked over or otherwise disturbed. An overshoot when the tool moves from set-down point to place point is even more troublesome, because here any overshoot causes the robot to attempt to penetrate the work surface with the object.

The distance d^{place} between the place position and the place surface, as measured along the approach vector, should be identical to the distance d^{pick} between the pick position and the pick surface:

$$d^{place} = d^{pick} \qquad (5.9)$$

Indeed, if $d^{place} < d^{pick}$, then an attempt will be made to penetrate the work surface when the object is placed, as the left in figure 5.7.

In this case the object will probably slip within the jaws of the tool gripper. Alternatively, if $d^{place} > d^{pick}$, then the place object will be unsupported when it is released, as shown at the right in figure 5.7. Gravity will cause it to fall to the work surface.

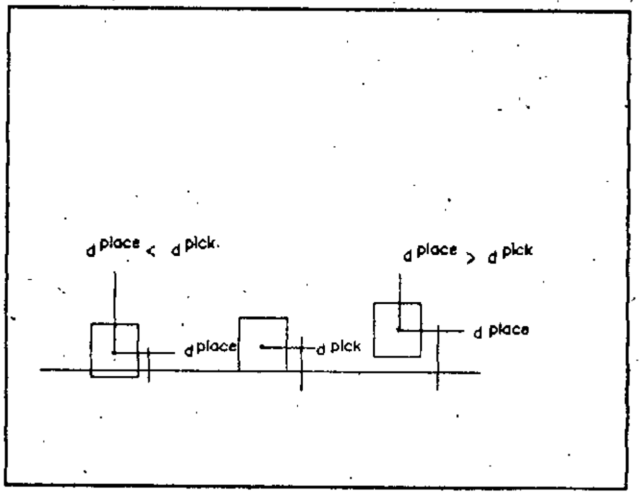

Figure 5.7 Place distance constraint

The simplest and most common case of place position constraint in equation (5.9) occurs when the object is picked up from and placed down onto a common horizontal work surface. In this case, the equation (5.9) becomes

$$p_3^{place} = p_3^{pick} \qquad (5.10)$$

Here P_3 denotes the position in the direction of z axis.

The pick position is often taken to be the center of mass, or the centroid of the object. The figure 5.8 shows the centroid of the object.

Alternatively, if the object is of an irregular shape, then the centroid of some feature of the object such as a handle might be used.

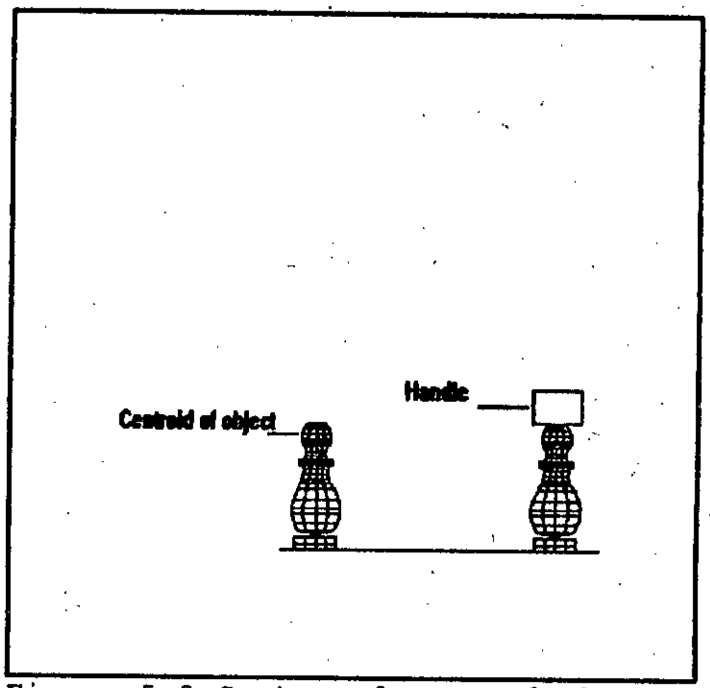

Figure 5.8 Center of mass of the object

5.3 Determination of Lift-off and Set-down Points

Once the pick point is determined, the associated lift-off point can be calculated from the equation 5.3, where v is the approach distance. As we can see from figure 5.8 the approach vector is orthogonal to the work surface on which the object is resting.

The tool rotation matrix is selected to ensure that the approach vector is orthogonal to work surface, where the surface is horizontal. This way the tool orientation remains fixed as the tool moves from lift-off point to the pick point. The tool position at the lift-off point is obtained by measuring a save a distance from the pick point. Therefore, the approach vector v is important to be measured exactly.

The work space of the robot may contain multiple objects and the objects may be of different heights. It is also not intended to have different lift-off and set-down points each time in the same work space because it requires some additional computational time which slows down the working environment. Therefore, it is convenient and justified to have a common lift-off and set-down point for all operations. Whenever the heights of all the objects are known, it is better to choose the maximum height of the object to avoid the additional computational time. Therefore, the desired lift-off and set-down point will be as in figure 5.9.

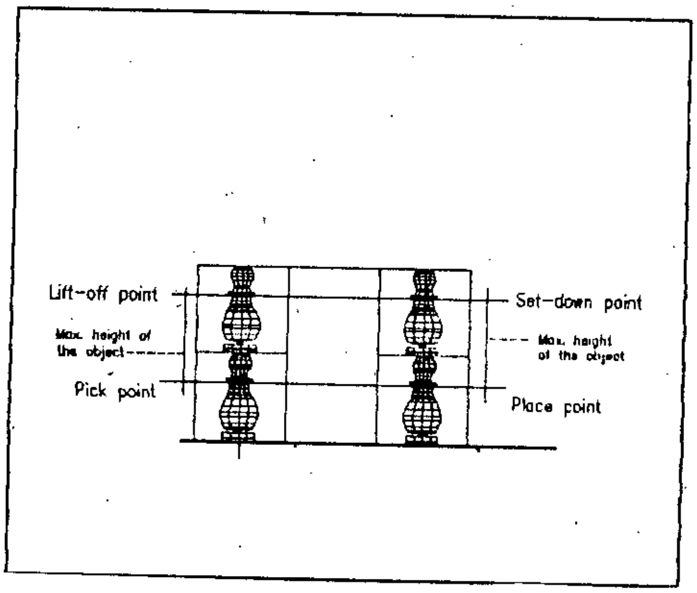

Figure 5.9 Lift-off and set-down point

The approach distance will be as follows:

Approach distance = Lower half of the object from the centroid
+
Upper half of the object from the centroid

Which can be rewritten as

$$V = \textit{Height of the Object} \tag{5.11}$$

After we have fixed the approach distance, the lift-off point and the set-down point can be determined as follows:

$$\begin{aligned} \textit{Lift-off point} &= \textit{Pick point} + V \\ \textit{Set-down point} &= \textit{Place point} + V \end{aligned} \tag{5.12}$$

5.4 Determination of the Via Points

Objects could be moved along the path shown in figure 5.9, but still there is a chance to collide against the obstacles within pick and place point. To avoid these undesired collision, a distance between the object to be moved and the obstacles should be maintained. A maximum distance is chosen so that the bottom of the object does not touch the top of the resting object between pick and place points. We call it a *safe distance.*

A safe distance is required because the accuracy of a manipulator is affected by computational errors, in the accuracy of the manipulator, flexibility effects due to bending of the links under gravitational and other loads, and gear backlash.

