DESIGN OPTIMIZATION AND SYNTHESIS OF MANIPULATORS BASED ON VARIOUS MANIPULATION INDICES

by

SANDIPAK BAGCHI

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2005

ACKNOWLEDGEMENTS

I would like to take the opportunity to sincerely express my gratitude to Dr. B.P. Wang for constantly supporting me all through my thesis tenure. The thought provocation suggestions and ideas provided by him helped me a lot in consolidating my thesis work. His Design Optimization course was the foundation for my design of manipulators. It was not just a support, but a constant source of inspiration that motivated me to step forward with new ideas.

I thank Dr. K.L. Lawrence for serving on my committee and also for his wonderful course in Finite Element Methods, so as to apply different techniques for the manipulator design and simulation. I would also like to express my appreciation to Dr. S. Nomura for his course in Mathematics that enabled me to build a strong foundation in analyzing and understanding the mathematical background of the design process. I would also like to express my thanks to my friend Aditya Apte for providing me with his optimization code to verify my results and also for his support at various stages of this research.

July 20, 2005

ABSTRACT

DESIGN OPTIMIZATION AND SYNTHESIS OF MANIPULATORS BASED ON VARIOUS MANIPULATION INDICES

Publication No. _____

Sandipak Bagchi, M.S.

The University of Texas at Arlington, 2005

Supervising Professor: Dr. B.P.Wang

Design optimization of serial manipulators involves striking a balance between an appropriate joint angle and exact link lengths. The optimization technique used in this thesis uses a unique algorithm that optimizes the joint angle through an Inverse Kinematics program. The synthesis problem involves setting up an accurate design length so as to reach a set of given target points inside the workspace without any singularity. The optimization process based on task specification focuses on the maximization of manipulability index, the objective function subjected to link length and joint angle constraints. The results obtained were plotted and animated to visualize the link movements in achieving the target locations.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Optimal Design of Manipulators

A **robot** is a mechanical device which performs automated physical tasks, either according to direct human supervision, a pre-defined program or, a set of general guidelines using artificial intelligence techniques. A robot is used to carry out certain tasks that are repetitive, difficult or dangerous for human beings. The choice of a robotic mechanism depends on the task to be performed, and consequently is determined by the position of the robots and by their dimensions and structure. This selection is done by intuition and experience; therefore, it is important to formulate a quantitative measure of the manipulation capability of the robotic system, which can be used in robot control and trajectory planning. In this perspective, Yoshikawa presented the concept of manipulability measure [7].

General task in any optimization process is to maximize the beneficial quantity or minimize the undesirable effect of certain parameters. Appropriate problem definition and its conjunction with suitable optimization tool can significantly improve the quality of product from technical and economical perspective [30]. The application of optimization concepts to the inverse kinematics solution is not only time efficient but also yields almost a precise solution. The use of FMINCON generates data that are compliant with the feasible solution and also the precision and accuracy of the end

effector is highly improved. This method has been applied in a nested format where the process starts with an initial guess for the design link lengths and subsequently evaluates the joint angle for each link with the assumed design length through inverse kinematics algorithm. In a way, this technique optimizes the link length as well as the joint angle at the same time.

However, the factor that determines the optimization process is the manipulability index, which is a measure of the manipulating ability of robotic end effectors as stated by Yoshikawa [8]. This property has been utilized to obtain better design parameters and also to obtain the best postures with the computed joint angles. The manipulator motion with the designed lengths and optimized joint angles can be visualized through animation where the robotic system is programmed to trace a given set of user defined locations.

1.2 Theoretical Background

Due to their scalability, numerical techniques often form a part of an inverse kinematics solver. So far research into the field of kinematics has failed to find a general non-numerical solution to the problem. Many researchers have proposed hybrid techniques yet these still rely on a numerical aspect. Meredith and Maddock proposed the Jacobian-based Inverse Kinematics solver as a real time solver [13]. This technique makes use of incrementally changing joint orientations from a stable starting point towards a configuration state that will result in the required end effector being located at a desired position in absolute space.

Design optimization of a robot (or a manipulator) has gained a lot of research interests as depicted in the works of various researchers. Design of manipulator links based on workspace optimization is reflected from the works of Shaik and Datseris [19]. According to them, for a fixed length of a generalized manipulator including both prismatic and revolute joints, optimization of workspace volume leads to specific manipulator configurations.

A method for planning of robotic assembly by numerical optimization of position and joint controller has been described by Prokop and Pfeiffer [20]. Khatami and Sassani [21] considered the kinematic isotropy as a performance evaluation criterion for optimal design of robotic manipulators. They developed a genetic algorithm to solve the minimax optimization formulation of robot design in order to find the optimal design parameters such as link lengths of the best isotropic robot configurations at optimal working points of the end effector [21]. One of the first measures for determining the specifications of a robotic mechanism was proposed by Yoshikawa [5]. Chedmail and Ramstein [31] used a genetic algorithm to determine the base position and type of a manipulator to optimize workspace reachability.

Prokop, Dauster and Pfeiffer suggested a model based optimization approach in order to deal with the impacts, friction and constrained motion while the robot interacts with the environment while manipulation [22]. This method is based on the detailed dynamic model of the manipulator, process dynamics of the task and interactions between these two. A simple index for the manipulator kinematic design optimization and best posture determination is presented by Mayorga and de Leon based on a simple

upper bound for a standard condition number of the Jacobian matrix [32]. A new method for computing numerical solutions based on combination of two nonlinear programming techniques and forward recursion formulas, with joint limitations of the being handled separately has been developed by Chun, Wang and Chen [23].

Vijaykumar, Tsai and Waldron demonstrated generic optimization of manipulator structures for working volume and dexterity. They have applied geometric optimization of manipulator structures based on kinematic geometry [24]. Manocha and Canny made use of algebraic properties of the system and symbolic formulation for reducing a multivariate problem to an univariate one and thus developing an efficient algorithm for inverse kinematics for a general 6R manipulator.

Manipulator design based on the manipulability index has been used as a major criterion in the optimal design of a robotic mechanism [10, 8, 11, 17, 12] and it has been proved to be a successful approach. Park and Kim presented a differential geometric analysis of manipulability for robot systems containing active and passive joints taking into account the dynamic characteristics of the robot system and manipulated object [33]. The concept of decomposed specific manipulability to solve inverse kinematic equation has been presented by Lee and Won [15].

1.3 Outline of Thesis

Manipulator Kinematics involving forward and inverse kinematics, link transformations, DH parameter definition for a manipulator (or a robot), the existence and uniqueness of inverse kinematics solutions and their solvability are discussed

briefly in Chapter Two of this thesis. It also takes into account the manipulability index and a brief applications and different types of the same.

Chapter Three focuses into various optimization techniques for solving constrained and unconstrained optimization problem and a brief analysis of these techniques, their applications and some advantages as well as disadvantages too.

Chapter Four is concerned with the problem formulation, the constraints and how they affect the optimization of the manipulator design. It also includes the assumptions for various design parameters and a precise algorithm for the optimal design process.

Chapter Five primarily discusses the simulation results with various case studies (such as different kinds of revolute manipulators) and their animation results. It compares the results obtained from various techniques of constrained optimization applied and selection of more accurate results.

Chapter Six focuses on the further development and modification and other criteria to be included so as to obtain a complete optimal design for a manipulator, such as manipulators with prismatic joints.

1.4 Software Tools

The software development for coding and programming is supported by MATLAB®. It helps in developing the optimization codes for generating the values for the design link lengths and the joint angles. It is considered to be a quite fast and accurate programming tool. MATLAB not only assists in producing results through

simulations, but also is a powerful tool for animating the motion of the robotic mechanism from the computed results. The animated motion enables us to obtain a detailed visualization of the manipulator inside the workspace in reaching the target locations when subjected to pre-defined constraints. The plots generated in each case can be utilized for data interpretation and analysis.

1.5 Design Optimization

The design optimization problem is formulated in two steps. In the first step, the design lengths are generated using a local optimization technique, fmincon in this case and then we use the optimized design length to generate a set of joint angles for reaching the set of user defined target points, satisfying the constraints in each case as well as the objective function too. However the optimized joint angles can be verified using the global optimization techniques. Animation of the results with optimized joint angles and design length enables the proper visualization of link orientations and positions.

The flowchart below represents the algorithm for optimal design

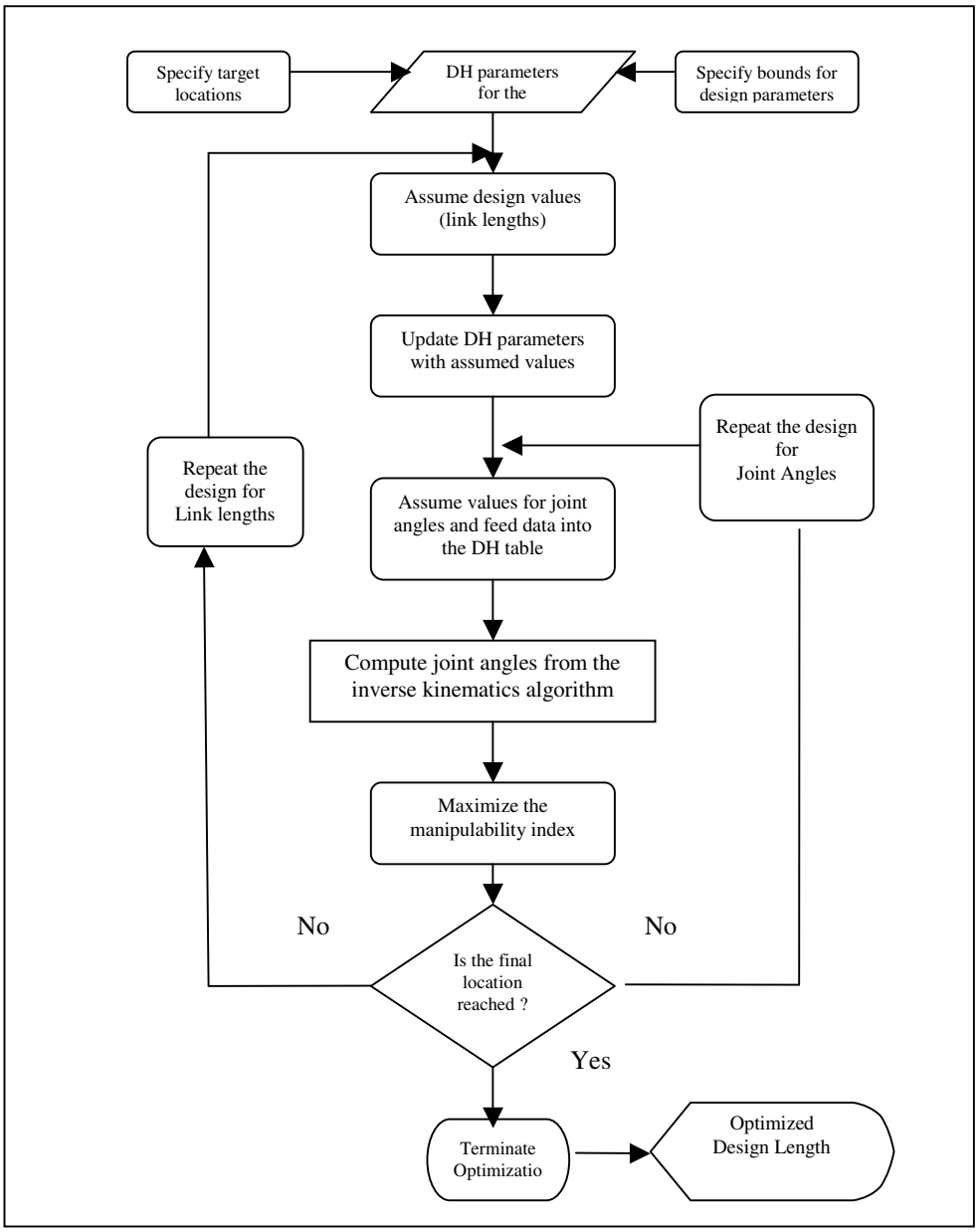

Figure 1.1 Flowchart for the optimization process

CHAPTER 2

MANIPULATOR KINEMATICS

2.1 Forward Kinematics

Kinematics is the science of motion which treats motion without regards to the forces causing it. The study of kinematics deals with the position, velocity, acceleration and all higher order derivatives of the position variable (with respect to time or any other variable). Hence kinematics basically refers to the geometric and time based properties of motion what we call position and orientation of a manipulator. The objective of Forward Kinematics is to determine the cumulative effect of the entire set of joint variables.

The basic concept of the manipulator lies in the fact, that it is composed of a set of rigid bodies, known as links, connected together at various joints. The joints are generally of two different categories, such as revolute (relative motion about a single axis) or a prismatic joint (linear motion, such as extension and contraction about a particular axis). In some cases, the joint can also be cylindrical, planar, screw or spherical. A revolute or a prismatic joint has only a single degree of freedom of motion. And the manipulator as a whole can be considered to have n-degrees of freedom, modeled from n-joints of one degree of freedom connected with n-links.

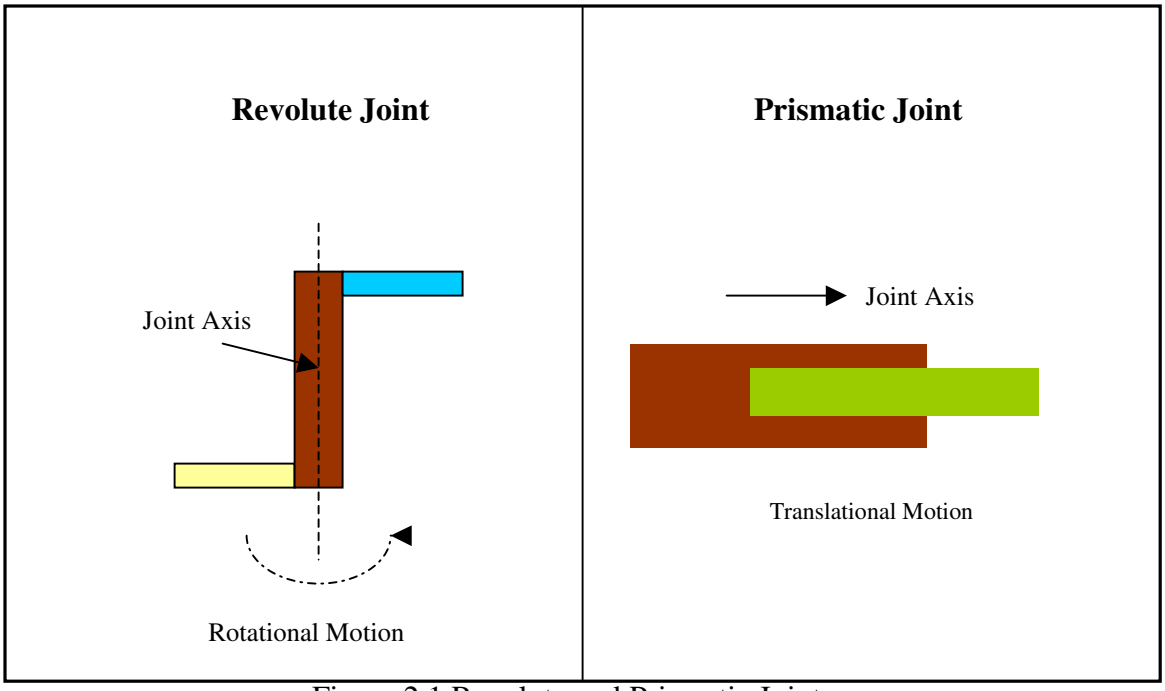

Figure 2.1 Revolute and Prismatic Joints

2.1.1 Link Parameters

Consider a manipulator consisting of n links connected serially by n-1 joints, with a single degree of freedom each, which may either be a revolute or a prismatic joint. As shown in Figure 2.2, the links are numbered 1, 2 …n, starting from the base or the reference frame. The base being considered to be immobile might be called as link 0. The first moving body is link 1 and so on to the free end of the arm, which is link n.

