Optimization-based Inverse Kinematics and Redundancy Resolution of Hyper-Redundant Robots in Cluttered Workspaces

A Dissertation
Submitted in partial fulfillment of the requirement for
the award of the degree of

DOCTOR OF PHILOSOPHY

in

MECHANICAL ENGINEERING

by

V V M J Satish Chembuly

(Roll No: 715029)

Under the supervision of

Dr. Hari Kumar Voruganti

Associate Professor

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY,

WARANGAL (TS) - 506004, INDIA.

JANUARY 2021

APPROVAL SHEET

This Thesis entitled "**Optimization-based Inverse Kinematics and Redundancy Resolution of Hyper-Redundant Robots in Cluttered Workspaces**"
by Mr. **V V M J Satish Chembuly** is approved for the degree of Doctor of Philosophy.

Examiners

_____

_____

_____

Supervisor

4/11/2021

Dr. Hari Kumar Voruganti
Associate Professor
Mechanical Engineering Department,
NIT Warangal.

Chairman

Prof. Adepu Kumar
Head, Mechanical Engineering Department,
NIT Warangal.

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY

WARANGAL (TS) – 506004, INDIA.

CERTIFICATE

This is to certify that the dissertation work entitled **"Optimization-based Inverse Kinematics and Redundancy Resolution of Hyper-Redundant Robots in Cluttered Workspaces"** which is being submitted by **Mr. V V M J Satish Chembuly (Roll No. 715029)**, is a bonafide work submitted to the Department of Mechanical Engineering, National Institute of Technology, Warangal in partial fulfillment of the requirement for the award of the degree of **Doctor of Philosophy** in **Mechanical Engineering.**

To the best of our knowledge, the work incorporated in this thesis has not been submitted elsewhere for the award of any degree.

4/11/2021

Dr. Hari Kumar Voruganti
(Research Supervisor)
Department of Mechanical Engineering
National Institute of Technology
Warangal- 506004

Prof. Adepu Kumar
Head, Department of Mechanical Engineering
National Institute of Technology
Warangal-506004

DECLARATION

This is to certify that the work presented in the thesis entitled **"Optimization-based Inverse Kinematics and Redundancy Resolution of Hyper-Redundant Robots in Cluttered Workspaces"** is a bonafide work done by me under the supervision of **Dr. Hari Kumar Voruganti,** and was not submitted elsewhere for the award of any degree. I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date: 04/1/2021

(V V M J Satish Chembuly)
(Roll No.: 715029)

Dedicated to

Almighty God

ACKNOWLEDGMENT

It gives me immense pleasure to express my deep sense of gratitude and thanks to my supervisor **Dr. Hari Kumar Vorugnati**, Associate professor, Department of Mechanical Engineering, National Institute of Technology Warangal, for his invaluable guidance, support, and suggestions. His knowledge, suggestions, and discussions helped me to become a capable researcher. He has shown me the interesting side of this wonderful and potential research area. His encouragement helped me to overcome the difficulties encountered in my research as well in my personal life also. He has also provided me opportunities to grow as a leader which may become the foundation for my future life.

I wish to express my sincere thanks to **Prof. N. V. Ramana Rao**, Director, NIT Warangal for his official support and encouragement.

I am very much thankful to **Prof. Adepu Kumar**, Head, Department of Mechanical Engineering for his constant encouragement, support and cooperation.

I also express my sincere thanks to **Prof. R. N. Rao, Prof. N. Selvaraj**, **Prof. P. Bangaru Babu** and **Prof. C.S.P Rao**, former Heads, Department of Mechanical Engineering for their constant encouragement, suggestions, support and cooperation.

I take this privilege to thank all my Doctoral Scrutiny Committee members, **Dr. P. Bangaru Babu**, Professor, Department of Mechanical Engineering, **Dr. Syed Ismail**, Assistant professor, Department of Mechanical Engineering and **Dr. T. P. Tezeswi**, Assistant Professor, Department of Civil Engineering for their detailed review, constructive suggestions and excellent advice during the progress of this research work.

Special heartful gratitude to my senior Ph. D scholar **Mr. Sangamesh Gondegoan, Vinay Kumar Ummidivarapu** and Junior Ph. D scholars **Mr. Rafaque Ahmad** and **Monex Sharma** for being supportive and standing alongside throughout the course.

I take this opportunity to convey my special thanks to the Faculty, Scholars, Nonteaching Staff, M. Tech friends (**Mr. Garvith Saxena, Mr. Subrata Mondal, Mr. Srinivas Bairy, Mr. Vamsi Krishna, Mr. Mayank Singh Parihar, Mr. Mitesh Gupta, Mr. Maneesh, Mr. Indrajith Biswas and Mr. Vaishak Ashok**) and all others in the Department and in the Institute who directly or indirectly helped me to reach goal.

I especially thank **Mr. G. Arun Manohar, Mr. G. Kartheek, Dr. Gudupudi Suresh, Mr. CH. Naresh, Mr. Praveen Oggu, Mr. Muzaffar, and Mr. Manoj Kumar Dundi** for being my friends throughout this course.

I render my respect to my mother (**Late. Sarva Lakshmi**) in this regard, her blessings made this possible. My heartfelt gratitude to my beloved family, (Mother-in-law (**M. Ravanamma**), Wife (**CH. Anusha**), Daughter (**CH. Ananya Samhitha**), Son (**Sai Ashrith Hrishikesh**)) for their encouragement and support during difficult times. I wouldn't finish this task without their cooperation throughout the course.

I acknowledge my gratitude to all my teachers and colleagues at various places for supporting and cooperating with me to complete the work. I render my respect to my parents for giving me support. Also, I take this opportunity to thank all my relatives, friends and well-wishers who are part of this journey directly and indirectly.

ABSTRACT

Hyper-redundant manipulators possess more degrees of freedom (DOF) than the required number of DOF to perform a particular task. The additional DOF enables to fulfill the secondary criterion such as obstacle avoidance, singularity avoidance, and joint-limit avoidance by satisfying the primary task of reaching end-effector to required task space location. Inverse kinematics (IK) of redundant manipulators remained challenging due to more number of unknown variables compared to their kinematic equations and equations are transcendental in nature.

This research proposed an optimization-based approach for determining IK solutions and redundancy resolution methods of hyper-redundant robots. Planar kinematic model have been developed and IK solutions are determined for different task space locations by avoiding polygonal obstacles. An effective collision detection scheme is devised and classical optimization algorithms were used for evaluating the IK solution. Redundancy resolution is performed by considering various performance metrics such as joint-distance minimization, maximization of manipulability measure, and minimization of power consumption. The task of redundancy resolution is posed as a constrained optimization problem for different end-effector paths and working environments. Simulations were performed on planar redundant manipulators with different performance measures.

Spatial representation of the redundant manipulator has been chosen for simulation which can be suitable for a real-time working environment. To accommodate the flexibility and manipulability in narrow regions, the joints were modeled with multiple degrees of freedom (DOF) and are considered as universal joints. Each universal joint has two orthogonal DOF which are made by pitch axis and yaw axis. The IK problem is multi-modal in nature and it has multiple solutions. A global search and Multi-start framework have been implemented to determine the multiple kinematic configurations for a given task location. Performance criteria such as joint-distance minimization, singularity avoidance, and collision avoidance have been chosen to perform the task of redundancy resolution. A classical non-linear constrained optimization technique has been implemented to perform the tasks of inverse kinematics and redundancy resolution. The 3D collision avoidance scheme was implemented with a collision detection algorithm by using a bounding box approach. Simulations were performed for 9-DOF spatial manipulators with 3D obstacles in the workspace. To compute the IK solution of robot working in complex working environments, restart procedure with

different initial guess is required with classical optimization algorithms. For such cases, a population-based TLBO algorithm is implemented to compute the joint configurations of the robot.

A realistic working environment has been modelled, similar to the industries at which redundant robots are deployed. IK simulations of 9 DOF spatial robot are performed for several cases such as pipeline inspection, welding of pipe joints, and pick and place applications in work facility layout and warehouses. Results of joint configurations are reported while avoiding the obstacles in the working domains. Results show that the proposed method is accurate and computationally efficient in determining the IK solution of spatial redundant manipulators in a multi-obstacle and restricted environment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

DOF	Degrees of Freedom
SCARA	Selective Compliance Assembly Robot Arm
FK	Forward Kinematics
IK	Inverse Kinematics
D-H	Denavit and Hartenberg
CCD	Cyclic Coordinate Descent
DLS	Damped Least Square
SDLS	Selective Damped least Squares
SQP	Sequential Quadratic Programming
TLBO	Teaching-Learning-Based Optimization algorithm
GA	Genetic Algorithm
PSO	Particle Swarm Optimization
TSL	Task Space Location

LIST OF SYMBOLS

J	Jacobian matrix
a_i	Link length
α_i	Link twist
θ_i	Joint angle
J^+	Pseudo-inverse of Jacobian
E	Current end-effector location
P	Desired task space location
D	Eucledian distance
L	Lagrangian
K	Total kinetic energy
P	Total potential energy
M	Inertial matrix
C	Centrifugal matrix
H	Coriolis matrix
G	Gravity matrix
${}^{o}T_i$	Homogenous Transformation matrix
τ_i	Generalized torque at joint i
I_i	Moment of inertia tensor
$\Delta\theta$	Joint deviation vector
$\boldsymbol{\Delta e}$	Error function of end-effector velocity
ω	Angular velocity
P_i	i^{th} Population
X_j	j^{th} Design variable
M_j	Mean value of the design variable
λ_i	Lagrangian multiplier

CHAPTER-I

1. Introduction

1.1. Overview:

Recent growth in automation and control improved the rate of advancement and productivity in industries. Significant part of automation research has been in the field of robotics. Industrial robots have been used in a wide range of applications for the past several decades and their usage has been increasing remarkably. These are being used in various applications such as material transfer, assembling of parts, packing, hazardous environments, etc.

Kinematic control of the robotic system is very crucial while it is working in complex environments. An essential requirement of a robot is to precisely reach the required location in the task space by avoiding the obstacles in the workspace. As the robot manipulators are usually controlled at a joint level, there is a need to transform the trajectory from the Cartesian space to joint space, which is known as inverse kinematics [1]. Inverse kinematics and trajectory planning for redundant manipulators is the main task in the development of robot technology. In many manipulator controllers such as resolved acceleration control [2] and nonlinear control [3], the IK solutions are used to determine the correction of joint motions for actuators to move the end-effector to the required pose. The IK problem has got different applications in other fields like Biomechanics [4] and computer graphics [5] etc. The motion of the robot is usually undergoing various constraints in both joint space and task space. Apart from the constraints related to kinematic and dynamic aspects, the performance metrics of the robot manipulator needs to be optimized.

This research deals with a redundant robotic manipulator working in complex environments subjected to task space constraints.

1.2. Evolution of robotics

The concept of the robot was first recognized by the Czech playwright Karel Capek in his drama in 1921. The term “robot” is derived from Czech word “robota” which means forced labourer. In 1940, the interaction between robots and humans was envisioned by the three fundamental laws of Isaac Asimov, the Russian science-fiction writer in his novel “Run-around”.

The early robots built in 1960’s originated from the combination of two technologies such as numerical control machines for manufacturing and tele-operators for remote radioactive

material handling. During the mid-20th century, the development of integrated circuits, digital computers and miniaturized components enabled computer-controlled robots to be designed and programmed. Then industrial robots became essential devices in the automation of flexible manufacturing systems in the late 1970's. Further to their wide applications in the automotive industry, industrial robots were successfully employed in general industry, such as the metal products, chemical, electronics and food industries.

In the 1980's, robotics was defined as the science that studies the intelligent connection between perception and action. The action of a robotic system is assigned to a locomotion apparatus (wheels, crawlers, legs, and propellers) to move in the environment or to a manipulation apparatus (arms, end effectors, and artificial hands) to operate on objects present in the environment. The perception is extracted from various sensors providing information on state of the robot (position and speed) and its surrounding environment (force and tactile, range and vision). The intelligent connection is assigned to programming, planning and control architecture that relies on the perception, which exploits learning and skill acquisition of robot.

In the 1990's research was boosted by the need to utilize the robots to address the challenges in human safety in hazardous environments (field robotics), enhance the human operator ability and reduce fatigue, develop products with wide potential markets aimed at improving the quality of life (service robotics).

Robotics has been rapidly expanding into the challenges of the human world in early 21st century. The new generation of robots is expected to safely co-habitat with humans in homes, workplaces, and communities, providing support in services, entertainment, education, healthcare, manufacturing, and assistance.

Artificial intelligence in robotics is revealing a much wider range of applications reaching across diverse research areas such as biomechanics, haptics, neurosciences, virtual simulation, animation, surgery, and sensor networks among others. In return, the challenges of the new emerging areas are proving an abundant source of stimulation and insights for the field of robotics. Robotics today is dealing with research and development in a number of inter-disciplinary areas including kinematics, dynamics, control, motion planning, sensing, programming, and machine intelligence.

1.3. Robot classification

This section presents the classification of robots mainly with focus on serial structures. Robots can usually be classified based on the number of degrees of freedom (DOF) and their kinematic configuration. The working capability of a robot can be evaluated by the number of DOF. A robotic manipulator usually requires a minimum of 6 DOF to reach any position and orientation in its three-dimensional workspace, whereas the planar manipulator needs 3 DOF for a specific position and orientation. For a specific application, one needs to design a robot manipulator based on the number of DOF and kinematic characteristic features of the robot.

Based on the kinematic configuration of the robot structure, robots are mainly categorized into robot manipulators and mobile robots shown in Fig. 1.1. The basic difference in the classification lies in the fact that the base link involving a robotic manipulator is fixed whereas all the links are movable in the task space for mobile robots. Typically, a robot manipulator is a serial type having an open-loop structure and a parallel type with a closed-loop structure. Moreover, there can be a hybrid structure that consists of both an open-loop or closed-loop structure. In general, the type of joints in a robot manipulator can be either prismatic (P) or revolute (R) whereas the link type may be either rigid or flexible. There are many combinations of these joints and links create a different configuration of a robot manipulator.

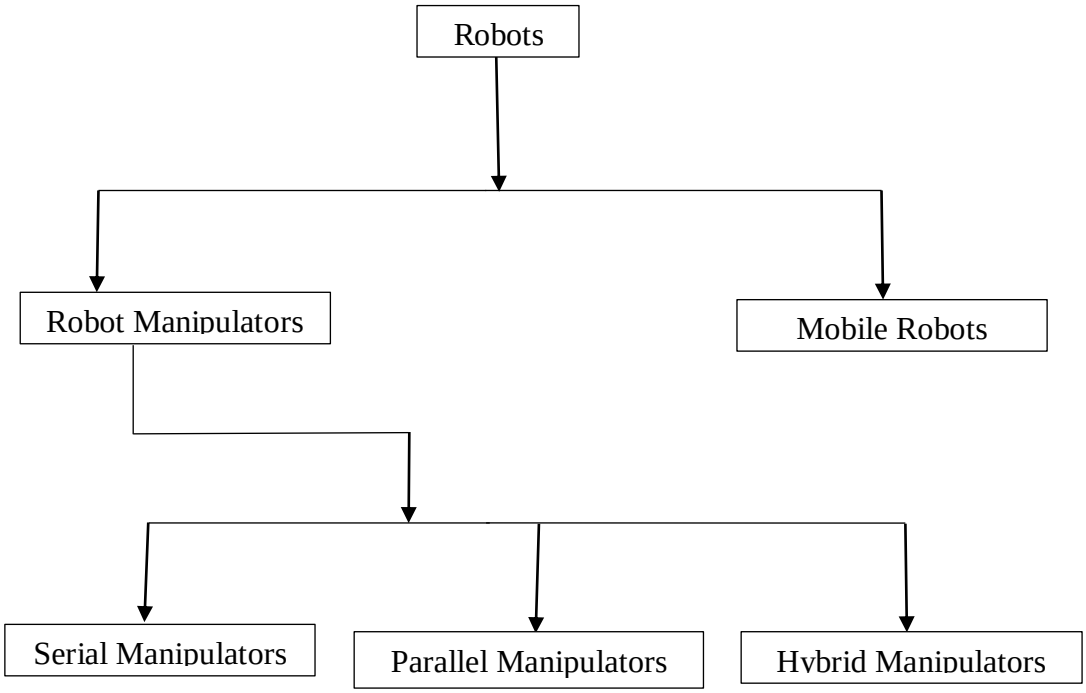


Fig. 1.1. Flow chart representing classification of robots.

The Fig. 1.1 represents the classification of robots when the links of the robot are connected in a sequential manner forming an open kinematic chain. If there is only one possible path to track from one end to the other end of the kinematic chain, then it is known as an open-loop kinematic chain whereas if there is more than one feasible path from one end to the other of the chain, it is called a closed-loop serial manipulator. The serial manipulator with planar linkages are easy to model due to their simple kinematic structures. Examples of open-loop industrial robots include PUMA and SCARA Robots. Serial robots are mainly used in industries for welding, machining, assembling, and material handling tasks, etc. Fig. 1.2 shows a PUMA robot and Fig. 1.3 depicts a robotic arm performing welding operation. Serial robots with more number of DOF than the required to perform a desired task is defined as redundant manipulators. Additional DOF of the robot improves the robot capability and allows the robot to work in various working conditions. A class of manipulators that denotes the combination of open-loop and closed-loop kinematic chains are referred to as hybrid manipulators. This thesis mainly focused on serial manipulators with redundant degrees of freedom.

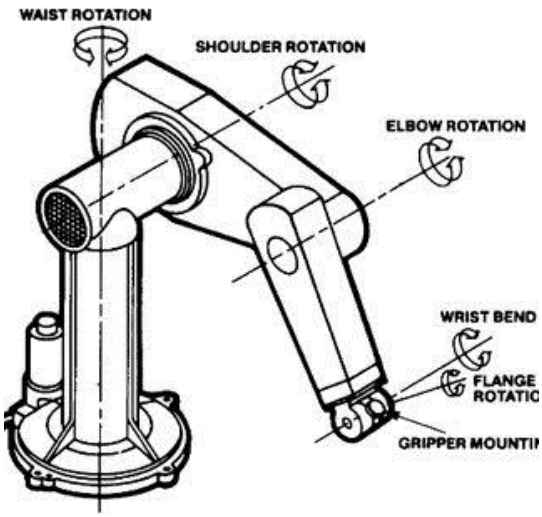


Fig. 1.2. PUMA 560 robot arm [6].

Fig. 1.3. Robot in industrial applications [7]

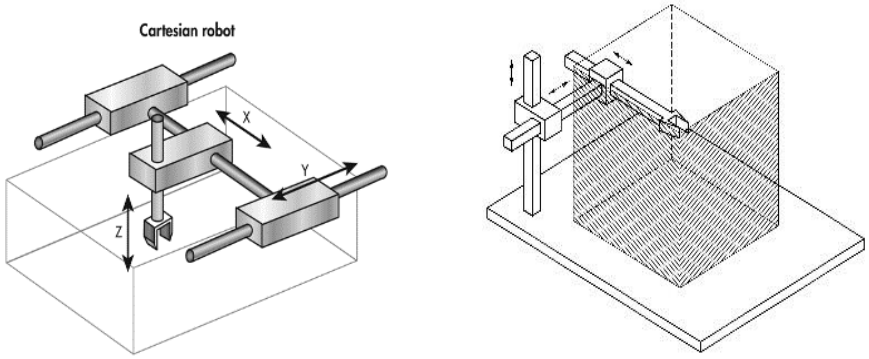
1.3.1. Classification based on workspace

The workspace of the robot is defined as the total volume covered by the end-effector while the manipulator completes the maximum possible movements. The workspace of the manipulator can be determined by the geometry of the robot structure and the limits of the joint parameters. There are two types of workspaces which are reachable and dexterous

workspace. Reachable workspace can be described as a space which the end–effector can reach the limited volume of the workspace in any arbitrary orientation, whereas in the dexterous workspace the end-effector can access any point in the workspace in any orientation. But practically dextrous workspace is appropriate for ideal geometries and generally, it does not suit industrial manipulators. Thus there is a need for the usage of redundant manipulators in different complex environments, these robots improve the dexterity of the manipulator. A classification of the robot is explained based on its kinematic configuration and work envelope of the robot.

Cartesian robot

Cartesian robots have three orthogonal perpendicular slides, giving linear motions along the three principal axes shown in Fig. 1.4 (a). The endpoint of the arm is capable to work in a cuboidal workspace shown in Fig. 1.4 (b). The Cartesian configuration can be used when a large work volume with low dexterity is the requirement.

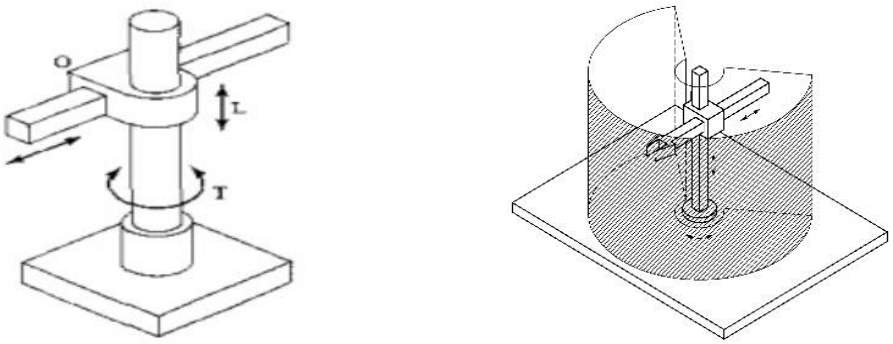


(a) (b)

Fig. 1.4. (a) Cartesian robot configuration [8] (b) Work volume [9].

Cylindrical robot

Cylindrical robots possess one revolute joint along with two prismatic joints (RPP), this arrangement creates cylindrical coordinates of end-effector shown in Fig. 1.5 (a). The workspace of this configuration is restricted as two concentric structures of a cylinder with finite length as shown in Fig. 1.5 (b). It is suitable to access narrow horizontal spaces and hence it can be used for assembly and machine loading operations.

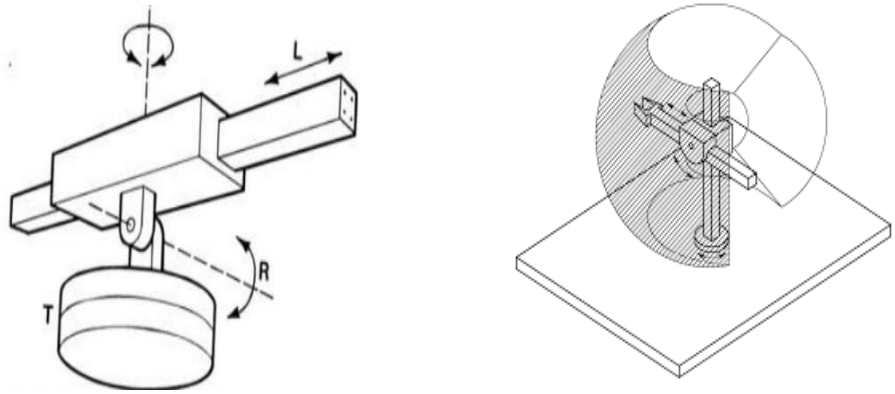


(a) (b)

Fig. 1.5. (a) Cylindrical robot configuration [10] (b) Work volume [9].

Spherical (polar) robot

It consists of a prismatic joint that can be raised or lowered about a horizontal revolute joint. These two links mounted on a rotating base. This arrangement is known as the RRP configuration shown in Fig. 1.6 (a), which allows the endpoint of the arm to move in a spherical workspace as shown in Fig. 1.6 (b). Polar configuration robot arms are mainly employed for industrial applications such as machining, spray painting, and so on.



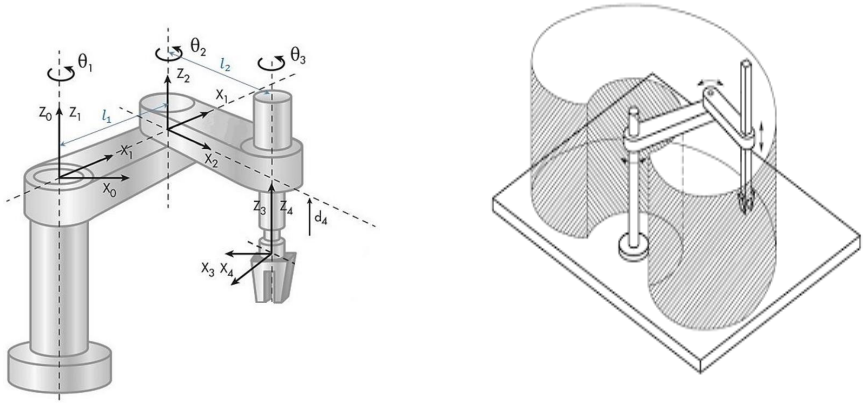
(a) (b)

Fig. 1.6. (a) Spherical robot configuration [10] (b) Work volume [9]

Selective compliance assembly robot arm (SCARA)

SCARA robot is designed for assembly tasks, which provide rigidity along the vertical axis and compliance along the horizontal axis. The SCARA configuration has vertical major axis rotations such that gravitational load, Coriolis, and centrifugal forces do not stress the structure. This makes the SCARA configuration suitable for assembly tasks. This configuration possesses three revolute joints and one prismatic joint knows RRRP shown in

Fig. 1.7 (a). The workspace of the SCARA configuration is a concentric hollow cylinder shown in Fig. 1.7 (b).