The via points are shown in figure 5.10, and can be determined as follows:

$$\textit{Via point} = \textit{Pick point} + V + \textit{Safe distance} \tag{5.13}$$

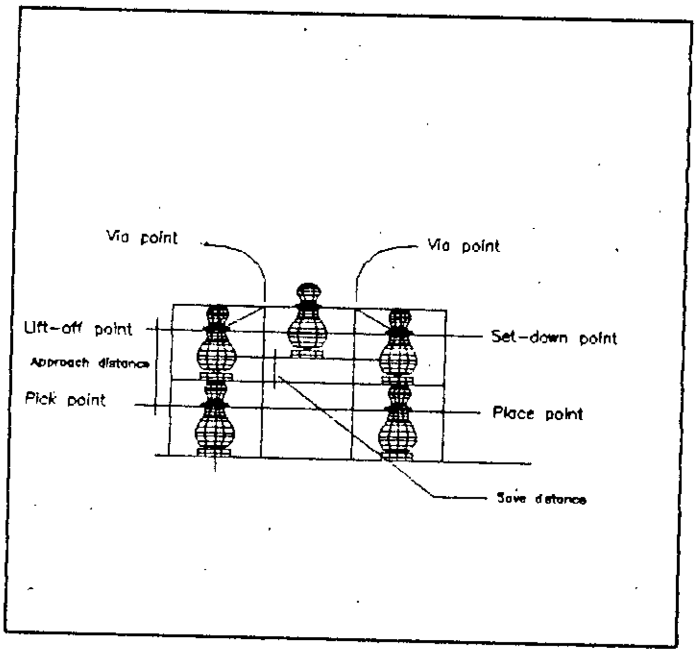

Figure 5.10 Via points

5.5. Algorithm for Pick-and-Place Trajectory

- Set $R^{lift} = R^{pick}$ that is the tool orientation remains fixed from lift-off point to pick point.

- Set $R^{set} = R^{place}$ that is the tool orientation remains fixed from set-down point to pace point.

- Determine the pick and place point using the inverse kinematics.

- Set approach distance v = maximum height of the object.

- Determine the lift-off and set-down points from the calculated pick and place point.

- Interpolate these points to form the trajectory.

The desired trajectory will be as in figure 5.11.

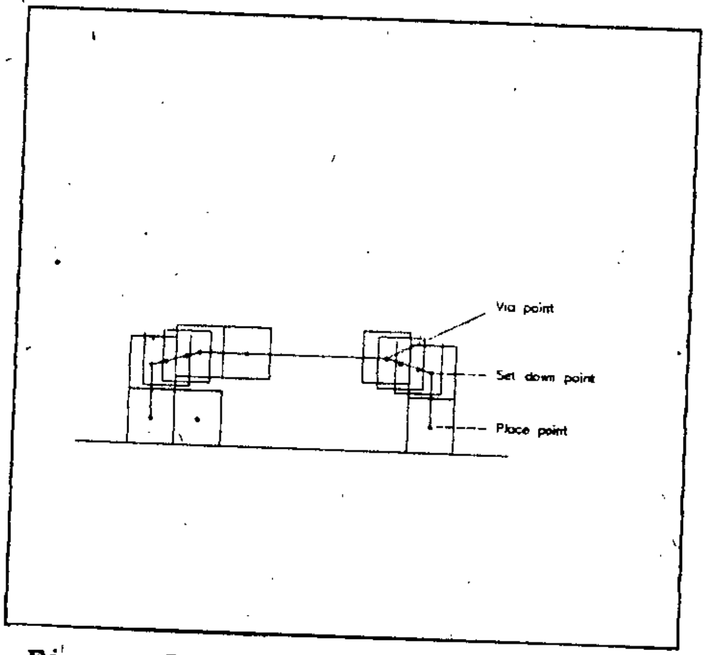

Figure 5.11 Desired trajectory

Chapter Six

Development of the System

6. Development of the System

The robot end-effector has to grasp a specified chessman and to move it from one square to another square of the chessboard. The user provides a move through the keyboard and the interface program generates two points on the chessboard one of which specifies where the chessman is currently located and the second point specifies where it has to be moved. These two points are then calculated into joint coordinate System which provides the position and orientation of the pick and place point of the trajectory planning system. The trajectory planning algorithm divides the path from pick point to the place point into segments by adding intermediate points or via points. These points are interpolated together to form the desired trajectory which is described in the previous section.

6.1 A Chess Application

To play chess we need a chessboard, chessmen, and an opponent. A chessboard looks like a draught board. There are 64 squares in a chessboard out of which 32 are white and 32 are black. When playing chess make sure that the white corner square is on your right side. The chessboard has a standard size of 12x12 in inch or 32x32 in centimeter i.e. each square has the size of 4x4 in centimeter or 1.5x1.5 in inches. The rows are numbered from 1 through 8 and the columns are numbered from a through h such that each square can be specified by a1 through h8. As for example a1 means row 1 of column a. Such a chessboard is shown in figure 6.1.

The chessmen in each set come in two colors. We always call lighter color White and darker color Black. The king is the tallest piece and about 9.5 cm tall, the queen is the second tallest and 8.5 cm tall, rook is 7.5 cm, knight is 6.5 cm, and pawn is the smallest chessman and about 4.5 cm tall.

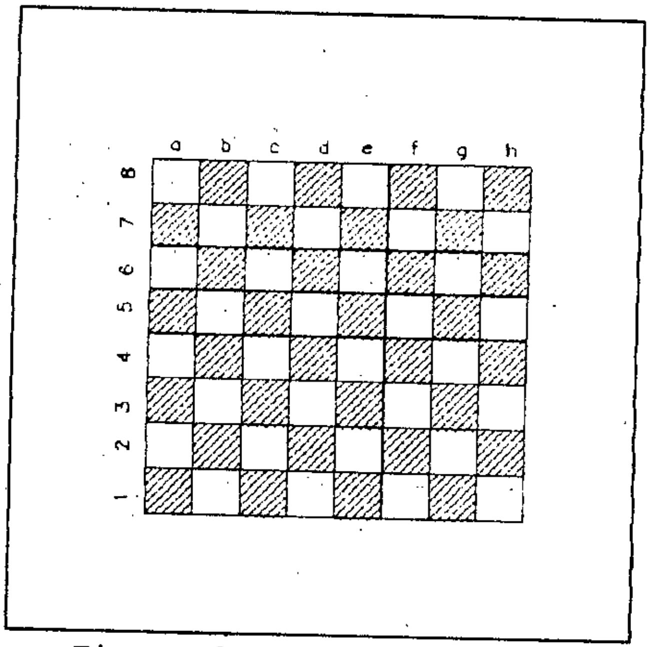

Figure 6.1 A chessboard

6.2 Mapping the Robot Workspace

In order to find a safe path, i.e., obstacle free path from the start point to the goal point the robot environment is first mapped. In Cartesian space the environment of the robot is easily represented since the ranges of the manipulator is subsequently divided into 64 squares of equal size. The columns are labelled A through H and rows are labelled 1 through 8. The squares are represented by A1 through H8.

The robot Workspace is now identical to the chessboard. A one to one mapping can be established between 64 squares of the robot environment and the 64 squares of the chessboard. From now on we shall replace the cluttered environment of the robot with a chessboard.

The chessboard should be placed in such a position so that it accommodates within the Workspace of the robot. Then we need to find the coordinate points of each square of the chessboard. Before we fix the coordinate points of the square, we can make good use of the symmetry of positions of the squares, if we just divide the board into two halves and position the robot in the middle of the x-axis. Then we require to rotate the base clockwise to position the end-effector in the left half and to rotate the base counter clockwise to position the end-effector in right half of the chessboard.

The use of symmetry of the board has the advantage that firstly it requires fixing of coordinate points of only 32 squares and the rest coordinates points of 32 squares are obtained by rotating the base in counter direction. Secondly, it requires less computation which ultimately saves time in writing large programs. Figure 6.2 shows the mapping of the chessboard.