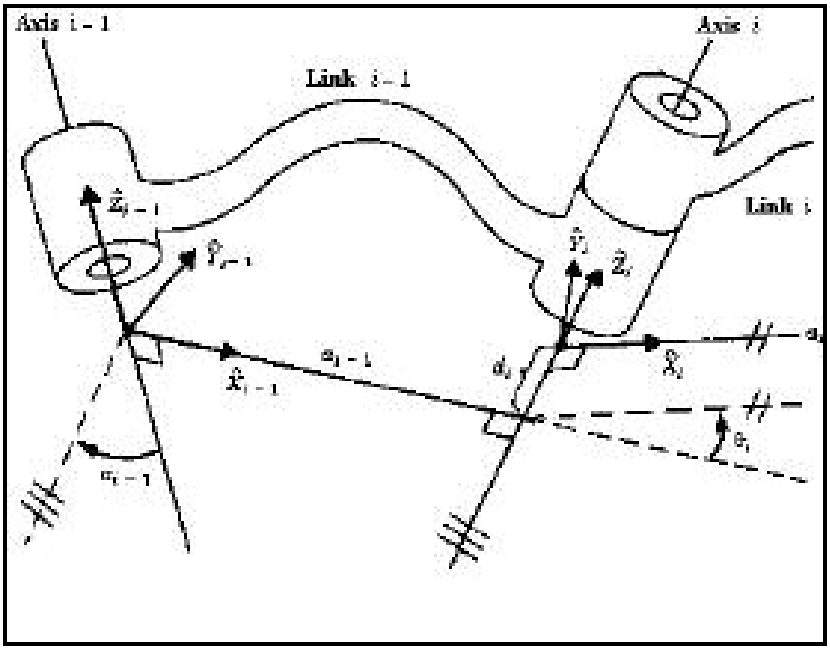

Figure 2.2 Link and Joint Configurations

For each joint i, the joint axis is defined as the vector with respect to which the joint either rotates (in case of a revolute joint) or translates (as in the case of a prismatic joint). However a link can be specified with two parameters which define the location of two axes in space. The distance is measured along a line that is mutually perpendicular to the both axes. Figure 2.2 shows link i-1 and the mutually perpendicular line along which the link length, a_{i-1} is measured. The other parameter that defines the relative location of the two axes is called the link twist, α_i. If we imagine a plane whose normal is the mutually perpendicular line just constructed, we can project both the axes i-1 and i onto this plane and measure the angle between them. The angle is measured from axis i-1 to axis i in the right hand sense about a_{i-1}. The relative positional reference between links i-1 and i at joint i can be described by the distance d_i between the feet of the two common normals on the joint axis i. The second factor that

determines the amount of rotation about this common axis between one link and its neighbor is called the joint angle, θ_i.

Figure 2.3 shows the interconnection of link i-1 and link i.

If a joint is revolute d_i is constant and θ_i is used to express the rotational angle of the joint; but if joint i is prismatic θ_i is constant and d_i represents the translational distance of the joint. As a result of this when the joint is revolute θ_i is adopted as the joint variable q_i, and when the joint i is prismatic we adopt d_i. The other three variables are constant and are called link parameters. This methodology of describing link mechanisms using $a_i, \alpha_i, d_i, \theta_i$ is called the Denavit Hartenberg notation.

2.1.2 Link Transformations

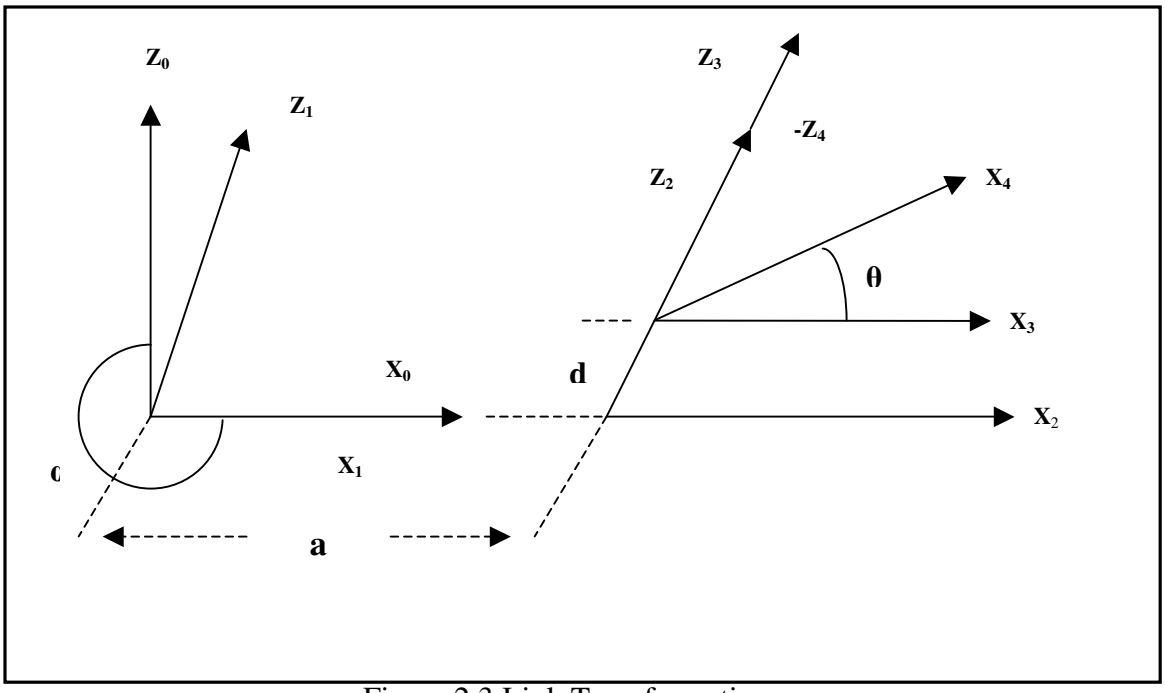

Figure 2.3 Link Transformations

The Denavit Hartenberg parameters are defined as follows:

a = The distance from original frame Z_{i-1} to Z_i along X_{i-1}

α = The angle measured from Z_{i-1} to Z_i along X_{i-1}

d = Distance from X_{i-1} to X_i along Z_i

θ = Angle from X_{i-1} to X_i along Z_i

The transformation between two adjacent frames can be represented as given below:

T = Rot(X_0, α). Trans(X_1, a). Trans (Z_2, d). Rot (Z_3, θ)

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\alpha & -s\alpha & 0 \\ 0 & s\alpha & c\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & a \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta & -s\theta & 0 & 0 \\ s\theta & c\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The final transformation matrix can be represented as a matrix multiplication of the individual matrices in the above equation.

$$^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & \alpha_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1}d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The main objective of forward kinematics is to obtain the final transformation matrix through the multiplication of individual transformation matrix and developing kinematic equations.

The transformation that relates the frame N to the frame 0 is given as

$$ {}^{0}_{N}T = {}^{0}_{1}T \; {}^{1}_{2}T \; {}^{2}_{3}T \; \ldots\ldots\ldots\ldots \; {}^{N-1}_{N}T $$

2.2 Inverse Kinematics

While forward kinematics is based on the manipulation with the structure, that is done by changing joint angles (in case of revolute joint) or relative displacement (in case of prismatic joint) within the controlled structure Figure 2.4, Inverse Kinematics is based on the direct manipulation with the end of the structure and the joint angles are derived from changes of the end effector of the manipulator Fig [4].

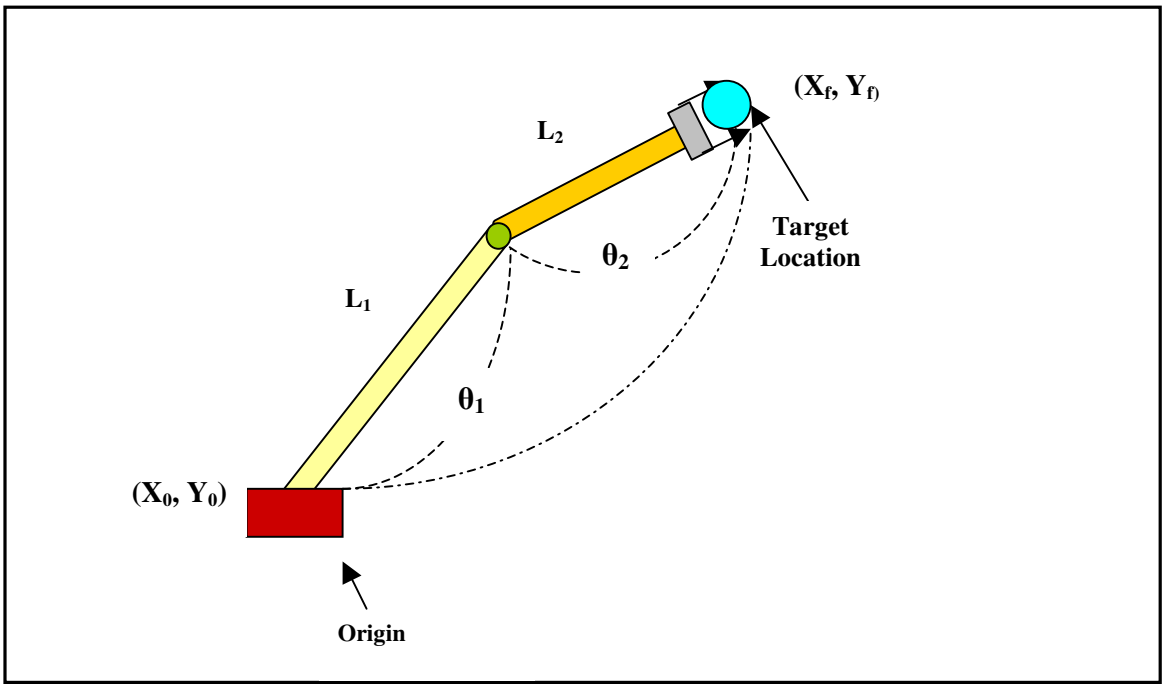

Figure 2.4 Inverse Kinematics Concept

Figure 2.1.1 illustrates the basic method for the inverse kinematics procedure for a general two-link manipulator, given the length of the links. The origin and final location along with link lengths determine the joint angles.

Since inverse kinematics makes it possible to manipulate the articulated structure by the end effector, it could be used in animation techniques for easy motion control. Inverse kinematics is used in several fields of applications and computer graphics is one of them that include animation. A number of methods and their combinations can be used to solve the inverse kinematics. Using each of them brings some advantages and disadvantages. Therefore, it is useful to combine them with various additional approaches.

2.2.1 Solvability

The problem of solving inverse kinematics of a manipulator is solving a non-linear set of equations. Given the numerical value of ${}^{0}_{N}T$ (the final transformation matrix that relates the end effector to the base frame) we determine the values of θ_1, θ_2, …..θ_N from inverse kinematics. However, the numbers of equations arising from the transformations are more than the number of variables. And it is important to group these equations so as to solve for θ. These equations are nonlinear, transcendental and sometimes very difficult to solve. Hence we should be aware of the existence of solution, multiple solutions and the method of solution.

2.2.2 Existence of Solution

Existence of solution comes into play when the workspace of the robot is taken into consideration. Workspace of a robot/manipulator is defined as the region in the space that is reachable by the end effector. It can be classified into two different types:

Dexterous Workspace: It is the volume of space that the robot end-effector can reach with all orientations [Craig]. That is at each point in the dexterous workspace, the end-effector can be arbitrarily oriented.

Reachable Workspace: It is defined as the volume of space which the robot can reach with at least one orientation. Dexterous workspace is a subset of the reachable workspace.

2.2.3 Method of Solution

Due to the non-linearity of the equations involved in solving the manipulator kinematics, it is very difficult to generate some general purpose algorithm for solving the inverse kinematics of manipulators. We can extend the concept of solvability only when it is possible to determine the values of all the joint variables at certain given position and orientation.

Generally, we come across two different methods of solution strategies:

Closed form solution and *Numerical* solution. Numerical solutions are iterative in nature and much slower as compared to their closed form solution. “Closed form” means a solution method based on analytic expressions or on the solution of a polynomial of degree 4 or less, such that non-iterative calculations suffice to arrive at a solution [1]. The closed form solution procedure can further be classified into two different

categories: Algebraic and Geometric. In this problem the results are determined using numerical optimization method and also verified with the closed form solutions.

2.3 Manipulability

Among the various performance indices proposed, the concept of service angle first introduced by Vinogradov et al. (1971), and the conditioning of robotic manipulators, as proposed by Yang and Lai (1985), finally emerged out to be manipulability, being introduced by Yoshikawa [5]. Manipulability is the measure of the manipulating ability of a robotic mechanism in positioning and orienting the end effectors [8]. The notion of manipulability has been particularly useful in the analysis and design of robot manipulators. Manipulability, or the ability to move and apply forces in arbitrary directions, has been used in applications ranging from the kinematic design of robotic fingers to the optimal positioning of the workpiece in a robot's workspace.