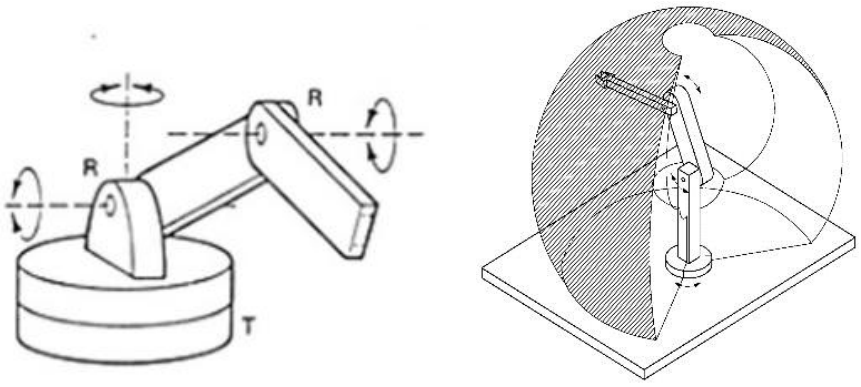


(a) (b)

Fig. 1.7. (a) SCARA robot configuration [8] (b) Work volume [9].

Articulated robot configuration

The articulated arm is the type of configuration that simulates a human arm and this type of arm is referred to as an anthropomorphic manipulator. It consists of two links corresponding to the human forearm and upper arm with two rotary joints corresponding to elbow and shoulder joints. These two links are mounted on a rotary table corresponding to the human waist. This configuration is defined as RRR configuration (shown in Fig. 1.8 (a)), the work volume of his configuration is spherical shape, with proper sizing of links and design of joints the endpoint of the arm can access full spherical space shown in Fig. 1.8 (b). The anthropomorphic structure is the most dexterous one and suitable for a wide range of industrial applications.



(a) (b)

Fig. 1.8. (a) Articulated robot configuration [10] (b) Work volume [9].

1.3.2. Classification of robots based on DOF

Robots are generally classified according to the number of degrees of freedom, degrees of freedom indicates the capability of the robot. A general task of robot consisting of arbitrary positioning and orienting the end-effector or tool can be achieved by six DOF or six-axis robots. Simple tasks such as painting and simple welding can be done with the five-axis robot and assembly robots often required four DOF to perform a given task

Redundant robots/ hyper-redundant robots

Redundant manipulators have more degrees of freedom than the required to perform a particular task in the workspace. This high DOF allows the robot to work in the cluttered environment by avoiding obstacles [11], singular configurations [12], and mechanical joint limits [13]. This in turn improves the dexterity of the robot in the workspace. Inverse kinematics of redundant manipulators have multiple solutions, which provides the flexibility to choose the best solution from the available solutions based on some performance criteria such as joint torque minimization [14], joint distance minimization [15] , and consumption of energy minimization [16].

Hyper-redundant robot possesses a large number of degrees of freedom shown in Fig. 1.9, such robots are similar to snakes or continuum robots and are useful in the operation of highly constrained environments. High degrees of freedom and greater maneuverability in the workspace makes these robots widely applicable in the fields of medical [17], aerospace [18], space exploration [19], orbital servicing such as space craft construction repair maintanence [20] etc, underwater exploration, and nuclear core reactors. The additional DOF of the redundant manipulator makes this robot more suitable to work in a wide range of applications. Some of the industrial applications of redundant robots are welding, Assembling, machining and additive manufacturing [21], shown in Fig. 1.10. Redundant robots has been employed in rescue missions, inspection and manipulation of complex pipe installation and nuclear plant installations [22] etc. Fig. 1.11 shows the robot employed in on-orbital servicing. Redundant robots working in space exploration shown in Fig. 1. 12. The kinematic structure of the redundant manipulators offers an advantage to applying various additional task constraints other than the primary task of reaching the end effector in the task space.

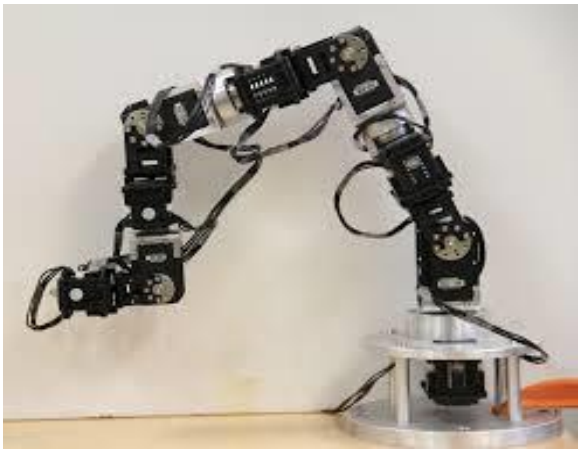


Fig. 1.9. Hyper-redundant manipulator [23].

Fig. 1.10. Redundant robots working in industrial applications [21]

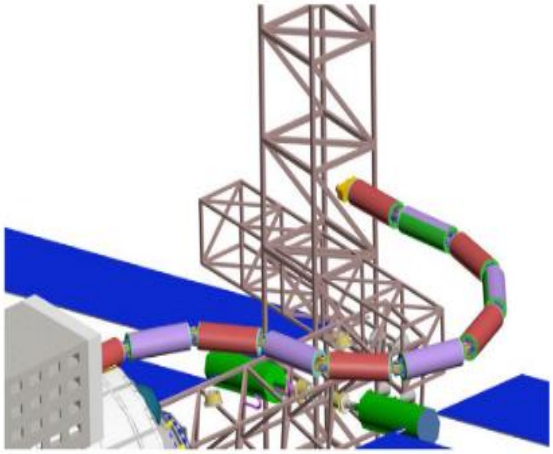


Fig. 1.11. Hyper-redundant manipulator working in on-orbital servicing applications [24]

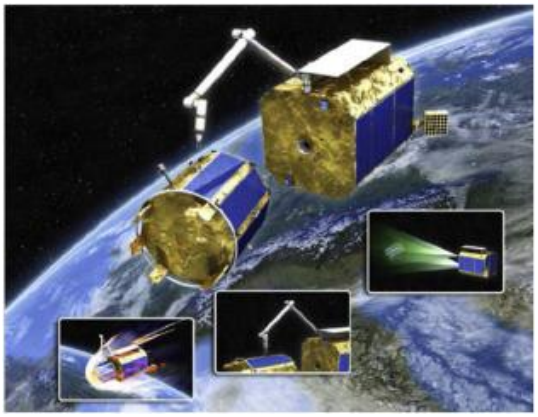


Fig. 1.12. Redundant robot working in space exploration [25]

Deficient manipulator

Deficient manipulators are referred to as those that possess less degrees of freedom than required to achieve admissible configuration in the workspace [26]. This type of manipulator can be used to perform certain tasks in the operational space, for which total DOF of the robot is not required to move the manipulator in a specified direction. Multiple kinematic deficient robots can be employed as cooperating robots to perform a common task in industrial application. Kinematic deficient manipulators can be used, unless kinematic redundancy is required in the task space and joint space to avoid obstacles and singularities. These manipulators gain the advantage over redundant manipulators in terms of manufacturing cost and compactness.

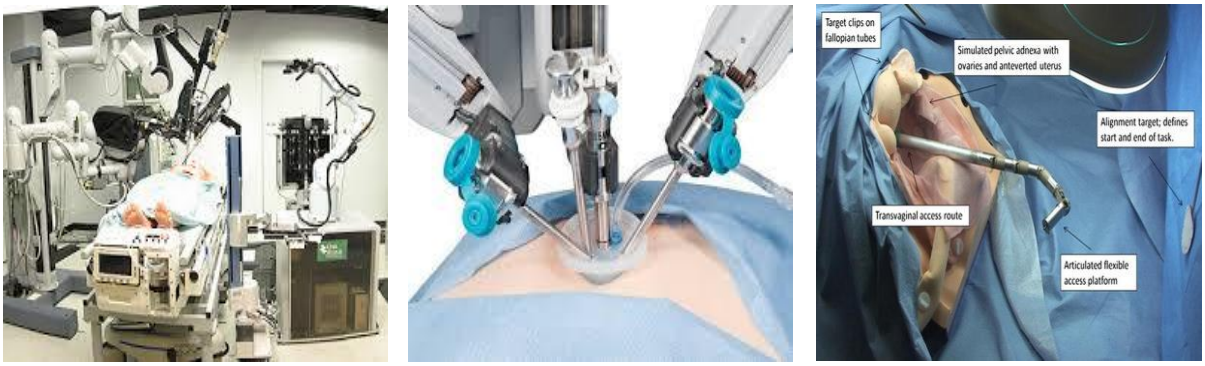
In this thesis, hyper-redundant manipulators working under different environments were presented.

1.4. Motivation

Generally, more complex serial robots are designed and built to suit a wide variety of applications. Many of them are modelled with multiple links/arms and they are redundant. This redundancy gives more flexibility and the robots can not only reach a particular position but also reach it with several configurations, thereby secondary goals can be achieved. More recently, there is renewed interest in the use of hyper-redundant manipulators due to their increased applications in all the fields. The use of redundant robots in different applications is shown in Fig. 1.13 (a-c). Fig. 1.14 (a-c) shows the use of hyper-redundant robots in medical and surgical applications. The characteristic equations of kinematic problems are non-linear and transcendental for which, closed-form solutions are not always possible. The variables in the equation are multi-valued and provide multiple solutions and they are configuration dependent. The task of resolving the best solution among the multiple solutions is known as redundancy resolution, which requires to choose objective satisfying task constraints.

(a) (b) (c)

Fig. 1.13. Hyper-redundant manipulator (a) Industrial application [27] (b) Tunnel inspection [28] (c) Under water inspection [29].



(a) (b) (c)

Fig. 1.14(a-c) Hyper-redundant manipulator in surgical applications [17, 30, 31].

There are several methods for redundancy resolutions. Some of the issues with redundancy resolution methods such as computation of Jacobian inverse for the under-determined system are computationally expensive and it is sensitive at singular configuration. The kinematic control of redundant manipulator, suitable to access narrow regions such as ducts and pipeline passages involves a certain difficulty in avoiding different shapes obstacles and accessing in different working environments. The task of modelling a suitable kinematic structure of a redundant robot for the required environment is crucial. The design of a redundant manipulator is suitable to work in narrow regions and hazardous working environments. Parameters to be considered while controlling and designing a redundant robot such as type of DOF, kinematic structure, mechanism, and type of material. The challenges involved in solving the inverse kinematics of redundant robots increased the attention of researchers towards the problem. The research presented here was focused on the IK solution and trajectory planning of a 9 degree-of-freedom manipulator, meant to use in an industrial manufacturing environment

1.5. Problem formulation

The general purpose of the IK problem of a manipulator is to find a configuration in the joint space so that the position and/or orientation of the end-effector(s) satisfy desired kinematic constraints.

The configuration of the end-effector is expressed by a vector function $\boldsymbol{X(\theta)}$ in terms of joint variables. The target location for the end-effector pose can be given by a vector $\boldsymbol{X}_d$. The aim is to find the feasible values of a vector $\boldsymbol{\theta}$ such that $\boldsymbol{X(\theta) = X}_d$. The solution of the IK problem leads to minimization of the Eucledian distance between current configuration of the end-effector and the configuration at the target location, this can be obtained by solving the following non-linear system of equations

$$E(\theta) = 0 \tag{1.1}$$

subjected to constraints

$$\theta_l \le \theta \le \theta_u \tag{1.2}$$

where θ_l and θ_u represents the lower and upper bounds of the joint variable and

$E(\theta) = E(X(\theta), X_d)$ is a positional error function vector, the detail of the error function vector will be discussed in chapter 3. This error function allows the robot to reach required task space location. In addition, to the primary task of reaching end-effector to the task space location an additional performance criterion such as Joint-distance minimization joint-torque minimization and singularity avoidance have been considered. The formulation of these performance criteria have been presented in subsequent chapters

1.6. Outline/ organization of the thesis:

The thesis is organized in the following manner,

I. Chapter 2 deals with literature related to inverse kinematics and redundancy resolution methods of hyper-redundant manipulators. This chapter also presents different techniques of obstacle avoidance, singularity avoidance, and joint limit avoidance and their limitations.

II. Chapter 3 describes kinematic and dynamic analysis of robot. This analysis helps to understand the fundamental concepts of manipulation and performance criteria of the robot.

III. Redundancy resolution techniques were presented in chapter 4. These techniques find the best solution among feasible solutions based on robot performance criterion.

IV. Collision avoidance is a crucial requirement for a redundant robot while working in a cluttered environment. Chapter 5 presents different types of collision detection and avoidance techniques of a redundant manipulator working in a planar and 3D environment.

V. The optimization techniques used for determining IK solutions are briefly explained in chapter 6. Global optimization techniques used to evaluate multiple solutions of the robot are also presented in this chapter.

VI. In chapter 7, IK simulation of the redundant robot in a planar workspace with convex and non-convex obstacles is presented. The task of redundancy resolution is implemented on a planar robot while avoiding both obstacles and singularities. The objective of minimizing power consumption of planar robot tracing different paths is performed

VII. In chapter 8, IK simulations are performed on a 9 DOF spatial redundant manipulator by avoiding 3D obstacles. Multiple solutions of the spatial robot are attained using a global optimization approach. Spatial redundant robot simulations are performed

while the robot is deployed in real-time applications such as work facility layout, pipe layout, and warehouse models.

VIII. Concluding remarks of the thesis and scope for the future research work is included in chapter 9.

CHAPTER-II

2. Literature Review

The advancement of robot technology and the increasing use of robots in various applications is gaining the attention of researchers in the field of robotics. The focus has been laid on the issues related to kinematic and dynamic aspects is important as the robots are used for different tasks. There is a large increase in the use of redundant manipulators in a wide range of applications due to their additional degrees of freedom (DOF). This extra DOF improves the capability of the robot to work in narrow and cluttered environments. The inverse kinematic (IK) solution is the primary task while exploiting different abilities of the manipulator such as obstacle avoidance, joint limit avoidance, and singularity avoidance, etc. Hence, this draws huge research attention to IK and control of redundant robots. A lot of research has been carried out in IK solution and redundancy resolution techniques of redundant manipulators. This chapter describes various IK solution, redundancy resolution, and obstacle avoidance techniques of redundant manipulators.

2.1 Kinematic modelling

A robot consists of a set of rigid bodies called links connected by joints. A simple type of joints that are generally used are revolute (rotational) and prismatic joints (translational). The kinematics of any robot can be described by its position and orientation of end-effector corresponding to the joint configuration. This is a mapping from joint space to task space and is known as forward kinematics. Therefore, kinematic modelling is commonly referred to as the forward kinematics of the manipulator. Several schemes have been proposed for the forward kinematics of robot manipulators.

2.1.1. Denavit-Hartenberg notation

The manipulator motion can be analyzed by considering a standard procedure of building a coordinate system on each link. This was first introduced by Denavit and Hartenberg (DH) in 1955 [32] which establishes coordinate frames attached to each link of the frame in a systematic manner. The DH representation for a given posture of end-effector specifies a displacement for the joint-link couple by defining four parameters the link length a_i , link twist α_i joint angle θ_i and link offset d_i, shown in Fig. 2.1. According to this convention, the position and orientation of each link in the coordinate system are related to the previous link is represented by a 4x4 homogenous transformation matrix.

$$T_i^{i-1} = \begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta\cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.1)$$

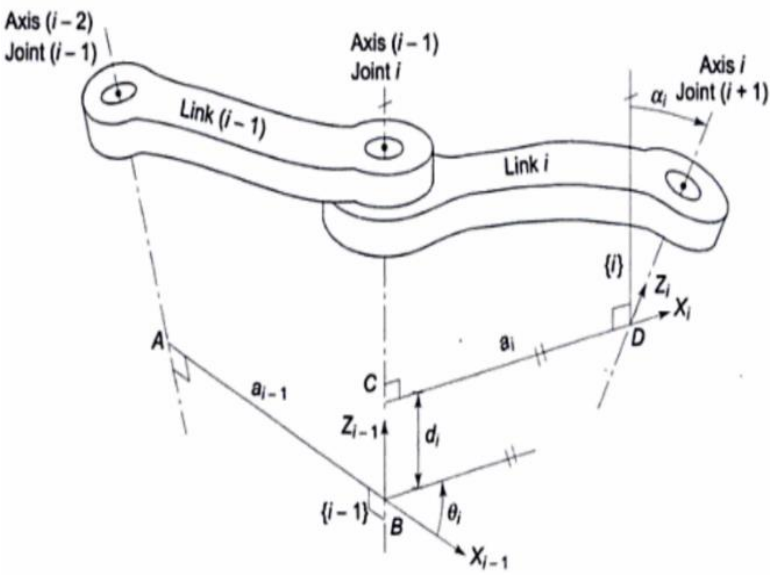


Fig. 2.1. Standard DH representation of a revolute joint [32].

Although DH notation represents convenient kinematic modelling, it is only suitable for some classes of robots because of constraints involved with an overdetermined system that is used to describe the displacement between two frames. This issue can be handled by providing some virtual joints on some links whose geometry violates these constraints.

2.1.2. Gupta notation

K C Gupta [33] introduced the zero reference position method as a new technique for handling the kinematic problem. The computation of the current posture of the link depends on the displacement of the mechanism from the zero reference position instead of joint-to-joint computations. Manipulator description in this approach gives a better understanding of motion. The disadvantage of this convention is that it cannot furnish itself with a robot functional classification.

2.1.3. Sheth and Uicker convention

This convention was developed by P. N. Sheth and J.J. Uicker [SU] in 1971 [34], which extends the use of DH parameters to all rigid link mechanisms. Due to the increased

flexibility in the choice of a coordinate system, this convention requires six parameters to define the shape of the link. SU is known as two frame convention since it requires two frames that are specifically defined and placed for each joint. This convention simplifies the kinematic description of an arbitrary mechanism and enables the computation of forward kinematics. In comparison with the standard DH convention, this convention establishes an extra frame for each joint and generality of the description can be attained.

2.2. Inverse kinematics solution techniques

Kinematic control of redundant manipulator mainly requires IK solver, Inverse kinematics approaches have been extensively investigated. IK methods are broadly categorized into analytical, geometrical, numerical, or iterative and evolutionary-based approaches. This section describes the inverse kinematic approaches of redundant manipulators by considering secondary criteria such as obstacle, singularity, and joint-limit avoidance.

2.2.1. Analytical approaches

Analytical or closed-form solutions for the IK problem can be obtained for a simple manipulator with less number of DOF. This approach is computationally fast and it can find all possible solutions. One of the first analytic solution was proposed by Piper in 1968 [35], presented closed-form techniques for six DOF manipulators when any three consecutive joint axes intersect at a common point.

It was observed that arms with serial links are best suitable for having an analytic solution with spherical wrist manipulation. In many of the arm chain models, the wrist partitioning technique has been used [36]. In this technique, the arm is assumed to be separated at the wrist joint and the hand is desired to move in the space so that the end-effector satisfies desired kinematic constraints. In this method, the position of the wrist center is determined according to the desired position and orientation of the end-effector. The remaining portion of the arm is assumed to trace the position of the hand so that the wrist joint is considered as an end-effector for the rest of the arm. Hollerbach and Sahar in 1983 [37] proposed analytical solutions for 6R human arm using wrist partitioning technique. Shimano and paul [38] also proposed an analytical IK solution for Stanford robotic arm. A unique closed-form IK formulation is not available for all 6 DOF manipulators. In such cases, numerical techniques have been implemented. Several innovative analytical approaches have been proposed for certain specific geometries. Mihelji [39] proposed an analytical approach for the IK solution of a human arm model robot used for rehabilitation purposes. Analytical approaches have

also been proposed for robots with more than 6 DOF. Shimizu et al. [40] Proposed an analytical IK solution for 7 DOF redundant manipulators by specifying joint limits, which furnish all the feasible IK solutions among the global configuration space. In this approach, a closed-form IK Solution was achieved using the parameterization method and later analytical technique has been developed for computing the IK solution within the joint limits. The redundancy resolution problem has also been solved using this approach in the position domain. The disadvantage with the analytical approach is that, these are configuration dependant methods and do not exist for a general manipulator with arbitrary geometry.

2.2.2. Numerical iterative approaches

The numerical iterative approach has been widely employed for IK of redundant manipulators, for those explicit closed-form solutions do not exist.

The most commonly used algorithm to solve the non-linear kinematic equations is the Newton-Raphson technique, which works based on simultaneous successive linear interpolation of non-linear equations [41]. In this method, the non-linear system is represented with a linear function by considering only first-order terms of the Taylor series of the function representing the kinematic constraints [42]. As this approach uses a first-order approximation to the non-linear equations, the convergence rate is slower when compared with second-order approaches. Although these techniques have faster convergence than the gradient descent method. The disadvantage with this approach is that the iterative procedures may diverge due to poor initial guess. Piper [35] used the Newton Raphson approach initially for solving the IK problem. In his approach, he proposed two different methods for the forward kinematics problem with different conventions. In one case, the forward kinematics is assumed as homogenous transformation matrix multiplication and in the other, it is viewed as a screw transformation matrix.

Jacobian based methods

The Jacobian of a manipulator is the matrix that linearly transforms joint velocities into the end-effector velocities [43]. Different Jacobian based approaches have been implemented for solving the IK problem, some of them are presented below

Gradient descent method

This approach tries to move the joint variable value towards the direction in which the square norm of error vector decreases with the maximum possible rate [44], the direction is defined as the steepest descent direction which is specified as

$$\delta = -\nabla g(\theta) = -J_e(\theta)^T . e(\theta) \tag{2.2}$$

where $e(\theta)$ is the vector representing the error residual function, J_e is the Jacobian matrix. $g(\theta)$ is the square of the norm of error residual function, given as.

$$g(\theta) = \| e(\theta) \|^2 \tag{2.3}$$

The joint angle update rule is represented as

$$\Delta\theta = \mu \cdot \delta \tag{2.4}$$

where μ indicates the value of step size.

The procedure continues till the norm converges to zero [45] i.e. all the kinematic constraints are fulfilled. This approach does not require Jacobian inverse, thus it is computationally inexpensive and there are fewer chances of facing divergence issues in the iterative process when compared with Newton Raphson algorithms. The limitation of this approach is that it has a slow convergence rate after some iterations.

Jacobian inverse

A solution to the IK problem aims to determine the roots of the nonlinear equation by using a Newton-Raphson technique.

The formulation of the IK problem creates *m* equations and *n* unknowns, where *n* is the number of joints and *m* is the number of elements corresponding to the task space locations, by assuming the equality of number of equations and number of unknowns. Solution of this

problem using Newton-Raphson approach leads to joint update rule

$$\Delta\theta = -J_e^{-1}.e \tag{2.5}$$

The Jacobian inverse method converges faster than the gradient descent method however it suffers from a limitation that the inverse of Jacobian is sensitive and ill-conditioned at the vicinity of the singularity.

Hessian based methods

Unlike the Jacobian based methods used to solve the system of kinematic equations, in optimization-based or Hessian based approaches, the IK problem is considered as a nonlinear optimization problem [46]. In this method, the problem is formulated in a way to find joint variable vector $\boldsymbol{\theta}$ that minimizes the objective function $f(\boldsymbol{\theta})$. Computation of the Hessian matrix is required to compute the solution of the IK problem. Hessian matrix deals with the second derivative of the error function with respect to the joint variable. Although this approach is numerically stable than Jacobian inverse methods, higher complexity is involved while calculating the Hessian inverse. The major issue with this approach is, the algorithm needs to perform the pseudo-inverse of Jacobian to exploit the redundancy of the manipulators. This increases the computational cost even further.

Heuristic search methods

The heuristic search method does not require gradient information for solving the IK problem, these methods are not computationally expensive but the local convergence rate is slow when compared with gradient-based methods

The cyclic coordinate descent (CCD) method is being used popularly for solving the IK problem of serial manipulators [47]. It is a numerical iterative approach that works to minimize positional and orientation errors by operating a single joint at a time, starting from end-effector to base. Although this approach is effective, it cannot handle the global constraints of the manipulator.

2.2.3. Hybrid analytical/numerical methods

Analytical methods for solving the IK problem involve complex trigonometric equations. For redundant manipulators, the problem is underdetermined and it is also required to satisfy additional task constraints. However, there are some approaches to exploit the features of analytical solutions for a manipulator with complex geometries. Some algebraic methods have been used for IK solutions using algebraic elimination which expresses one of the joint variables as the roots of the polynomial equation and other joint variables are determined in terms of a known variable using closed-form procedures [48]. These methods are sometimes referred to as analytical solutions. As the degree of the polynomial is greater than four, an iterative procedure must be employed to solve the given problem. However numerical algorithms for solving the roots of the polynomial equations are fast and reliable. These

solutions are not available for all categories of the manipulator, but they are less configuration dependent when compared with pure analytical approaches. Algebraic techniques can give all feasible solutions for a given IK problem. These approaches can also be used for the symbolic computation of the IK solutions. Ananthanarayanan and Ordonez [49] presented a hybrid approach by combining analytical and numerical approaches for computing the IK solution of generalized (2n+1) hyper-redundant manipulators. In this approach, a spherical joint is considered at a wrist with obstacle avoidance and joint-limit avoidance as a secondary criterion. This method utilizes the simplicity of the analytical approaches to improve the speed of the numerical solvers that operate on each joint position individually, thus this method considerably improves computational efficiency.