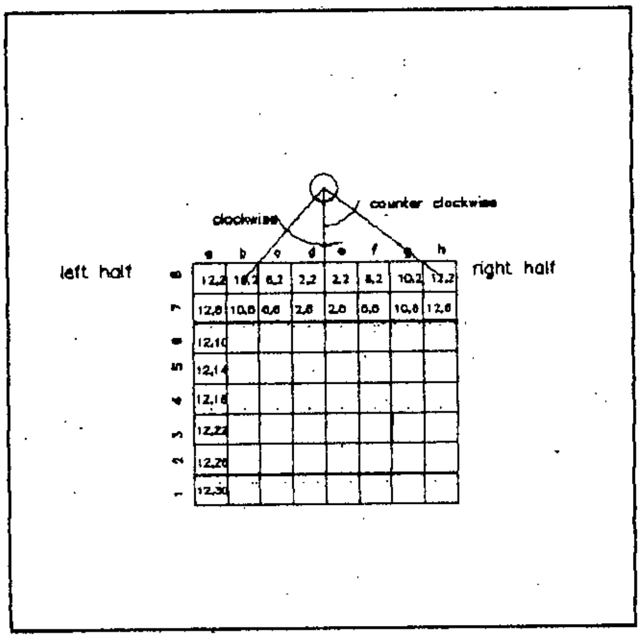

Figure 6.2 Mapping of the chessboard

After the coordinate points are fixed for each square in both halves, we need to find all possible valid combinations of joint angles of the manipulator and then transform them into coordinates.

As for example, if the chessboard has the size of 32x32 cm, then each square has the size of 4x4 cm. The coordinate points of A1 through D8 is shown in figure 6.2 and the mapping is as follows:

```
A1 -------> 12,2
.
.
.
A8 -------> 12,30
.
.
.
D1 -------> 2,2
.
.
.
D8 -------> 2,30
```

The coordinate points of the other half can be found by simply mirroring the points A1 through D8 about the border line of D column, i.e., the coordinate points of H1 through E8 will be same for corresponding squares of the other half, as follows:

```
E1 ------> D1 -------> 2,2
.
.
.
E8 ------> D8 -------> 2,30
.
.
.
```

```
H1 ------> A1 -------> 12,2
.
.
.
H8 ------> A8 -------> 12,30
```

6.3 Calculation of the Joint Angles

After we have fixed the coordinate points of each square of the chessboard, we can now calculate the joint angles. The first joint angle of the manipulator for each square, i.e, for every pick, lift-off, via, set-down, and place point can be calculated. Then the third joint angle for each square, i.e., for every pick, lift-off, via, set-down, and place points, should be calculated followed by the second joint angle of the manipulator for the same joint angles. The desired trajectory then could be obtained by interpolating these points. It is, therefore, a very tedious work to calculate all the joint angles each time. Therefore, an alternative technique should have to be found to eliminate this tedious calculation.

The first joint angle for pick, lift-off, and via points will be the same as well as for place, set-down, and via points since they all have the same orientation. Only thing is that it requires to specify the pick distance and place distance in z-direction over the x-y plane. The lift-off and set-down points can be obtained by adding the approach distance to pick point and place point respectively. Similarly the two via points are found by adding the safe distance to lift-off point and set-down point respectively.

The first joint angle for pick, lift-off and via points can be calculated by using the equation (3.48) derived in section 3.2.3. For example, let D1 be the pick point. Here the center point of squares are considered whose coordinates are known in Cartesian coordinate system. The first joint angle of the manipulator for D1 is as follows:

$$D1 : \theta_1 = \tan^{-1}(\frac{2}{2}) = 45^\circ \tag{6.1}$$

Here (2,2) is the coordinate point of D1.

Similarly the first joint angle for place, set-down, and via points can be calculated by using the same equation. That is, the first joint angles for all squares A1 through H8, either pick, lift-off, via points or place, set-down, via points are easily obtained.

The third joint angle has the same orientation for pick, lift-off and via points and different positions for pick, lift-off and via points. The third joint angle for pick, lift-off, and via points can be caculated by using the equation (3.52) or (3.55) derived in section 3.2.3. As can be seen from the both equations that three different values of Θ_3 will be found for pick, lift-off and via points depending upon the different values of z. If z equals the distance of the object centroid from the plane, i.e., $z = d^{pick}$, then it is the pick point. If z equals the height of the object plus the distance of the object centroid i.e. z = height of object + $d^{pick} = d^{lift\text{-}off}$, then it is the lift-off point. If z equals to $d^{lift\text{-}off}$ plus the safe distance, i.e., $z = d^{lift\text{-}off} + d^{safe} = d^{via}$, then it is the via point. The third joint angle for any square can be calculated as follows:

$$\theta_3 = \tan^{-1}(\pm\frac{\sqrt{1-D^2}}{D})$$
$$\text{where } D = \frac{r^2+z^2-a_2^2-a_3^2}{2a_2a_3} \tag{6.2}$$

Here r^2 equals $x^2 + y^2$ mentioned earlier. a_2 and a_3 are the upper arm and lower arm lenght of the manipulator respectively. Two solutions for θ_3 corresponds to the elbow-up position and elbow-down position, respectively. Only the elbowup solution is prefered, and it is recovered by choosing the positive sign in the equation.

For example, let H3 be the pick point. The third joint angle for pick point can be obtained by setting $z = d^{pick}$ in equation (6.2) as follows:

$$D = \frac{r^2+(d^{pick})^2-a_2^2-a_3^2}{2a_2a_3}$$

$$H3 : \theta_3 = \tan^{-1}\left(+\frac{\sqrt{1-D^2}}{D}\right) \tag{6.3}$$

r^2 equals $Hx^2 + HY^2$ in equation (6.3).

Similarly, the third joint angle for lift-off point can be found by setting $z = d^{lift\text{-}off}$ as follows:

$$D = \frac{r^2+(d^{lift\text{-}off})^2-a_2^2-a_3^2}{2a_2a_3}$$

$$H3 : \theta_3 = \tan^{-1}\left(+\frac{\sqrt{1-D^2}}{D}\right) \tag{6.4}$$

In the same way we can caculate the third joint angle for via point by setting $z = d^{via}$ as follows:

$$D = \frac{r^2+(d^{via})^2-a_2^2a_3^2}{2a_2a_3}$$

$$H3 : \theta_3 = \tan^{-1}\left(+\frac{\sqrt{1-D^2}}{D}\right) \qquad (6.5)$$

Similarly, the third joint angle can be calculated for place, set-down and via points by using the same equations and setting values for z = d^{place}, z = $d^{set\text{-}down}$ and z = d^{via} in the above equation respectively.

The second joint angle Θ_2 can be obtained by using the equation (3.57) derived in section 3.2.3. The second joint angle actually depends on values of z, and on the corresponding third joint angle Θ_3. If z = d^{pick} and corresponding third joint angle is θ_3^{pick}, then the second joint angle for pick point is as follows:

$$\theta_2 = \tan^{-1}\left(\frac{ra_3\sin\theta_3^{pick}-d^{pick}(a_3\cos\theta_3^{pick}+a_2)}{r(a_3\cos\theta_3^{pick}+a_2)+d^{pick}a_3\sin\theta_3^{pick}}\right) \qquad (6.6)$$

Similarly, by setting z = $d^{lift\text{-}off}$ and corresponding θ_3 the second joint angle for lift-off point can be found as follows:

$$\theta_2 = \tan^{-1}\left(\frac{ra_3\sin\theta_3^{lift}-d^{lift}(a_3\cos\theta_3^{lift}+a_2)}{r(a_3\cos\theta_3^{lift}+a_2)+d^{lift}a_3\sin\theta_3^{lift}}\right) \qquad (6.7)$$

In the same way second joint angle for via point is obtained by setting z = d^{via} and corresponding θ_3.

$$\theta_2 = \tan^{-1}\left(\frac{ra_3\sin\theta_3^{via} - d^{via}(a_3\cos\theta_3^{via} + a_2)}{r(a_3\cos\theta_3^{via} + a_2) + d^{via}a_3\sin\theta_3^{via}}\right) \tag{6.8}$$

Simialrly, the second joint angle for place, set-down and via points can be calculated in the same way.

6.4 Hardware

The Fischertechnik system consists of a three-axis articulated robot and an interface. The Fischertechnik interface is hooked up to the parallel printer port of the computer. A flat ribbon cable from the robot is attached to the interface. At the end of the cable is a 20-pin connector which fits into an adapter. The adapter carries a second connector which fits to the printer port of the computer. The interface expects an unfiltered direct current between 6 and 10 volts.