Various factors should be taken into consideration for an optimum robot manipulator design for performing a given task in the workspace – the factor that enables changing the position and orientation is known as Manipulability. Manipulability however helps in determining the structure and organization of manipulators suited for various task performances. Yoshikawa's proposed concept of manipulability can be mathematically represented as

$$M = \sqrt{det(\boldsymbol{J}\boldsymbol{J}^T)}$$

where $\boldsymbol{J}$ is the Jacobian matrix and depends on the configuration of the manipulator given by joint vector $\boldsymbol{q}$.

For evaluating quantitatively this ability of manipulators, we need to extend our knowledge to the concepts of manipulability ellipsoid and manipulability measure.

2.3.1 Manipulability ellipsoid

Let us consider a manipulator with 'n' degrees of freedom. The joint variables represented by n-dimensional vector 'q' and position/orientation represented by another m-dimensional vector 'r', such that $r = [r_1, r_2 \ldots\ldots r_m]^T$ and $(m \leq n)$. The kinematic relation between q and r is assumed to be

$r = f_r(q)$

The relation between velocity vector *v* corresponding to *r* and the joint velocity *q'* is

$v = J(q)\dot{q}$

where $J(q)$ is the Jacobian Matrix.

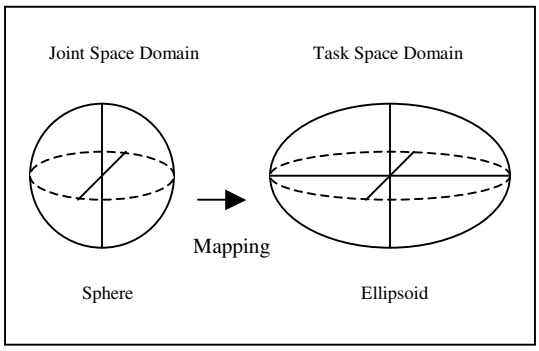

Figure 2.5 Manipulability Ellipsoid

Let us consider the set of all end effector velocities that are realizable by join velocities such that

$$\| \dot{q} \| = (\dot{q}_1^2 + \dot{q}_2^2 + \ldots\ldots\ldots\ldots + \dot{q}_n^2)^{1/2}$$

satisfies $\| \dot{q} \| \leq 1$. This set is an ellipsoid in the m-dimensional Euclidean space [10]. The end effectors can move at higher speed along the major axis compared to that of the minor axis. Only in some special cases where the ellipsoid is a sphere the end effector can move uniformly in all directions. Since this ellipsoid represents an ability of manipulation, it is called manipulability ellipsoid and is represented as follows

$$\{v \,/ v^T (J^+)^T J^+ v \leq 1 \text{ and } v \quad R(J)$$

where J^+ is the pseudo-inverse of the

matrix J [2], and $R(J)$ denotes the range of J.

The manipulability measure w has the following properties:

(i) $w = \sqrt{\det J(q) J^T(q)}$

(ii) When m = n, then w reduces to $w = | \det J(q) |$

(iii) Generally $w \geq 0$ holds and

$w = 0$ if and only if rank $J(q) \leq m$, which implies the manipulator to be in singular configuration.

The size and shape of the ellipsoid are used to determine the amplification between joint space and task space.

2.3.2 Manipulability in Conventional Task space

Manipulability can be classified into two kinds of dexterity in motion:

global manipulability and specified manipulability [15]. The global manipulability is concerned with the ease of arbitrary changing of position and orientation of the end

effector while the specified manipulability relates to the changing of position and orientation along a particular direction in the workspace.

2.3.3 Classification of Manipulability

Manipulability can be classified as Kinematic manipulability and Dynamic manipulability. The principal objective of kinematic manipulability lies in the quantification of velocity and force transmission characteristics of the manipulator for performing a specific task in a particular orientation and position. But the Dynamic manipulability is concerned with the acceleration of the end effector in arbitrary directions. This concept was developed by Yoshikawa.

2.3.4 Advantages and Disadvantages of Manipulability

The advantages of manipulability are as explained below:

I. The manipulability measure helps in design and control of robots and task planning as they yield quantitative measure of the easiness of arbitrary changing of position and orientation of the manipulator.
II. The other benefit is the measure of fast recovery ability from the escapable singular point for redundant manipulators.

But however there are some disadvantages of manipulability too.

I. Scale and Order dependencies. These factors prevent fair comparison among manipulators with different dimension and make it impossible to derive the physical meaning of manipulability [11].
II. The Jacobian that relates the joint space to the task space is dependent on the configuration of the manipulator and is considered to be a local performance

index that varies randomly which in turn makes the manipulability index variable too.

III. The Jacobian that relates the joint space to the task space is dependent on the configuration of the manipulator and is considered to be a local performance index that varies randomly which in turn makes the manipulability index variable too.

2.3.5 Global Manipulability

To define the posture independent of kinetostatic index, we need a global index instead. This can be done in the same way as that of a magnitude of a vector is defined as a sum of squares of its components, where the distribution of manipulability is considered over the entire workspace. However the value of manipulability is not constant and it depends on the specific location in the workspace. Integrating any of the various local manipulability measures over the task space yields the global manipulability index.

$$M_{global} = \frac{\int_W M dw}{\int_W dw}$$

where W is the workspace and dw is a differential area in which the manipulator is considered as constant.

The closer to unity the index M implies better the manipulability behavior of the system.

2.3.6 Uniformity of Manipulability

Uniformity of manipulability over the entire workspace is defined as the ratio of the minimum and maximum values of manipulability.

$$U = \frac{M_{\min}}{M_{\max}}$$

where M_{min} *and* M_{max} are the minimum and maximum values of manipulability.

Uniform manipulability coupled with global manipulability enables global evaluation of performance.

2.3.7 Applications

Manipulability is proved to be a true measure of accuracy performance. It can also be used in the numerical analysis to estimate position and orientation error of platform caused by the amplified position error of the actuator [11].

2.4 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. *Repeatability* is a measure of how closely a manipulator can return to a previously taught point [4].

Most present day manipulators are highly repeatable but not very accurate. Since there is no direct measurement of the end effector position and orientation. As a result of this one must rely on the assumed geometry of the manipulator and its rigidity to infer the end effector position from the measured joint angles. Accuracy is therefore affected by computational errors, machining accuracy in the construction of the manipulator,

flexibility effects such as the bending of links under gravitational and other loads, gear backlash and a host of other static and dynamic effects. Accuracy as defined as the ability of the robot to precisely move to a desired position in 3-D space as defined by Conrad, Shiakolas and Yih and also depicted in Figure 2.4.1. Absolute accuracy and repeatability describe the ability of a robot to move to a desired location without any deviation. Dynamic accuracy and repeatability describe the ability of a robot to follow a desired trajectory with little or no variance. Additionally, as in all robotic applications zero overshoot is a necessity to avoid disastrous collisions with other parts in the work-cell. Ideally, both the absolute and dynamic accuracy and repeatability can be minimized to the attainable resolution [39].

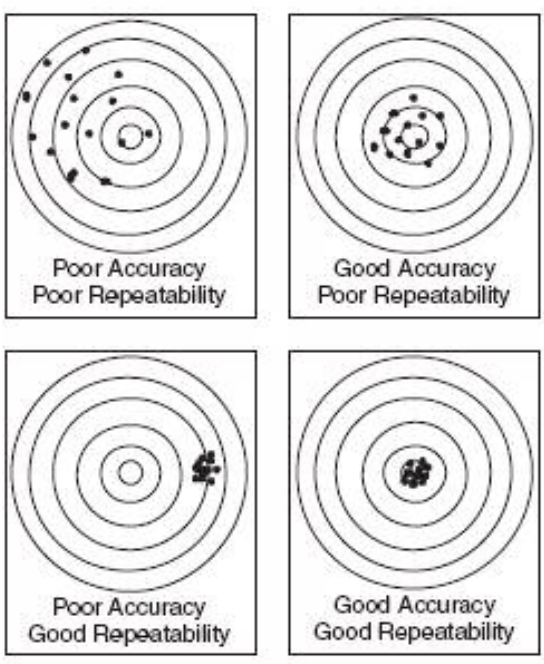

Figure 2.6 Accuracy and Repeatability [39]

From Figure 2.6 we can see in the first case that the manipulator traces certain points that are way apart from the desired center location, indicating that both the accuracy as

well as repeatability are poor. However, in from the second section of Figure 2.6, it is quite prominent that although the points lie close to the desired location yet the repeatability is not improved. But things are just opposite in the next sub section of the above figure where the repeatability increases drastically at the expense of accuracy causing the points to be located at a considerable distance from the center taught point. But the last one in the figure is an ideal case where the manipulator traces back to the same position every time and being centered about the desired location.

2.5 Animation

To '*animate*' is literally '*to give life to*'. 'Animating' is moving something which can't move itself. Animation adds to graphics the dimension of time which vastly increases the amount of information which can be transmitted. In order to animate something, the animator has to be able to specify, either directly or indirectly, how the 'thing' is to move through time and space. The basic problem is to select or design animation tools which are expressive enough for the animator to specify what they intend, yet at the same time are powerful enough fro animators from specifying any details that the animator is not interested in. The appropriateness of a particular animation tool depends on the effect desired by the animator. An artistic piece of animation will probably require different tools that an animation intended to simulate reality.

Computer based animation can be classified into *computer-assisted animation* and *computer-generated animation*. Computer-assisted animation refers to system

consisting of one or more two dimensional planes that computerize the traditional animation process. While computer generated animation specify the motion in three dimensional environment. The motion specification for computer-generated animation is divided into two categories: *low level techniques* (techniques that aid the animator in precisely specifying motion), and *high level techniques* (techniques used to describe general motion behavior).

Low level techniques consist of techniques, such as shape interpolation algorithms (in-between), which help the animator fill in the details of the motion once enough information about the motion has been specified by the animator. When using low level techniques, the animator usually has a fairly specific idea of the exact motion that he or she wants.

High level techniques are typically algorithms or models used to generate a motion using a set of rules or constraints. The animator sets up the rules of the model, or chooses an appropriate algorithm, and selects initial values or boundary values. The system is then set into motion, so to speak, and the motion of the objects are controlled by the algorithm or model. The model-based/algorithmic approaches often rely on fairly sophisticated computation, such as physically based motion control.

Images convey a lot of information because the human visual system is a sophisticated information processor. It follows, then, that moving images have the potential to convey much more information. When animation is recorded for later viewing, it is typically presented in film or video formats by recording a series of still

images. This is possible because the eye-brain assembles a sequence of images and interprets them as a continuous movement. Persistence of motion is created by presenting a sequence of still images at a fast enough rate to induce the sensation of continuous motion.

2.5.1 Real-Time versus Single Frame Animation

Animation can either be generated in *real-time* or in *single-frame mode*. Real-time implies that the images are being generated at a fast enough rate to produce the perception of persistence of motion. For general purposes, the rate capable of producing this perception is usually taken to be 1/24th of a second; the actual rate depends on the types of images being viewed and on the specific viewing conditions. If the imagery cannot be produced at a fast enough rate to provide real-time animation, then it can be generated a single frame at a time and each frame can be recorded on some medium so that it can be played back at animation rates later (i.e., rates fast enough to produce persistence of motion). The difference between **real-time animation** and **single-frame animation** is dependent on image quality, the computational complexity of the motion, and the power of the hardware that is being used to calculate the motion and render the images. Model-based motion control algorithms can require processing that is too intense to be done in real-time. Sometimes it is possible to pre-compute the motion and then render in real-time.

In this case some applications of animation are carried out based on single frame animation. The animation of the link movements of the manipulator both in two and three dimensions depending on the location of the target points enables the observer to

have a pretty clear idea about the trajectory and motion sequence. It is basically considered to be a tool for verifying the simulation results with the optimized design [6].

CHAPTER 3

OPTIMIZATION TECHNIQUES AND PROBLEM FORMULATION

Formulation of an optimum design problem involves transcribing a verbal description of the problem into a well defined mathematical expression. The formulation process however depends on the design variables that are used to describe a system [J S Arora]. All systems are designed to perform within a given set of constraints which includes limitations on resources, material failure, response of the system, member sizes, etc. The constraints must be influenced by the design variables of the system. So the design variables and the constraints play a complimentary role in the process of design optimization. Another criterion called the objective function determines the performance of a particular design with respect to the other. It is influenced by the variables of the design problem, i.e. it must be a function of the design variables.

3.1 Methods of Optimization

3.1.1 Steepest Descent Method

This is the simplest and oldest method of computing the search direction for unconstrained optimization problem. It exploits the properties of gradient and objective function [JA]. The earliest reference to this method is given by Cauchy.

Let us consider a design vector X_k, we need to choose a downhill direction $\mathbf{d}$ and then a step size $\alpha > 0$ such that a new design variable $\mathbf{x_k + \alpha d}$ is better than $\mathbf{x_k}$ and f($\mathbf{x_k +}$

$\boldsymbol{\alpha}\mathbf{d}) < f(\mathbf{x_k})$. The direction vector is given by $\mathbf{d_k} = -\nabla\ \mathbf{f}\ (\mathbf{x_k})$ and once it is determined we can easily evaluate $\boldsymbol{\alpha}$ by minimizing the objective function $f\ (\boldsymbol{\alpha}) \equiv f\ (\mathbf{x_k} + \boldsymbol{\alpha}\mathbf{d})$. However, this algorithm is not very efficient, so it is not recommended for general applications.

3.1.2 Conjugate Gradient Method

This is a simple and effective modification of the Steepest Descent Method. This method was first implemented by Fletcher and Reeves. In steepest descent method the directions at two consecutive steps are orthogonal to each other. This tends to slow down the steepest descent method in spite of being convergent. But in case of conjugate gradient method the directions are not orthogonal to each other and tend to cut diagonally through the orthogonal steepest descent directions. And hence this process improves the rate of convergence. The conjugate gradient directions $\mathbf{d^{(i)}}$ are orthogonal with respect to a symmetric and positive definite matrix $\mathbf{A}$, i.e. $\mathbf{d^{(i)T}\ A\ d^{(j)}} = \mathbf{0}$ for all i and j and $i \neq j$.

3.1.3 Newton's Method

In the steepest descent method we used only first order derivative information in the definition of the objective function but introduction of a second order derivative not only improves the search direction but also enhances the rate of convergence. As this method has a quadratic rate of convergence it uses Hessian matrix for the function. It basically uses Taylor's series expansion of the function about the current design point.

3.1.4 Simulated Annealing

In the conventional methods of minimization, we seek to update a point when the function has a lower value. This strategy generally leads to a local minimum. A robust method for seeking global minimum must adopt a strategy where a higher value of function is acceptable under some conditions. Simulated annealing provides such a strategy [BC-1]. This method is similar to random search method but the difference lies in the selection of the design vector.