2.2.4. Geometric approach

The geometrical approach has been largely used for computing the IK solution of redundant robots. It has the benefit of good geometric intuition and less computational cost . Chirikjian and Burdick [50] adopted a geometric technique for computing the IK of hyper-redundant manipulators by representing them as a characteristic curve of a robot, known as the backbone curve. The discrete manipulator was modelled as a continuous curve. The IK analysis was performed by fitting the physical manipulator to the backbone curve. In this approach, obstacle avoidance has been performed by considering tunnels as obstacles. This approach is suitable for a discrete planar manipulator with high DOF and also well suits for the continuous manipulator. Xu et al. [24] proposed a modified modal approach for computing the mission-oriented IK of hyper-redundant manipulators that are used for orbit servicing applications. To represent the geometry of the manipulator the 3D backbone curve is specified using the mode functions. The equivalent link of the robot is fitted to the appropriate backbone function. Joint angles were determined by evaluating the position of each node. The coordinate points of each node can be obtained by fusing the complete length of the manipulator and mode functions. An additional parameter called arm angle is used for the redundancy problem. This approach is implemented for 12 DOF robots satisfying additional tasks such as obstacle avoidance and singularity avoidance. Menon et al. [51] presented a geometric approach for motion planning of hyper-redundant manipulators. This is an optimization-based approach where the motion of the links of the manipulator was supposed to take the motion of the Tractrix curve. The velocity of the curve lies along with the link and it is least among all the possible velocities. The hyper-redundant manipulators are represented by splines and a Tractrix based algorithm is employed to obtain length

conserving motion of the hyper-redundant robot. In this approach, better visualization of motion can be obtained due to localized shape control and higher-order continuities that are available in splines. Sardana et al. [52] presented a simple geometric technique for computing the inverse kinematics of a four-link redundant manipulator of an in-vivo robot for medical applications. In this approach, redundancies are introduced by providing twisting joints to some of the joints. This approach provides an IK solution easily for a surgical robot used for biopsy application and it is computationally efficient. Yahya et al. [53] used a geometric approach for computing the IK solution of redundant manipulators for a given path. Manipulability measure has been calculated to show how far the manipulability value is from the singularity configuration. The advantage of this approach is, the angle between two adjacent links is assumed to be the same. This makes the movement of links stable and hence controlling of links is easier. The limitation with the geometrical approach is that a closed-form solution should be available geometrically for the first three joints and computing the IK solutions for these joints are time-consuming.

2.2.5. Evolutionary approaches

Computation of IK solution by the evolutionary approach is popular and they have been widely used for different robot configurations. Parker et al. [54] make use of genetic algorithms to compute the IK problem of redundant robots by minimizing joint displacement. This approach has a limitation of poor precision in the solution. Koker [55] applied neural networks and genetic algorithms jointly to determine the solution to the IK problem of the Stanford robot by minimizing the positional error at the end-effector. This approach combines the features of both neural networks and evolutionary approaches and results in more precise solutions. Better accuracy of the solution can be achieved by training three Elman neural networks using separate training sets. One of these trained sets gives better results than the other two training sets and the floating portion of each set is placed in the initial population of the genetic algorithm. Bingul et al. [56] applied a neural network by employing a back-propagation algorithm to compute the IK problem of an industrial robot with an offset wrist. The limitation of this approach is, it cannot give multiple solutions to the IK equations. The neural network approach typically requires large training sets whose size increases exponentially with the number of DOFs. Nearchou [57] proposed an evolutionary approach of binary-coded genetic algorithm to find out the unique solution to the IK equations of the redundant and non-redundant robots. This work shows that the evolutionary approach was superior to the pseudo-inverse method and simple binary-coded genetic

algorithm. Further, it was shown by Deb and Agrawal [58], is that for continuous search spaces real-coded genetic algorithms were more suitable.

One of the popularly used evolutionary approach which mimics the teaching learning environment in a class room is teaching-learning-based optimization (TLBO). TLBO algorithm has been applied in various fields of science and engineering. The main advantage of TLBO is that, it doesn't require any algorithm-specific tuning parameters like crossover and mutation rate in GA, acceleration constants like c_1 and c_2 as in PSO, etc. Rao et al. [59] proposed the Teaching-Learning-Based Optimization (TLBO) algorithm for the optimization of mechanical design problems.The effectiveness of the method is validated on several problems. The examples considered include five benchmark optimization test functions, four mechanical design problems and six real-world mechanical design optimization problems. The obtained results of the proposed algorithm outperformed the previous results obtained by other population-based optimization algorithms. The convergence rate and computational effort are better for TLBO over other evolutionary optimization methods. Rao et al. [60] applied TLBO for large scale non-linear optimization problems. The efficiency of the TLBO algorithm was compared with other popular optimization algorithms (GA, ABC, PSO, HS, DE, Hybrid-PSO) by considering several different benchmark problems with different characteristics. The results obtained by TLBO algorithm showed its applicability for large scale problems. Awatef et al. [61] adopted TLBO technique for navigation problem of mobile robots, this technique is employed for optimum trajectory and minimum travelling time of the robot to reach the goal. To show the efficacy of the approach, the TLBO approach has been compared with other evolutionary approaches. Savsani et al. [62] made a comparative study of seven different metahueristic methods for the trajectory planning of 3 DOF robotic arm by minimizing joint travelling time, joint travelling distance and total joint Cartesian lengths simultaneously. Results show that TLBO algorithm is significant over other optimization algorithms. In the present thesis, TLBO algorithm is used for solving IK problem of hyper-redundant manipulator working in complex environment. In general, classical optimization techniques can be employed for solving IK problems. But in complex domains, classical approaches fails to give solution with single initial guess. Thus, TLBO algorithm is employed for solving the IK problem in complex domains. The detailed explanation of TLBO approach is presented in chapter 6. IK simulation results using TLBO algorithm are reported in chapter 8.

2.3. Methods of redundancy resolution

To take the advantage of redundancy, numerous computational schemes have been adopted. Many researchers have implemented the idea of the pseudo-inverse technique [42] to attain the joint velocities which minimize the joint rates in the least square sense.

2.3.1. Jacobian pseudo-inverse and its variants

One of the extensively used redundancy resolution techniques is the optimized inverse of Jacobian, known as least square or Moorse-Penrose inverse [63]. This inverse can minimize the cost function instantaneously. The cost function is defined as the Euclidean norm of the joint velocity vector

$$g(\Delta\boldsymbol{\theta}) = \sum_{j=1}^{j=n} (\Delta\theta_j)^2 \tag{2.6}$$

This cost function is better approximated when compared with the Newton-Raphson technique. Hence, this approach leads to faster convergence. In the pseudo-inverse approach, redundancy is utilized to minimize the joint velocities or norm of deviation of joint velocities from the current configuration.

This method is also utilized at the joint acceleration level by adding the null-space term to satisfy the additional secondary criteria such as obstacles [64], singularity [65], and joint-limit avoidance [66], and joint torque minimization [14].

The idea of the extended Jacobian method was first proposed by Baillieul [67], this approach is used to execute obstacle avoidance based on optimizing a distance criterion. In this approach, the equations of velocity have been provided with additional equations equal to the number of unknown joint velocities. To make the gradient as zero in the null space of Jacobian, *n-m* rows are added to the Jacobian matrix. Maciejewski and Klein [64] proposed an extension to Jacobian based IK formulation to integrate obstacle avoidance. This approach is to evaluate the joint angle velocities of redundant manipulator working with multiple goals such as the primary target of reaching specified end-effector trajectory and the secondary criterion as obstacle avoidance. This was achieved by decomposing the solution into a particular and homogenous solution, which will exactly represent the priority of the tasks. The end-effector control is achieved by maximizing the distance of links from obstacles. In general, Jacobian methods lack sensitivity at joint limits and have singularities. Kircanski and

Petrovic [68] proposed an IK solution for 7 DOF Robot avoiding singularities. This approach reduces the computational burden by combining the analytical and pseudo-inverse approach.

Liegieos [69] proposed the idea of generalized inverses by utilizing the null space of the rectangular Jacobian. The null space combines the homogenous solution that provides joint motion and does not contribute to end-effector motion. A performance function is being projected on to the null space and the obtained IK solution can optimize the function value. A potential function can be chosen to fulfill the performance requirement of a redundant manipulator. For the task of joint limit avoidance, the joint midrange is selected as a performance metric that measures the deviation of joints from the mid-value, which is given as

$$g(\theta) = \sum_{i=1}^{n} \left(\frac{\theta_i - \theta_i^{mid}}{\theta_i^{max} - \theta_i^{min}} \right)^2 \tag{2.7}$$

Similarly, a performance measure of redundant manipulators has been chosen, that avoids singular configurations. The main limitation of the gradient projection method is that it cannot guarantee that a given configuration is feasible or not since the performance criteria are expressed by a scalar metric. A feasible configuration might be exist corresponding to a higher value of the potential function when compared to a non-feasible configuration.

A new redundancy resolution technique using task priority proposed by Chiaverini [70], this approach can deal with kinematic and algorithmic singularities. This technique has been implemented for 7 DOF redundant robot arm. Huo and Baron [71] presented a twist decomposition algorithm avoiding joint limits and singularities of a redundant robot in a welding application. Singla et al. [72] presented a high index norm minimization approach for redundancy resolution of serial manipulators. In this approach, they have interpreted different norms of intermediate indices and observed that 8-norm solutions are found to be better than other minimum norm solutions.

2.3.2. Configuration control

Another redundancy resolution technique is the configuration control method proposed by seraji [73]. This approach combines the forward kinematics of manipulator with a set of kinematic functions in joint space or Cartesian space that consider the additional task of the redundant robot.

2.3.3. Penalty function method

A popular approach implemented for redundancy resolution is the penalty function method [74] in this approach penalty function is being added to the objective function. The algorithm searches for the minimum value while the penalty function increases its value when a specific condition violates. This approach can be used to avoid joint limits and obstacle avoidance. Zhang et al [75] proposed a similarity in position-based and velocity-based redundancy resolution approaches and found they were equivalent. A classical optimization-based approach has been presented for the IK solution of hyper-redundant manipulator avoiding both obstacles and singularities [76] was implemented at a joint position level. In this approach, there is no Jacobian computation thus reduces the computational burden of the IK solution.

2.4. Trajectory planning

The redundant manipulators while operating in different environments, it is important to attain smooth trajectories that avoid mechanical vibrations of the robot. Thus trajectory planning of redundant manipulators is an important aspect to be considered in the kinematic analysis of robots. Menasri et al [77] formulated a bi-level optimization problem for trajectory planning of redundant manipulators and solved using a bi-genetic algorithm.

B-spline interpolation [78, 79] techniques are another class of approaches for solving trajectory planning. In these approaches, trajectory planning is executed without using the inverse of the Jacobian matrix. The idea of using optimal motion planning with optimal time as a performance criterion has been chosen for path planning under the B-Spline assumption in the task space [80]. A new idea based on the variational approach [81] was attempted. In this approach, the trajectories of the robot joint space were represented as a B-spline curve and the measure of performance is directly integrated through the trajectory of the end-effector. Xidias [82] implemented an optimal time trajectory planning for hyper-redundant manipulator in 3-Dimensional workspaces with obstacles. This approach evaluates an optimization problem to determine the joint trajectories with minimum time consumption while executing the required tasks.

Numerous works have been carried out on the optimization of robot trajectories by minimizing the total energy consumption [83] and torque minimization [14]. Devendra and Manish [84] presented an approach to optimize the torque applied at the joints and they have implemented a genetic algorithm to solve the problem. Hirakawa and Kawamura [85] adopted

a new scheme for achieving the trajectory of a redundant robotic arm that minimizes the total energy consumption of the robot. Saramago and Stefen [86] implemented the trajectory planning of a robotic arm to minimize energy consumption and travel time. Zhang et al. [87] implemented a unified quadratic programming-based dynamic system for minimizing the applied torque of redundant robots.

2.5. Obstacle avoidance techniques

Collision avoidance is an important criterion to be considered in IK and motion planning of robots. Different Collision avoidance techniques have been adopted for redundant robots. The most popular method proposed for obstacle avoidance is an artificial potential field approach [88]. In this approach, an artificial potential field was developed and geometric modelling of obstacles was done using analytical primitives. This potential field forces the manipulator towards the target location while moving the links of the robot away from the obstacles. The task of collision avoidance is considered as high-level planning which is executed as an effective component of low-level real-time control. This approach has been used for robot manipulators and mobile robots. The limitation with the potential field approach is that the solution traps in the local minima. The Pseudo inverse of Jacobian and its variants such as extended Jacobian [67] and task priority approaches [89] have been widely used for local control of redundant robots avoiding obstacles. The main limitation of the Jacobian-based methods is that the kinematic and algorithmic singularities are involved in IK computation.

Chirikjian and Burdick [90] implemented a geometric approach to manipulate the robot in tunnels based on discrete modal technique. Differential geometry methods are used to formulate the equations of manipulator that locates inside the tunnels without colliding the obstacles. The limitation of this approach is defining the modal functions, which require several sets of modes to span the entire robot workspace, and proper mode switching mechanisms need to be performed. Choset and Henning [91] proposed a road map based generalized Voronoi graph for motion planning of a serpentine robot. This approach offers an advantage since it uses the follow the leader approach to represent the backbone curves through the environment. This approach reduces the computational burden associated with highly redundant manipulators. Menon et al. [51] presented an optimization-based approach for motion planning of hyper-redundant manipulators avoiding obstacles. In this approach, the algorithm is purely geometric and the obstacles were defined by smooth and differentiable functions that are suitable for a gradient-based optimized algorithm.

Another class of obstacle avoidance methods was performed by implementing polyhedral collision detection techniques. These techniques works based on computational geometry algorithms. The polyhedral interference-based approaches have been used for collision detection and computation of the distance between two objects [92]. Hubbard [93] implemented a method for collision detection by modelling polyhedral objects with bounding spheres, which are hierarchically arranged for the representation of the robot model. In this approach, the condition for collision detection was performed by checking the distance between the centers of two spheres is smaller than the sum of their radii. The limitation of this approach is that the accuracy of collision detection can be improved by approximating the geometric model with more number of spheres, which is computationally expensive. The bounding box approach has been used extensively for collision detection of robots. Van Henten et al. [94] implemented a bounding box approach for representing the links of the robot, and later the bounding boxes were transformed into hierarchical bounding spheres with different levels of refinement. This technique is computationally fast but accuracy in collision interference is low. The collision avoidance techniques presented in the literature are adopted for different types of obstacles. The limitation with these approaches is, obstacle modelling and collision interference techniques are quite complex.

2.6. Multi-modal optimization

The modularity of robots has been practiced in the design of robot mechanisms for flexibility and ease in maintenance. A modular, reconfigurable robot consists of a set of standardized modules that can be arranged to different structures and DOFs for diverse task requirements [95]. The multi-modal nature of the IK problem offers multiple solutions. Kalra et al. [96] solved the multi-modal IK problem of an industrial robot by using a real-coded genetic algorithm, which is a fitness sharing niching method. The task of redundancy resolution in this approach is performed by selecting the joint configuration that is closest to the current robot configuration in the joint space. The limitation of this approach is to set more unknown parameters. These parameters rely significantly on the nature of search space, which makes the approach configuration dependent. Tabandeh et al. [97] proposed an adaptive niching technique to solve the IK of a serial robotic manipulator with an application to modular robots, which was able to determine multiple IK solutions. This approach combines the real coding and clustering process, this improves the accuracy of the end-effector for random task points. Unlike other niching algorithms, this approach needs a few parameters to be set. This feature permits the algorithm to be applied for computing the IK solution of any robot

configuration. From the literature, it is observed that the use of the niching based approach for computing the multiple IK solutions of industrial and redundant robots. Though the method is efficient it is computationally expensive. Espinoza et al. [98] proposed an inverse kinematic solution of a 10 DOF modular hyper-redundant robots, multiple solutions of the robots are evaluated using exhaustive and error-optimization approaches. In this approach, a comparison has been made among different global optimization approaches. In this thesis, classical optimization methods have been used for evaluating the global minimum and multiple IK solutions. This significantly reduces the time, and it can also be applied to a wide variety of robot configurations.

2.7. Observations from the literature

Some of the major points observed in the literature

I. Due to increase in the use of redundant manipulators in all the fields of science and engineering, there is a huge research scope in analysis and development of these manipulators in different working environments. Since IK is the basic requirement for the control of robots, a lot of research has been carried out in inverse kinematics and redundancy resolution methods of hyper-redundant manipulators.

II. The pseudo-inverse of Jacobian is extensively used to determine the IK solution of a redundant manipulator. The task of redundancy resolution is implemented by satisfying different secondary requirements using the null space of the Jacobian matrix. The limitation of this technique is sensitivity at singular configurations.

III. Another popular approach used for IK solution is the geometric approach, this is used because of its good geometric intuition and low computational cost. The difficulty with this approach is that a closed-form solution must prevail geometrically at least for a few joints.

IV. In the literature, several collision detections and avoidance techniques have been implemented for collision avoidance of redundant manipulators in different working environments, these approaches are quite complex. Hence, there is a need for simple and effective collision avoidance.

V. Evolutionary approaches have also been used for solving the inverse kinematics problem. These approaches are computationally expensive and lack in precision.

VI. The characteristic feature of the IK problem is multi-modal in nature, hence global optimization techniques need to be implemented to determine multiple solutions.

These multiple solutions are helpful to evaluate the configurations of reconfigurable robots.

VII. Multiple configurations of the robot were determined using the niching based approach, which is a fitness sharing method. These approaches require more algorithmic control parameters to initiate the process.

VIII. In the literature [24,52,71], redundant robots has been employed in orbital servicing, medical and welding applications. The issues related to kinematics, dynamics, and design of these manipulators remained challenging. Hence, there is a need for analyzing IK Solutions and kinematic control of redundant robots with different kinematic structures working in complex environments.

2.8. Objectives of the Thesis

The main focus of the thesis is

I. To develop a computationally efficient method for the IK solution of hyper-redundant robots working in complex environments.

II. To improve the performance of redundant manipulators while reaching the desired task space location by optimizing different performance criterion such as
- Joint distance minimization
- Singularity avoidance
- Joint torque minimization

III. To develop an effective collision avoidance scheme for redundant robots with polygonal and 3D obstacles in the working environments.

IV. To analyze multiple IK solutions of the robot that are suitable for working in diverse task space requirements.

V. To explore the performance of spatial redundant robots that are adaptable for working in realistic environments such as plant layout, pipe-line inspection, work facility layout, and warehouse application, etc.

Research hypothesis

Research hypothesis is to employ classical optimization techniques to solve IK problem of hyper-redundant manipulators in complex working environments.

The charecteristic feature of IK problem is of multi-modal in nature. The task of evaluating multiple IK solutions has been performed using classical optimzation techniques. This approach is computationally efficient when compared with global optimization approaches.

Multi-start approach have been performed to evaluate multiple IK solutions of the spatial robot.

2.9. Overview of the Thesis

This section describes the overview of the thesis. In this work, IK simulation of hyper-redundant robots working in planar and spatial workspace. A classical optimization approach have been considered for the evaluating IK solution. Different secondary performance criterion have been considered for redundancy resolution. The over of work has been represented in block diagram shown in Fig. 2.2. It shows different techniqes adopted for solving IK problem and redundancy resolution of hyper-redundant robots.

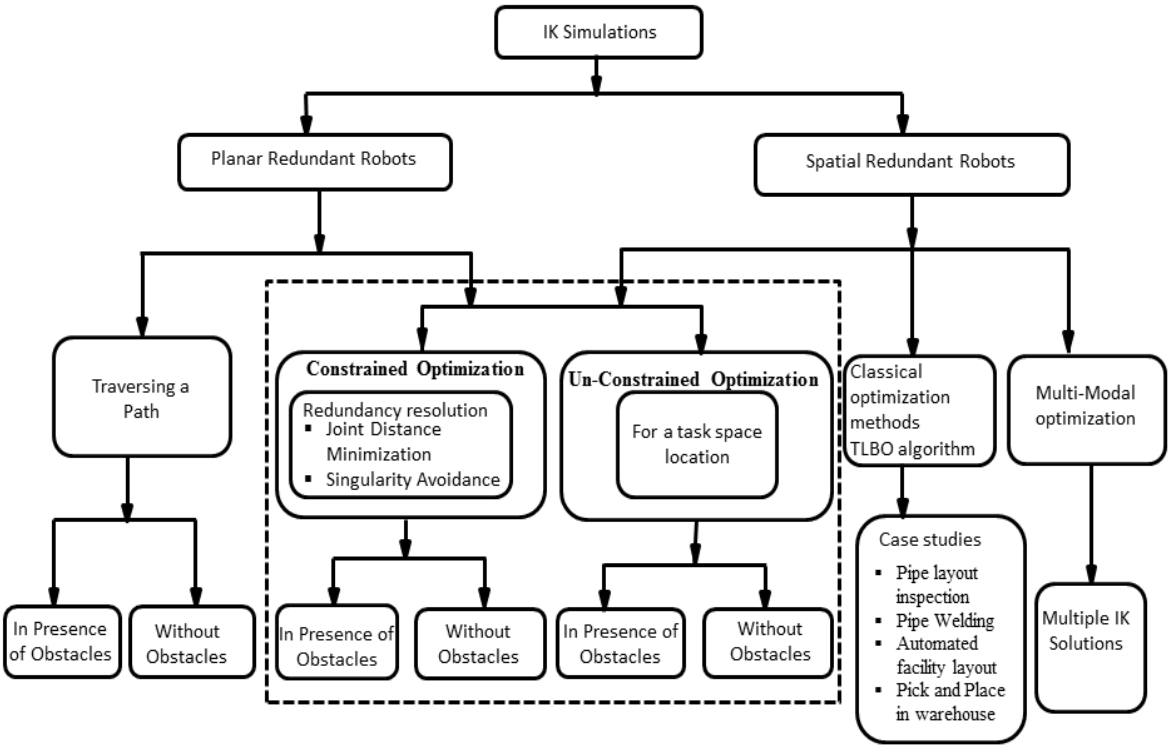


Fig. 2.2. Flow chart representing the overview of the IK simulations.

CHAPTER-III

3. Fundamentals of Robot Manipulation

The description of the manipulator movement involves kinematics, dynamics, trajectory planning, and control of the robot. The forward and inverse kinematics analysis of a robot is a basic requirement for mathematical modelling of kinematic structure and its control. For the redundant manipulators, the IK problem is a more complex and challenging aspect for modelling and control. Several methods have been employed for solving the IK problem, which is explained in chapter 2. In this thesis, IK computation of hyper-redundant manipulators working in planar and 3D workspaces are presented. There is a focus on redundancy resolution techniques that are implemented for achieving secondary criteria such as obstacle and singularity avoidance while satisfying the primary task requirement. An additional performance criterion, joint torque minimization and power consumption have also been chosen in the IK analysis of redundant manipulator. To satisfy all these performance metrics, it is required to analyze the kinematic and dynamic aspects of the robot. This chapter describes the fundamental concepts of robot manipulation.

3.1. Kinematic modelling

A robot consists of a set of rigid bodies called links which are connected by joints. The kinematics of any robot can be described by the position and orientation of the end-effector corresponding to the joint configuration, known as forward kinematics. Therefore, kinematic modelling is commonly referred to as the forward kinematics of the manipulator.

3.1.1. D-H Parameters:

The motion analysis of the robotic manipulator can be performed by considering a standard procedure to model a coordinate system on each link. This was first introduced by Denavit and Hartenberg [23] in 1955 (Denavit) which systematically establishes a relationship of each coordinate frame with respect to the previous frame attached on the link. The coordinate frame assignment for the links of the manipulator using this convention can be performed by the following operations,

- Consider z_{i-1} as the axis for the joint, between link $i-1$ and link i to define the homogenous transformation between frames $i-1$ and i
- Set axis z_i as the axis of rotation or displacement of joint $i+1$

- Find the common normal line a_i between z_{i-1} and z_i and define the origin of frame *i* as the intersection of a_i and z_i, which is o_i
- Axis x_i is defined in the direction of a_i, from joint i to $i+1$
- Axis y_i is defined as the mutual perpendicular to the axis z_i and x_i by following the right-hand rule.

The DH representation of a particular pose describes a displacement for i^{th} joint-link pair in a mechanism by defining four parameters: The link length a_i , link twist α_i joint angle θ_i and link offset d_i.

According to this convention, the transformation of i^{th} frame relative to the $(i-1)^{th}$ frame is expressed by a generalized homogeneous transformation matrix. It is given by

$$T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

The homogenous transformation for a given pose is a function of DH variable i. e. θ for revolute joints and *d* for prismatic joints as the most of the robot joints are either prismatic or revolute. Assignment of coordinate frames and identification of joint-link parameters are shown in Fig. 3.1.

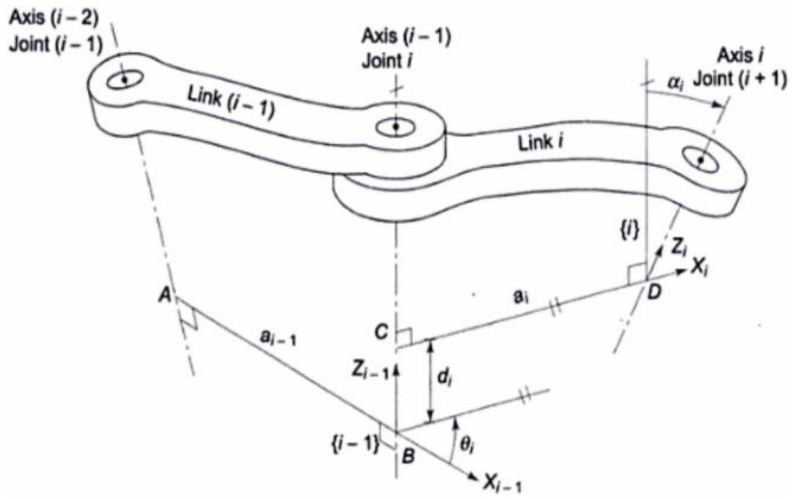


Fig. 3.1. Description of DH parameters for a joint *i* and link *i* [32].