D.C. motors serve for driving the robot. The three large sized motors are used for actuating the robot's base, shoulder and arm respectively and the smaller motor causes the opening and closing of the gripper. D.C. motors have the features that they generate high torque at comparatively low current and produces quick motion of the robot. With other types of motors, D.C. motors share a common disadvantage that they can move the manipulator joints with the help of gear units and the position of the manipulator joint is only approximately known. Therefore, only the period during which the motors are active can be controlled by computers. Consequently, the same period of actuation must not always ensure the same displacement of the manipulator joint. Such a situation is not desired for a precision instrument like robot. Therefore, the positon of the manipulator joint must be measured by an independant control system.

6.4.1 Control System of Robot

Control of robot requires knowledge of the position and velocity of each joint. In general, the position of the robot's manipulator can be measured directly. We usually measure the position of each joint and apply the kinematic transform to find the position of the manipulator.

Measurement of position is easy with stepper motor driven joints. The angular position is known simply by knowing how far the joint has been commanded to move on either direction. But most often we are confronted with the problem of measuring an angle of D.C. motor. For such applications, some mechanism for sensing position is necessary. There are two different techniques used for sensing position of the joint: Digital technique and Analog technique.

By analog technique angular positions are measured with a potentiometer and A/D converter. Velocity can be determined by either differentiating the potentiometer signal or by sensing an analogue tachometer with A/D converter. Use of differentiating circuit is cheaper than tachometer but they are very noise sensitive. However, their use is not recommended.

There are a number of digital techniques to measure the position and velocity of the joints. Among the most used and more accurate technique is the optical shaft encoder and A/D converter to interface the computer. There are three basic type of optical shaft encoder:

- Absolute shaft encoder
- Incremental shaft encoder
- Tachometer

However, in some applications absolute shaft encoder is easier to use and incremental shaft encoder provide a high degree of resolution at relatively low cost.

Incremental Optical shaft encoder

The incremental encoder is a glass disc. It is imprinted only with one circular row of slots, all of the same size and distance apart. The number of slots in the circular row increase with the required resolution. Three sensors are used. Two of which are used to sense clockwise and counter clockwise directions. The third sensor is focused on a reference point which is used to determine the position. The length of time required for a single pulse to be completed represents the velocity. Brief description of incremental shaft-encoder can be found in [7].

Absolute Optical Shaft-encoder

This type of encoder consists of a circular glass disc imprinted with rows of broken concentric arcs of opaque material. A light source is assigned to each row with a corresponding detector on the opposite side of the disc. The arcs and sensors are arranged so that, as light shines through the disc, the pattern of activated sensors is a unique encoding of the position of the shaft. However, instead of representing the angular shaft-position in the standard set of sequential binary numbers, absolute encoders usually employ a special Gray code. An absolute encoder requires twelve or more separate sensors to gain good resolution and is therefore rather expensive.

Tachometers

Tachometers can only be used as angular position sensors in situations where the rotation of the shaft never reverses as tachometers cannot distinguish the sense of rotation. They are relatively simple, and therefore cheap devices and consists of n disc incorporating either n number of inserts made of ferrous metal to generate n number of optically reflective marks on which light is reflected and returned on a photo-transistor of the shaft, or n numbers of holes through which light passes and detected by the photo-transistor that generates n number of pulses per revolution of the shaft.

The angular resolution of a tachometer is given by

$$r = 360°/n \text{ degrees}$$

Thus, if we use an electronic pulse counter, angular position of the shaft may be measured. The angular velocity is given by

$$\omega = 2\pi \times f/n$$

Where f is the measured frequency of pulses, n is the number of marks on the disc.

6.4.2 Positioning System of the Fischertechnik Robot

The positioning system installed in the Fischertechnik robot for this purpose consists of fork-type photo-interrupter as shown in the figure 6.3. The driving shaft of the gear holds a cup-shaped wheel on the circumference of which black strips have been imprinted at a regular distance. There are a total of 32 black strips. Now the transmission type photo-interrupter overlaps the rim of the wheel. One side of the photo-interrupter is equipped with a light-emitting diode emitting infra-red light. The other side of the photo-interrupter is provided with a photo-transistor which is infra-red light sensitive. When the wheel rotates driven by the motors, infra-red light is interrupted by the black strips and photo-transistor produce a low signal at the output. Otherwise infra-red light passes through the disc to the photo-transistor and it produces a high signal at the output. With the running motor, a sequence of pulses will appear at the output.

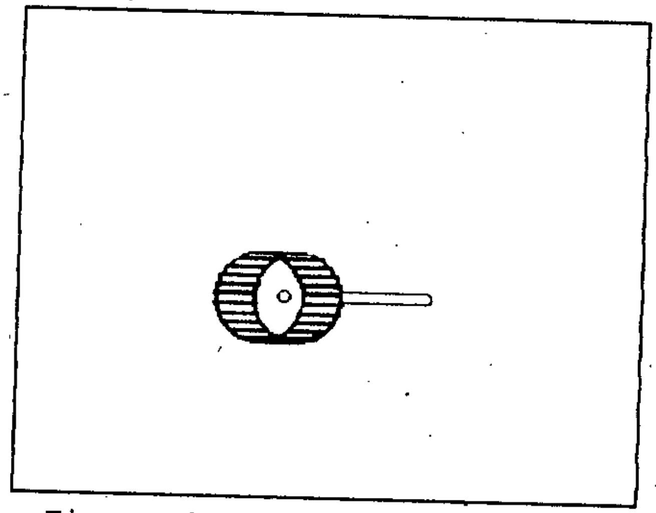
Figure 6.3 Optical encoder

6.4.3 Precision, Accuracy, Repeatability, and Positioning of the Fischertechnik Robot

The precision of a robotic manipulator is a measure of spatial resolution with which the tool can be positoned within the robot's work space. If the tool tip is positioned at point A, as shown in figure 6.4, and the next closest position that it can be moved to is B, then the precision along the dimension will be the distance between A and B.

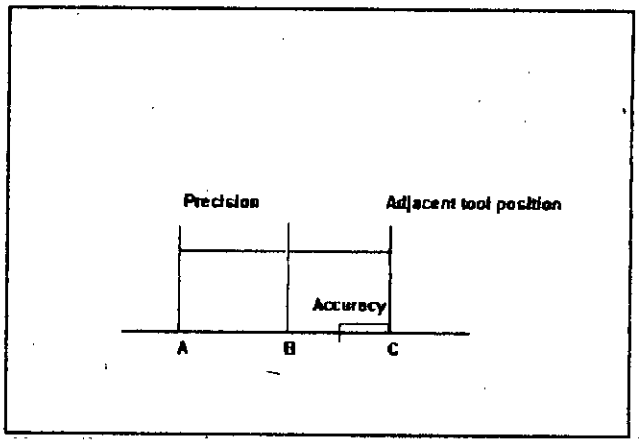

Figure 6.4 Precision and accuracy

More generally, in three-dimensional space the robot tool tip might be positioned anywhere on a three-dimensional grid points within the workspace. This grid or lattice of points where the tool tip can be placed is not uniform, because the shape of a horizontal grid element is not square and grid elements near the outside surface of the work space envelop are larger than grid elements near the inside surface of the work space. The overall precision of the robot is the maximum distance between neighboring points in the grid. With articulated robots, both the horizontal and vertical precision vary over the work space.

One of the virtues of an articulated robot is that the precision varies throughout the work space, the state of one axis influences the precision along another axis. That is, a small computational error, associated with one of the joints, can result in larger errors in the successive joints.

The best way to specify the precision of a robot is to specify the precision of individual joints. The precision or spatial resolution of the individual joint is affected by the drive system, the position sensors, and the power transmission, including gears, sprockets, and cables.