3.1.4 Genetic Algorithm

It is defined as an optimization technique that revolves around the genetic reproduction process and 'survival of the fittest' strategies [1]. As compared to conventional techniques, GA uses a population by population approach to evaluate many individuals in parallel. Due to this parallelism they are less likely to be deceived by false optimum [D-13]. It is based on a natural selection and natural genetics [N-13]. GA follows the natural order of obtaining the maximum. The problem formulation for this algorithm can be mathematically formulated as:

Maximize $f(x)$

Subject to $l_i \leq x_i \leq u_i \; i = 1 \text{ to } n$

This evaluation technique consists of creating an initial population, an evaluation phase where the values of the variables are extracted. The subsequent stage is the creation of a mating pool through reproduction process where the weaker members are replaced by stronger ones based on fitness values. This step is followed by a crossover operation in

the same mating pool to produce off springs that undergo random mutation to generate highest fitness value of the variables.

GA is considered to be an efficient and powerful tool in the optimization process, which provides robust solution to continuous and discrete problems [1]. This process doesn't require any gradient or higher order differentiation as used in direct search techniques.

3.1.4 Differential Evolution

Differential Evolution is an improved version of GA for faster optimization. Unlike simple genetic algorithm that uses binary coding for representing problem parameters, DE uses real coding of floating point numbers [2]. Differential Evolution as developed by Rainer Storn and Kenneth Price (1996) is one of the best evolution algorithms, and is proven to be a promising candidate to solve real valued optimization problems [N-4]. It is considered to be a search type algorithm. Among the DE's advantages are its simple structure, ease of use, speed and robustness [2].

3.2 Problem Formulation

We use numerical optimization technique to determine the theta value taking into consideration the manipulability and errors. We start from an initial guess of theta given the upper and lower bounds for the same, minimize the least square objective function using an optimization subroutine. A brief algorithm used to solve this problem is shown below.

3.2.1 Solution of Inverse Kinematics by Optimization

In the case of inverse kinematics the desired end effector positions are known (the points defined by the user within the workspace) and θ (joint variable) values are unknown. We need to determine the optimum value of θ, so as to achieve the best possible. We can use forward kinematics concept to calculate the error between the desired and the final location of the end effector.

Let, $P_{desired}$ = The desired end effector location

P_{actual}= Actual location of the end effector

The given problem is to minimize, given $P_{desired}$, to find the joint angles θ, minimizing

$$f = \sum \left(P_{\text{desired}} - P_{actual} \right)^2$$

Equation () is an unconstrained minimization problem. It can be solved as follows:

1. Define the Denavit-Hartenberg (DH) parameters for the specific type of robot or manipulator.
2. Define the desired locations of the end effector
3. Start with an initial guess for all the joint angles θ.
4. Define the bounds for each of the joint angles within the workspace.

5. The function FMINCON automatically evaluates the joint angle from the transformation matrix given in the equation () for the minimum value of the function.

The factor that is taken into consideration for the feasible manipulator design is known as manipulability. The manipulability measure or manipulability index is given as the square root of the determinant of the Jacobian and Jacobian transpose which is given in the equation (). As a result of this, we consider manipulability index to be our objective function which has to be maximized for the optimum design of link parameters, which in this case is the link length.

3.2.2 Robot Synthesis problem by Optimization

Now the optimum problem can be formulated can be presented in a different way as explained in the following algorithm.

Manipulability index: $m = \sqrt{\det\left(J.J^T\right)}$

1. Define DH parameters, final desired location.
2. Initial guess and bounds for the joint angles are made as above.
3. Initial guess for the Design parameters are also made.
4. FMINCON is used to minimize the objective function

$$\boxed{f = -m^2}$$

this leads to the maximization of the manipulability index taking into factor the given constraints.

5. The manipulability index function now calculates the maximum manipulability, the final position and Jacobian for the optimal design.

3.3 Sequential Quadratic Programming for Design

SQP (Sequential Quadratic Programming) methods represent the state of the art in nonlinear programming methods. This method has been implemented and tested which reveals that it outperforms every other tested method in terms of efficiency, accuracy, and percentage of successful solutions, over a large number of test problems [C/B-1].

This method is pretty close to Newton's method for constrained optimization as is applied for unconstrained optimization. At each major iteration, an approximation is made of the Hessian of the Lagrangian function using a quasi-Newton updating method. This is then used to generate a QP subproblem whose solution is used to form a search direction for a line search procedure. The general method, however, is stated here. The principal idea is the formulation of a QP subproblem based on a quadratic approximation of the Lagrangian function. Since SQP may converge to local minimum, the accuracy of this algorithm decreases with increase in number of variables.

The algorithm is as depicted in the following flowchart.

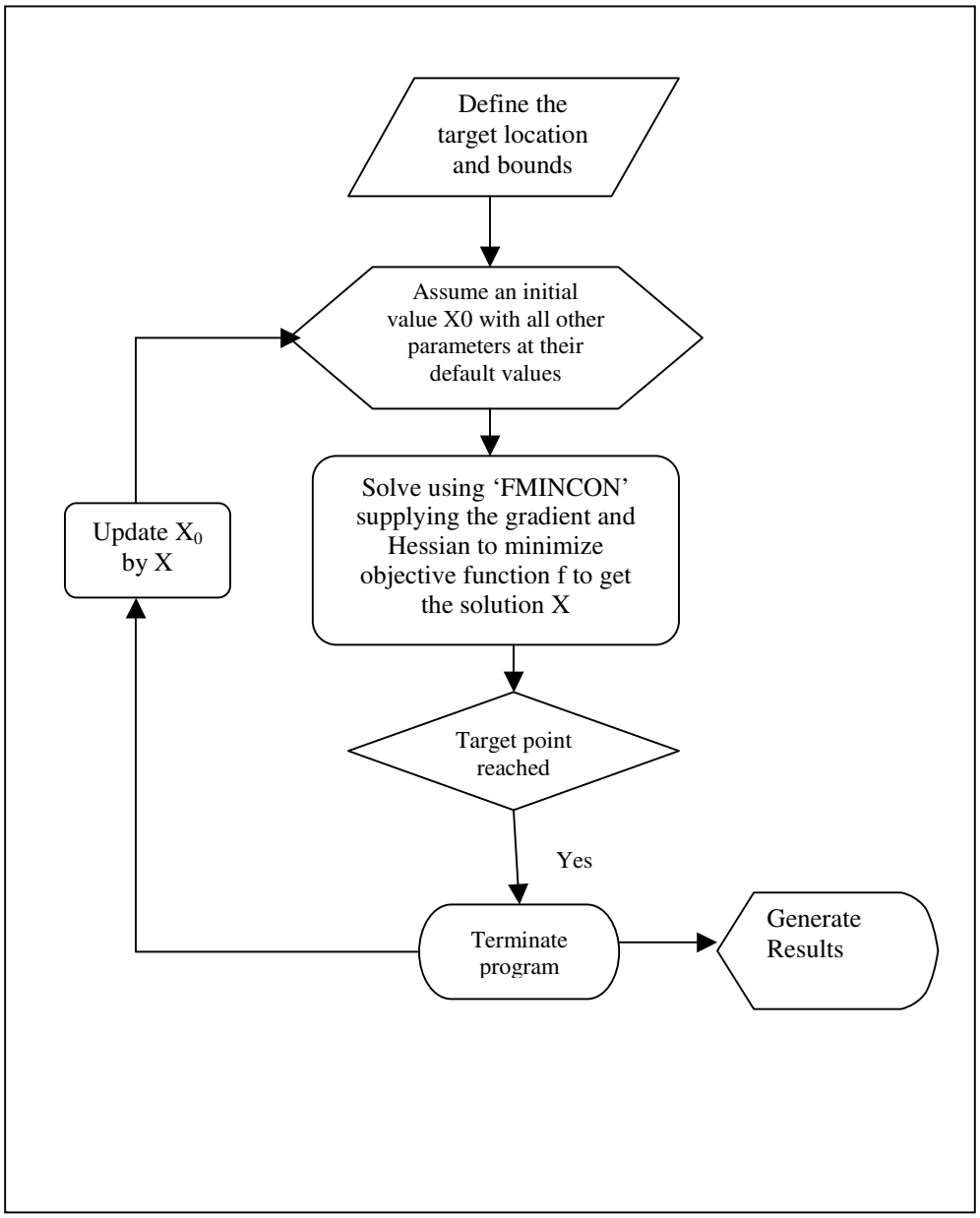

Figure 3.1 Flowchart for design optimization

3.4 FMINCON

Fmincon finds a constrained minimum of a scalar function of several variables starting at an initial estimate. This is generally referred to as *constrained nonlinear optimization* or *nonlinear programming*.

Find a minimum of a constrained nonlinear multivariable function

$$\min \; f(x)$$

subject to

$$c(x) \le 0$$
$$c_{eq}(x) = 0$$
$$A.x \le b$$
$$A._{eq}\, x = b_{eq}$$
$$lb \le x \le ub$$

where *x, b, beq, lb,* and *ub* are vectors, *A* and *Aeq* are matrices, *c(x)* and *ceq(x)* are functions that return vectors, and *f(x)* is a function that returns a scalar. *f(x)*, *c(x)*, and *ceq(x)* can be nonlinear functions.

3.5 Problem Definition

Our problem can be formulated on the basis of FMINCON as follows

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options,P1,P2,...)

where fun = objective function that maximizes the manipulability index

x0 = initial value of the design variable (i.e. D0)

A, b, Aeq, Beq = Null vectors in this case

lb = lower bounds for the design variable, min link lengths or the joint angles

ub = upper bounds for the design variable max link lengths or the joint angles

nonlcon = no nonlinear constraints considered in this case, hence null vector

options = maximum no of iterations

P1 = DH parameters

P2 = Final target locations

P3 = Optional parameter (the optimized length for inverse kinematics used only to calculate the joint angles)

P1, P2, P3 being the problem dependent parameters passed on directly to the function.

The constraints specified in the computation of the design link lengths and joint angles from inverse kinematics are represented in the tables [3.5.1 and 3.5.2] below:

Table 3.1 Constraints for the Link Lengths

CONSTRAINTS	SCARA	3-LINK ELBOW	6-LINK ELBOW	PUMA 560
LINK LENGTHS	**[L1 L2]**	**[L1 L2 L3]**	**[L1 L2 L3 L4 L5 L6]**	**[L1 L2 L3 L4 L5 L6]**
UPPER BOUNDS	25 , 15	22, 15, 10	35, 20, 20, 10, 10, 6	40, 30, 20, 5, 5, 10
LOWER BOUNDS	0, 13	15, 8, 5	15, 10, 10, 4, 5, 2	25, 20, 10, 5, 5, 5

Table 3.2 Constraints for the Joint Angles

CONSTRAINTS	SCARA	3-LINK ELBOW	6-LINK ELBOW	PUMA 560
JOINT ANGLES	**[θ_1 θ_2]**	**[θ_1 θ_2 θ_3]**	**[θ_1 θ_2 θ_3 θ_4 θ_5 θ_6]**	**[θ_1 θ_2 θ_3 θ_4 θ_5 θ_6]**
LOWER BOUNDS	-90, 0	-90, 0, 0	-90, 0, 0, 0, 0, 0	-90, 0, 0, 0, 0, 0
UPPER BOUNDS	360, 90-	360, 90, 90	360, 180, 180, 90, 90, 90	360, 180, 180, 90, 90, 90

CHAPTER 4

MANIPULATORS FOR CASE STUDY: A BRIEF DESCRIPTION

4.1 SCARA Manipulator

The SCARA (Selective Compliant Articulated Robot for Assembly) as shown in figure [], which is aimed mainly for assembly has gained popularity over the last few years. A SCARA planar manipulator consists of two planar revolute joints that are controlled by motor drives and a prismatic joint at the end that is mainly used for positioning the work-piece. The DH table for this is also modeled as presented in the table. The constraint values for the joint angles and the design parameters, i.e. the link length.

The schematic diagram for the SCARA workspace is as shown below.

Figure 4.1 SCARA (from ADEPT)

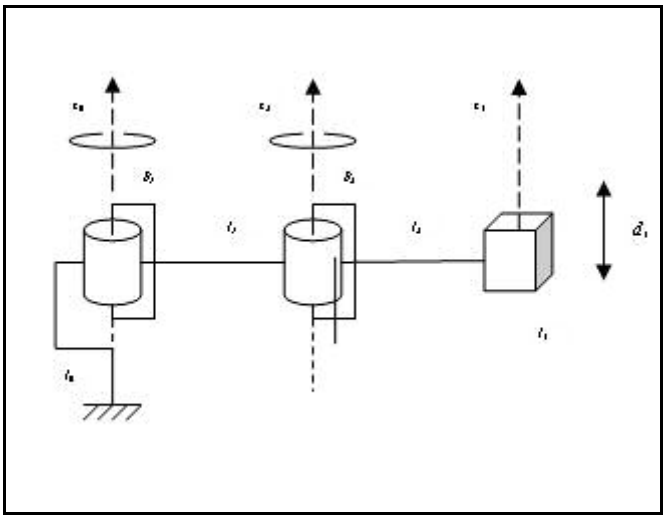

Figure 4.2 SCARA schematic

Table 4.1 DH Table for SCARA Manipulator

Frames	**α**	**A**	**d**	**θ**
0 – 1	0	0	0	θ_1
1 – 2	0	L_1	0	θ_2
3 – prismatic joint	90	L_2	0	0

4.2 Articulated Configuration

A manipulator comprising all revolute joints is known as an articulated or anthropomorphic manipulator. The common revolute designs are the elbow type such as a PUMA, as shown in figure 4.3.1 and the parallelogram linkage as Cincinnati Milacron T^3 735. This type of configuration provides relatively large freedom of movement in a compact space. Since the motor drive is born by link 1, the other two subsequent links can be more lightweight and the motors themselves can be less powerful [4].

Here for our simulation purpose we consider three types of articulated manipulators:

3-link elbow manipulator, 6-link elbow manipulator (such as GMF S-110, not considered specifically) and a special type of articulated manipulator the PUMA 560.

4.2.1 3-Link Elbow Manipulator

A 3-link elbow manipulator is as shown in the figures [4.2.1 & 4.2.2] below. It consists of three revolute joints for articulated motion. An assumed data as considered for such type of robot is as represented in the following DH table 4.2.1.