3.1.2. Forward and inverse kinematics of serial manipulators

Forward kinematics can be defined as the mapping from the joint space (θ) to the Cartesian space (x, y, z), mathematically defined as

$$Q \subseteq R^m \rightarrow W \subseteq R^n, \tag{3.2 a}$$

Inverse kinematics is the mapping from the Cartesian space *(x, y, z)* to the joint space (θ).

$$W \subseteq R^n \rightarrow Q \subseteq R^m, \tag{3.2 b}$$

where Q = subset of joint variables as the solution of the inverse kinematics equations, W = subset of Cartesian space, R^m = universal set of joint variables, R^n = universal set of Cartesian space.

3.1.3. Forward Kinematis of serial planar redundant robot

A generalized n linked robot is modelled with link length $l_1, l_2, \ldots l_n$ and the angle between link and *x-axis* is $\theta_1, \theta_2, \ldots \theta_n$. The forward kinematic model is represented by a simple geometric and analytical interpretation of the robot.

End effector position in workspace is given as

$$E_x = l_1 \cos\theta_1 + l_2 \cos\theta_2 + l_3 \cos\theta_3 + \ldots l_n \cos\theta_n \tag{3.3 a}$$

$$E_y = l_1 \sin\theta_1 + l_2 \sin\theta_2 + l_3 \sin\theta_3 + \ldots l_n \sin\theta_n \tag{3.3 b}$$

3.1. 4. Kinematic model of a spatial redundant manipulator

The joints of the spatial redundant robot are modeled with universal joints for better accessibility and control in the workspace. These joints have two orthogonal DOFs that are formed by the pitch axis and yaw axis at all the joints except at the joint1. A generalized spatial hyper-redundant manipulator is shown in Fig. 3.2. The forward kinematic analysis begins with the assignment of coordinate frames at the robot joints, and these frames are used to describe the position and orientation of one frame relative to another frame. The frame assignment of the hyper-redundant manipulator is shown in Fig. 3.2. The Denavit-Hartenberg (D-H) parameters for the assigned frames are given as per the standard convention.

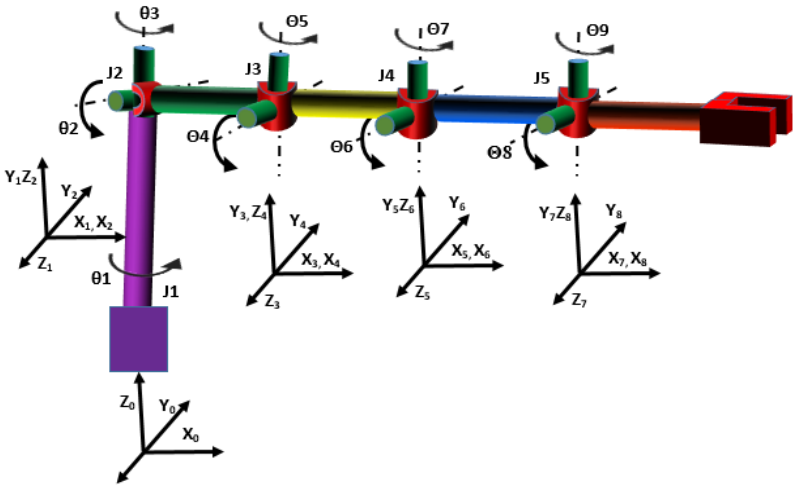


Fig. 3.2. Schematic diagram of spatial hyper-redundant manipulator.

The transformation of the i^{th} frame relative to the $(i\text{-}1)^{th}$ frame is derived from the generalized homogeneous transformation matrix. It is given by Eq. 3.1.

For the robot shown in Fig. 3.4, the position of the end-effector with respect to the base frame is given by

$$\boldsymbol{T}_9^0 = \boldsymbol{T}_1^0\ \boldsymbol{T}_2^1\ \boldsymbol{T}_3^2\ \boldsymbol{T}_4^3\ \boldsymbol{T}_5^4\ \boldsymbol{T}_6^5\ \boldsymbol{T}_7^6\ \boldsymbol{T}_8^7\ \boldsymbol{T}_9^8 \tag{3.4}$$

From the above equation of the homogeneous transformation matrix, the final end-effector position and orientation can be determined. Which is taken as the required inputs to perform the IK analysis.

3.2. Robot positioning

A joint configuration of the robot is to be provided to position the robot end-effector in the task space. The task of computing joint variables was carried out by the inverse kinematics of the robot. This section describes the inverse kinematics methods and challenges involved in IK problem of serial robots.

3.2.1. Analytical method for inverse kinematics

An analytical approach is recommended while attempting inverse kinematics of serial robots. The forward kinematic equations of the serial robot in a planar workspace are often easy to obtain, whereas in complex robots DH parameters might be useful for arriving forward kinematic equations. The Kinematic equations involve the trigonometric function, which makes the problem complex. Moreover, multiple solutions of trigonometric functions need to

be considered. An example of a 3DOF planar robot illustrates the IK Model using an analytical approach.

3.2.2. A 3 DOF Manipulator with revolute joints

For the 3R manipulator shown in Fig. 3.3, the analytical approach for the IK problem is determined by understanding its forward kinematic model. As the joints are revolute and it is a planar robot, only one joint variable is involved, the forward kinematic model of this can be easily determined. By analyzing its geometry, the forward kinematic relation is given by i.e the position and orientation of the end-effector (x, y).

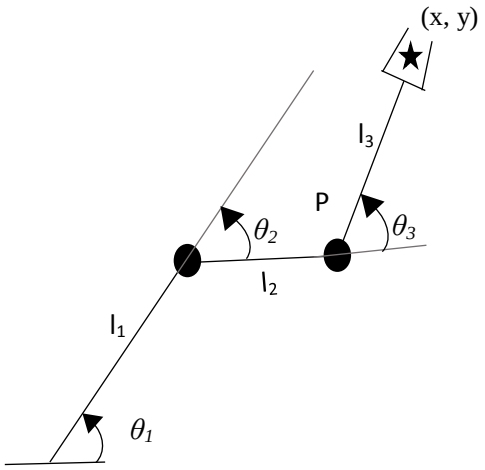


Fig. 3.3. A 3 DOF Serial robot arm

From the geometry of the robot, forward kinematic equations are given as

$$x = l_1 cos(\theta_1) + l_2 cos(\theta_1 + \theta_2) + l_3 cos(\theta_1 + \theta_2 + \theta_3) \quad (3.5\ a)$$

$$y = l_1 sin(\theta_1) + l_2 sin(\theta_1 + \theta_2) + l_3 sin(\theta_1 + \theta_2 + \theta_3) \quad (3.5\ b)$$

$$\phi = \theta_1 + \theta_2 + \theta_3 \quad (3.5\ c)$$

The location of point *P* can be computed as

$$P_x = x - l_3\ cos(\phi) = l_1 cos(\theta_1) + l_2 cos(\theta_1 + \theta_2) \quad (3.6\ a)$$

$$P_y = y - l_3\ sin(\phi) = l_1 sin(\theta_1) + l_2 sin(\theta_1 + \theta_2) \quad (3.6\ b)$$

Squaring the two sides of Eq (3.6 a-b) and adding, gives

$$cos(\theta_2) = \frac{P_X{}^2 + P_Y{}^2 - l_1{}^2 - l_2{}^2}{2 l_1 l_2} \quad (3.7\ a)$$

The existence of solution for Eq. (3.7a) imposes the condition, $-1 \leq \mathrm{Cos}(\theta_2) \leq 1$, beyond this range the point is outside the reachable workspace of the arm, then set

$$sin(\theta_2) = \pm\sqrt{1-cos(\theta_2)^2} \tag{3.7 b}$$

where the positive sign is with reference to elbow up posture and the negative sign is the elbow down posture. Hence, the angle θ_2 is computed as

$$\theta_2 = a\tan 2\ (sin(\theta_2), cos(\theta_2)) \tag{3.8}$$

where *atan2* is the function that computes the value of $\tan^{-1}$ in the appropriate quadrant.

θ_1 can be determined by expanding $cos(\theta_1+\theta_2)$ and $sin(\theta_1+\theta_2)$ of Eq.(3.5) and rearranging them

$$P_x = (l_1 + l_2\cos(\theta_2))\cos(\theta_1) - l_2\sin(\theta_1)\sin(\theta_2) \tag{3.9 a}$$

$$P_y = (l_1 + l_2 cos(\theta_2))sin(\theta_1) + l_2 cos(\theta_1)sin(\theta_2) \tag{3.9 b}$$

To evaluate θ_1 Eq. (3.9a) is multiplied by $l_2\sin(\theta_2)$ and Eq. (3.9b) by $l_1 + l_2\cos(\theta_2)$, followed by the subtraction of former from latter, this gives the value

$$sin(\theta_1) = \frac{(l_1+l_2cos\ (\theta_2))P_y - l_2 sin(\theta_2)P_x}{l_1^2+l_2^2+2l_1l_2cos(\theta_2)} \tag{3.10 a}$$

Similarly

$$cos(\theta_1) = \frac{(l_1 + l_2cos(\theta_2))P_x - l_2 sin(\theta_2)P_y}{l_1^2 + l_2^2 + 2l_1l_2cos(\theta_2)} \tag{3.10 b}$$

The solution θ_1 can be obtained by

$$\theta_1 = atan2\ (sin(\theta_1), cos(\theta_1)) \tag{3.10 c}$$

Finally, the angle is found from the expression

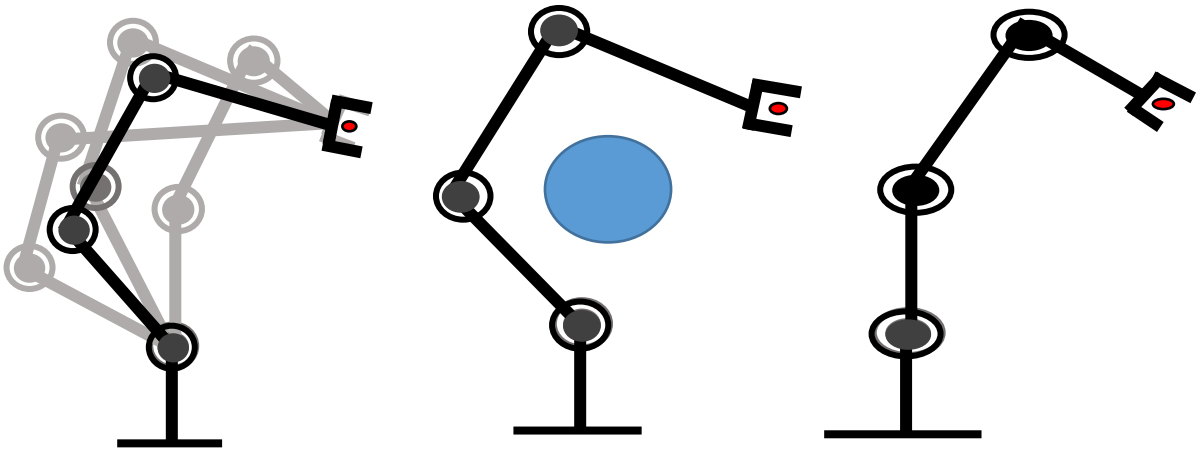
$$\theta_3 = \varphi - \theta_1 - \theta_2 \tag{3.11}$$

These equations have two solutions θ_1 as they are involved with trigonometric functions, therefore two sets of solutions $\theta_1, \theta_2, \theta_3$ can be achieved, the two solutions corresponding to different joint configurations give the same end-effector position. It is implied from the above derivation the complexity of the analytical solution increases with the complexity of robot geometry and the number of DOFS i.e for redundant robots. As this approach is geometry dependent, the IK solution for a general manipulator with arbitrary geometry does not exist. Due to the limitations in analytical approaches, some of the numerical approaches have been used for the IK solution of different robot configurations with redundant DOF. A detailed discussion of IK solution methods was presented in the literature review. The

redundancy of the robot, despite its advantages with respect to additional capabilities, computational complexity involves solving IK problems using analytical approaches is high. In such cases, numerical methods are often employed for the IK solution.

3.3. Manipulation of redundant robots

Redundant manipulators have more DOFs, than the required to reach a task space location. In the case of 2D planar manipulating systems with position and orientation as required targets, redundant manipulator possess more than three DOF. The additional DOFs allow multiple solutions in joint space for a given task space position and orientation. Availability of multiple solutions leads the controller of the manipulator to choose the best solution from the feasible solutions that satisfy secondary criteria. Fig. 3.4(a) depicts multiple solutions of a redundant system and selecting the solution based on the performance criteria. Fig. 3.4(b) shows joint configuration while avoiding obstacles. While Fig 3.4(c) shows configuration that minimizes joint rotation with respect to home position.



a b c

Fig. 3.4. IK solution of redundant robot (a) Multiple solutions (b) Avoiding obstacle (c) Minimizing joint distance.

3.3.1. Robot differential kinematics

For redundant manipulators, kinematic equations are highly non-linear and closed-form solutions are not always possible. Thus, numerical approximations are used to solve the IK problem. Differential velocity kinematics is used while predicting these approximations. In differential kinematics, the Jacobian matrix is used which maps from joint space velocity to Cartesian space velocity. The forward velocity relation is given by

$$\boldsymbol{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} = \dot{\boldsymbol{X}} \tag{3.12}$$

where $\dot{\theta}$ represents joint rates and $\dot{X}$ is the end-effector velocities. The Jacobian matrix of 6 DOF manipulator is represented by 6 x *n* matrix, n being the number of manipulator's joints. The first three rows of the Jacobian matrix indicate joint velocities corresponding to linear task space velocities and the last three rows relate joint velocities corresponding to angular joint space velocities.

$$\mathbf{J}(\theta)=\begin{bmatrix} \mathbf{J}_{v_1} & \mathbf{J}_{v_2} & .... & \mathbf{J}_{\omega_n} \\ \mathbf{J}_{\omega_1} & \mathbf{J}_{\omega_2} & .... & \mathbf{J}_{\omega_n} \end{bmatrix} \tag{3.13}$$

The i^{th} column of the Jacobian matrix corresponds to i^{th} joint velocities by combining both linear and angular velocities, a column of the Jacobian matrix for a revolute joint is given by

$$\begin{bmatrix} J_{v_i} \\ J_{\omega_i} \end{bmatrix}=\begin{bmatrix} z_i \times o_i \\ z_i \end{bmatrix} \tag{3.14}$$

where the joint linear velocity is the cross product of the joint's motion axis vector z_i with the distance between the i^{th} joint origin and the end-effector, o_i. In the case of a prismatic joint, an i^{th} joint column of Jacobian is given by

$$\begin{bmatrix} J_{v_i} \\ J_{\omega_i} \end{bmatrix}=\begin{bmatrix} z_i \\ 0 \end{bmatrix} \tag{3.15}$$

3.3.2. Jacobian inverse methods

The Jacobian matrix inverse methods have been used extensively in IK computations and it has been performed in several ways. Based on kinematic structure and configuration, if the kinematic configuration produces full rank i.e. square Jacobian matrix, matrix inverse can be computed using standard methods.

By considering equation 3.10 with joint space dimension is equal to task space dimension, the joint velocities can be obtained by inversion of Jacobian matrix [100]

$$\dot{\theta}=J^{-1}(\theta)\,\dot{X} \tag{3.16}$$

If the initial posture of the robot is known, the joint positions can be computed by integrating velocities over time

$$\boldsymbol{\theta}(t) = \int_0^t \dot{\boldsymbol{\theta}} dt + \boldsymbol{\theta}(0) \tag{3.17}$$

$$\boldsymbol{\theta}(t_{k+1}) = \boldsymbol{\theta}(t_k) + \dot{\boldsymbol{\theta}}(t_k)\Delta t \tag{3.18}$$

If a kinematic structure is redundant when the Jacobian matrix is a rank deficit or non-square, an alternative formulations for Jacobian inverse were used. A possible solution method is to formulate the problem as constrained optimization problem.

Once the end-effector velocity and Jacobian are known it is desired to find the solution theta that satisfy the equation 3.10 and minimize the quadratic cost function of joint velocities

$$g(\dot{\boldsymbol{\theta}}) = \frac{1}{2}\dot{\boldsymbol{\theta}}^T \boldsymbol{W} \dot{\boldsymbol{\theta}} \tag{3.19}$$

where W is a suitable symmetric positive definite weighting matrix. This problem can be solved by using method of Lagrange multipliers

3.3.3. Pseudo-inverse of Jacobian

The pseudo-inverse matrix or Moore-Penrose [100] is the most widely used in IK computation of redundant robots i.e. dealing with the rank deficit and non-square Jacobian matrices. The pseudo-inverse is given by

$$\boldsymbol{J}^+ = \boldsymbol{J}^T (\boldsymbol{J}\boldsymbol{J}^T)^{-1} \tag{3.20}$$

This computes the IK problem by minimizing the magnitude of joint angle variation required to achieve an end-effector posture in the least square sense. This property is beneficial since the minimization of joint velocity tends to reduce kinematic motion requirements. The computation cost involved in determining pseudo inverse is high and it also increases with an increase in the number of DOF. The pseudo-inverse approach performs well in general, but it suffers when the robot configuration is close to singularities, where the Jacobian loses its rank.

3.3.4. Jacobian transpose

This method attempts to move the joint values in the direction of the square of the norm of error vector which reduces in the highest rate [99]. This Jacobian transpose is the least computationally expensive. The joint motion is updated by a small displacement in the joint vector which is given by

$$\Delta\boldsymbol{\theta} = \boldsymbol{J_e(\theta)^T} \,.\, K \,.\, \boldsymbol{e(\theta)} \tag{3.21}$$

where $e(\theta)$ is the positional error function vector which represents the required kinematic constraint, $\Delta\theta$ is the incremental joint configuration vector and J_e is the error Jacobian matrix, K is a stiffness constant that pulls the end-effector towards the target location.

This approach has a limitation, of slow convergence after a few iterations because Jacobian transpose inherently takes large steps of joint increments. IK solution does not minimize the norm of joint angle.

3.4. Position based inverse kinematics

For a long period, there was a conception that closed-form IK solutions for redundant robots are highly complicated and difficult. Thus, the IK problems have been approached by linearizing the configuration space to velocity space. That is, the problem is mapped into velocity space of end effector by using linearized derivatives of the joint space, which is represented by the Jacobian matrix. Numerical approaches used for the velocity-based IK problem suffers from a sort of limitations such as higher computational time and joint velocities at singular configurations. Position based inverse kinematics techniques are faster, reliable, and more accurate. IK problem has been solved using analytical, geometric, and evolutionary-based approaches. The IK problem in the position-based approach can be modelled as an optimization problem. This gives the Joint configuration that minimizes the Euclidian distance of end-effector location and desired location in the workspace.

3.4.1. Desired position and error representation

The Inverse kinematics solution of a redundant manipulator is computed by formulating it as an optimization problem with the objective of minimizing positional error i.e. distance between the current position of end-effector and desired position in the task space. The objective function is the reachability of the manipulator in the task space. This is measured as the total Euclidean distance *(D)*, i.e., the distance between the current position of the end-effector ***(E)*** and the task space location ***(P)***, as shown in Fig. 3.5, and the vectorial representation of distance is given by.

$$D = \| \boldsymbol{E} - \boldsymbol{P} \| \tag{3.22}$$

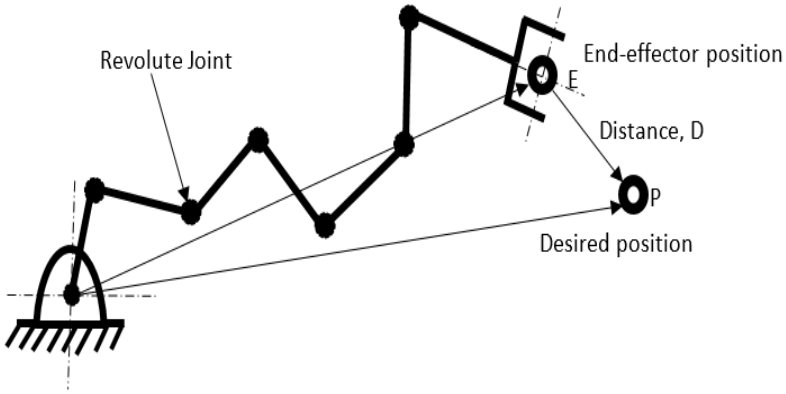


Fig. 3.5. Schematic diagram of serial hyper-redundant manipulator representing the objective.

For a given joint configuration ***(θ)***, the end effector position ***E*** *(E_x, E_y, E_z)* can be calculated using the forward kinematic relationship. The current end-effector position is a function of joint angle ***(θ)***, hence the distance can be represented as

$$D = || \boldsymbol{E}(\boldsymbol{\theta}) - \boldsymbol{P} || \tag{3.23}$$

where, $\boldsymbol{\theta}$=[$\theta_1, \theta_2, \theta_3, \theta_4, \ldots\ldots\ldots\ldots, \theta_n$]

To reach the specified task space location, ***P*** *(P_x, P_y, P_z)* from the end-effector position ***(E)***, the distance between ***E*** and ***P*** should be zero. Hence, the problem is posed as minimization of a square of total Euclidean distance, which is given as

$$\textit{Minimize: } f = D^2 = \left((E_x - P_x)^2 + (E_y - P_y)^2 + (E_z - P_z)^2 \right) \tag{3.24}$$

In this thesis, IK computation has been performed for both planar and spatial redundant manipulators. The IK computation of the planar robot does not include the Z-coordinate term. Thus the equation is simplified as

$$\textit{Minimize: } f = D^2 = \left((E_x - P_x)^2 + (E_y - P_y)^2 \right) \tag{3.23}$$

The minimization of the objective function given in Eq. (3.20-3.21) results in a joint configuration that reaches the manipulator to the desired position and orientation in task space.

3. 5. Robot dynamics

During the performance of a task, the manipulator needs to carry payloads and undergo different accelerations, sometimes moves at a constant speed and declaration. The variation

of position and orientation with respect to time is termed as the dynamic behaviour of the robot. Torques need to apply at the joints to balance the internal and external forces of the robot. The internal forces are due to the inertia of links, Coriolis forces, and frictional forces. External forces are due to payload and forces caused by the environment such as gravity. This section describes the mathematical model for the dynamic behaviour of the robot.

The dynamic equations are often referred to as a set of the equation of motion that represents the dynamic response of the manipulator to input actuator torques. The dynamic model of the manipulator is convenient for the computation of torques and forces required for the execution of the end-effector task. The complex dynamic system of serial link manipulator can be modelled systematically by using physical laws of Lagrangian mechanics or Newtonian mechanics.

Approaches such as Lagrangian-Euler formulation, energy-based and Newton-Euler approach, based on force-balance can be applied to the manipulator equations of motion. The Newton-Euler and Lagrangian-Euler formulations provide computationally expensive closed-form solutions. To improve the computational speed recursive methods and approximate methods have been developed. This section provides the basic formulation of manipulator dynamics using the Lagrangian-Euler approach [1, 100].

3.5.1. Lagrangian mechanics

A scalar function called Lagrangian L is defined as the difference between the total kinetic energy K and potential energy P of a mechanical system

$$L=K-P \tag{3.25}$$

The Lagrange-Euler formulation is based on a set of generalized coordinates to describe the variables of the system. In generalized coordinate, displacement q is used as a joint variable, which represents a linear displacement d for a prismatic joint and angular displacement θ for a rotary joint. Similarly, q describes linear velocity and angular velocity for prismatic and rotary joints. A generalized torque τ is required at the joint to produce desired dynamics. Force f represents for a prismatic joint and torque τ for a revolute joint. For the robot in Fig. 3.6, ${}^{i}r_i$ be the point fixed and at rest on a link i

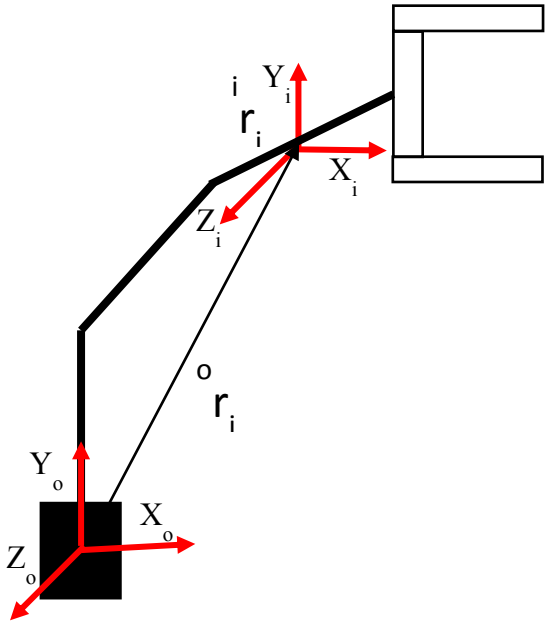


Fig. 3.6. A 3 DOF manipulator showing base coordinate frame and homogenous coordinate frame.

The homogenous coordinates with respect to the coordinate frame of i^{th} link are given as

$$^{i}\boldsymbol{r}_i = (x_i, y_i, z_i, 1)^T \tag{3.26}$$

The position of point *r* with respect to the base coordinate frame is

$$^{0}\boldsymbol{r}_i = {}^{0}\boldsymbol{T}_i \, {}^{i}\boldsymbol{r}_i = ({}^{0}\boldsymbol{T}_1 \, {}^{1}\boldsymbol{T}_2 \ldots {}^{i-1}\boldsymbol{T}_i) \, {}^{i}\boldsymbol{r}_i \tag{3.27}$$

The velocity of point *r* with respect to the base frame is obtained as

$$^{0}\boldsymbol{v}_i = \boldsymbol{v}_i = {}^{0}\dot{\boldsymbol{r}}_i = \left[\sum_{j=1}^{i} \frac{\partial \, {}^{0}\boldsymbol{T}_i}{\partial q_j} \dot{q}_j \right] {}^{i}\boldsymbol{r}_i \tag{3.28}$$

The transformation matrix $^{o}T_i$ involve complex trigonometric terms and its partial derivatives with respect to q_j, in Eq. 3.25 involves complex computation. The following steps simplify the computation of the partial derivative of the homogenous transformation matrix.