On many robots, increamental encoders are used to determine shaft position rather than absolute encoders such as potentiometers. Incremental encoders typically consists of a slotted disk with infrared emitter-detector pairs. As the shaft turns, the infrared beams are interrupted or not, depending upon the position of the slotted disk. If two or more emitter-detectors are used, then both the direction and the magnitude of rotary motion can be determined.

The accuracy of a robotic manipulator is a measure of the ability of the robot to place the tool tip at an arbitrarily precised location in the work space. Robotic arm accuracy is not nearly as easy to analyze as precision. Accuracy depends, for example, on mechanical inaccuracies such as gear backlash and deflection in components and in the links due to loading of the gripper. However, a simple bound can be placed on the error or inaccuracy, in terms of precision, namely

$$e \geq \frac{Precision}{2}$$

Here *e* denotes error or inaccuracy.

Repeatability is another important characteristics. Repeatability is the measure of ability of robot to position the tool tip in the same place repeatedly. On account of such things as gear backlash and flexibility in links, there will always be some repeatability error. The Fischertechnik robot shows a very poor repeatability and it is on the order of a milimeter.

6.5 Software

The Fischertechnik interface is normally operated under MS-DOS operating system version 2.0 and higher. A machine language program

called INTERFAC.COM which drives the interface is supplied with the system. The driver program itself is written in BASIC. The function of the driver program is to write a short machine language program to a memory location of the computer. The machine language program occupies the memory range from FF00 to FFFF. The memory range available to BASIC is thus shortened by 256 bytes which, however, does not normally cause a noticeable limitation. The memory location from FF00 to FFFF cannot be used for other purposes.
The machine language program generates the standardized parameters. A source listing of the machine language program is given in appendix B. Details of the hardware and software can be found in [32].

6.6 Development of the Algorithm

Since trajectory generation in WCS repeatedly needs kinematics which is computationally more time consuming, trajectory planning in JCS is intended. Obviously this requires a sufficiently large number of points to be transfered to JCS using inverse kinematics. To avoid large number of calculations which is computationally expensive, the following strategies are taken:

i) Do not involve the computer in calculation of the joint angles during execution since it slows down the operation of the program.

ii) Use predetermined values of the joint angles in the program to drive the robot faster.

The trajectory planning system is developed according to the algorithm in section 5.5 which is shown in figure 6.5.

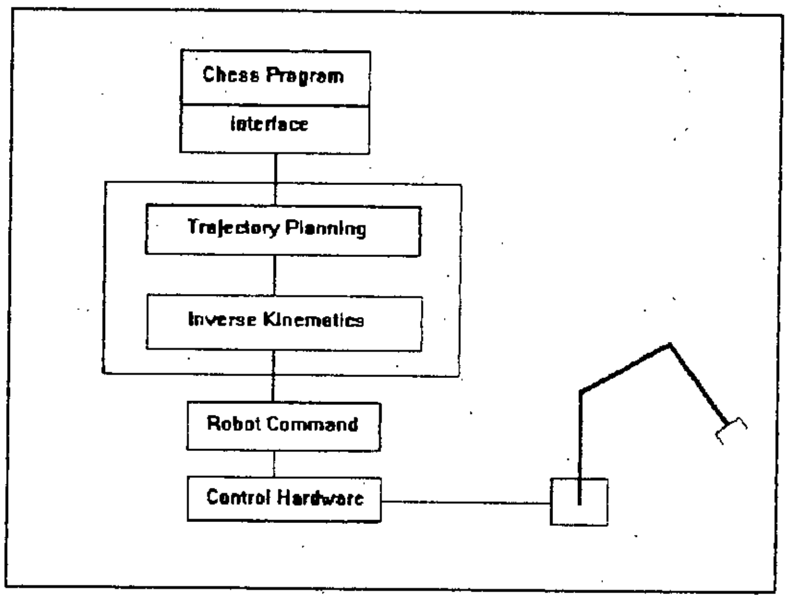

Figure 6.5 Trajectory planning system

Chapter Seven

Suggested Future Study and Conclusion

7. Suggested Future Study and Conclusion

The common feature of all robots which have been discussed so far is their ability to repeat a preprogrammed sequence of operations as long as necessary. However, these robots are unable to sense and respond to any change in their environment. For example, if a robot was programmed to grip an object at a certain point, the robot will always close gripper jaws at that point, even if an object is not there. If an obstacle is inserted in the robot's work space, the robot will collide with it and will not move around the obstacle to avoid collision.

In order to operate in a changing environment, the robots must be equipped with sophisticated sensors such as vision systems and have some degree of artificial intelligence (AI). Based upon the data received from sensors, the AI algorithm makes real-time decisions that might change the programmed sequence of operations of the robot.

7.1 Introducing Vision System

Most vision systems are equipped with one or two video cameras linked to a frame grabber board and processing system. The vision system takes a snapshot, digitizes the image and analyzes it to define the object.

The main applications of vision systems are in handling, assembly, object classification, and inspection. In handling and assembly the vision systems are used to recognize the position and orientation of objects to be handled or assembled. The systems can also determine the presence and absence of objects and detect particular features of objects e.g. diameter.

The simplest type of vision systems are able to identify isolated objects and their orientation against a flat surface. No overlapping objects are permitted, and the number of stable positions of each object is limited. Constant lighting conditions must be maintained so as to clearly define the silhouette of dark object on a white background or vice versa. Diffuse front lighting may prove satisfactory if the objects contrast with the background.

There are three basic types of cameras which are used in vision system: linear arrays, vidicons, and solid state cameras. Actually, in-depth treatment of these equipments is beyond the scope of the present work. Cameras can be mounted either on the robot or can be mounted externally. When the camera is mounted on the arm, the robot is moved based upon an error signal derived from the object's location within the field of view. If the camera is mounted on the robot arm, different views of the object can be observed by changing the location of the arm, and three-dimensional information can be obtained. A disadvantage of this type of system is that it requires a camera to be mounted on the robot where it is subject to rough handling. The camera also increases the robot arm loading and requires running extra caables to the robot. The alternative method of implementing visual feedback is to mount the camera externally and use it to observe the relative position between the robot end effector and objects in the work space. The error between the observed and desired positions is used to control the robot.

Finally, instead of discretizing the robot work space and coordinate points to locate object a vision system can be implemented that will provide the position and orientation of the object and guide the manipulator by using a nonoverlapping object algorithm.

7.2. Conclusion

Although robotic manipulators are not nearly as accurate as their expensive and more specialized hard automation counterparts, they do provide increased flexibility. It is one of great challenges of robotics to make use of this increased flexibility to compensate for less than perfect accuracy. In particular, clever algorithms and control strategies are needed that make use of external sensors to compensate for uncertainties in the environment and the uncertainties in the exact position of the robot itself.

All the techniques used in inverse kinematics are sensitive to coputational errors, mainly due to the coupling between the various axis parameters. That is, a small coputational error in one joint results in large errors in the successive joints.

The optical encoder used by the Fischertechnik robot allows the manipulator a positional increment of 5.62 degree in the rotary axis which would have been on the order of 0.01 degree. That is, an angular displacement smaller than 5.62 degree cannot be detected.

The light-emitting diode used by the optical encoder emits infra-red beam which must penetrate and pass through the plastic material of the photo-interrupter to produce the pulses. But sun-light and light of neo-tubes also contain a certain portion of infra-red light which interferes the pulses. Therefore, for the operation an optimal adjustment of the photo-interrupter should be sought.

In general, the accuracy of the present system will not be that good. The robot may gradually drift out of calibration with prolonged operation due to computational errors, positional accuracy and environmental effects. To maintain reasonable accuracy, robots are to be periodically reset to a standard hard home position using the limit switches.

References

References

[1] Douglas R. Malcolm, Jr.: Robotics: An Introduction; Breton Publishers, Boston, Massachusetts, 1985.

[2] Yu. Kozyrev: Industrial Robots Handbook; Mir Publishers, Moscow, English Translation, 1985.

[3] Yoran Koren: Robotics for Engineers; McGraw-Hill Book Company, International Edition, 1987.