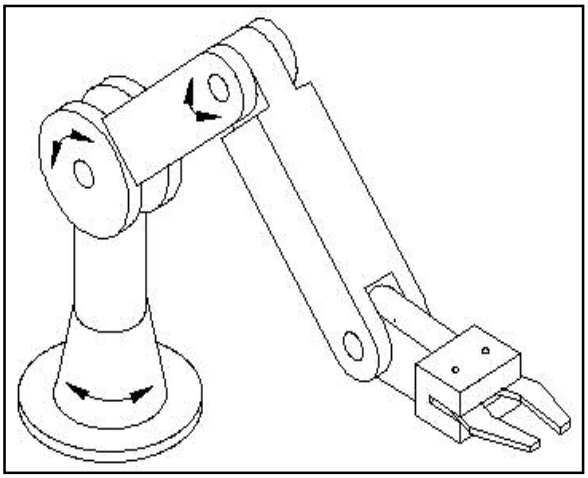

Figure 4.3 3-link Elbow Manipulator [www.learningconcepts.net/aw1_robot.htm]

Table 4.2 DH Table for 3-Link Elbow Manipulator

Frames	**α**	**A**	**d**	θ
0 – 1	0	0	0	θ_1
1 – 2	0	L_1	0	θ_2
2 - 3	0	L_2	0	θ_3
3 – end effector	90	L_3	0	0

4.2.2 6-Link Elbow Manipulator

A 6-link elbow manipulator such a general robot with 6 DOF is as represented in the DH tables shown in the following table 4.2.2. An example of a 6-link manipulator, such as GMF S-110 is as shown in the figure 4.2.3. This robot is the same as that of a 3-link elbow manipulator but in this one we have the last three links that are used for orientation rather that positioning of the end effector and are skewed about a single joint.

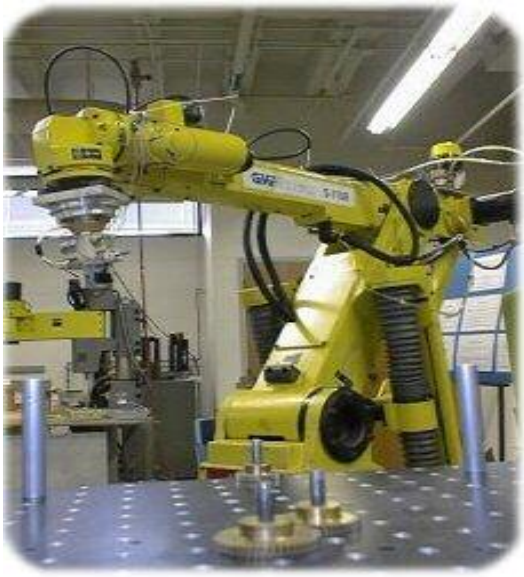

Figure 4.4 GMF S-110

Table 4.3 DH Table for 6-Link Elbow Manipulator

Frames	**α**	**A**	**d**	**Θ**
0 – 1	0	0	0	θ_1
1 – 2	0	L_1	0	θ_2
2 – 3	90	L_2	0	θ_3
3 – 4	0	L_3	0	θ_4
4 – 5	0	L_4	0	θ_5
5 – 6	0	L_5	0	θ_6
6 – end eff	90	L_6	0	0

4.3 PUMA 560 Manipulator

The PUMA 560 resembles a human arm in its shape and capabilities. Each member is mechanically linked to the others and can rotate around an axis. The manipulator is endowed with six rotational degrees of freedom, which allow it to achieve complete dexterity within its workspace, so as to reach any point within the workspace with arbitrary orientation. To each motor shaft is coupled an incremental quadrature encoder, which is used to track the change in angular position of the motor shaft and allows to know the angular position of the corresponding link. The motors driving the major joints (i.e. the first three joints) are equipped with electromagnetic brakes, which, once engaged, do not allow the shaft to rotate. The Puma 560 is a six degree of freedom robot manipulator. The end-effector of the robot arm can reach a point within its workspace from any direction. The six degrees of freedom are controlled by six brushed DC servo motors [http://www.cvrl.cs.uic.edu/resources.html].

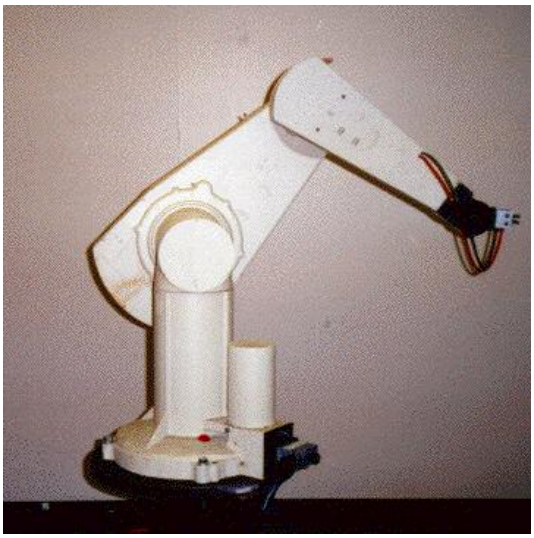

Figure 4.5 PUMA 560 [iel.ucdavis.edu/projects/ imc/Hardware.html]

The schematic diagram for the PUMA 560 is as shown in the figure 4.3.2.

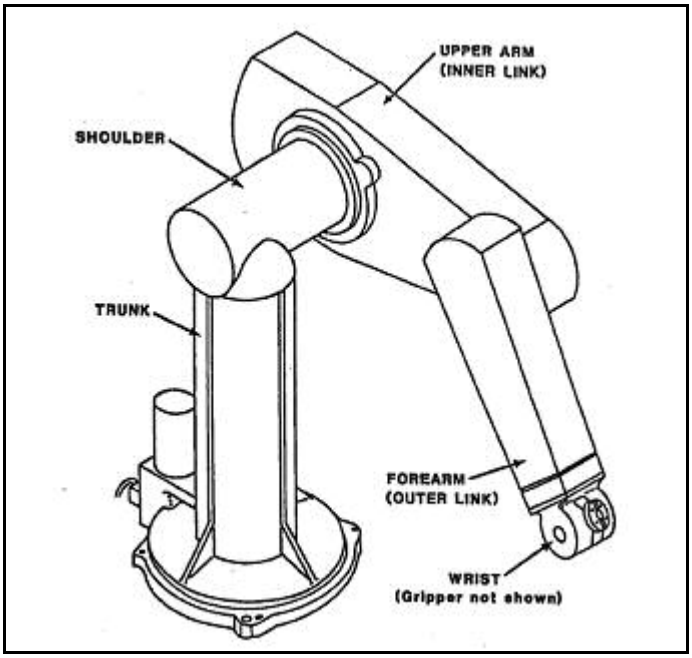

Fig 4.6 Schematics of a 6 DOF PUMA 560
[www.virtual.unal.edu.co/rsos/artes/2003259]

The DH table for a PUMA manipulator can be realized from the following DH table for the same.

Table 4.3 DH Table for PUMA 560

Frames	α	a	d	θ
0 – 1	0	0	0	θ_1
1 – 2	-90	0	0	θ_2
2 – 3	0	L_2	5	θ_3
3 – 4	-90	L_3	5	θ_4
4 – 5	90	1	0	θ_5
5 – 6	-90	1	0	θ_6
6 – end effector	0	L_6	0	0

CHAPTER 5

SIMULATION RESULTS AND DISCUSSIONS

5.1.SCARA Manipulator

5.1.1 Optimal Design link lengths for SCARA

The target locations set for the SCARA Manipulator is as listed in the table below

Table 5.1 Target Locations for SCARA

Target	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
X	22	16	8	2	-5	-14	-20	-24	-21
Y	12	18	23	27	22	19	10	2	-8

Target	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
X	-18	-12	-2	6	12	18	25	25
Y	-18	-22	-26	-24	-20	-16	-5	4

Table 5.2 Design Data for SCARA from IK solution

Objective Function	**MI 1**	**MI 2**	**MI 3**
Design Lengths	29.2965 16.7229	30.0000 17.0000	29.7457 17.0000

Table 5.3 Design Data for SCARA from closed form solution

Objective Function	**MI 1**	**MI 2**	**MI 3**
Design Lengths	29.4386 17.0000	29.5822 17.0000	29.7845 17.0000

The SCARA manipulator discussed earlier in the section 4.1 was evaluated to determine the design variables for performing the required task based on the given

target locations. Both closed form solutions and the solutions from the general algorithm were carried out. However, both the solutions yielded the same results.

5.1.2 Joint Angles from Inverse Kinematics

Joint angle calculation for the SCARA was carried out using two different algorithms. One of these algorithms used the fmincon to solve for the joint angles from Inverse Kinematics while the other method is based on solving the joint angles with the generated design lengths using API code for Ant Colony Optimization technique. These two methods have two different types of solutions indicating various orientations and the position of the manipulator to reach a particular target point in space.

Table 5.4 Joint Angles 1

Joint	**FMINCON**		**ACO method**	
	θ_1	θ_2	θ_1	θ_2
1	-5.6564	54.7608	-5.9476	123.3021
2	14.7006	62.8793	13.8233	126.5370
3	38.3128	60.7468	36.2578	125.6540
4	68.3742	32.6444	51.5825	116.4772
5	63.0930	73.9412	68.6127	131.7341
6	90.7138	66.5604	91.9440	128.2103
7	112.9820	75.2891	119.2732	132.3890
8	141.5684	62.8824	140.7208	126.5442
9	160.8130	74.5416	166.6513	132.0266
10	197.6891	51.1362	190.4809	121.9367
11	212.1226	54.7609	206.8309	123.2727
12	241.6416	44.9078	231.1453	119.8110
13	253.2623	57.5475	249.4892	124.3328
14	264.1818	68.5959	266.5443	129.1708
15	284.6992	62.8811	283.8370	126.5964
16	321.5800	50.7631	-45.8511	121.8163
17	341.0844	52.4250	-25.4213	122.4064

Table 5.5 Joint Angles 2

Joint	DE Method		PSO Method	
	θ_1	θ_2	θ_1	θ_2
1	-5.9409	123.2728	354.0575	123.2731
2	13.8293	126.5664	13.8289	126.5659
3	36.2659	125.6606	36.2659	125.6609
4	51.5602	116.4608	51.5603	116.4603
5	68.5765	131.7149	68.5778	131.7155
6	91.9052	128.1943	91.9048	128.1936
7	119.2791	132.3970	119.2790	132.3972
8	140.6989	126.5664	140.6986	126.5659
9	166.6585	132.0176	166.6585	132.0176
10	190.4773	121.9386	-169.5223	121.9383
11	206.8370	123.2726	-153.1631	123.2734
12	-128.8335	119.8414	-128.8344	119.8413
13	249.4741	124.3565	249.4741	124.3564
14	266.5339	129.1287	266.5339	129.1299
15	-76.1705	126.5635	-76.1710	126.5661
16	-45.8279	121.8037	-45.8275	121.8051
17	334.5559	122.4034	-25.4449	122.4033

5.1.3 Plot Results

The plot of the link orientation and position to locate a target point in a plane is as shown in 3-D view in the figure 5.1. Plotting of the link positions along with the animation enables the user to have a proper understanding of the system and necessary changes to be made to reach the goal point from a given base position and orientation. It also enables us to determine the deviation of the actual position from the original target position in case of large errors by eye estimation only. Even the singular configurations can be determined from the plot results. However the SCARA plot from the results doesn't have any singularities or significant errors. The objective of this optimization process was to design the link lengths such that they do not exceed a

certain value of the total link lengths (50 units in this case) and the minimum should also be more than the shortest distance between the origin of the first link to the target location so as to avoid singular configuration and hence preventing any loss in the degrees of freedom.

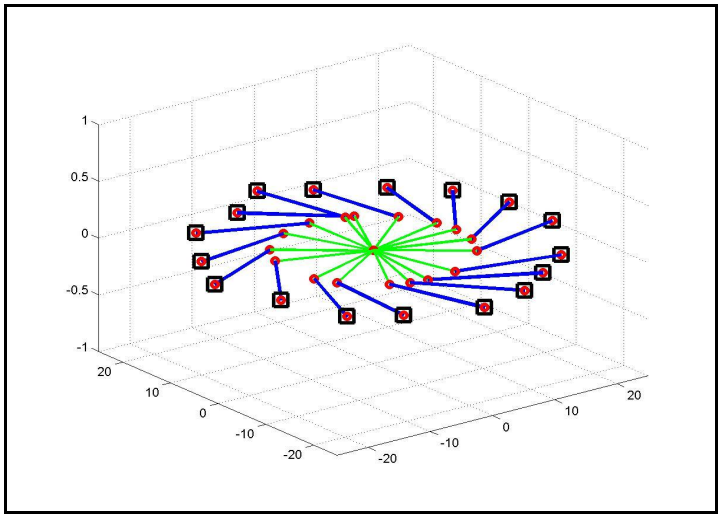

Figure 5.1 SCARA plot

A plot of the manipulability implies which target point has got the maximum manipulability. The graph represents the manipulability index as a function of the target points. The manipulability graph in the figure 5.2 for the SCARA shows that the target (-20, -10) has got the highest manipulability. Table 5.1.4 shows the actual position, target position and the values of manipulability index for each point.