Consider the transformation matrix for link *j* for a rotary joint

$$^{j-1}\boldsymbol{T}_j = \begin{bmatrix} C\theta_j & -S\theta_j C\alpha_j & S\theta_j S\alpha_j & a_j C\theta_j \\ S\theta_j & C\theta C\alpha_j & -C\theta_j S\alpha_j & a_j S\theta_j \\ 0 & s\alpha_j & C\alpha_j & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.29}$$

Partial derivative with respect to θ_j

$$\frac{\partial {}^0\boldsymbol{T}_i}{\partial q_j} = \begin{bmatrix} -S\theta_j & -C\theta_j C\alpha_j & C\theta_j S\alpha_j & a_j C\theta_j \\ c\theta_j & -S\theta C\alpha_j & S\theta_j S\alpha_j & a_j S\theta_j \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.30}$$

The partial derivative of ${}^{j-1}\boldsymbol{T}_j$ with respect to θ_j can be obtained without differentiating the terms, the same result can be attained using matrix operations, by pre multiplying ${}^{j-1}\boldsymbol{T}_j$ with $\boldsymbol{Q}_j$

$$\boldsymbol{Q_j} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \textit{(for revolute joint)} \quad \boldsymbol{Q}_j = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \textit{(for prismatic joint)} \tag{3.31}$$

$$\frac{\partial {}^0\boldsymbol{T}_i}{\partial q_j} = \boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_j \tag{3.32}$$

Since $\frac{\partial {}^0\boldsymbol{T}_i}{\partial q_j} = {}^0\boldsymbol{T}_1 \, {}^1\boldsymbol{T}_2 .... {}^{j-2}\boldsymbol{T}_{j-1} \frac{\partial {}^{j-1}\boldsymbol{T}_j}{\partial q_j} {}^j\boldsymbol{T}_{j+1} .... {}^{i-1}\boldsymbol{T}_i$

Using Eq. 3.29 the partial derivative can be simplified as

$$\frac{\partial {}^0\boldsymbol{T}_i}{\partial q_j} = {}^0\boldsymbol{T}_1 \, {}^1\boldsymbol{T}_2 .... {}^{j-2}\boldsymbol{T}_{j-1} (\boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_j) \, {}^j\boldsymbol{T}_{j+1} .... {}^{i-1}\boldsymbol{T}_i = {}^0\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_j \tag{3.33}$$

The result is valid only for $j \le i$, hence for *i=1,2,……n*

$$\frac{\partial {}^0\boldsymbol{T}_i}{\partial q_j} = \begin{cases} {}^0\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_j & \textit{for } j \le i \\ 0 & \textit{for } j > i \end{cases} \tag{3.34}$$

The link velocity is thus simplified as

$$\boldsymbol{v}_i = \sum_{j=1}^{i} {}^0\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_i \dot{\boldsymbol{q}}_j \, {}^i\boldsymbol{r}_i \tag{3.35}$$

Kinetic energy

The kinetic energy of the differential mass dm_1 on link i, located at ${}^0\boldsymbol{r}_i$ and moving with the velocity ${}^o\boldsymbol{v}_i$ with respect to the base frame is

$$dk_i = \frac{1}{2} dm_i (\boldsymbol{v}_i)^2 \tag{3.36}$$

The trace operator is used to obtain $(v_i)^2$ as

$$\boldsymbol{v}_i^2 = Tr({}^o\dot{\boldsymbol{r}}_i \, {}^o\dot{\boldsymbol{r}}_i^T) = Tr(\boldsymbol{v}_i \boldsymbol{v}_i^T) \tag{3.37}$$

Substituting v_i from Eq .3.32 in Eq.3.34, the kinetic energy of differential mass is obtained as

$$dk_i = \frac{1}{2} Tr\left[\left(\sum_{j=1}^{i} {}^0\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_i \, \dot{\boldsymbol{q}}_j \, {}^i\boldsymbol{r}_i \right) \left(\sum_{k=1}^{i} {}^0\boldsymbol{T}_{k-1} \boldsymbol{Q}_k \, {}^{k-1}\boldsymbol{T}_i \dot{\boldsymbol{q}}_k \, {}^i\boldsymbol{r}_i \right)^T \right] dm_i \tag{3.38}$$

The total kinetic energy of link i is then

$$K_i = \int dk_i$$

$$K_i = \frac{1}{2} Tr\left[\sum_{j=1}^{i} \sum_{k=1}^{i} {}^0\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_i \int {}^i\boldsymbol{r}_i \, {}^i\boldsymbol{r}_i^T dm_i \left({}^0\boldsymbol{T}_{k-1} \boldsymbol{Q}_k \, {}^{k-1}\boldsymbol{T}_i \right)^T \dot{\boldsymbol{q}}_j \dot{\boldsymbol{q}}_k \right] \tag{3.39}$$

The integral term $\int {}^i\boldsymbol{r}_i \, {}^i\boldsymbol{r}_i^T dm_i$ is the moment of inertia tensor $\boldsymbol{I}_i$, which is given as

$$\boldsymbol{I}_i = \begin{bmatrix} \int x^2{}_i dm_i & \int x_i y_i dm_i & \int x_i z_i dm_i & \int x_i dm_i \\ \int x_i y_i dm_i & \int y^2{}_i dm_i & \int y_i z_i dm_i & \int y_i dm_i \\ \int x_i z_i dm_i & \int y_i z_i dm_i & \int z^2{}_i dm_i & \int z_i dm_i \\ \int x_i dm_i & \int y_i dm_i & \int z_i dm_i & \int dm_i \end{bmatrix} \tag{3.40}$$

Therefore K_i is, $K_i = \frac{1}{2} Tr\left[\sum_{j=1}^{i} \sum_{k=1}^{i} ({}^0\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_i) \boldsymbol{I}_i \left({}^0\boldsymbol{T}_{k-1} \boldsymbol{Q}_k \, {}^{k-1}\boldsymbol{T}_i \right)^T \dot{\boldsymbol{q}}_j \dot{\boldsymbol{q}}_k \right]$ (3.41)

Thus for the n-DOF manipulator, the total kinetic energy of the manipulator is

$$K = \sum_{i=1}^{n} K_i = \frac{1}{2} \sum_{i=1}^{n} Tr\left[\sum_{j=1}^{i} \sum_{k=1}^{i} ({}^0\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \, {}^{j-1}\boldsymbol{T}_i) \boldsymbol{I}_i \left({}^0\boldsymbol{T}_{k-1} \boldsymbol{Q}_k \, {}^{k-1}\boldsymbol{T}_i \right)^T \dot{\boldsymbol{q}}_j \dot{\boldsymbol{q}}_k \right] \tag{3.42}$$

Exchanging the trace and sum operations

$$K = \frac{1}{2}\sum\sum\sum T_r\left[\left({}^0\boldsymbol{T}_{j-1}\boldsymbol{Q}_j\,{}^{j-1}\boldsymbol{T}\right)\boldsymbol{I}_i\left({}^0\boldsymbol{T}_{k-1}\boldsymbol{Q}_k\,{}^{k-1}\boldsymbol{T}_i\right)^T\right]\dot{\boldsymbol{q}}_j\dot{\boldsymbol{q}}_k \tag{3.43}$$

The kinetic energy *k* of the manipulator is a scalar and is a function of joint position and velocity.

Potential Energy

The potential energy P_i of link *i* in a gravity field *g* is

$$P_i = -m_i g({}^0\bar{\boldsymbol{r}}_i) = -m_i g\,{}^0\boldsymbol{T}_i\,{}^i\bar{\boldsymbol{r}}_i \tag{3.44}$$

The total potential energy of the manipulator is the sum of the potential energy of the links, that is

$$P = \sum_{i=1}^{n} P_i = -\sum_{i=1}^{n} m_i g\,{}^0\boldsymbol{T}_i\,{}^i\bar{\boldsymbol{r}}_i \tag{3.45}$$

Equation of motion

The Lagrangian is given by $L = K - P$, from Eq. (3.40) and (3.42)

$$L = \sum_{i=1}^{n}\sum_{j=1}^{i}\sum_{k=1}^{i} T_r\left[\left({}^0\boldsymbol{T}_{j-1}\boldsymbol{Q}_j\,{}^{j-1}\boldsymbol{T}\right)\boldsymbol{I}_i\left({}^0\boldsymbol{T}_{k-1}\boldsymbol{Q}_k\,{}^{k-1}\boldsymbol{T}_i\right)^T\right]\dot{\boldsymbol{q}}_j\dot{\boldsymbol{q}}_k + \sum_{i=1}^{n} m_i g\,{}^0\boldsymbol{T}_i\,{}^i\bar{\boldsymbol{r}}_i \tag{3.46}$$

According to the Lagrange-Euler dynamic formulation, the generalized torque τ_i of the actuator at joint i, to drive link *i* of the manipulator, is given by

$$\tau_i = \frac{d}{dt}\left[\frac{\partial L}{\partial \dot{q}_i}\right] - \left[\frac{\partial L}{\partial q_i}\right] \tag{3.47}$$

$$\tau_i = \sum \boldsymbol{M_{ij}(q)}\ddot{\boldsymbol{q}}_j + \sum\sum h_{ijk}\,\dot{\boldsymbol{q}}_j\,\dot{\boldsymbol{q}}_k + G_i \quad \textbf{for i=1,2,.....n} \tag{3.48}$$

where

$$M_{ij} = \sum_{p=\textbf{max (i,j)}}^{n} T_r\left[\boldsymbol{d}_{pj}\boldsymbol{I}_P\boldsymbol{d}_{pi}{}^T\right] \tag{3.49}$$

$$h_{ijk} = \sum_{p=\,\boldsymbol{max\,(ijk)}}^{n} T_r \left[\frac{\partial \left(\boldsymbol{d}_{pk} \right)}{\partial q_p} \boldsymbol{I}_p \boldsymbol{d}_{pi}^{\,T} \right] \tag{3.50}$$

$$G_i = -\sum m_p \boldsymbol{g} \boldsymbol{d}_{pi}\, {}^{p}\bar{\boldsymbol{r}}_p \tag{3.51}$$

$$d_{ij} = \begin{cases} {}^{0}\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \; {}^{j-1}\boldsymbol{T}_i \; for \; j \le i \\ 0 \qquad\qquad for \; j > i \end{cases} \tag{3.52}$$

$$\frac{\partial \boldsymbol{d}_{ij}}{\partial q_k} = \begin{cases} {}^{0}\boldsymbol{T}_{i-1} \boldsymbol{Q}_J \; {}^{j-1}\boldsymbol{T}_{k-1} \boldsymbol{Q}_k \; {}^{k-1}\boldsymbol{T}_i \; for \; i \ge k \ge j \\ {}^{0}\boldsymbol{T}_{k-1} \boldsymbol{Q}_k \; {}^{k-1}\boldsymbol{T}_{j-1} \boldsymbol{Q}_j \; {}^{i-1}\boldsymbol{T}_i \; for \; i \ge j \ge k \\ 0 \qquad\qquad\qquad\qquad for \; i < j \; or \; i < k \end{cases} \tag{3.53}$$

Eq. (3.45) is the dynamic model of the manipulator and gives a set of *n* nonlinear, coupled, second-order ordinary differential equations for *n*-links of the *n*-DOF manipulator. In this work, optimum joint trajectories were computed with an objective of minimization of power consumption. For which, computation of torques is required, which are evaluated from the equations of motion.

3.6. Trajectory planning

During the motion of the robot, it needs to be provided with initial and final positions, intermediate locations and travelling time along a defined path. Specification of robot position as a function of time is called trajectory planning. In general, trajectories are completely specified for a task such as tracing a path of robot motion during its operation or a task can be moving from one position to another. Trajectory planning can be implemented in joint space and also in Cartesian space. A geometric path cannot be completely specified. A reduced number of path parameters are specified such as initial and final position, intermediate positions, constraints on the maximum accelerations and velocities. Based on the above inputs trajectory planning generates variables of motion in a sequence of time which describes the end-effector position and orientation by respecting the task constraints. As the control action of the manipulator is performed in joint space, a suitable IK algorithm is to be implemented to generate the time sequence of joint variables corresponding to the task space location of the end-effector.

This section describes joint space trajectory planning techniques [9] for different polynomial functions and problem formulation for optimum trajectory planning. Each point in the task space can be specified by the desired position and orientation of the end-effector. These points are transformed into a set of desired joint angles by the use of inverse kinematics. Then a smooth polynomial function is interpolated for each of these joints while satisfying different kinematic constraints. Kinematic constraints are position, velocity, and acceleration conditions at the start, via, and endpoints in the joint space.

3.6.1. Cubic polynomial function

Inverse kinematics allows determining a set of joint angles corresponding to the target position and orientation. A polynomial function for each joint value at time t_i is the initial position of the joint and the value at t_f is the goal position of that joint, a smooth polynomial function might be used to interpolate the joint motion between starting and end position.

For a smooth polynomial function, a minimum of four constraints on $\theta(t)$ is required. Two constraints on the function values are chosen as initial and final position values

$$\theta(0) = \theta_i \tag{3.54 a}$$

$$\theta\left(t_f\right) = \theta_f \tag{3.54 b}$$

Additional constraints are taken to satisfy the function continuous in velocity, in general, initial and final velocities are taken as zero

$$\dot{\theta}\,(0) = 0 \tag{3.55 a}$$

$$\dot{\theta}\,(\mathrm{t}_f) = 0 \tag{3.55 b}$$

These four constraints can be satisfied with the polynomial function of the third degree, a cubic polynomial function has the form

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{3.56}$$

The joint velocities and accelerations are given as

$$\dot{\theta}(t) = a_1 t + 2a_2 t + 3a_3 t^2 \tag{3.57 a}$$

$$\ddot{\theta}(t) = 2a_2 + 6a_3 t \tag{3.57 b}$$

Combining the Eq 3.56 & 3.57 with the four desired constraints gives four equations in four unknowns

$$\theta_0 = a_0 \tag{3.58 a}$$

$$\theta_f = a_0 + a_1 t_f + a_2 {t_f}^2 + a_3 {t_f}^3 \tag{3.58 b}$$

$$0 = a_1 \tag{3.58 c}$$

$$0 = a_1 + 2a_2 t_f + 3a_3 {t_f}^2 \tag{3.58 d}$$

By solving these equations, the coefficients are

$$a_0 = \theta_0 \tag{3.59 a}$$

$$a_1 = 0 \tag{3.59 b}$$

$$a_2 = \frac{3}{{t_f}^2}(\theta_f - \theta_0) \tag{3.59 c}$$

$$a_3 = -\frac{2}{{t^3}_f}(\theta_f - \theta_0) \tag{3.59 d}$$

3.6.2. Cubic path for via points

In the trajectory planning with cubic polynomial functions, we have considered the motions described by a set of constraints such as initial position, final position and duration of time. In general, it is required to allow the paths that include intermediate via points. Normally the manipulator can pass through the via points without stopping. For this, a generalized cubic fit that satisfies the path constraints is required. Each of these via points is converted into a set of joint configurations by the application of inverse kinematics. Then the smooth joint trajectories that connect via points can be computed.

If desired velocities of the joints at the via points are known, then cubic polynomials functions can be approximated. However, the velocity constraints at each end are not zero, rather some known velocity. The velocity constraints are

$$\dot{\theta}(0) = \theta_o \tag{3.60 a}$$

$$\dot{\theta}(t_f) = \dot{\theta}_f \tag{3.60 b}$$

The equations describing general cubic spline are

$$\theta_0 = a_0 \tag{3.60 a}$$

$$\theta_f = a_0 + a_1 t_f + a_2 {t_f}^2 + a_3 {t_f}^3 \tag{3.60 b}$$

$$\dot{\theta}_o = a_1 \tag{3.60 c}$$

$$\dot{\theta}_f = a_1 + 2a_2 t_f + 3a_3 t_f^2 \tag{3.60 d}$$

Solving these equations for the coefficients, we obtain

$$a_0 = \theta_0 \tag{3.61 a}$$

$$a_1 = \dot{\theta}_0 \tag{3.61 b}$$

$$a_2 = \frac{3}{{t_f}^2}(\theta_f - \theta_0) - \frac{2}{t_f}\dot{\theta}_0 - \frac{1}{t_f}\dot{\theta}_f \tag{3.61 c}$$

$$a_3 = -\frac{2}{t_f^3}(\theta_f - \theta_0) + \frac{1}{t_f^2}(\dot{\theta}_f + \dot{\theta}_o) \tag{3.61 d}$$

If the desired joint velocities are known at each via point, then the coefficients in the equation can be applied to determine the required cubic functions for each segment.

3.6.3. Higher-order polynomials

Higher-order polynomials are sometimes used for path segments. For which, the position, velocity, and acceleration at the beginning and end of a path segment need to be specified. A quintic polynomial function is represented as

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \tag{3.62}$$

where the constraints are given as

$$\theta_0 = a_0 \tag{3.63 a}$$

$$\theta_f = a_0 + a_1 t_f + a_2 {t_f}^2 + a_3 {t_f}^3 + a_4 t_f^4 + a_5 t_f^5 \tag{3.63 b}$$

$$\dot{\theta}_o = a_1 \tag{3.63 c}$$

$$\dot{\theta}_f = a_1 + 2a_2 t_f + 3a_3 t_f^2 + 4a_4 t_f^3 + 5a_5 t_f^4 \tag{3.63 d}$$

$$\ddot{\theta}_0 = 2a_2 \tag{3.63 e}$$

$$\ddot{\theta}_f = 2a_2 + 6a_3 t_f + 12a_4 t_f^2 + 20a_5 t_f^3 \tag{3.63 f}$$

These constraints specify a linear set of six equations with six unknowns, whose solution is

$$a_0 = \theta_0 \tag{3.64 a}$$

$$a_1 = \dot{\theta}_0 \tag{3.64 b}$$

$$a_2 = \frac{\ddot{\theta}_0}{2}, \tag{3.64 c}$$

$$a_3 = \frac{20\theta_f - 20\theta_o - (8\dot{\theta}_f + 12\dot{\theta}_o)t_f - (3\ddot{\theta}_0 - \ddot{\theta}_f)t_f^2}{2t_f^3} \tag{3.64 d}$$

$$a_4 = \frac{30\theta_0 - 30\theta_f + (14\dot{\theta}_f + 16\dot{\theta}_o)t_f + (3\ddot{\theta}_0 - 2\ddot{\theta}_f)t_f^2}{2t_f^4} \tag{3.64 e}$$

$$a_5 = \frac{12\theta_f - 12\theta_0 - (6\dot{\theta}_f + 6\dot{\theta}_o)t_f - (\ddot{\theta}_0 - \ddot{\theta}_f)t_f^2}{2t_f^5} \tag{3.64 f}$$

In this thesis, the redundant robot is simulated for different paths and the corresponding joint configurations were evaluated. A suitable trajectory planning scheme has been implemented to determine the joint trajectories by considering the IK solutions at the via points of the path. This work also focuses on optimum trajectory planning in the joint space of a planar redundant robot, traversing in a constrained path and point to point applications. Trajectory planning of spatial redundant manipulators is also performed with a performance criterion of singularity avoidance.

CHAPTER-IV

4. Redundancy Resolution Techniques

In the redundant manipulators, the number of DOF is greater than the number of end-effector kinematic constraints. This extra DOF makes some of the joint parameters redundant. Displacement of these redundant joint variables causes the change in robot configuration, but it does not necessarily produce a change in end-effector pose. Redundant joint parameters were chosen arbitrarily for the desired end-effector location. The task of selecting the best IK solution from the available feasible solutions is known as redundancy resolution. The additional DOF of the manipulator can accomplish the desired objectives, which can be considered as secondary goals.

4.1. Kinematic redundancy

Kinematic redundancy is crucial in adopting the utilization of robots in versatile applications, in the areas of automotive industries, medical field, and space explorations. This chapter describes kinematic redundancy and redundancy resolution techniques.

4.1.1. Kinematic null space

The null space of a matrix $\boldsymbol{A}$ is a subspace of vectors for which $\boldsymbol{AX} = 0$ written as

$$Null(\boldsymbol{A}) = \left\{ \boldsymbol{X} \in R^{n}; \boldsymbol{AX} = 0 \right\} \tag{4.1}$$

where *n* is the column dimension of A. In inverse kinematics, the null space of the Jacobian is referred to as a set of all vectors of θ which satisfy the equation

$$Null(\boldsymbol{J}) = \left\{ \dot{\boldsymbol{\theta}} \in R^{n}; \boldsymbol{J}\dot{\boldsymbol{\theta}} = 0 \right\} \tag{4. 2}$$

The vector $\dot{\boldsymbol{\theta}}$ in the Eq. 4.2 is in the null space of matrix J, which does not cause any movement to the end-effector with the change in joint velocity.

The null space of a kinematic system exists only when the system has a Jacobian matrix ***J,*** $\boldsymbol{J} \in R^{mxn}$ with $m < n$, in which the system is redundant and underdetermined. The generalized pseudo inverse formulation [93] describes the kinematic null space in a way that it can be utilized by projecting a vector onto the IK solution via a null space projector.

4.1.2. Generalized pseudo inverse

The generalized pseudo-inverse matrix for a system is derived from the forward kinematic equation

$$J\dot{\theta} = \dot{X} \tag{4.3}$$

Premultiplying by J^T to get a square matrix JJ^T leads to

$$J^T J\dot{\theta} = J^T \dot{X} \tag{4.4}$$

To get the joint velocity term on the left side of the equation, premultiply by $(J^T J)^{-1}$

$$(J^T J)^{-1} J^T J\dot{\theta} = (J^T J)^{-1} J^T \dot{X} \tag{4.5}$$

$$\text{further, } I = (J^T J)^{-1} J^T J = J^+ J \tag{4.6}$$

where $I \neq JJ^+$ or $I - JJ^+ \neq 0$, from this it is observed that there is an additional term called null space projection operator

$$(I - J^+ J)z \tag{4.7}$$

From the Eq. 4.8 it can be shown that the IK solution with secondary velocity vector z is not effecting the end-effector velocity

$$\dot{\theta} = (I - J^+ J)z \Rightarrow \dot{X} = J\dot{\theta}$$

$$\dot{X} = J(I - J^+ J)z$$

$$\dot{X} = (J - JJ^+ J)z \tag{4.8}$$

$$\dot{X} = (J - J)z$$

$$\dot{X} = 0$$

assembling the homogenous and particular solution, the generalized inverse is

$$\dot{\theta} = J^+ \dot{X} + (I - J^+ J)z \tag{4.9}$$

4.2. Velocity based redundancy resolution

The redundancy resolution techniques have been implemented to improve manipulator performance. This has been achieved through motion control at the velocity level and position level. Research has been carried out in redundancy resolution techniques to meet performance criteria such as improvement of manipulability, joint torque optimization, joint limit avoidance, and collision avoidance.

4.2.1. Jacobian pseudo-inverse

One of the early implemented techniques for redundancy resolution is to use the optimized inverse of Jacobian. The pseudo-inverse of generalized Jacobian [42] can be used to minimize a scalar function *g* which is the Euclidean norm of joint velocity vector or joint deviation vector $\Delta\boldsymbol{\theta}$

$$g(\Delta\boldsymbol{\theta}) = \sum_{j=1}^{j=n} (\Delta\theta_j)^2 \tag{4.10}$$

Subjected to the constraints

$$\Delta\boldsymbol{e} = \boldsymbol{J}.\Delta\boldsymbol{q} \tag{4.11}$$

where *e* is the error function related to end-effector velocity, **J** is the Jacobian for a particular configuration.

In the Jacobian pseudo-inverse method, redundancy is utilized to minimize the norm of deviation from the present configuration or joint speeds.

4.2.2. Extended Jacobian method

In the extended Jacobian method [67, 102], the degree of redundancy i. e. *n-m* rows are augmented to the Jacobian matrix to make gradient *g(θ)* as zero in the null space of Jacobian, where *g(θ)* represented as an objective of the secondary goal cost function

$$\Delta x = \begin{bmatrix} \boldsymbol{J}_e(\boldsymbol{\theta}) \\ \boldsymbol{J}_{ext}(\boldsymbol{\theta}) \end{bmatrix} \Delta\boldsymbol{\theta} \tag{4.12}$$

where $\boldsymbol{J}_e$ is the end-effector Jacobian matrix $\boldsymbol{J}_{ext}$ is the extended Jacobian matrix

$$\boldsymbol{J}_{ext} = \frac{\partial f_c(\boldsymbol{\theta})}{\partial\theta} \tag{4.13}$$

$f_c(\boldsymbol{\theta}) = \boldsymbol{N}^T \frac{\partial g}{\partial\theta}$, where *g(θ)* is the scalar kinematic objective function and N is the null spacematrix of ***J*** that correlates the self-motion of the redundant manipulator.

4.2.3. Gradient projection method

The gradient projection method proposed by Liegeois [69] used to exploit the redundancy to avoid joint limits, the pseudo-inverse solution can be extended and the general solution for the IK problem can be expressed as

$$\delta\theta = J^{+}\delta x + (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\,\mathbf{Z} \tag{4.14}$$

where $\delta\theta$ is the differential joint angular motion, δx is the differential variation in the end-effector motion J^{+} is the pseudo-inverse of Jacobian of the manipulator, $(I\text{-}J^{+}J)$ is the projector matrix and $\mathbf{Z}$ is an arbitrary vector.