[4] K. S. Fu; R. C. Gonzalez; C. S. G. Lee: Robotics: Control, Sensing, Vision, and Intelligence, McGraw-Hill Book Company, International Edition, 1987.

[5] Robert J. Schilling: Fundamentals of Robotics: Analysis and Control, Prentice Hall, Englewood Cliffs, New Jersey, 1990.

[6] F. L. Lewis; C. T. Abdallah; D. M. Dawson: Control of Robot Manipulators, Macmillan Publishing Company, 1993.

[7] Wesley E. Snyder: Industrial Robots: Computer Interfacing and Control, Prentice Hall Inc., Englewood Cliffs, New Jersey, 1985.

[8] Mark W. Spong ; M. Vidyasagar : Robot Dynamics and Control; John Wiley & Sons; New York, Chihester:, Brisbane Toronto Singapore; 1989.

[9] G. Hadley: Linear Algebra; Narosa Publishing House; New Delhi; Madras Bombay; Calcutta; 5th Reprint 1992.

[10] G. Hadley : Linear Programming : Narosa Publishing House Madras, Bombay; 2nd Reprint 1988.

[11] Goldenberg, A. A.; Benhabib, B., and Fenton, R. G.:A Complete Generalized Solution to the inverse Kinematics of Robots, IEEE J. Robotics and Automation, Vol. RA-, No. 1, pp. 14-20, 1985.

[12] J Pieper D. L. , The Kinematics of Manipulators under Computer Control: Ph.D. Theses Stanford University, 1968

[13] Shahinpoor, M. : The Exact Inverse Kinematics Solutions for the Rhino XR-2 Robot, Robotics Age, Vol.7,No.8,pp.6-14, 1985.

[14] J Paul, R. P., Shimano, 8.,and Mayer G. Kinematic Control Equations for simple Manipulators : IEEE Trans. System, Man, and Cybernetics, Vol. SMC-II, No. 6, 1981.

[15] Mikell P. Groover; Mitchell Weiss ; Roger N. Nagel; Nicholas G. Odery : Industrial Robotics Technology, Programming, and Applications : McGraw-Hill Book Company; International Edition; 1986.

[16] Kang G. Shin, and Neil D. Mckay : Minimum-time Control of Robotic manipulators with Geometric path constraints; IEEE Transactions on Automatic Control; Vol. AC-30; No. 6; June 1985. pp. 531-541.

[17] J. Y. S. Luh; Charles E. Campbell Jr. : Minimum Distance Collision-Free path planning for Industrial Robots with a Prismatic Joint; IEEE Transactions on Automatic Control Vol. AC-291 No. 8. August 1984. pp. 675-680.

[18] J. Y. S. Luh : An Anatomy of Industrial Robots and Their Controls; IEEE Transactions on Automatic Control, Vol.AC-28,No.2 February 1983.pp.133-153.

[19] Chun-Shin Lin ; Po-Rang Chang, and J.Y.S. Luh : Formulations and Optimization of Cubic Polynomial Joint Trajectories for

Industrial Robots; IEEE Transactions on Automatic Control Vol. AC-28; No. 12, December 1983. pp. 1066-1073.

[20] Zeungnam Bien and Jihong Lee : Minimum-Time Trajectory Planning Method for two Robots ; IEEE Transactions on Robotics and Automation; Vil. 8, No.3, June 1992. pp. 414-418.

[21] Bertrand Tondu and Loic Urbain : Functional Scheme of a Numerical Robot Controller Based on an On-line Trajectory Generation Approach; Proceedings of the 24th International Symposium on industrial Robots, November 1993, Tokyo, Japan. pp.

[22] H. Hatwal and A. K. Mallik : Tracking and Planning of Gross Motion of End-Effectors ; Robotics : Proceeding of National Workshop, April , 1987, Department of Science and Technology, Government of India.

[23] R. Bhattachariya, A. K. Dutta, S. K. Mukherjee and A. Bhattachariya : Robotics; Proceedings of National Workshop, April, 1987, Department of Science and Technology, Government of India.

[24] Sukumar Bhattachariya and Ashitava Ghosal : A Neural Architecture for Concurrent Generation of Configuration Space Trajectory ; Proceedings of the 1992 International Conference on Artificial Neural Networks (CANN-92), Brington, UK, 4-7 September, 1992. pp. 571-574.

[25] Yasuhiro Woob and Mitsuo Kawato : A Neural Network Model for Trajectory Formulation of Arm Movement by Using Forward and Inverse Dynamics Models; Proceedings of the 1992 International Conference on Artificial Neural Networks (CANN-92), Brington, UK, A-7 September 1992. pp. 575-578.

[26] M. E. Kahn and B. Routh : The Near-minimum-time Control of Open-loop Artificial Kinematic Chains ; Journal of Dynamic System and Measurements; Vol. 93; pp. 164-172 ; September 1979.

[27] C. Widdoes A. Heuristic collision Avoider for the Stanford Robot Arm; Stanford University; Stanford CA, C.S. Memo 227; 1974.

[28] S. M. Udupa : Collision Detection and Avoidance in Computer Controlled Manipulators; Ph.D. Dissertation, California Institute of Technology, 1977.

[29] T. Lozano-Perez: Automatic Planning of Manipulator Transfer Movements; IEEE Transaction on System, Man, and Cybernetics Vol. SMC-II, pp. 681-698; Oct. 1981.

[30] R. A. Brooks ; Solving the Find-path Problem by Good Representation of Free Space ; 2nd Annual National Conference on Artificial Intelligence ; Pittsburgh ; PA ; August 18-20, 1982, pp. 381-386.

[31] M. N. H. Siddique , A. K. M. Mostafa Kumal ; Mohammad Kaykobad :Obstacle-free Trajectory Planning of a Three-axis Articulated Robot for Pick-and-place Operation; Proceedings, Second Annual Paper Meet, Mechanical Engineering Division, IEB September 28-30, 1995, Rajshahi.

[32] Fischetechnik Computing Manual.

[33] T. S. Ganguli; S. Ghosh ; A. R. Saha : Design and Construction of Simple and Inexpensive Optical Encoder Controlled by Sophisticated Solid State Integrated Circuitry using Digital Resolution System for High-Technology Robot System Controller ; Proceedings of National Workshop on Robotics April 1987.

[34] A. K. M. Mostofa Kamal: Application of Robot Technology inIndustry - An Unavoidable Tredn to Compete in the Global Market; A Seminar Presentation, Organized by the Mechanical Engineering Division; Institution of Engineers, Bangladesh, Dhaka, October 10, 1994.

Appendix

APPENDIX A

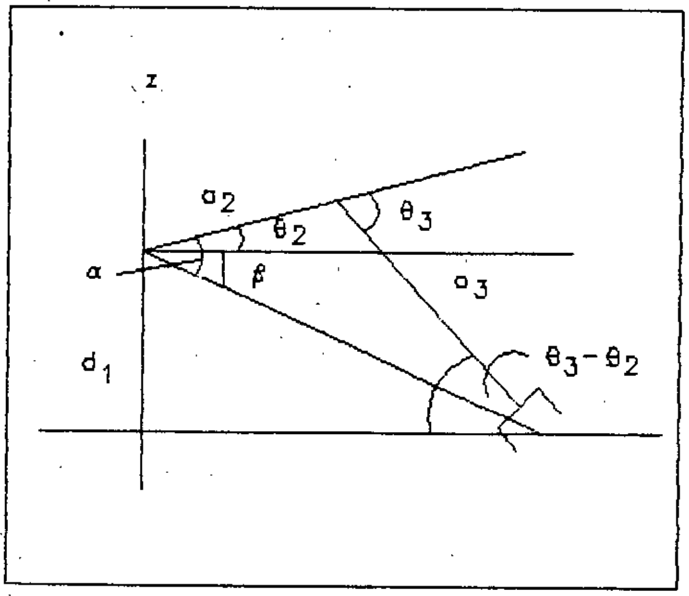

Figure A1 Projection onto plane formed by link2 and link3

From the figure A1 we can find the position of r as follows

$$r=a_2\cos\theta_2+a_3\cos(\theta_3-\theta_2)$$
$$z=a_2\sin\theta_2+a_3\sin(\theta_3-\theta_2)$$