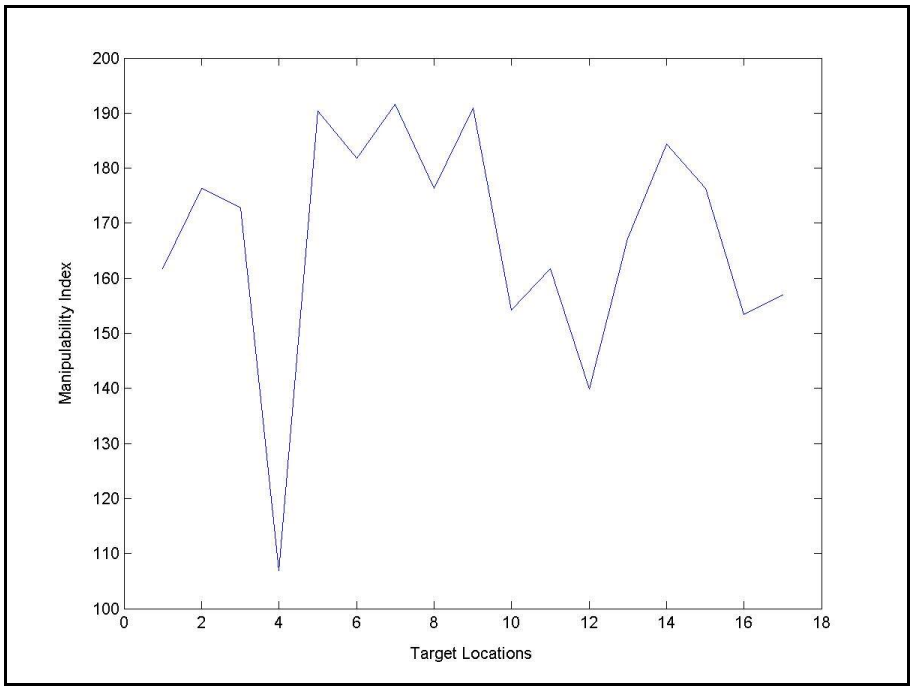

Figure 5.2 Manipulability Index for SCARA Manipulator

Table 5.6 Locations and Manipulability Index

Target Location (X,Y)	**Actual Position**	**Manipulability Index**
22, 12	22.000 12.000	161.7932
16,18	16.001 18.003	176.3132
8, 23	8.002 23.002	172.8308
2, 27	2.001 27.002	106.8566

Table 5.6 – Continued

-5, 22	-5.000 22.000	190.3639
-14, 19	14.000 19.001	181.7471
-20, 10	-20.0003 10.0002	191.6002
-24, 2	-23.9999 2.0001	176.3179
-21, -8	-21.0001 -8.0000	190.9277
-18, -18	-18.000 -18.0000	154.2438
-12, -22	-12.0000 -22.0000	161.7935
-2, -26	-2.0001 -26.0000	139.8480
6, -24	6.0000 -24.0000	167.1589
12, -20	11.9998 -19.9999	184.4314
18, -16	18.0000 -16.0001	176.3160
25, -5	25.0001 -5.0002	153.4311
25, 4	25.0000 4.0000	157.0005

5.2 3-link Elbow Manipulator

5.2.1 Optimal Design Link Lengths

A brief description of the 3-link manipulator used for analysis purpose has been discussed in the section 4.2.1. The design bounds for the 3link manipulator considered to lie between a total combined maximum length of 50 and a minimum length of 28 units. The target locations for this type of manipulator is shown in the Table 5.5 while the designed link length satisfied our criteria and the results are tabulated as shown in table 5.6

Table 5.7 Target Locations for 3-Link Elbow Manipulator

Target	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
X	22	16	8	2	-5	-14	-20	-24	-21
Y	12	18	23	27	22	19	10	2	-8

Target	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
X	-18	-12	-2	6	12	18	25	25
Y	-18	-22	-26	-24	-20	-16	-5	4

Table 5.8 Design Length for 3-Link Elbow Manipulator

Objective Function	**MI 1**	**MI 2**	**MI 3**
Design Lengths	20.0000 10.0000 10.0000	20.0000 10.0000 10.0000	24.9882 15.0000 5.0000

5.2.2 Joint Angles from Inverse Kinematics

Joint angle calculation for the 3-Link Elbow manipulator was carried out using the same two algorithms using fmincon and API code for Ant Colony Optimization technique. The results obtained from both are as tabulated in the table 5.2.3.

Table 5.9 Joint Angles 1

Joint	**By FMINCON**			**By API Code**		
	θ_1	θ_2	θ_3	θ_1	θ_2	θ_3
1	-10.7690	69.5708	53.2632	-3.1182	43.9468	86.2641
2	7.6675	73.8145	52.8507	359.9999	105.8698	4.7675
3	29.8696	75.0143	49.5735	33.9084	60.7314	69.3703
4	47.4537	68.9443	42.3404	48.1363	65.7918	46.7595
5	59.7336	82.0039	49.5386	57.9364	89.3446	38.2847
6	83.6957	81.2365	44.7152	83.7902	80.8473	45.2844
7	109.2113	86.3488	44.0816	112.2627	75.5426	60.2461
8	132.2192	83.4967	38.4928	132.8388	80.7104	42.7195
9	156.2089	88.1983	40.5812	159.6844	75.4396	59.7215
10	183.1227	81.7211	33.1016	184.0513	76.9841	40.2597
11	198.8957	83.6587	32.4812	199.7752	79.1232	39.3992
12	224.1139	82.2093	28.5858	230.2025	55.7963	65.9570
13	240.8001	86.6724	29.6954	240.1989	90.6745	23.2847
14	256.2683	90.0000	32.8168	262.1141	67.4412	66.1835
15	274.6320	87.1123	32.8988	285.0629	49.8656	84.7955
16	306.5959	83.1278	30.6928	312.2799	58.9787	65.1395
17	-10.7690	69.5708	53.2632	-29.2423	65.7902	57.0335

Table 5.10 Joint Angles 2

Joint	**DE Method**			**PSO Method**		
	θ_1	θ_2	θ_3	θ_1	θ_2	θ_3
1	357.8997	40.6942	90.0000	-11.1330	70.9995	51.2459
2	15.2817	49.1870	85.6447	3.9829	90.9003	26.8934
3	27.5570	85.4344	33.9318	35.2518	56.4326	74.9852
4	54.2081	42.5400	76.3009	48.1992	65.4979	47.1899
5	56.2542	98.3147	23.8759	58.1902	88.1680	40.1167
6	81.3061	93.1927	26.0753	87.3542	67.9292	63.9275
7	116.8319	62.0807	79.1336	112.3187	75.3555	60.5048
8	141.1459	52.1779	81.9380	140.6959	53.5297	80.2379
9	165.3787	58.7827	82.8993	155.8073	89.9731	37.8093

Table 5.10 – Continued

10	194.7962	38.6598	90.0000	186.3059	67.3278	54.1083
11	210.2049	42.1577	88.3202	199.9680	78.2037	40.7693
12	224.1605	81.9135	29.0461	229.4027	58.4967	62.4141
13	252.9310	42.3243	89.9990	242.5274	77.9168	43.0756
14	-90.0000	48.1222	90.0000	260.6971	72.1282	59.6161
15	-85.6135	88.4700	30.7641	-78.5189	60.7722	70.8649
16	309.4890	69.3910	50.9996	-54.3321	89.5396	20.4898
17	-26.1750	55.0167	71.2468	331.8844	61.6958	62.5557

5.2.3 Plot Results

The plot of the 3-link Elbow Manipulator is as shown in the figure below. The target locations considered in this case consists of points lying on a 2-D plane, let us consider for example a plane surface of a table where the manipulator is used for pick and place operations. The plot shows the alignment of each link to reach the given points avoiding singularity.

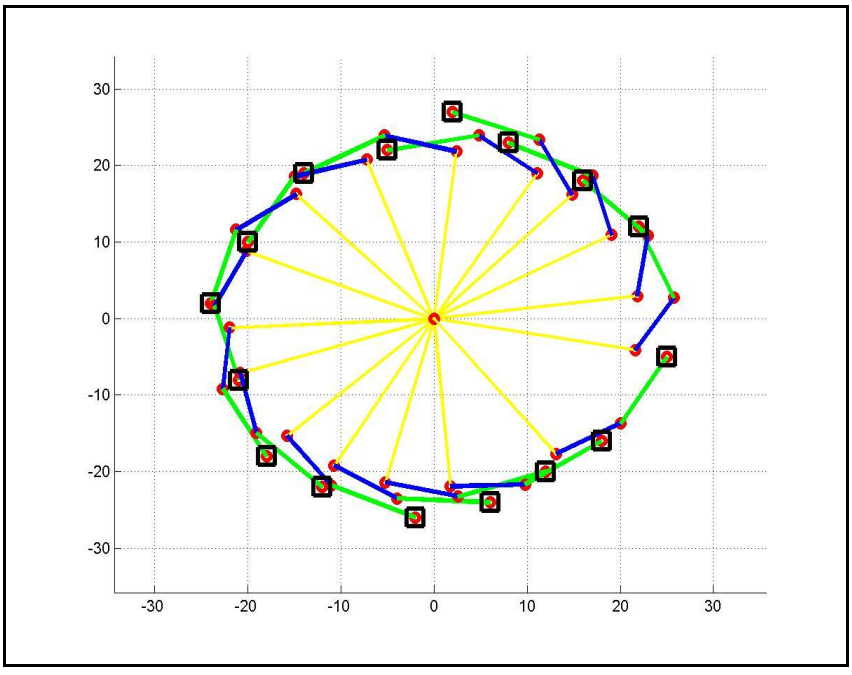

Figure 5.3 Plot of a 3-Link Elbow Manipulator

The manipulability plot as shown in the figure 5.2.2 below represents a series of curves, each peak representing a better manipulability index and the ability of the manipulator to reach that particular point on the plane.

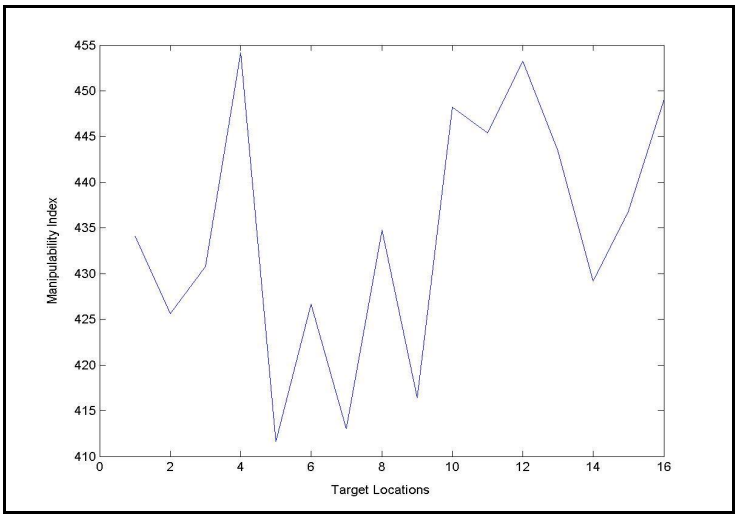

Figure 5.4 Manipulability Index for 3-link Elbow Manipulator

A table is shown below that depicts the deviation of the actual positioning of the manipulator using the algorithm and the desired end effector position. It also gives us a brief idea about the manipulability index for each point traced by the links.

Table 5.11 Locations and Manipulability Index

Target Location (X,Y)	**Actual Position**	**Manipulability Index**
22 , 12	22.0000 11.9999	434.1002
16 , 18	16.0000 18.0000	425.6119
8 , 23	8.0000 23.0000	430.7819
2 , 27	1.9999 27.0002	454.1585
-5 , 22	-4.9999 22.0000	411.6505

Table 5.11 – Continued

-14 , 19	-14.0000 19.0001	426.6529
-20 , 10	-20.0000 10.0000	413.0377
-24 , 2	-24.0000 2.0000	434.7529
-21 , -8	-21.0001 -8.0000	416.3886
-18 , -18	-18.0000 -18.0000	448.2093
-12 , -22	-12.0000 -22.0000	445.3816
-2 , -26	-2.0000 -26.0000	453.2199
6 , -24	6.0000 -24.0000	443.4055
12 , -20	11.9994 -20.0005	429.2054
18 , -16	18.0000 -16.0000	436.7899
25, -5	25.0000 -5.0000	449.0534

5.3 6-Link Elbow Manipulator

A more practical design involves the design of a 6-Link manipulator as they are more in practical use than two-link or three-link devices. These types of manipulators are designed so as to provide the manipulator with more degrees of freedom to achieve better manipulability. The design variables calculated using this algorithm is as shown in Table 5.9 and the target locations are in the Table 5.8.

Table 5.12 Target Locations for 6-Link Elbow Manipulator

Target	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
X	22	16	8	2	-5	-14	-20	-24	-21
Y	12	18	23	27	22	19	10	2	-8

Target	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
X	-18	-12	-2	6	12	18	25	25
Y	-18	-22	-26	-24	-20	-16	-5	4

5.3.1 Optimal Design Link Lengths

Table 5.13 Design Length for 6-Link Elbow Manipulator from IK solution

Objective Function	**MI 1**	**MI 2**	**MI 3**
Design Lengths	25.0001 4.9998 20.0000 10.0000 7.4999 6.0000	26.7185 25.0612 11.2386 6.8714 6.0491 3.1072	26.7185 25.0612 11.2386 6.8714 6.0491 3.1072

The objective was to achieve a design such that the design length is sufficient enough to provide adequate room for each and every link to retain all the degrees of freedom in order to avoid singularity. The design so far obtained is well within the bounds satisfying all the criteria.

5.3.2 Joint Angles from Inverse Kinematics

Joint angles computed for a 6-Link Elbow manipulator using the same two algorithms, fmincon and API code are as shown in table 5.3.2. The results obtained from both are as tabulated in the table 5.2.3.

Table 5.14 Joint Angles from FMINCON

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	-79.7874	33.5358	128.0242	23.8810	15.0022	18.5168
2	24.2257	57.6086	158.8419	75.6856	20.8587	15.4125
3	37.6680	71.5956	168.3621	58.8783	22.1158	21.6541
4	56.6702	61.0627	170.8931	77.3111	16.2214	15.9684
5	64.9570	93.6354	172.4294	32.9190	30.0841	35.7340
6	77.5136	97.1327	176.4895	33.2143	30.2618	37.1882
7	87.8373	63.0161	171.5551	75.3189	15.6912	19.9599
8	78.8281	88.5381	167.4778	26.6111	34.8912	42.1210
9	104.7916	86.2482	174.2788	42.6761	28.9669	38.6261
10	138.6368	20.0000	180.0000	15.0000	15.0000	50.3828
11	146.2895	26.1827	150.6591	18.5878	28.2644	57.2619
12	168.6743	23.7079	124.5313	35.3950	35.8962	54.9679
13	269.5480	57.1677	161.4262	71.5664	37.5791	75.8814
14	231.3241	59.3925	170.2550	29.7585	44.3041	85.7422
15	240.2161	51.7783	155.1454	71.5459	42.6084	80.2111
16	241.1942	39.7346	133.2448	90.0000	39.3577	69.1705

Table 5.15 Joint Angles from ACO Method

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	187.3735	154.4517	38.8563	52.6918	22.3072	23.9399
2	220.1781	150.2587	20.2662	75.2006	64.4966	19.3080
3	228.5082	153.9327	65.2470	18.9236	24.7080	64.5899
4	250.2496	149.8827	25.5631	50.5569	79.9772	38.7575
5	267.1187	151.6389	29.9362	50.7879	50.3582	28.9970
6	-66.2800	159.1621	50.0977	41.0700	30.5292	70.0975
7	319.3301	155.7936	31.4534	49.8739	59.2697	77.2938
8	-11.2886	149.9290	39.1175	64.2539	54.6123	41.1388
9	24.8866	137.8405	66.6405	20.5970	65.3855	18.9118
10	58.0951	144.3900	58.4865	17.0694	63.8673	25.3526
11	87.8492	170.9696	128.9099	61.2824	58.6322	20.8694
12	131.1034	163.8697	111.3884	59.8006	63.2643	38.7883
13	-44.4757	21.3246	165.0087	54.9515	17.3749	36.0009
14	171.6652	164.8480	118.7196	34.7028	75.4069	42.5632
15	200.1261	159.5687	106.0160	53.6751	69.4346	27.4841
16	189.4513	156.7055	21.0643	71.1267	45.5530	29.8469