First-term in the equation is the least norm solution and the second term is the null-space solution, which is orthogonal to least norm solution. The null space solution is the self-motion of the manipulator which produces no end-effector motion. For the desired end-effector motion a null space or homogenous solution is chosen in such a way that the resulting joint configuration optimizes the performance measure, known as $h(\theta)$, $\mathbf{Z}$ is chosen to be

$$\mathbf{Z} = \pm \mathrm{K} \nabla h\,(\theta) \tag{4.15}$$

where K is the positive real number and $\nabla h(\theta)$ is the gradient of $h(\theta)$, a positive sign indicates the criterion is to be maximized and a negative indicates minimization of criterion.

The potential function has been chosen for different secondary goals of redundant manipulator such as obstacle avoidance and singularity avoidance

4.2.4. Singularity avoidance at velocity level

In singularity avoidance, the potential function is selected as the manipulability index, which can be expressed as

$$\mu = \sqrt{\det(JJ^{T})} \tag{4.16}$$

The redundancy can be used to maximize this measure so that the algorithm avoids the singular configurations. Some of the methods such as Damped Least Square (DLS) [45,103] and Selective damped least squares (SDLS) [45,104] are used to improve the dexterity of redundant manipulators based on the IK solution which deals with singularity avoidance and does not use the null space through the projection operator. The DLS inverse is the variation

of pseudo-inverse which reduces the effect of singularities on IK solutions. The DLS conditions the pseudo inverse by the formula given by

$$\dot{\boldsymbol{\theta}} = \boldsymbol{J}^{\boldsymbol{T}}(\boldsymbol{J}\boldsymbol{J}^{\boldsymbol{T}} + \lambda^2 \boldsymbol{I})^{-1}\dot{\boldsymbol{x}} \tag{4.17}$$

where I is the n x n identity matrix and λ is damping constant with non-zero value instead of minimizing $\dot{\theta}$ alone which solves the IK Problem, DLS minimizes the quantity

$$\| \boldsymbol{J}\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{X}} \| + \boldsymbol{\lambda} \| \dot{\boldsymbol{\theta}} \|^2 \tag{4.18}$$

This approach improves the IK solution near the singularities by limiting the projection of task space velocity onto joint space velocity when the Jacobian is near-singular. The value of damping parameter λ should be selected very large, such that it improves the behaviour of the solution near singularities. Selectively damped least squares and numeric filtering are singular value decomposition-based approaches. DLS method uses a single and fixed damping constant, whereas SDLS uses variable damping constants. These variable constants are automatically adjusted based on the error distance of the end-effector target.

4.2.5. Collision avoidance

Collision avoidance is often implemented for controlling redundant manipulators, when they were work in complex workspaces with obstacles. The task of collision avoidance is solved using different redundancy resolution methods. Collision avoidance of robots is very crucial for avoiding collisions with obstacles and also for preventing self-collisions. Several redundancy resolution methods have been developed for collision avoidance. The most popular approach, proposed by Maciejewski and Klien [64] includes calculating both the minimum distance between the obstacle and redundant manipulator and the point on the manipulator closest to the obstacle.

Any set of joint rotations that obtains goal configurations without collisions is considered as collision avoidance. Obstacle avoidance scheme has been implemented by considering vector *z*, which specifies velocity i.e collision free joint space vector, the solution is given as

$$\dot{\boldsymbol{\theta}} = \boldsymbol{J}^{+}\dot{\boldsymbol{X}} + (\boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J})\boldsymbol{z} \tag{4.19}$$

The homogenous part of the solution is used to reconfigure the manipulator to be nearer to the collision-free joint configuration.

The obstacle avoidance approach is to identify the point on the manipulator that is closest to an obstacle which is referred to as an obstacle avoidance point. A velocity is assigned to this obstacle avoidance point in a direction away from the obstacle. The primary target of specified end-effector velocity and secondary criterion of obstacle avoidance is given by

$$\begin{aligned} J_e\dot{\theta} &= \dot{x}_e \ \textit{and} \\ J_0\dot{\theta} &= \dot{x}_0 \end{aligned} \tag{4.20}$$

where J_e= Jacobian of end-effector

J_0= Jacobian of obstacle avoidance point

$\dot{x}_e$ =Specified end-effector velocity

$\dot{x}_0$ =Specified velocity of obstacle point

For obstacle avoidance, the matrix has been modified by adjoining the two Jacobians. The two equations in 4.20 were modified into a single equation

$$\begin{bmatrix} J_e \\ \hline J_0 \end{bmatrix}\dot{\theta} = \begin{bmatrix} x_e \\ \hline x_0 \end{bmatrix} \tag{4.21}$$

The set of solutions that exactly satisfy the primary goal and secondary goal of obstacle avoidance is given by

$$J_0 J_e^{+}\dot{X}_e + J_0(I - J_e^{+}J_e)z = \dot{x}_0 \tag{4.22}$$

This equation can be solved for the desired homogenous solution. A solution that increases minimum obstacle distance is given by

$$z = \left[J_0(I - J_e^{+}J_e)\right]^{+}(\dot{X}_0 - J_0 J_e^{+}\dot{X}_e) \tag{4.23}$$

The desired solution satisfying two goals satisfying the constraints imposed by the available degrees of freedom

$$\dot{\theta} = J_e^{+}\dot{x}_e + (I - J_e^{+}J_e)\left[J_0(I - J_e^{+}J_e)\right]^{+}(\dot{X}_0 - J_0 J_e^{+}\dot{X}_e) \tag{4.24}$$

the solution can be simplified as

$$\dot{\theta} = J_e^{+}\dot{x}_e + \left[J_0(I - J_e^{+}J_e)\right]^{+}(\dot{X}_0 - J_0 J_e^{+}\dot{X}_e) \tag{4.25}$$

Several collision avoidance techniques have been implemented using the pseudo-inverse of Jacobian such as the extended Jacobian technique and task priority approach. The details and limitations of these approaches are presented in chapter 2.

4.3. Position based redundancy resolution

Different redundancy resolution methods at the velocity level have been presented. Velocity level methods determine the required joint velocities while achieving specified end-effector velocity. These methods are not able to give directly the joint positions that result in a specified end-effector position. To find the joint positions, joint rates that are evaluated by the redundancy resolution at velocity level must be integrated.

Mathematically, the redundancy resolution in the position level is described by finding θ which results in best configuration that fulfills the performance criterion of robot and reaching desired task space location, given as

$$x_d = f(\theta) \tag{4.26}$$

where x_d is the desired end-effector position in the task space, θ is the joint configuration corresponding to x_d

As the problem at the position level is to be solved by the integration of joint velocities, for which an initial condition is required. The initial condition is described by defining the initial configuration of the manipulator from which the motion towards the target location begins [75]. The initial posture $\boldsymbol{\theta}_1$, when the end-effector is located at $\boldsymbol{x}_1$

$$x_1 = f(\theta_1) \tag{4.27}$$

The end-effector path is assumed between the initial position x_1 and the desired position x_d. The simplest assumption for this path is a line segment joining two positions in the workspace. This line segment is divided into *N* smaller segments for numerical integration. At any position on the line segment, the joint rates required to move the end-effector along the path can be determined.

These joint velocities are integrated sequentially till the end-effector reaches the target location. The redundancy resolution algorithm is as follows

1. Assume an initial configuration $\boldsymbol{\theta}_1$ of the manipulator and calculate the initial position of the end-effector $\boldsymbol{x}_1$
2. Plan a trajectory from $\boldsymbol{x_1}$ to $\boldsymbol{x_d}$ and assume time period of motion, T
3. Determine the velocity at an interval *k* that moves the end-effector towards the desired position

$$\dot{\boldsymbol{x}}_k = \alpha \frac{\boldsymbol{x}_d - \boldsymbol{x}_k}{(N+1-k)\Delta t}, \Delta t = \frac{T}{N} \tag{4.28}$$

4. Joint rates are generated for the specified end-effector velocity at the interval k, joint velocity using the pseudo inverse of Jacobian

$$\dot{q}_k = J^{+}_{e}(\theta_k)\,\dot{x}_k \tag{4.29}$$

5. Determine the θ_k at the next interval by numerically integrating

$$\theta_{k+1} = \theta_k + \dot{\theta}_k \Delta t \tag{4.30}$$

6. The new end-effector position

$$x_{k+1} = f(\theta_{k+1}) \tag{4.31}$$

7. Repeat the steps 2-6 for time interval *k=1:n*

In this work, redundancy resolution at position level has been carried out while the robot is traversing a path in different working environments. Performance metrics such as joint-rotation minimization, singularity avoidance, joint-torque minimization have been considered. The task of redundancy resolution was performed for a robot while moving along a specific path and the path is discretized into several points. A non-linear constrained optimization algorithm is implemented for every point on the path by assuming an initial guess, which is the home configuration of the robot. A performance measure has been chosen for the redundancy resolution. The optimization process results in the best configuration that satisfies the required performance criteria. Different performance metrics chosen for computing an IK solution are discussed below.

4.3.1. Joint distance minimization

Minimization of the sum of individual joint rotation between the home configuration, which is assumed as initial configuration, and the configuration corresponding to end-effector task space location i.e., final configuration, used as the optimization criterion given in Eq. (4.32). The reachability of the end effector in task space is chosen as constraints of the problem given in Eq. (4.33). The IK problem formulated as a constrained optimization problem in the 2D workspace is stated as

Minimize: $$f = \sum_{j=1}^{n} (\theta_{ij} - \theta_{fj})^2 \tag{4.32}$$

Subject to: $$g = \left((E_x - P_x)^2 + (E_y - P_y)^2\right) = 0 \tag{4.33}$$

where θ_{i_j} is the initial joint configuration of the corresponding j^{th} joint; θ_{f_j} is the final joint configuration of the corresponding j^{th} joint.

In the 3D workspace, the objective of individual joint distance minimization remains the same whereas reaching of end-effector to task location is stated as

$$\textit{Subjected to: } g=\left((E_x-P_x)^2+(E_y-P_y)^2+(E_z-P_z)^2\right)=0 \tag{4.34}$$

The task of redundancy resolution was performed using the sequential quadratic programming technique. This criterion has been implemented for planar and spatial redundant manipulators traversing specified paths with and without obstacles in the workspace. Obstacle avoidance was also included in this scheme using the penalty approach.

4.3.2. Singularity avoidance at position level

Singular configurations are defined as the configurations of the robot at which the required joint rates to achieve an end-effector motion along one or more directions are extremely high. At singular configuration, Jacobian loses its full rank. Singularities of serial manipulators are of two types, boundary singularity, and interior singularity. Boundary singularities are observed when the robot is fully stretched out in such a way that the end-effector is very near to the boundary of the workspace. Interior singularities occurred by lining up two or more joint axes. Because of this, robot performance is affected. Hence, these singular configurations are to be avoided.

The measure of manipulability can be used as the potential function for singularity avoidance, which is given as follows

$$\mu=\left(\sqrt{\left|\boldsymbol{J}\boldsymbol{J}^{\boldsymbol{T}}\right|}\right) \tag{4.35}$$

The measure of manipulability is non-negative at non-singular configurations. It becomes zero only at singular points. The higher the manipulability measure, the robot is away from the singular configuration. Here, the problem of singularity avoidance is performed by maximizing the manipulability measure with the constraint of reaching the task space location shown in Eq. 4.34. The optimization problem is solved and the configurations avoiding singularities have been obtained for a redundant robot while traversing a path in the planar and 3D workspace. Results are also shown when the obstacles are included in the workspace.

4.3.3. Optimum trajectory planning

Optimum trajectory planning of redundant manipulator has been evaluated, while the robot is performing the task and the power consumed to move the joints of the robot is being minimized. The end-effector of the robot is commanded to traverse a path in the workspace. The Lagrangian formulation is used to determine the joint torques and equations of motion, which is given by

$$\frac{d}{dt}\left[\frac{\partial L}{\partial \dot{q}_i}\right]-\left[\frac{\partial L}{\partial q_i}\right]=\tau_i \tag{4.36}$$

where, *L* represents the Lagrangian, which is defined as the difference between total kinetic energy and total potential energy of the manipulator. The joint torque applied is specified as τ_i .

The total torque of the manipulator is put in the matrix form after applying Lagrangian equations of motion, is given by

$$\begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{bmatrix} + \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \\ \dot{\theta}_3^2 \end{bmatrix} + \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1\dot{\theta}_2 \\ \dot{\theta}_1\dot{\theta}_3 \\ \dot{\theta}_2\dot{\theta}_3 \end{bmatrix} + \begin{bmatrix} G_{11} \\ G_{22} \\ G_{33} \end{bmatrix} \tag{4.37}$$

where M, C, H, G represents inertia, centrifugal, Coriolis, and gravity matrix. A 3 DOF planar manipulator is chosen for optimal trajectory simulation.

The trajectory of each joint is interpolated with a quintic polynomial function

$$\theta_i(t) = a_i t^5 + b_i t^4 + c_i t^3 + d_i t^2 + e_i t + f_i \tag{4.38}$$

where the coefficients of the polynomial equations are to be determined, the values of the coefficients should minimize the objective of power consumption while satisfying the end conditions.

The objective of the optimum trajectory planning problem is to manipulate the end-effector of the planar redundant robot along a given path with minimum power consumption.

The objective function of the optimization problem is stated as

$$f = \sum \left| \left(\omega_i T_i \right)^2 \right| = \sum \left| \left(\dot{\theta}_i T_i \right)^2 \right| \tag{4.39}$$

Subjected to constraints

$$g_1 = \left(x_{te} - x_{tp} \right)^2 + \left(y_{te} - y_{tp} \right)^2 = 0 \tag{4.40}$$

$$g_2 = \ddot{\theta}_i \tag{4.41}$$

where, ω_i is the angular velocity of the i^{th} joint, τ_i is the torque applied at the i^{th} joint, (x_{te}, y_{te})are the coordinates of end-effector (x_{tp}, y_{tp}) are the coordinates of task space and $\ddot{\theta}_i$represents angular acceleration at the i^{th} joint. Trajectory planning is implemented for the 3DOF planar robot for two different types of point-to-point motion and continuous path motion.

In this thesis, the minimization of power consumption at each joint was analysed. Thus the individual joint torques were minimized. The task of minimization of power consumption was performed by formulating the dynamic equations of a planar redundant manipulator which gives the equations of motions in terms of joint torques. Joint motions are interpolated with a polynomial equation. Coefficients of the polynomial equations were chosen as the variable of the optimization problem. An optimization algorithm is implemented, which gives the values of coefficients of the polynomial equation which results in minimum total power consumption.

CHAPTER V

5. Collision Avoidance Techniques

5.1. Collision avoidance

Redundant robots performing the required task need to avoid collisions with obstacles in the workspace. The collision avoidance requires collision detection i.e., to find whether a robot is in a collision with any obstacle at a given configuration. Collisions in the workspace occur in two aspects namely, the collision of a link with obstacles and other links. Collision is an undesirable effect, as it causes a loss of energy and damage to the parts of the manipulator. This section demonstrates collision avoidance techniques for planar and spatial redundant robots with polygonal and 3D obstacles.

5.1.1. Collision avoidance for planar robots

Collision avoidance requires collision detection, for which links of a robot are modelled as line segments and obstacles are modelled as polygons shown in Fig. 5.1. The problem of detecting collisions between links and obstacles boils down to finding the intersection of line segments and a polygon. The line segments are discretized into a set of points proportionate to their link length shown in Fig. 5.1 a. Collision detection of link with polygon can be obtained by checking whether the points on the link lie inside, outside, or boundary of the polygon. This problem can be considered as a detection of point-in-polygon.

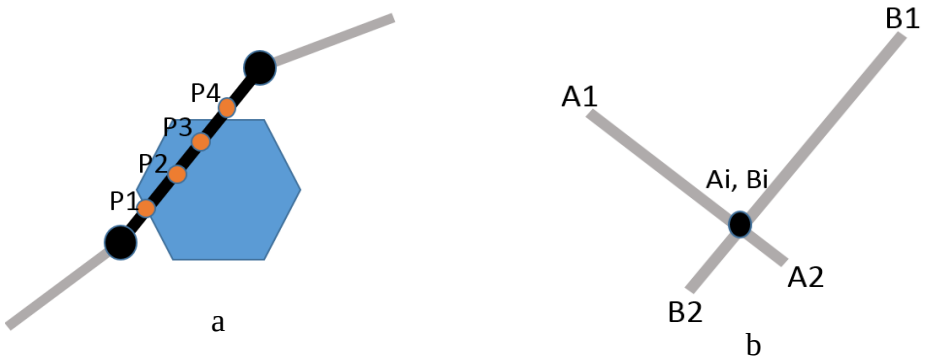


Fig. 5.1. Schematic sketch of collision detection scheme (a) Link and polygon (b) Self-intersection of links

Point-in-polygon detection is a computational geometry technique used in computer graphics and motion planning application [105, 106]. Generally, this check is performed using two types of algorithm such as winding number and ray casting algorithm.

5.1.1.1. Winding number algorithm

Winding number is defined as the number of times a curve travels around a point [107]. This algorithm states that for a point in a polygon this number will be non-zero. To determine the winding number, it is to calculate the angle subtended by each side of the polygon with the query point. This is indicated by angles θ_1, θ_2, θ_3 and θ_4 with the edges of polygon AB, BC, CD, and DA respectively shown in Fig. 5.2. If the summation of these angles adds up to 2Π the point lies inside the polygon, for a query point P_1 shown in Fig and if the sum is 0, the point lies outside, for a query point P_2, shown in Fig. 5.2.

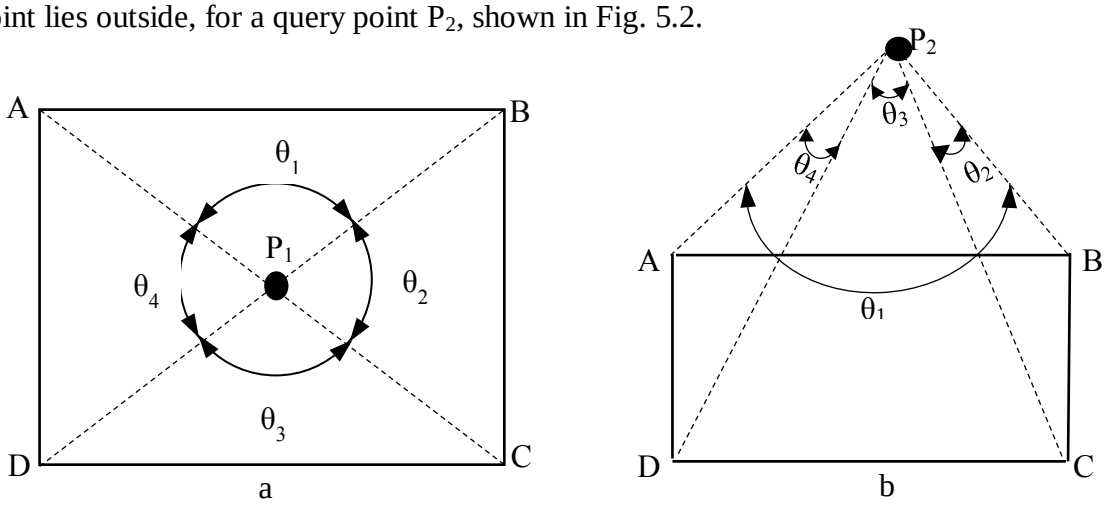


Fig. 5.2. Illustration of point detection in polygon (a) Point located inside (b) Point located outside.

5.1.1.2. Ray casting algorithm

Ray casting algorithm is also known as the crossing number algorithm [106,108]. This algorithm determines if a point located inside or outside a polygon by finding how many times a ray (starting from a point and going in a fixed direction) intersects the edges of the polygon. The condition to check if a point lies outside the polygon, the ray will intersect the edges of the polygon for an even number of times. If a point lies inside, the ray intersects the edges of the polygon for odd number of times. The illustration of this technique is shown in Fig. 5.3.

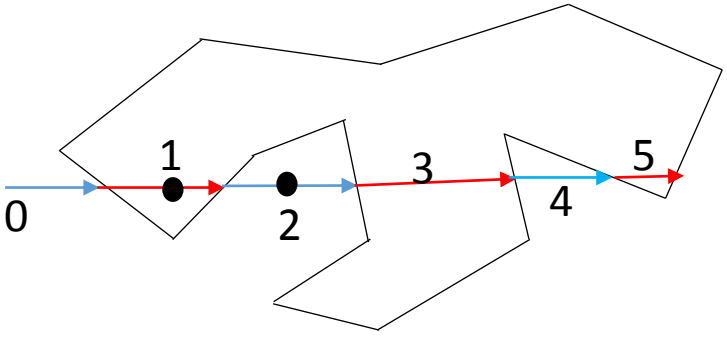


Fig. 5.3. Illustration of ray casting algorithm for a point-in-polygon.

The task of computing a point in a polygon is performed using ***inpolygon*** function of MATLAB. This MATLAB function works on the principle of winding number algorithm [107]. This algorithm evaluates a given set of points on the links whether they lie inside, outside, or on the boundary of the polygonal obstacles for a given configuration.

Winding number and ray casting algorithms are popularly referred as "*Point in Polygon"* algorithm which have been implemented in different areas such as computer graphics [109], computer vision, geographical information system [110], motion planning of robots [111], CAD [112]. Huang [113] proposed comparison of different point in polygon methods. In his research, it is shown that among the different approaches, ray casting method and winding number methods are well suited for non-convex shaped polygons.

Some of the limitations with this approach are

I. The computational complexity increases with an increase in the number of nodes of a polygon.
II. Although it can deal with all kinds of polygons, it is difficult to detect if the investigated point lies on the polygon circumference or if the ray intersects a polygon vertex.
III. Evaluation of point in polyhedron is difficult for 3D obstacles when the boundary of a geometric object is represented as a triangulated surface.

Although this approach has certain limitations, the advantages of computational time and suitability for convex and non-convex obstacles made this approach to select this method for collision detection of polygonal obstacles.

In robotic applications, polygons with a limited number of nodes without self-intersection have been considered. For this application ray casting approach is appropriate and computational complexity is less.[113].

However, the limitation of 3D obstacles with ray casting method has been overcome by adopting bounding box approach in the thesis.

Most of the obstacle avoidance methods of redundant manipulators use pseudo-inverse techniques [67], configuration space approach [73] and artificial potential field approach [88]. The above mentioned approaches are velocity based methods which work by assigning velocity to the critical point on the robot and directs away from the obstacle.

These methods suffer from following limitations

1. Sensitivities at the singular configuration and are known to be computationally expensive.
2. Difficulty with local minima.

By considering the above limitations in the velocity based schemes the proposed approach adopts ray casting algorithm for collision detection of planar robot avoiding convex and non-convex obstacles.

The Problem of self-intersection of two links (detecting whether a pair of links collide in a particular configuration) boils down to finding the intersection of two line segments, which is illustrated by Fig. 5.1(b). This collision is detected using ***polyxpoly*** function of MATLAB, which returns the coordinates of the intersecting points of links in the planar workspace.

Once the configurations with collisions are identified, these are to be avoided. Collision avoidance of obstacles and self-collisions are handled with a penalty approach. An optimization problem is being solved for computing joint configurations, which minimizes the objective function. If the collision occurs in a particular configuration, a penalty value is being added, which increases the value of the objective function. Further optimization algorithm attempts to minimize the increased objective function. This results in a robot configuration reaching task space by avoiding obstacles.

The modified statement of the optimization problem has been used when the workspace has obstacles, it is given by

$$f = \left((E_x - P_x)^2 + (E_y - P_y)^2\right) + \sum_{i=1}^{m} C_i \tag{5.1}$$

where c_i is the ith penalty, m is the number of collisions

5.2. Obstacle avoidance of 3D obstacles

Robots performing a required task in a real-time working environment need to avoid 3D obstacles. This section describes the collision avoidance scheme of redundant robots avoiding 3D obstacles. The task of collision detection has been carried out by using a bounding box approach. In this work, 3D obstacles such as spheres, cylinders, and cones were considered and the boundaries of these solids are enveloped by a box. For generating bounding boxes, the obstacles were represented as a set of points on solid boundaries. From the point set, the extremum coordinates of the points are determined. By using these coordinates, the vertices of the box are determined. The facet information i. e. the vertices that are used to form a particular face are computed using the convex hull algorithm [114]. The facets and their corresponding vertices are used to model the bounding boxes used in collision avoidance. Once the obstacles in 3D space were surrounded by a bounding box, the configurations that

lead to collision need to be detected. The collision detection scheme is illustrated through the Fig. 5.4.

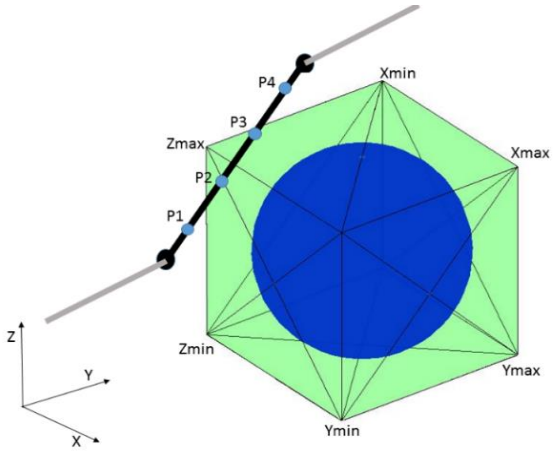


Fig. 5.4. Schematic representation of 3D collision detection scheme.