Using the following trigonometric identities

$$\cos(A-B)=\cos A\cos B+\sin A\sin B$$
$$\sin(A-B)=\sin A\cos B-\cos A\sin B$$

We can rewrite the equations as follows

$$r=a_2\cos\theta_2+a_3(\cos\theta_2\cos\theta_3+\sin\theta_2\sin\theta_3)$$
$$z=a_2\sin\theta_2+a_3(\sin\theta_2\cos\theta_3-\cos\theta_2\sin\theta_3)$$

Squaring the both sides of the above equations and adding them together, we obtain

$$r^2+z^2=a_2^2+a_3^2+2a_2a_3\cos\theta_3$$

From the above equation we can find $\cos\theta_3$

$$\cos\theta_3=\frac{r^2+z^2-a_2^2-a_3^2}{2a_2a_3}$$

Defining α and β as in shown in figure A.1

$$\tan\alpha=\frac{a_3\sin\theta_3}{a_3\cos\theta_3+a_2}$$
$$\tan\beta=\frac{z}{r}$$

Using the following trigonometric identity

$$\tan(A-B)=\frac{\tan A-\tan B}{1+\tan A\tan B}$$

From the figure we find that

$$\theta_2=\alpha-\beta$$

Setting the values of alpha and beta we get

$$\tan\theta_2=\tan(\alpha-\beta)=\frac{\tan\alpha-\tan\beta}{1+\tan\alpha\ \tan\beta}$$

$$\tan\theta_2=\frac{ra_3\sin\theta_3-z(a_3\cos\theta_3+a_2}{r(a_3\cos\theta_3+a_2)}$$

APPENDIX B

To write programs in a language other than BASIC, speed them up through complex procedures in machine language, or to extend the functions of the interface the following ideas will be helpful.

The Fischertechnik interface handles a number of tasks which we would like to discuss with the aid of the block diagram shown in figure B.1. The signals from and to the computer are shown in the left side of the figure. There are four outputs M1, M2, M3, and M4 and ten inputs E1, E2, E3, E4, E5, E6, E7, E8, EX, and EY. It is clearly seen that the number of data lines available at the computer port is significantly lower than the number of lines required on the interface. This limited number of data lines must, therefore, be employed in such a way so that all signals to the interface can be controlled easily. The concept employed is that of multiple use of the data lines with the aid of shift regiters. In this way, for example, only three data lines are required for controlling the output. A parallel connection scheme would have required eight data lines. Let's take a closer look at connections M1 to M4. The data lines required are designated DATA OUT, CLOCK, and LOAD OUT. If there is an output, the data for all four motors are transmitted in each case, i.e., a whole byte because each of four motors requires two bits for controlling the direction of rotation. The motor outputs to which the signal does not apply are thus once supplied with the current state which is buffered in the computer as an output word.

The bits of the output word are sequentially fed (with the most significant bit first) to the DATA OUT line. When the signal at the CLOCK output goes from low to high, the bit is transferred to the shift register, and then the next bit at DATA OUT follows and is likewise transferred to the shift register with the next pulse. The previous bit has been shifted one position to the right in the

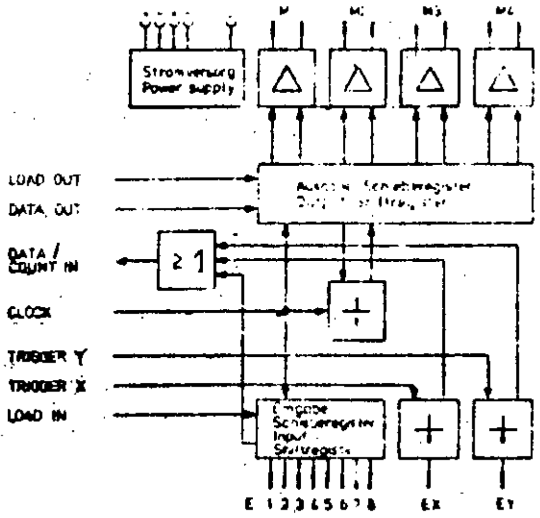

Figure B1: Block diagram of the Interface

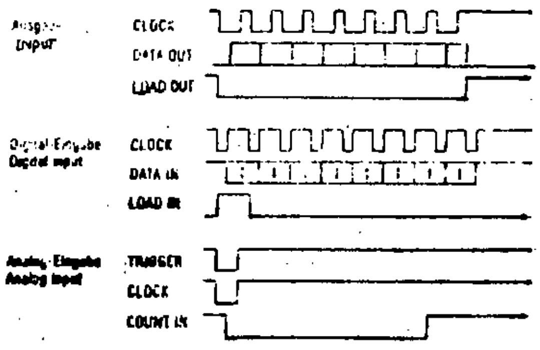

Figure B2: Pulse diagram of the Interface

shift register to make room for the subsequent bit. After a total of eight such data transfers, the whole output word has been transferred to the shift register. The bit first transferred has been shifted all the way to the right in course of data transfer. The activity of the data transfer has no effect on the outputs of the shift register. The output amplifiers are not directly controlled by the shift register, rather via an in-line storage register which is integrated in the shift register module. When the LOAD OUT line goes from low to high, only then data are transferred to storage register. The timing of the signals are shown in the pulse diagram.

Transfer of data to power amplifiers depends on the enabling circuits which is again controlled by a monostable multivibrator. The circuit generates an enabling signal with a duration of half a second if there is a pulse on the CLOCK line. That is, the power amplifiers receive a signal first since the data were just transferred with the aid of the CLOCK line. If no more data are transmitted within the next half second, the monostable multivibrator will flip back to stable state and the enabling signal is removed. The monostable multivibrator can be retriggered, i.e., the time of half a second is always calculated from the time of the last CLOCK pulse.

The monostable multivibrator has an enabling input. The output to the amplifiers can be immediately inhibited via this input. On the Fischertechnik interface this occures when an invalid data pattern which would command the connected motor to simulteneously rotate clockwise and counter clockwise, applies at the output of the storage register.

To control the transfer of the digital signals to inputs E1 through E8 is reversal of the output process described above. The output signal LOAD IN causes the transfer of the data applying at the inputs to the input shift register. This involves all eight inputs, even though only one of them is to be interrogated. When applying to the shift register, each pulse of the CLOCK line will cause the transfer of one bit on the input line DATA IN, the bit from E8 first and then one from E1 last. By testing this line, the computer can collect the bits and reassemble them into a data word. The desired bit is subsequently filtered out and transferred to the BASIC program.

Since the same CLOCK line is used for data transmission as for output, the digital input will also activate the monostable multivibrator, which controls the enabling signal for the output data. Malfunctioning of the output shift register caused by the multiple function of the CLOCK line is not to be expected since the current output data are not contained in the output shift register, but in the storage register. The former is controlled by the CLOCK pulses, unlike the latter, which only reacts to the LOAD OUT signal.

For the analog inputs EX and EY, Potentiometers or other variable resistors are used as timing elements in two additional monostable multivibrators circuits. A low resistance value is converted to a short pulse and a high resistance is converted to a pulse with long duration. The pulse is itself is triggered by the starting signals TRIGGER-X and TRIGGER-Y (with negative logic), respectively, and then appears in COUNT IN line. A machine language program determines the pulse duration by means of the number of loops which can be executed during the pulse duration. This number is fed back to the BASIC program which calls this function. There is no direct relationship between the analog value and the angle position of the resistance of the potentiometer. The clock rate of the microprocessor is, however, involved. The AT microprocessor can go through a greater number of loops during the duration of the pulse than the XT microprocessor. For this reason, the count is adapted to the required value range by shifting it to the right. There is a linear relationship between the number determined at the end and the resistance. If required, this value can be converted into angular degrees or resistance values by means of calibration.