Table 5.16 Joint Angles from DE Method

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	236.9655	22.7110	55.0301	80.1952	79.3971	56.2824
2	1.3125	76.5236	170.7007	89.9322	27.5318	22.2124
3	19.8178	59.0267	166.0213	55.5852	82.7031	43.9911
4	50.3977	48.9875	173.8857	72.8939	17.3420	67.8737
5	52.4292	87.9875	171.7822	18.9895	23.9523	63.0824
6	48.0739	134.8374	175.2249	48.0402	16.7853	60.9176
7	60.8682	29.5131	34.0316	69.6181	88.0811	59.0946
8	39.9699	32.3790	64.6236	86.3692	23.2200	25.4929
9	65.9000	108.7942	174.3796	74.1251	82.4908	23.3473
10	159.6290	176.2295	73.1655	79.0593	16.2824	80.3288
11	91.5372	28.7777	90.3118	30.0856	32.4221	50.7375
12	146.7990	36.5966	172.3440	72.4774	23.0213	66.5000
13	238.2340	32.4294	70.4146	71.7408	88.0646	80.9710
14	207.8975	62.0703	140.2345	20.9779	88.9765	51.6824
15	194.3588	21.7832	73.5652	20.5273	57.0109	15.9315
16	229.2944	25.5283	179.4905	78.6240	82.3380	15.0111

Table 5.17 Joint Angles from PSO Method

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	242.6270	85.7930	175.0356	52.1741	46.2158	25.9259
2	-29.9074	54.1410	30.9592	15.0492	50.4312	56.0564
3	2.3192	38.4146	36.5388	68.5663	78.2354	55.1265
4	29.9922	24.6509	26.3212	72.2722	74.1883	54.4203
5	25.7398	48.4701	65.3190	55.8810	79.3596	43.6581
6	53.8933	92.6279	150.2009	58.0698	54.6685	72.4994
7	52.1226	26.3219	57.6732	17.8130	74.9077	40.9684
8	39.4697	26.6984	92.5447	39.9232	42.2749	55.7967
9	69.1828	105.3773	160.9909	81.1059	66.2566	38.3723
10	120.7924	126.2654	90.3946	70.5324	51.3094	62.9618
11	120.9463	39.9498	135.6999	48.8051	62.7394	27.9122
12	130.1482	74.6931	168.7230	84.7266	34.9350	36.1204
13	253.2490	64.9521	172.3863	45.8117	49.3883	61.9151
14	200.1493	45.9945	94.5610	52.8925	68.5124	41.1398
15	226.0188	36.2580	168.2054	48.4541	80.6340	56.4996
16	219.7653	47.4240	179.4022	89.6813	51.1224	53.6680

5.3.3 Plot Results

Figure 5.3.1 shows the plot of such a manipulator in 3-D space. The 3-D plot depicts the positioning of the end effector much more distinctly for any kind of operations and especially when most of the practical operations involve 3-D points inside the work space.

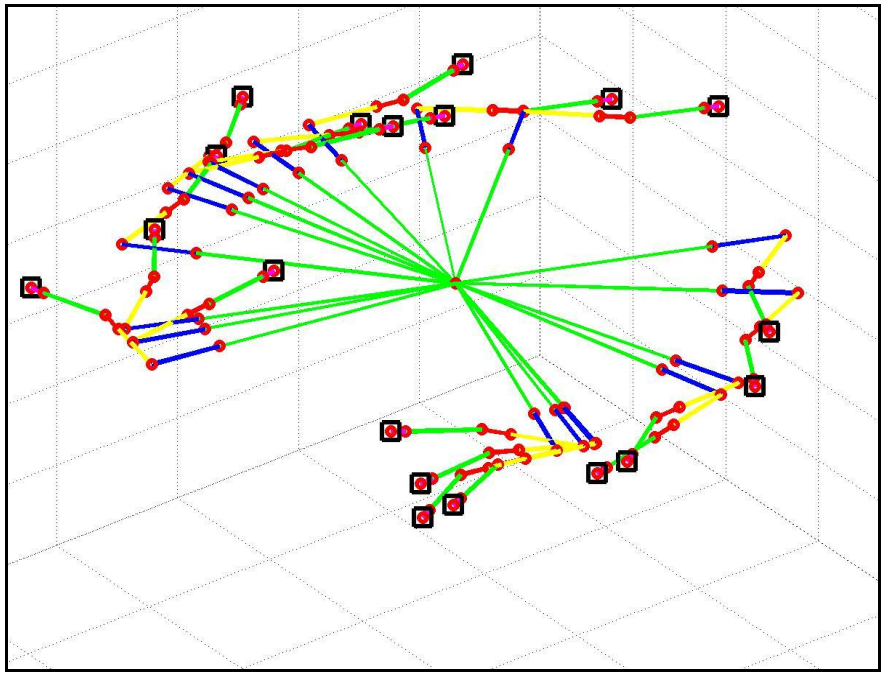

Figure 5.5 Plot of a 6-Link Elbow Manipulator

The manipulability plot as shown in the figure 5.3.2 below represents a series of curves, each peak representing a better manipulability index and the ability of the manipulator to reach that particular point in the workspace which is a 3 dimensional one.

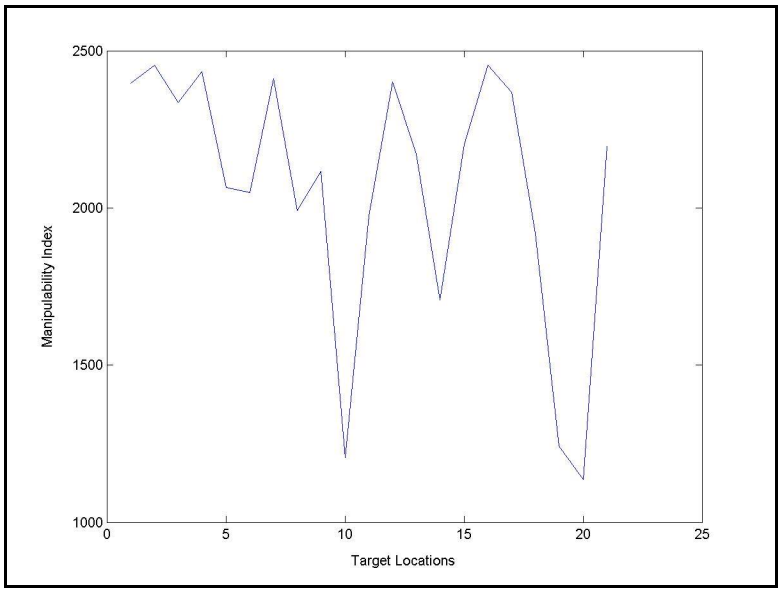

Figure 5.6 Manipulability Index for 6-link Elbow Manipulator

The target positions, actual end effector location and the manipulability index for the 6-link manipulator are as shown in the table 5.2.4. The manipulability index as observed is better than two or three link mechanisms.

Table 5.18 Locations and Manipulability Index

Target Location (X,Y)	**Actual Position**	**Manipulability Index**
-15 , -50	-15.0000 -50.0000	2397.6
55 , 15	55.0000 15.0000	2454.1
45 , 25	45.0000 25.0000	2334.6
35 , 44	35.0000 44.0000	2434.2
25 , 35	25.0000 35.0000	2066.1
15 , 40	15.0000 40.0000	2048.0
7 , 55	7.0000 55.0000	2411.1

Table 5.18 – Continued

18 , 37	18.0000 37.0000	1991.7
-5 , 45	-5.0000 45.0000	2117.2
-17 , 17	-17.0000 17.0000	1204.8
-23 , 35	-23.0000 35.0000	1974.7
-45 , 33	-44.9999 33.0000	2400.3
-10 , -50	-10.0000 -50.0000	2172.3
-30 , -25	-30.0000 -25.0001	1706.0
-35 , -38	-35.0000 -38.0000	2202.7
-40 , -45	-40.0000 -45.0000	2454.6

5.4 PUMA 560

5.4.1 Optimal Design Link Lengths

The Unimation PUMA 560 is a six degree of freedom robot with all rotary joints. The link and frame assignments are as shown in the previous chapters. In this manipulator first three links are known as the arms of the manipulator that are used for positioning, while the last three links are known as wrists and they are mainly used for orientation purposes. The design length for the manipulator is shown in table 5.13 which is generated for the given set of points in Table 5.12 and lies well within the bounds.

Table 5.19 Target Locations in 3-D space for PUMA

Target	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
X	46	38	27	19	5	-15	-28	-38
Y	5	25	30	38	45	37	28	-11
Z	-6	-8	-5	-8	-7	-5	-9	-8
Target	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
X	-35	-25	-5	14	20	24	25	46
Y	-10	-31	-23	-22	-19	-7	4	5
Z	-9	-6	-4	-5	-6	-5	-4	-5

Table 5.20 Design Length for PUMA

Objective Function	**MI 1**	**MI 2**	**MI 3**
Design Lengths	11.0000	11.0000	11.0000
	39.7969	38.6745	38.6745
	10.7522	10.9832	10.9832
	5.0000	5.0000	5.0000
	5.0000	5.0000	5.0000
	9.8432	10.2315	10.2315

5.4.2 Joint Angles from Inverse Kinematics

Joint angles computed for a PUMA using the fmincon and API code are as shown in table 5.4.2. The results obtained from both are as tabulated in the table 5.2.3.

Table 5.21 Joint Angles from FMINCON

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	7.0224	93.0645	180.0000	26.1172	38.9960	23.0157
2	34.3070	101.3722	179.9236	28.0794	42.8404	24.3668
3	48.9118	103.1488	167.6666	26.4444	40.4335	23.0657
4	64.6315	109.0042	174.3867	29.5397	44.8854	25.7430
5	83.5673	101.2549	177.8880	25.1848	45.3912	19.1804
6	112.0736	107.3954	165.9007	25.6982	45.8039	19.7222
7	136.0068	120.6739	169.8637	32.5663	54.0116	27.5281
8	164.0521	120.1501	167.8641	30.6597	55.4556	23.1551
9	196.4897	131.9299	164.6510	36.1846	62.9840	29.2982
10	228.7055	117.3393	164.5961	26.9047	58.6327	10.2992
11	255.9041	104.0153	124.0143	30.9414	59.3388	16.8593
12	296.6792	95.5163	126.2945	27.5611	69.4944	7.6463
13	309.8597	89.5467	127.1933	27.3351	74.2658	5.8669
14	334.4998	96.6302	123.1502	24.9185	74.0318	3.0038
15	359.5193	101.2585	124.1965	22.4720	75.9205	2.0874
16	360.0000	78.4796	165.7524	18.8938	90.0000	1.3322

Table 5.22 Joint Angles from ACO Method

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	199.7463	158.4292	41.3223	65.6416	69.7481	55.0648
2	226.5053	155.7832	41.2259	28.0422	51.3838	52.2745
3	242.9445	156.4630	72.3510	26.5458	71.7654	73.0307
4	257.8303	152.4469	59.2330	63.6055	39.4424	86.3934
5	276.7550	156.5829	42.9728	23.7161	40.8000	35.1831
6	-53.2492	155.1062	73.1240	25.2129	18.1144	42.3786
7	-29.3604	150.1771	73.3983	42.7938	72.5048	28.3509
8	145.1181	23.9606	179.9815	52.9316	18.7453	47.4787
9	32.3397	149.6651	89.5663	16.7313	81.0394	69.4123
10	66.0762	154.7791	73.7632	16.1829	63.3798	16.5845
11	99.0568	169.8579	151.9159	81.5726	49.3322	57.5186
12	281.8084	20.0019	164.9856	77.0139	33.9669	60.8564
13	157.5491	167.3116	151.1954	62.2957	34.4976	86.8540

Table 5.22 Continued

14	-37.3028	20.0074	163.7153	74.5563	46.0833	36.6982
15	-11.3482	20.0145	164.3314	27.3425	28.9529	25.9800
16	199.5936	159.8648	40.7578	45.1312	60.4880	47.7747

Table 5.23 Joint Angles by DE Method

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	236.9655	22.7110	55.0301	80.1952	79.3971	56.2824
2	1.3125	76.5236	170.7007	89.9322	27.5318	22.2124
3	19.8178	59.0267	166.0213	55.5852	82.7031	43.9911
4	50.3977	48.9875	173.8857	72.8939	17.3420	67.8737
5	52.4292	87.9875	171.7822	18.9895	23.9523	63.0824
6	48.0739	134.8374	175.2249	48.0402	16.7853	60.9176
7	60.8682	29.5131	34.0316	69.6181	88.0811	59.0946
8	39.9699	32.3790	64.6236	86.3692	23.2200	25.4929
9	65.9000	108.7942	174.3796	74.1251	82.4908	23.3473
10	159.6290	176.2295	73.1655	79.0593	16.2824	80.3288
11	91.5372	28.7777	90.3118	30.0856	32.4221	50.7375
12	146.7990	36.5966	172.3440	72.4774	23.0213	66.5000
13	238.2340	32.4294	70.4146	71.7408	88.0646	80.9710
14	207.8975	62.0703	140.2345	20.9779	88.9765	51.6824
15	194.3588	21.7832	73.5652	20.5273	57.0109	15.9315
16	229.2944	25.5283	179.4905	78.6240	82.3380	15.0111

Table 5.24 Joint Angles from PSO Method

Joint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	242.6270	85.7930	175.0356	52.1741	46.2158	25.9259
2	-29.9074	54.1410	30.9592	15.0492	50.4312	56.0564
3	2.3192	38.4146	36.5388	68.5663	78.2354	55.1265
4	29.9922	24.6509	26.3212	72.2722	74.1883	54.4203
5	25.7398	48.4701	65.3190	55.8810	79.3596	43.6581
6	53.8933	92.6279	150.2009	58.0698	54.6685	72.4994
7	52.1226	26.3219	57.6732	17.8130	74.9077	40.9684
8	39.4697	26.6984	92.5447	39.9232	42.2749	55.7967
9	69.1828	105.3773	160.9909	81.1059	66.2566	38.3723
10	120.7924	126.2654	90.3946	70.5324	51.3094	62.9618

Table 5.24 – Continued

11	120.9463	39.9498	135.6999	48.8051	62.7394	27.9122
12	130.1482	74.6931	168.7230	84.7266	34.9350	36.1204
13	253.2490	64.9521	172.3863	45.8117	49.3883	61.9151
14	200.1493	45.9945	94.5610	52.8925	68.5124	41.1398
15	226.0188	36.2580	168.2054	48.4541	80.6340	56.4996
16	219.7653	47.4240	179.4022	89.6813	51.1224	53.6680

5.4.3 Plot Results

Figure 5.4.1 shows the plot of a PUMA in 3-D space. The 3-D plot for PUMA only depicts the positioning of first three links and the end effector since the other three links are oriented mutually perpendicular to each other.