Several points that are uniformly distributed on the robot link are considered say, P_1, P_2, …..P_n and a check is performed whether these points lie within the bounding box of an obstacle. If any of these points lie in the bounding box, then the collision occurs. The algorithm for collision detection is shown in Table 5.1. This algorithm takes the extremum coordinates of the box as input and checks for a set of points on the links of the robot if they lie within the bounding box. In the algorithm, *n* represents the number of query points on each link for collision check.

Table 5.1. Algorithm for collision detection

Algorithm: Collision detection
1. Input: Extremum coordinates of the bounding box of the obstacle X_{min}, X_{max},Y_{min},Y_{max}, Z_{min}, Z_{max} and point P_i(P_{ix}, $P_{iy,}$ P_{iz}) 2. Collision check for points, P_1……P_n for i=1: *n* **If** X_{min}<= P_{ix}<= X_{max}&& Y_{min}<= P_{iy}<= Y_{max} && Z_{min}<= P_{iz}<= Z_{max} Collision **Else** No Collision End End

By using the above algorithm, configurations leading to the collision have been identified. These configurations need to be avoided, and the robot should move away from the obstacle. The penalty approach has been implemented for obstacle avoidance. The penalties are imposed for the configurations that are interfering with obstacles by augmenting them with the objective function. The modified statement of the optimization problem has been used when the workspace has obstacles, which is given by

$$\textit{Minimize:} \qquad f = \left((E_x - P_x)^2 + (E_y - P_y)^2 + (E_z - P_z)^2\right) + \sum_{i=1}^{m} C_i \tag{5.2}$$

where c_i is the ith penalty, m is the number of collisions.

In practice, the immediate surroundings of an obstacle may get in contact with links. The bounding box approach considers an additional space beyond the volume of the obstacles, which make the robot to maintain some clearance with the obstacles while moving in the workspace. Thus the bounding box model is appropriate for different types of obstacles.

In this thesis, a collision-avoidance scheme has been implemented for hyper-redundant robots working in 2D and 3D workspaces. Simulations have been performed for a robot working in an environment with different shapes of (non-convex and convex) obstacles. Hyper-redundant robots are deployed to work in narrow, confined, and hazardous workspace. A realistic working environment with 3D obstacles has been modelled and a collision avoidance scheme is implemented for robots working in these environments. Case studies of hyper-redundant robot avoiding complex obstacles are presented in chapters 7-8 of this thesis.

Collision avoidance scheme of planar and hyper redundant robots is represented with a flow chart shown in Fig. 5.5

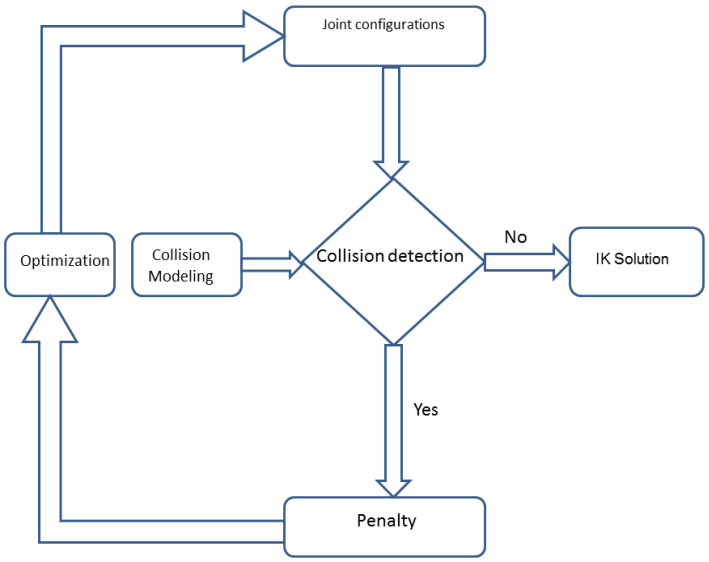


Fig. 5.5. Flow chart showing obstacle avoidance scheme of hyper-redundant robots

CHAPTER-VI

6. Optimization Methods

IK problem of the redundant manipulator is posed as an optimization problem. Optimization techniques employed for solving the IK problem are briefly reviewed in this section. Initially, the IK problem posed as an unconstrained problem i.e. without obstacles, later obstacles have been considered and solved using the penalty approach. An unconstrained and constrained IK problem in planar and 3D environments have been solved using optimization approaches.

6.1. Nelder-Mead's simplex search algorithm

The IK problem posed as a non-linear optimization problem without constraints has been attempted to solve using Nelder and Mead simplex search method [115], this method is a direct search and derivative-free method. This approach is preferable for non-linear and multivariable problems without constraints. This method executes by developing a geometric simplex, which is a geometric figure formed *(n+1)* vertices, where *n* is the number of variables of the optimization problem. If the points of the simplex are equidistant then the simplex is regular. In two-dimensional space, the simplex is a triangle and in three dimensions it is a tetrahedron. The idea in this method is to check the values of the objective function at the *(n+1)* vertices and move the simplex gradually towards the optimum value during the iterative process. At each iteration, the worst value of the vertex is evaluated first. Then a new simplex is formed from the existing simplex by a rule that moves the search away from the worst value of the simplex. Four different situations may occur based on the function values in every iteration. The algorithm carries out the operations such as reflection, extension, and contraction to determine a new vertex, shown in Fig. 6.1.

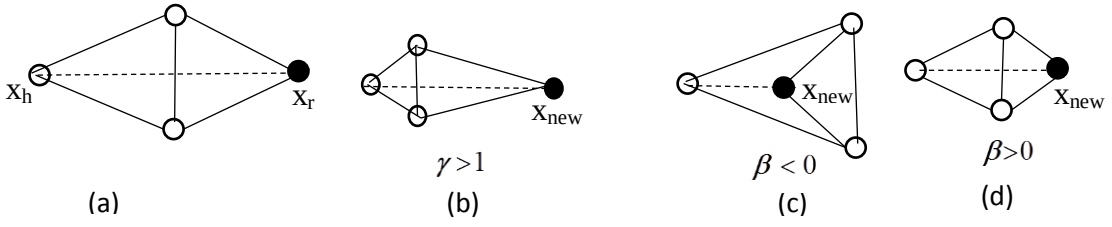


Fig. 6.1. Illustration of simplex search method (a) Reflection (b) Expansion (c), and (d), Contractions.

Simplex method at first, evaluates the centroid of the worst vertex. Then the worst point in the simplex is reflected about the centroid and the new point x_r is found, shown in Fig. 6.1(a).

If the function value at the reflected point is better than the current best point in the vertex, then the reflection leads to a good region in the search space. Thus an expansion along the direction from the centroid to the reflected point is performed shown in Fig. 6.1(b). The amount of expansion is governed by a factor γ . If the function value at the reflected point is worse than the current worst point in the simplex, the reflection is taken the point to the bad region of the search space. Thus a contraction in the direction from the centroid to the reflected point is made shown in Fig. 6.1(c). The amount of contraction is governed by a factor β . Contractions with positive and negative values β are shown in fig 6.1(c-d). The algorithm continues iteratively and the obtained new point replaces the worst point in the simplex. These operations continue till the desired minimum is obtained. This method is used for solving the IK problem without constraints and results are reported.

Nelder-mead simplex method is one of the well-known direct search algorithm for multi-dimensional unconstrained optimization problems [116]. Nelder and mead proposed two ways for handling constraints by transforming the scale of variables and modifying the function value such that it takes the high function value when the constraints are violated. The limitation in this approach in handling of constraints is the necessity for the initial simplex to lie in the feasible region.

Sakai and Iwane [117] proposed a methodogy that overcomes the limitation of handling the constraints, this approach involves the independent treatment for constraint violation and objective function. The effectiveness of simplex search algorithm equipped with constraint handling method has been compared with evolutionary methods by Mehta and Dasgupta [118]. This approach has been implemented on various benchmark problems. Results show that the proposed method performs much better than the several evolutionary algorithms. The strategy of assigning the values to an infeasible point, performs better with simplex method in comparison with GA based approach. Dasgupta et al. [119] proposed an approach to exploit classical optimization algorithms for multi-modal optimization, which gives multiple solutions of the problem.

The limitation of attaining single local solution has been resolved with this approach. Due to the advantages of the Nelder-Mead simplex method it is implemented in the proposed approach for solving IK problem of hyper-redundant manipulator. As the IK problem have multiple solution, this algorithm is used to find multiple IK solutions of the problem. From the results it is inferred that the computational time for solving the IK problem of hyper-

redundant robot in complex environment is less and it is about 2 minutes. Multiple IK solutions of robot is also achieved even with less computational time (60 seconds).

6.2. Multi-start method for multiple optima

The optimization algorithms implemented for solving IK and redundancy resolution are local search algorithms. These algorithms often suffer from getting trapped in a local optimum point. Multi-modal optimization deals with the task of evaluating multiple local optimal solutions while optimizing multi-modal functions. Generally, global optimization algorithms are employed to determine multiple solutions and these algorithms search through more than the single basin of attraction. The task of evaluating multiple IK solutions has been performed using global search and multi-start framework which performs the optimization process with the generation of a number of starting points and use local optimization solver to find the optimal solutions. These optimization routines can be applied to the problems with a smooth objective and constraint function and the solvers search for a global minimum or for a set of local minima.

6.2.1. Optimization workflow

The optimization solver can be employed to determine global or multiple local solutions using a sequence of operations [120]. The optimization workflow starts with the creation of a problem structure. A problem structure specifies a local optimization problem and its solver is used to minimize a given problem. A set of input parameters needs to be defined while creating a problem structure. Parameters such as local solver, objective, constraints, and options of the solver are to be supplied. The structure of the problem is created using *createoptimproblem* function in MATLAB. After the creation of the problem structure, the solver object is to be created which contains the preferences of a global portion of the object. Once the solver objects were created the number of start points from which the optimization process starts is to be defined. The start points can be defined using random start point generation or custom start point generation. Then run solver allows the optimization process to execute and arrives at multiple local solutions and global solution. The workflow of the optimization process is illustrated in Fig. 6.2.

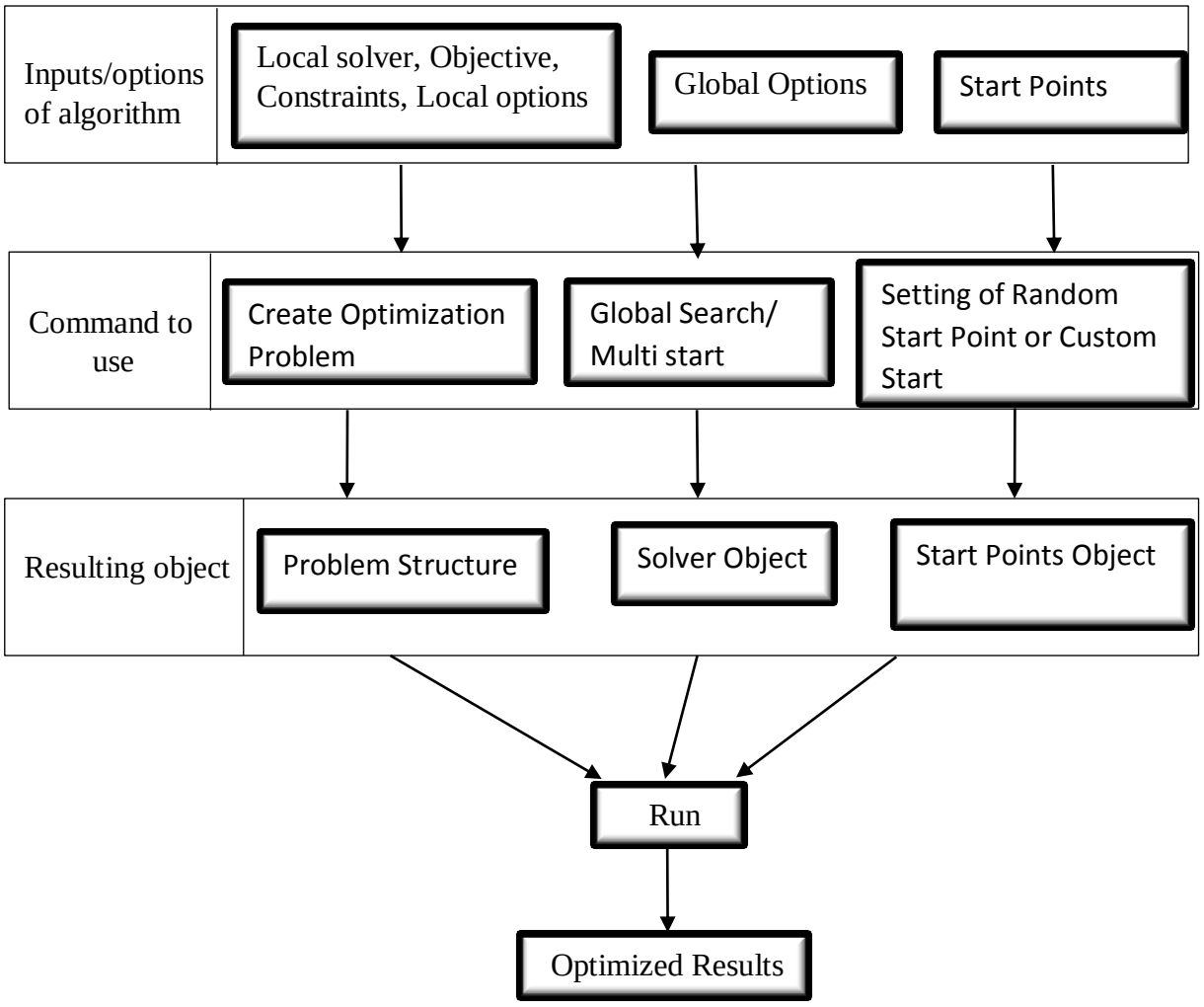


Fig. 6.2. Work flow of global optimization problem.

6.2.2.Multi-start framework

The Multi-Start method [121] has two phases. In the first phase, the solution is generated, and in the second phase, the solution is usually improved. Then, each iteration produces a solution, and the best of overall solutions is the output. The algorithm of the multi-start procedure is described below in Table 6.1. This method generates uniformly distributed points [122] within the search space (*S*) and starts a local solver from each of these points. In general, this approach converges to a global solution when there are a large number of start points in search space, and there is also a chance of arriving at the same local solution many times. To overcome this difficulty, some potential start points that are close to the previous solution have to be eliminated.

Table 6.1. Algorithm for multi-start procedure

Algorithm: Multi-start Procedure
1. Initialize i=0
2. **While** (stopping condition)
(a) Select a decision variable x_i from the multiple start point
(b) Apply a local search optimization algorithm to improve x_i.
(c) Let the x_i^* be the solution obtained
If (x_i^* improves the function value)
(d) Update the best local minimum obtained so far
i=i+1
End
Output: Obtained all local minima

The elimination procedure is performed by generating uniformly distributed start points in *S*, and the objective function value is evaluated at each point. The points are sorted according to their function value and the best points are retained. A local solver starts from each point of the reduced sample, except if there is another sample point within a certain critical distance that has a lower function value. The local solver is not started from the sample points that are very close to a previously discovered local minimum. Then, again additional uniformly distributed points are generated, and the procedure is applied to all the points which are retained from previous iterations and newly generated set of points. The implementation of this algorithm provides multiple IK solutions. A few cases of multiple configurations of spatial hyper-redundant robot were reported for a given task space location. These multiple solutions can be considered as suitable kinematic configurations of a robot working in diverse environments.

6.2.3. Global search algorithm

A global search algorithm has been implemented to compare the solutions that are obtained through the multi-start approach are close to the global minimum. This algorithm generates start points using a scatter search mechanism [123]. The main feature of this algorithm is that it analyses start points and eliminates the points that are not likely to improve the function value. Initially, it generates potential start points then it evaluates score function for a set of trial points. The points with the best score function have been chosen and use that as an initial

point for the local solver. If the remaining trail points satisfy function score and constraint filters, the global search runs the local solver. Finally, this process creates a global optimum solution vector. Joint configurations corresponding to the global minimum for different task space locations are reported using this approach.

6.3. Sequential quadratic programming

The optimization problem with constraints was handled by a non-linear constrained optimization technique called sequential quadratic programming (SQP). This method mimics the Newtons method for constrained optimization problem which is a quadratic approximation of the Lagrangian function with linearization of constraints [124,125]. A quadratic sub-problem is formulated and solved to develop a search direction. The line search can be performed with respect to two alternative merit functions, and a modified BFGS formula updates the Hessian matrix.

A quadratic programming subproblem is formulated based on a quadratic approximation of the Lagrangian function. It is given as

$$L(\boldsymbol{X},\lambda) = f(\boldsymbol{X}) + \sum_{i=1}^{m} \lambda_i . g_i(\boldsymbol{X}) \tag{6.1}$$

where λ_i is a Lagrangian multiplier

The solution vector $\boldsymbol{\Delta X}$ is treated as the search direction $\boldsymbol{S}$, and subproblem of quadratic programming is stated as

$$Q(\boldsymbol{S}) = \nabla f(\boldsymbol{X})^T \boldsymbol{S} + \frac{1}{2} \boldsymbol{S}^T [\boldsymbol{H}] \boldsymbol{S} \tag{6.2}$$

Subjected to

$$\beta_j g_j(\boldsymbol{X}) + \nabla g_j(\boldsymbol{X}^T)\boldsymbol{S} \le 0,\ j = 1,2,\ .....,\ m \tag{6.3a}$$

$$\overline{\beta} h_k(\boldsymbol{X}) + \nabla h_k(\boldsymbol{X}^T)\boldsymbol{S} = 0,\ k = 1,2,\ .....,\ p \tag{6.3b}$$

where $\boldsymbol{H}$ is a positive definite matrix that is initially considered as the identity matrix and it is updated in the subsequent iterations so as to converge to the Hessian matrix of the Lagrangian function, β_j and $\overline{\beta}$ are the constants that ensure the linearized constraints lie in the feasible space.

Once the search direction $\boldsymbol{S}$ is found by solving the problem given in Eq. 6.2, the design vector is updated as

$$X_{J+1} = X_J + \alpha S \tag{6.4}$$

where α is the step length parameter along the direction S.

The step length parameter is evaluated by an appropriate line search procedure so that a required decrease in the merit function can be achieved, Hessian matrix can be updated by any quasi-Newton methods, BFGS method has been used in this approach for updating the Hessian matrix [125].

A non-linear constrained optimization problem can be solved in a few number of iterations than an unconstrained problem using SQP. The reason is because of the limits of feasible search space, the solver can make a decision regarding search direction and step length. The SQP implementation mainly consists of three stages namely updating the Hessian matrix, solution of quadratic programming, line search, and merit function.

Hessian matrix update

Hessian matrix is updated at each iteration as a positive definite quasi-Newton approximation of the Hessian of the Lagrangian function. The H is evaluated using the BFGS method

$$H_{k+1} = H_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{H_k^T s_k^T s_k H_k}{s_k^T H_k s_k} \tag{6.5}$$

where

$$S_k = X_{k+1} - X_k \tag{6.6}$$

$$q_k = \left(\nabla f(x_{k+1}) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x_{k+1}) \right) - \left(\nabla f(x_k) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x_k) \right) \tag{6.7}$$

A positive definite Hessian is maintained by providing $q_k^T s_k$ positive at each update and H is initialized with a positive definite matrix.

Quadratic programming solution

At each iteration of the SQP method, an active set strategy is being implemented. The QP solution procedure involves two phases, the first phase performs the calculation of a feasible point. If it exists, then the second phase involves the generation of an iterative sequence of feasible points that converge to the solution.

Line search and merit function

The solution to the QP problem produces a vector ***S***, which is used to form a new iteration given in Eq. 6.4. The step length parameter α is determined in order to produce a sufficient decrease in the merit function. This allows the positive contribution from constraints that are inactive in the QP solution. A penalty parameter is initially set. This ensures a larger contribution to the penalty parameter from constraints with smaller gradients, which would be the case for active constraints at the solution point.

By using this approach a few simulations of robot tracing a specified path with constraints are presented in chapters 7-8.

6.4. Teaching learning based optimization

TLBO is an evolutionary algorithm which mimics the teaching and learning environment for the optimization process. TLBO is a population-based method which uses a population of the solution to arrive at a global solution. It is a simple and fast converging algorithm. The performance of the algorithm is improved due to the absence of algorithmic tuning parameters that are present in GA and PSO. A group of learners has been considered as a population in this approach [59]. In general, the population of the optimization algorithm consists of design variables. In TLBO, the number of students in the classroom is considered as population size. Design variables are analogous to subjects furnished to the learners. The total marks secured in all subjects by each learner is equivalent to the fitness of the function value. The teacher is regarded as the best solution for the whole population. The operation in the TLBO technique is divided into two phases, i.e, the Teacher phase and Learner phase illustrated in Fig. 6.3.

Teacher phase

In this phase, the teacher attempts to improve the mean result of the class in his subject. A good teacher tries to bring the learners up to his level with respect to his knowledge. But in reality, this is not feasible and a teacher can only improve the mean of the class up to some level depending on the potentiality of the class, which follows a random process based on many factors. At any iteration *i*, consider M_i be the mean and T_i be the teacher. T_i will try to move the mean M_i with regard to its own level, now the new mean will be T_i represented as M_{new}. The solution is upgraded according to the difference between the prevailing and new mean which is stated as

$$Difference\_mean\ i = r_i \left(M_{new} - T_F Mi \right) \tag{6.8}$$

where, TF is a teaching factor that determines the value of the mean to be modified, and r_i is a random number in the range [0, 1]. The value of TF can be either 1 or 2, which is again a heuristic step and determined randomly with the same probability as *TF = round [1 + rand (0, 1) {2 − 1}].*

This difference mean changes the existing solution, as stated by the following expression

$$X_{new,i} = X_{old,i} + Difference\_Meani \tag{6.9}$$

This is repeated for the entire population. After updating the design variables, the fitness values of the population are computed and are compared with the old fitness values. The best fitness values and their corresponding design variables are selected and defined as the teacher phase set. The best solution among the teacher phase set is selected as the present 'teacher'.

Learner phase

Learners improve their knowledge in two different ways: one is through the instruction from the teacher and the other through interaction among themselves. A learner communicates randomly with other learners for increasing their knowledge. A learner learns something new if the other learner has higher knowledge than him or her. Learning phenomenon is expressed for a population size n.

Randomly select two learners X_i and X_j such that i ≠ j.

$$X_{new} = X_i + r_i\left(X_i - X_j\right),\ if\ f\left(X_i\right) < f\left(X_j\right) \tag{6.10}$$

$$X_{new} = X_i + r_i\left(X_j - X_i\right),\ if\ f\left(X_j\right) < f\left(X_i\right) \tag{6.11}$$

Accept X_{new} if it performs better.

This task is repeated for the whole population. The modified values of the design variables are used to evaluate the new fitness value. The fitness values obtained in the learner's phase has been compared with the values in the teacher's phase and the best values are selected. Finally, the best value among the population is chosen as the best solution for the current iteration. This ends the learner's phase. The process is repeated until the convergence criteria is achieved.

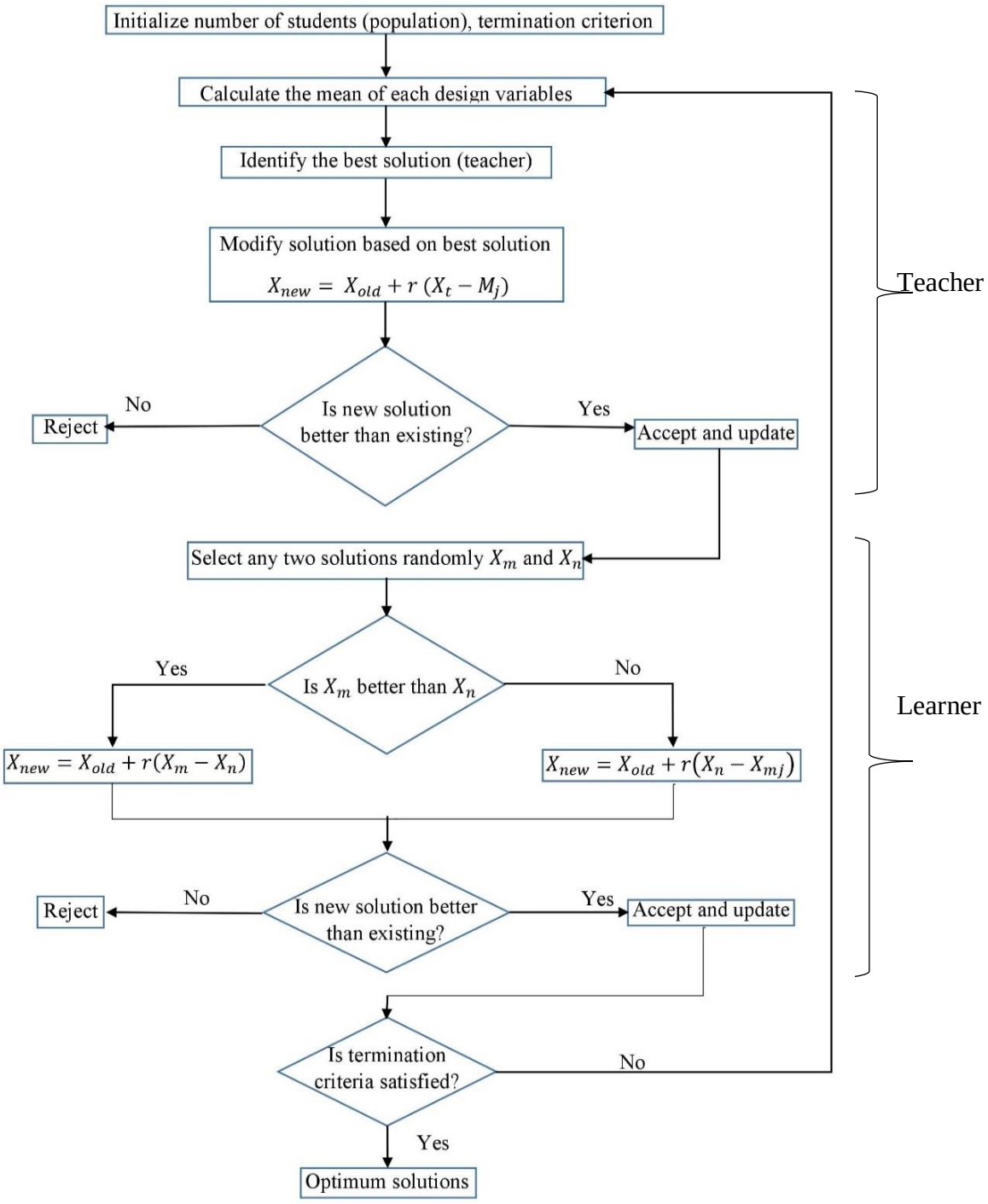


Fig. 6.3. Flow chart of TLBO algorithm.