The connection between the interface and the computer's parallel printer port is given as follows.

Connection of the interface to the printer port:

Interface	Printer port	Pin number
OV	GND	17-25
LOAD-OUT	Data 1	2
LOAD-IN	Data 2	3
DATA-OUT	Data 3	4
CLOCK	Data 4	5
TRIGER X	Data 4	6
TRIGER Y	Data 5	7
DATA/COUNT-IN	Busy	11

Out of the eight data lines of this interface, the lower six lines are used for the output signals discussed above. The parallel printer port has only one input line available to all PC's. This is the active signal from the printer, the BUSY line. There is no conflict, if all input lines are combined using an OR citcuit. Since the machine language program knows which input function it has been requested. It has the capabilty of interpreting the signals on this input line correctly. The following machine language program controls the process described above. For the parameter transfer to the machine language program and back, CALL and USR functions are used in BASIC respectively.

The following source program uses the printer port address of 03BCH which is stored at address FFFA and in the subsequent memory addresses. This can be modified by the program if required due to the configuration of the computer. When the Fischertechnik interface is used with faster microprocessors, the required number of right shift commands is inserted by the program.

```
;=============================================================
;Interface Driver Routine
;Version 1.2
;File INTERFAC.COM
;
;Control of the Fischertechnik Interface:
;Output and Input using the CALL command the USRO function. This
;program requires a Centronics parallel printer interface as I/O
;port. The machine language program is located above the BASIC
;program and will not be destroyed any command like NEW or LOAD.
;The program requires 256 bytes of RAM taken from BASIC memory.
;
;Output Control:
;
;CALL  [motor number](operation)
;motor numbers are m1, m2, m3, and m4
;Operations are CW for Clockwise, CCW for Counter Clockwise and
;MOFF for Motor off
;
;CALL INIT initializes the interface and switches of all motors
;
;Input Control:
;Digital Input Control Commands:
;
;USR(digital input line)
;Digital input lines are E1, E2, E3, E4, E5, E6, E7, and E8
;Analogue Input Control Commands:
;USR(analogue input line)
;Analogue input lines are EX, and EY
;========================================================
;Routine for Output control
;Entry via CALL command
;=======================================================
     MOV BX,0100     ;INIT
     CLI             ;DISABLE INTERRUPT
     JMP LAB1
     MOV AL,03       ;OUTPUT BITS FOR MOTOR 1
```

```
      JMP LAB2
      MOV AL,0C        ;OUTPUT BITS FOR MOTOR 2
      JMP LAB2
      MOV AL,30        ;OUTPUT BITS FOR MOTOR 3
      JMP LAB2
      MOV AL,C0        ;OUTPUT BITS FOR MOTOR 4
LAB2:
      CLI              ;DISABLE INTERRUPT
      MOV BL,[FFBE]    ;GET PREVIOUS VALUS
      OR  BL,AL
      PUSH BP
      MOV BP,SP
      MOV SI,[BP+06]   ;GET OPERATION CODE
      MOV AX,[SI]
      POP BP
      MOV BX,00
      AND AH,AL
      XOR BL,AH        ;EVALUATE MOTOR BIT
LAB1:
      MOV [FFBE],BL    ;SAVE NEW VALUE
;==================================================
;ROUTINE FOR INTERFACE CONTROL
;OUTPUT CONTROL
;OUTPUT BIT PATTERN IN BL
;USES AL, BL, CX, AND DX
;==================================================
      MOV CX,0008      ;LOOP COUNTER
      MOV DX,[FFBA]    ;ADDRESS OF THE PRINTER PORT
      MOV AL,30        ;STATIC BIT PATTERN FOR MOTORS
LAB3:
      RCL BL,1         ;LEAST OUTPUT WORD
      JNB LAB11        ;DATA-OUT LOW
      OR AL,04         ;OUTPUT WORD IS SEQUENTIALLY FED TO
                                   ;DATA-OUT WITH CLOCK PULSE
 LAB11:
      OUT DX,AL        ;OUTPUT TO PORT
      OR AL,08          ;DATA ARE TRANSFERRED WITH  CLOCK
      OUT DX,AL         ;SEND BITS TO PORT
      LOOP LAB3         ;END OF LOOP
      MOV AL,39         ;OUTPUT IS STORED IN SHIFT REGITER
       ;VIA A STORAGE REGISTER WHEN LOAD-OUT LINE GOES HIGH
      OUT DX,AL         ;OUTPUT TO PORT
      STI               ;ENABLE INTERRUPT
      CMP BH,01         ;TEST WHERE YOU CAME FROM
      JZ LAB4
       RETF 0002        ;RETURN FROM MOTOR MN OPERATION
LAB4: RETF              ;RETURN FROM INIT
;=============================================================
;THIS PORTION IS NOT USED IN THE PROGRAM
;ROUTINE FOR INTERFACE CONTROL
;INPUT CONTROL
;ENTRY VIA USRO FUNCTION
;=============================================================
      CLI               ;DISABLE INTERRUPT
      MOV AL,[BX]       ;GET ARGUMENT OF USR0
      MOV DX,[FFBA]     ;ADDRESS OF PRINTER PORT
      CMP AL,A0         ;ANALOGUE INPUT
      JZ  LAB5          ;GO TO LAB5
      CMP AL,90         ;ANALOGUE INPUT
      JZ  LAB5
      MOV [FFBF],AL     ;SAVE INPUT BIT MASK
;=============================================================
;INTERFACE CONTROL
;USES AL, CX, AND DX
;=============================================================
      MOV CX,0008       ;loop counter
      MOV AL,32         ;LOAD-IN high
      OUT DX,AL         ;output to port
      OR  AL,08         ;set CLOCK
lab8:
      OUT DX,AL         ;output to port
      RCL AH,1          ;clear LS bit
```

```
     MOV DX,[FFBC]  ;address busy-line
     IN  AL,DX      ;read DATA-IN
     AND AL,80      ;mask DATA-IN
     CMP AL,80      ;test DATA-IN
     JNZ lab7       ;DATA-IN high
     OR  AH,01      ;DATA-IN low
lab7:
     CLC
     MOV DX,[FFBA]  ;address of printer port
     MOV AL,30      ;static bit pattern
     OUT DX,AL      ;output to port
     MOV AL,38      ;set CLOCK
     OUT DX,AL      ;output to port
     LOOP lab8      ;end of loop
     MOV AL,[FFBF]  ;get bit mask
     AND AL,AH      ;mask bit
     CMP AL,00      ;test bit
     JZ  lab9       ;return 0
     MOV AL,01      ;return 1
lab9:
     MOV [BX],AL    ;deposite in FAC
     STI            ;enable interrupt
     RETF           ;return from USRO
;==========================================================
;Analogue input
;The argument of the USRO function either being 90 (input EX)
;or AC (input EY) will lead the program flow to this sub-routine
;Uses AX, CX, and DX
;==========================================================
lab5:
     MOV CX,00FF    ;loop counter
     OUT DX,AL      ;triger one-shot
     MOV AL,38      ;set CLOCK
     OUT DX,AL      ;out-put to port
     MOV DX,[FFBC]  ;address busy line
     IN  AL,DX      ;read COUNT-IN
     RCL AL,1          ;test COUNT-IN
     JNB lab10         ;COUNT-IN low
     LOOP lab6         ;continue testing
     MOV AX,03FF       ;evaluate analogue value
     SUB AX,CX
     NOP               ;space for SAR AX,1
     NOP
     NOP
     NOP               ;space for SAR AX,1
     MOV [BX],AL       ;deposite in FAC
     MOV [BX+1],AH
     STI               ;enable interrupt
     RETF              ;return from USR0
;--------------------------------------------------------------
;Address of printer port FFBA
;Address of busy line FFBC
;Memory location of output variable FFBE
;Memory location of bit mask FFBF
;------------------------------------------------------------
```