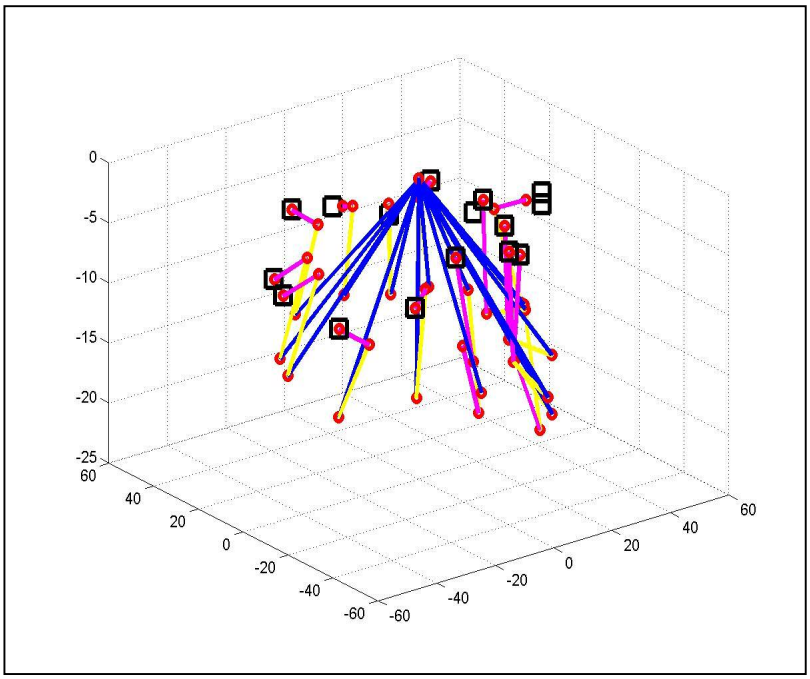

Figure 5.7 Plot of PUMA 560 links in 3-D workspace

The manipulability plot as shown in the figure 5.8 below represents a series of curves, each peak representing a better manipulability index and the ability of the manipulator to reach that particular point in the workspace which is 3 dimensional.

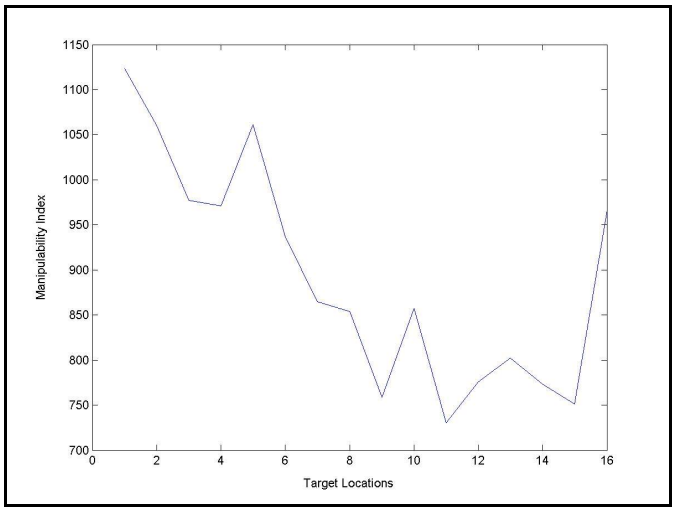

Figure 5.8 Manipulability Index for PUMA 560

The target positions, actual end effector location and the manipulability index for the 6-link manipulator is as shown in the table 5.4.4. The manipulability index as observed is better than two or three link mechanisms.

Table 5.25 Locations and Manipulability Index

Target Location (X,Y)	**Actual Position**	**Manipulability Index**
46 , 5 , -6	46.0000 5.0000 -6.0000	1123.5
38 , 25 , -8	38.0001 25.0000 -7.9999	1060.0
27 , 30 , -5	27.0000 30.0000 -5.0000	977.40
19 , 38 , -8	19.0000 38.0000 -8.0000	971.30
5 , 45 , -7	5.0000 45.0000 -7.0000	1061.3

Table 5.25 – Continued

-15 , 37 , -5	-14.9999 37.0000 -5.0001	937.10
-28 , 28 , -9	-28.0000 27.9999 -9.0000	864.70
-38 , 11 , -8	-38.0002 11.0001 -7.9998	853.50
-35 , -10 , -9	-35.0000 -10.0000 -9.0000	758.70
-25 , -31 , -6	-25.0000 -30.9999 -5.9998	857.20
-5 , -23 , -4	-5.0000 -23.0001 -4.0000	730.40
14 , -22 , -5	-14.0000 -22.0000 -5.0001	775.60
20 , -19 , -6	-20.0000 -19.0000 -6.0002	802.00
24 , -7 , -5	23.9998 -7.0000 -4.9998	773.20
25 , 4 , -4	24.9999 4.0000 -3.9997	751.20
46 , 5 , -5	46.0001 5.0000 -4.9999	966.30

However, using the fmincon algorithm for design optimization purposes pretty much satisfactory results are obtained and also a lot of time is saved as the computation process is also carried out within desirable limits. The main objective of maximizing the manipulability index is achieved using this technique and at the same time one can gain

control of the singularities by defining the bounds for each and every joint angles so that the limit is not exceeded and the manipulator also doesn't lose any degree of freedom for a given task inside the workspace.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

The design optimization carried out in this research is based on certain task specification by generating some random target points and analyze the ability of the manipulator or robotic device to reach those locations with the deigned length and the optimized joint angles. The objective function being the manipulability index is subjected to certain design constraints such as the maximum and minimum link lengths as well as the freedom for the link movements limited by the joint angles.

The animation and the plots generated with the designed parameters determine the error in locating the object and the accuracy with which the target is achieved. The simulations were carried out with various manipulators such as the SCARA and three different types of articulated manipulators (3-Link and 6-Link Elbow, PUMA 560) that are prevalent in most industrial usage. As seen from the simulation results most of the manipulators are able to achieve the target location with accuracy and precision without attaining any singular configuration.

6.2 Future Work

This process can be considered as stepping stone towards a complex design optimization process. However, there is an ample scope for future work to be carried out in this context as explained below:

1. The case studies considered in this case consist of only serial link manipulators with revolute joints. This concept can be extended to the design optimization of both serial and parallel manipulators with prismatic joints too. Combination of both the types may yield a better design for industrial purposes.

2. The results obtained can be modified or improved using various other evolutionary techniques.

3. Additional manipulator analysis should be carried out to develop a more generalized algorithm for optimization.

4. The design considered for our optimization purposes is involved with locating the object. But for a better and feasible solution the design for the cross section of the manipulators can be taken into account so as to minimize the weight as well as the joint torques.

APPENDIX A

MANIPULATION INDICES

MI 1 = min (svdJ) / max (svdJ)

MI 2 = min (svdJ)

MI 3 = sqrt (det ($J. J^T$))

Where, J = Jacobian Matrix

The matrix that relates the Cartesian space velocities to the Joint space velocities.

svdJ = Singular value of the Jacobian J

APPENDIX B

TERMINOLOGIES

Pseudoinverse:

In linear algebra, the pseudoinverse A^+ of a $m \times n$ matrix is a generalization of the inverse matrix. The pseudoinverse is defined and unique for all matrices whose entries are real or complex numbers. Usually, the pseudoinverse is computed using singular value decomposition.

Singularity:

It is defined as the condition where the manipulator loses one degree of freedom. In this case the Jacobian matrix becomes non-invertible and hence loses its rank. The determinant of the Jacobian matrix in singular condition tends to infinity.

Service Angle:

It is defined to be the synonym for Manipulability which was in use before the term manipulability was pioneered by Tsuneo Yoshikawa.

REFERENCES

[1] A.D. Belegundu & T.R.Chandrupatla, “Optimization Concepts and Applications in Engineering”, Pearson Education Publication.

[2] J.S. Arora, “Optimization for Engineers”.

[3] John J. Craig, “Introduction to Robotics”, 2nd Edition, Pearson Education Publication.

[4] M.W. Spong & M. Vidyasagar, “Robot Dynamics and Control”, John Wiley and Sons Publication.

[5] T. Yoshikawa, “Foundation of Robotics: Analysis and Control”, The MIT Press, Cambridge, Massachusetts, London.

[6] Rick Parent, “Computer Animation: Algorithms and Technique”, Morgan Kaufmann Publishers.

[7] N.M.Fonseca Ferreira & J.A. Tenreiro Machado, “Manipulability Analysis of Two-Arm Robotic Systems”.

[8] Tsuneo Yoshikawa, “Manipulability and Redundancy Control of Robotic Mechanisms”, IEEE, 1985.

[9] Y.W.Sung, D.K.Cho, M.J. Chung & K.Koh, “A Constraint-based Method of the Inverse Kinematics fro Redundant Manipulators”.

[10] Tsuneo Yoshikawa, “Translational and Rotational Manipulability of Robotic Manipulators”, IEEE, 1991.

[11] H.H. Pham & I-Ming Chen, “Optimal Synthesis for Workspace and Manipulability of Parallel Flexure Mechanism”, Proceedings of the 11th World Congress in Mechanism and Machine, Aug 18-21, 2003.

[12] M.T. Rosenstein & R.A. Grupen, “Velocity Dependent Dynamic Manipulability”, Department of Computer Science, University of Massachusetts, Amherst, MA, USA.

[13] M. Meredith & S. Maddock, “Real-Time Inverse Kinematics: The Return of the Jacobian”, Dept of Computer Science, University of Sheffield, UK.

[14] D. Manocha and J.F. Canny, “Efficient Inverse Kinematics for General 6R Manipulators”, IEEE Transaction on Robotics and Automation.

[15] J. Lee & K.T. Won, “Inverse Kinematics Solution based on Decomposed Manipulability”, Proceedings of the 1999 IEE, Intl. Conf. on Robotics & Automation, May 1999.

[16] Jihong Lee, “A Study on Manipulability Measures of Robot Manipulators”, IEEE, 1997.

[17] C. Chevallereau & W. Khalil, “A New Method for the Solution of the Inverse Kinematics of Redundant Robots”, IEEE, 1988.

[18] C.C. Cheah, M. Hirano, S. Kawamura & S. Arimoto, “Approximate Jacobian Control for Robots with Uncertain Kinematics and Dynamics”, IEEE, 2003.

[19] M.A. Shaik & P. Datseris, “A Workspace Optimization Approach to Manipulator Linkage Design”, IEEE, 1986.

[20] G. Prokop & F. Pfeiffer, “Improved Robotic Assembly by Position and Controller Optimization”, IEEE, 1996.

[21] S. Khatami & F. Sassani, “Isotropic Design Optimization of Robotic Manipulators Using a Genetic Algorithm Method”, Proceedings of IEEE, Intl. Symposium on Intelligent Control, Oct 27-30, 2002.

[22] G. Prokop, K. Dauster, F. Pfeiffer, “Application of Model-Based Optimization in Robotic Manipulators”, IEEE, 1999.

[23] L.C.T. Wang & C.C. Chen, “A Combined Optimization Method for Solving the Inverse Kinematics Problem of Mechanical Manipulators”, IEEE Transaction on Robotics and Automation, Aug, 1991.

[24] R. Vijaykumar, M.J. Tsai & K.J. Waldron, “Geometric Optimization of Manipulator Structures for Working Volume and Dexterity”, IEEE 1985.

[25] K.L. Doty, C. Melchiorri, E.M. Schwartz & C. Bonivento, “Robot Manipulability”, IEEE Transactions on Robotics and Automation, Vol 11, No. 3, June 1995.

[26] J.T.Y. Wen, L.S. Wilfinger, “Kinematic Manipulability of General Constrained Rigid Multibody Systems”, IEEE Transactions on Robotics and Automation, Vol 15, No. 3, June 1999.

[27] K. Miller, “Optimal Design and Modeling of Spatial Parallel Manipulators”, Intl. Journal of Robotics Research, Vol 23, No. 2, Feb 2004.

[28] J.M.Ahuactzin & K. Gupta, “Completeness Results for a Point-to-Point Inverse Kinematics Algorithm, Proceedings of the 1999 IEEE, Intl. Conf. on Robotics and Automation, May, 1999.

[29] S.K. Chan & P.D. Lawrence, “General Inverse Kinematics with the Error Damped Pseudoinverse”, IEEE, 1988.

[30] G.N. Vanderplaats, “Numerical Optimization a powerful tool for Engineering Optimization, Computer Modeling and Simulations in engineering, 1, 1996, 107-126.

[31] P. Chedmail and E. Ramstein, “Robot Mechanism Synthesis and Genetic Algorithm”, Proceedings of IEEE, International Conference on Robotics and Automation, Vol 4, 1996.

[32] R.V. Mayorga & E.D. de Leon, “Optimal Upper Bound Conditioning for Manipulator Kinematic Design Optimization”**,** IEEE, 1997.

[33] F.C. Park & J.W. Kim, “Manipulability and Singularity Analysis of Multiple Robot Systems: A Geometric Approach”, IEEE, 1998.

[34] Master’s Thesis of Dharmeshkumar H. Koladiya, “Application of Differential Evolution Optimization to Robotics and Mechanism Dimensional Synthesis”, 2002.

[35] Master’s Thesis of Aditya Apte, “Finite Element and Numerical Methods for Inverse Problems in Optical Medical Imaging”, 2004.

[36] Master’s Thesis of Chirag Natvarial Patel, “Differential Evolution a Method of Global Optimization”, 2002.

[37] Dat Tan Pham, “Design Optimization using Fast Annealing Evolution Algorithms”, 2003.

[38] S.H. Lee, “Manipulability-Based Design and Analysis of a Hybrid Manipulator for Highway Applications”, Mechanics Based Design of Structures and Machines, 33:99-118, 2005.

[39] K.L. Conrad, P.S. Shiakolas, T.C. Yih, “On the Accuracy, Repeatability and Degree of Influence of Kinematic Parameters for Industrial Robots”, International Journal of Modeling and Simulation, 2002.

BIOGRAPHICAL INFORMATION

Sandipak Bagchi received his Master of Science in Mechanical Engineering from the University of Texas at Arlington. Before joining UTA as a master's student he completed his Bachelor of Engineering from Siddaganga Institute of Technology, Tumkur, India affiliated to Visveshwaraiah Technological University, Belgaum. His research interests include design optimization and robotics.