TLBO algorithm is used for simulating the IK problem of spatial redundant robots avoiding complex 3D obstacles in realistic working environments. IK problem with obstacle avoidance in narrow regions cannot be computed in a single attempt using a classical optimization

algorithm. For complex cases, classical algorithms need to restart several times with different initial guesses for solving the problem. Hence a global optimization based TLBO algorithm is used for computing the IK problem in restricted areas. The results are reported for the cases of robot deployed in inspection and welding of a pipeline, pick and place operation in work facility layout.

6.5. Overview of optimization techniques

The overview of optimization techniques implemented for IK simulation of hyper-redundant robot is described in flow chart given below.

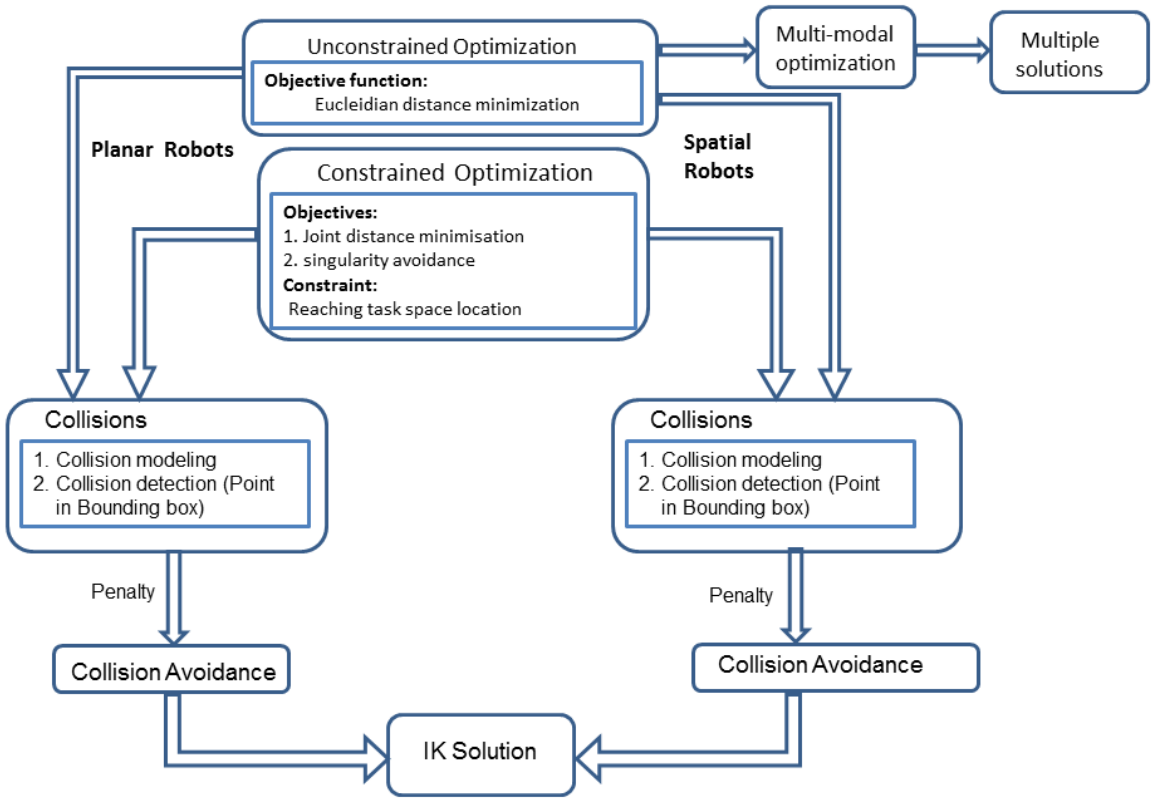


Fig. 6.4. Flow chart of optimization techniques for IK simulation of hyper-redundant robots.

CHAPTER-VII

7. Results of IK Solution of Planar Hyper-Redundant Manipulators

In this chapter, simulation results of the IK solution of planar hyper-redundant manipulators working with and without obstacles in the workspace have been presented. A redundancy resolution scheme is implemented for a planar manipulator traversing along a path by avoiding obstacles in the workspace. The task of redundancy resolution (finding the best solution among the available solutions) is performed using an optimization criterion such as joint-distance minimization and singularity avoidance. Optimum trajectory planning of redundant manipulator is also performed to evaluate joint trajectories while the robot is traversing a straight-line path.

7.1. IK solutions of planar hyper-redundant manipulators

This section shows joint configurations of a robot while the end-effector is commanded to move along the path. The links of the robot are considered uniform in length. Simulations have been performed for 5 DOF and 10 DOF planar redundant robots. Following cases were considered for IK simulation.

Table 7.1. Cases performed for IK simulation of robot in 2D workspace.

<table>
<tr><th>Joint Configurations</th><th>Task Performed</th><th>No of DOF</th><th>Workspace</th></tr>
<tr><td rowspan="7">IK Solution of Planar Redundant robots</td><td rowspan="3">Traversing a path</td><td rowspan="2">5 DOF</td><td>Without obstacles</td></tr>
<tr><td>Two polygonal obstacles</td></tr>
<tr><td rowspan="2">10 DOF</td><td>Without obstacles</td></tr>
<tr><td>Two polygonal obstacles</td></tr>
<tr><td rowspan="2">Specific task space location</td><td rowspan="2">(5-10) DOF</td><td>Convex polygonal obstacles</td></tr>
<tr><td>Non-convex polygonal obstacles</td></tr>
<tr><td>Redundancy resoluton of planar robots</td><td>5 DOF</td><td>Avoiding complex polygonal obstacles</td></tr>
<tr><td>Minimizaton of Joint Power consumption</td><td>3 DOF</td><td>Without obstacles</td></tr>
</table>

Different cases of IK simulation and redundancy resolution of planar hyper-redundant robots working in various environments has been shown in the flow chart shown in Fig. 7.1. Results of the following simulations have been presented in subsequent sections.

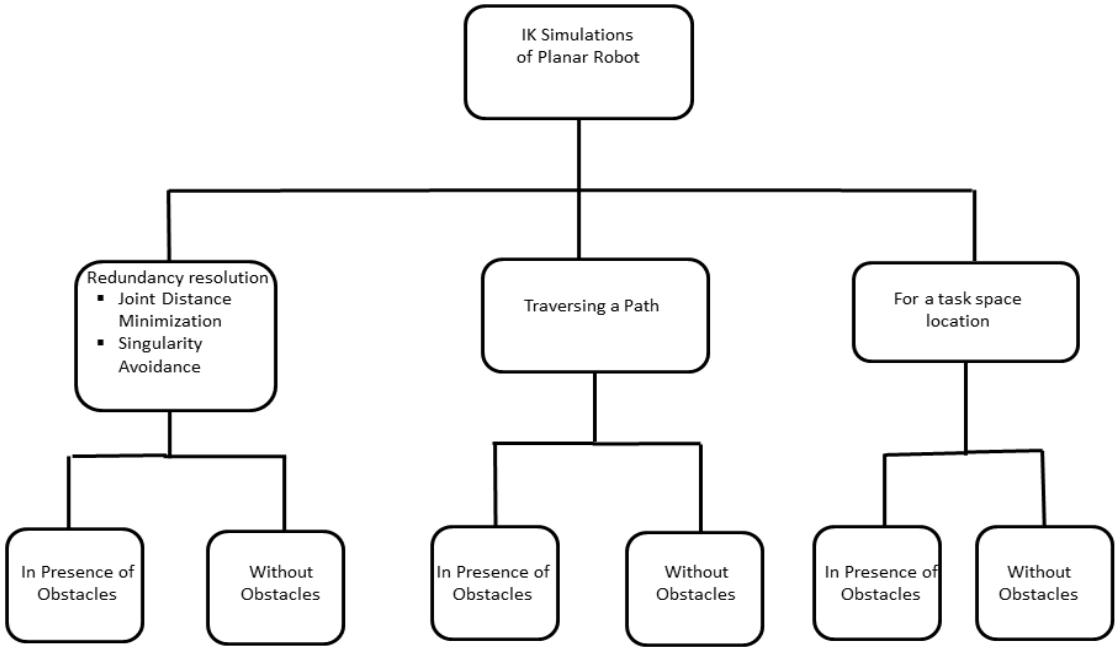


Fig. 7.1. Flow chart representing the IK simulations and redundancy resolution of hyper-redundant robots

7.1.1. Without obstacles in the workspace

IK simulations are performed for 5 linked and 10 linked robot configurations without obstacles in the workspace. A path is chosen in the workspace and a set of points were chosen on the path. The coordinates of the points were given as a task space location (TSL) of the end-effector. Joint configurations are shown for the given task space coordinates. The IK problem is posed as a distance minimization problem i.e. distance between the current location and target location of the end effector, given in Eq. 3.21. The forward kinematic relationship gives the current end-effector location as a function of joint variables. The robot with 5 DOF has five unknown variables and two equations corresponding to the coordinates of the target location in the workspace. In a given path, ten points were chosen and the optimization problem is solved for each point to determine the joint configuration to reach a specific point. The red circle in Fig. 7.2 represents the boundary of the workspace. For this simulation, the home position of robot links is assumed as zero degrees with X-axis. Fig. 7.2 shows the joint configurations while the end-effector is tracing a path. Table 7.2 shows the angles of the links for each robotic configuration.

The values of the positional error in Table 7.2 infers that the end-effector is reaching the desired TSL accurately. The number of iterations required to minimize the objective function value is 20 as shown in Fig. 7.4 (a). The computational time for the IK solution of a robot for

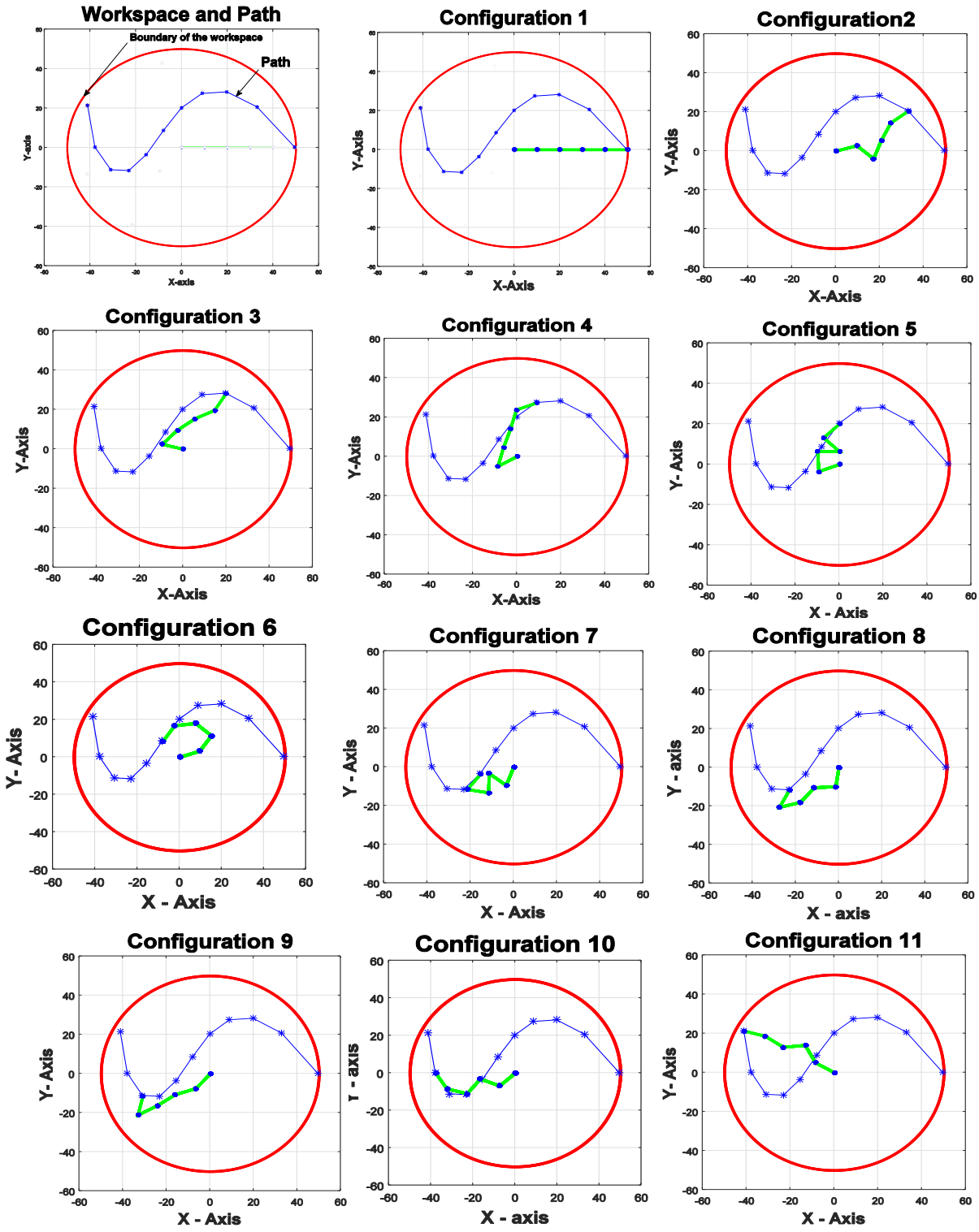


Fig. 7.2. A 5-linked robot configurations without obstacles in the workspace.

all the TSL's on the path is 32.542 seconds, performed on a PC with Intel Xeon E5 processor @3.50 GHz with 32 GB RAM.

Table 7.2. Joint configuration of 5 DOF robot traversing a path without obstacles in workspace.

S.NO	Path Coordinate (mm)	Angle of links in degrees					Positional Error (mm)
		θ_1	θ_2	θ_3	θ_4	θ_5	
1	(33,21)	4.82	325.07	62.64	73.30	42.19	3.10×10^{-06}
2	(20,28)	159.52	46.36	21.38	31.94	60.01	2.38×10^{-06}
3	(9,27)	220.66	81.06	69.19	85.54	27.82	5.33×10^{-08}
4	(0,20)	198.87	87.08	175.58	139.60	45.46	2.48×10^{-06}
5	(-8,9)	8.568	40.21	137.31	178.40	221.37	3.30×10^{-06}
6	(-16,-4)	256.30	139.10	284.56	175.31	54.84	4.01×10^{-06}
7	(-24,-13)	258.12	183.81	231.63	194.48	62.16	5.51×10^{-07}
8	(-31,-12)	235.13	202.17	214.22	220.49	77.26	1.17×10^{-06}
9	(-38,0)	246.73	158.72	225.47	160.48	140.79	1.74×10^{-06}
10	(-44,-22)	141.02	124.07	188.25	154.26	166.70	2.72×10^{-06}

An IK simulation for 10 DOF robot is also performed without obstacles in the workspace while traversing the same path, this simulation is performed to observe the computational time taken to perform the IK problem with the increase in the number of DOF. Fig.7.3 shows two robot configurations corresponding to the task location on the path. Fig.7.4 (b) shows the number of iterations required to minimize the objective function value is 180. Computational time for the solution to this problem is 66.232 seconds.

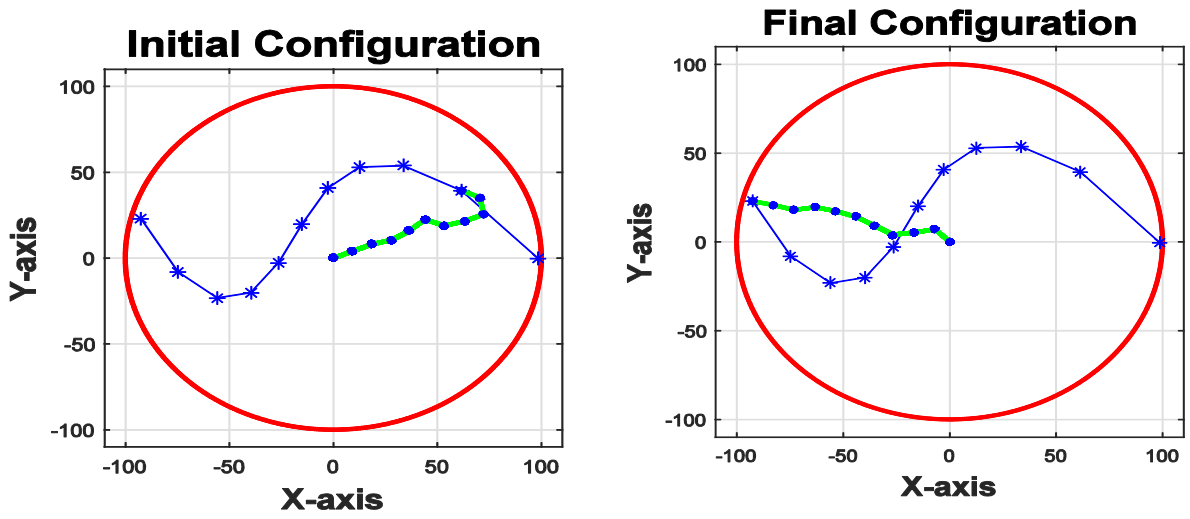
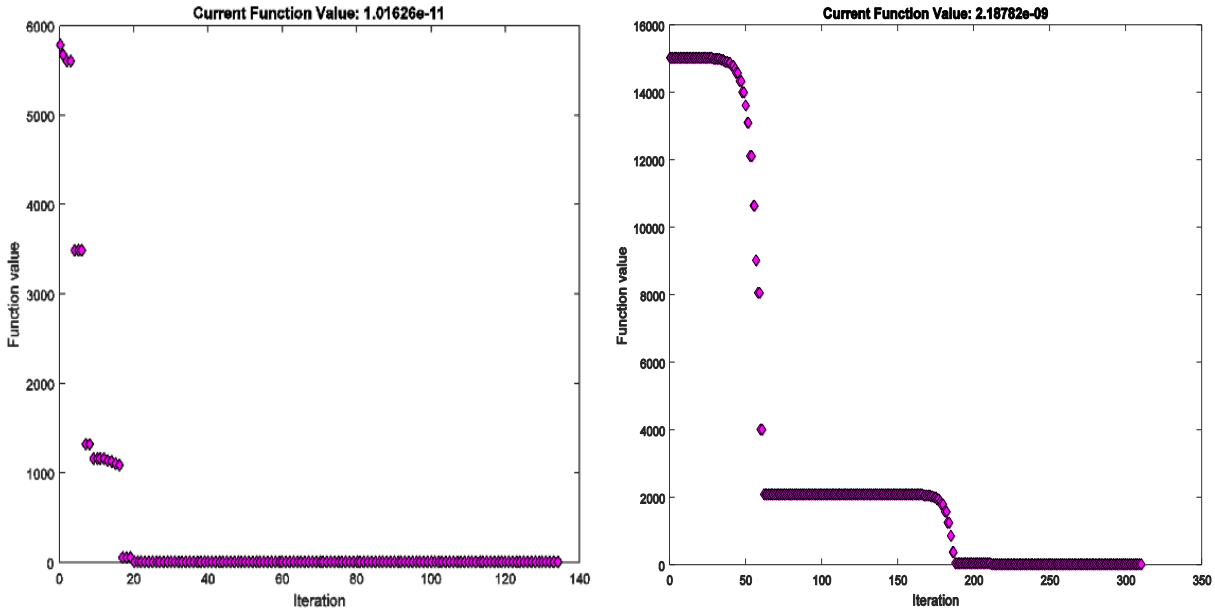


Fig.7.3. A 10-linked robot configuration without obstacles.



(a) (b)

Fig. 7.4. Convergence plot of IK solution without obstacles (a) 5 DOF robot (b) 10 DOF robot.

7.1.2. Polygonal obstacles in the workspace

A workspace with polygonal obstacles has been considered for the IK simulation of redundant robots. Initially, a 5 DOF robot is considered and two polygonal obstacles such as triangle and hexagon have been chosen. A curved path is taken, which is passing in between the obstacles. IK simulations are performed for the TSL chosen on the path. Fig. 7.5 shows the configurations of the robot while traversing a path avoiding obstacles in the workspace.

The task of collision detection is performed using the point-in-polygon technique discussed in section 5.1. Collision avoidance of redundant robot is implemented by adding a penalty value to the objective function given in Eq. 5.1. Further optimized to given IK solution of robot avoiding obstacles. The computational time for the solution to this problem is 62.323 seconds.

An IK simulation of 10 DOF robot avoiding obstacles have shown in Fig. 7.6. The computational time for the solution to this problem is 120.452 seconds. Fig.7.7 shows that the number of iterations for convergence of solution for 5 DOF robot is 120 whereas, for 10 DOF robot, it is 180.

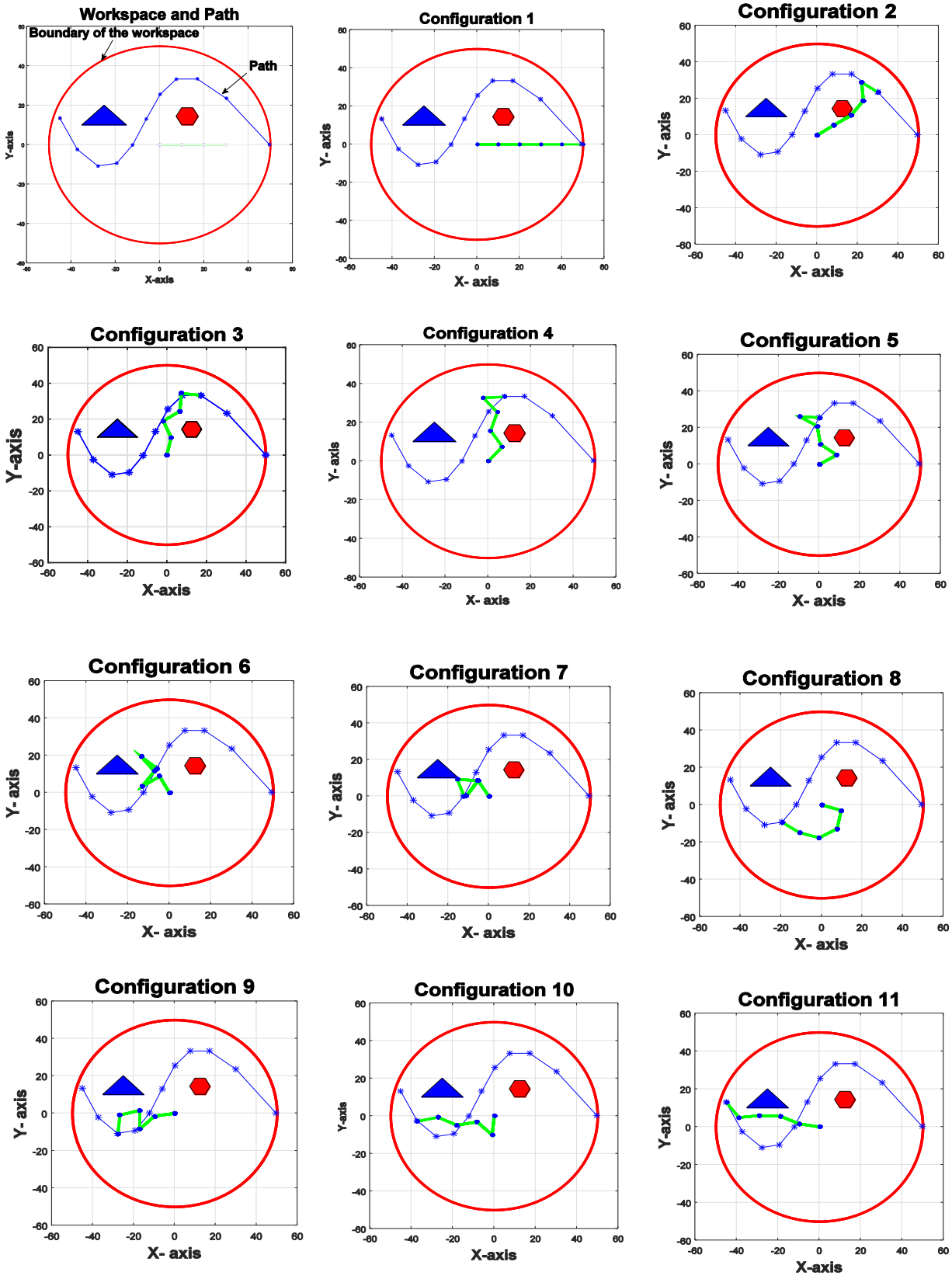


Fig. 7.5. A 5-linked robot configuration with two obstacles in workspace.

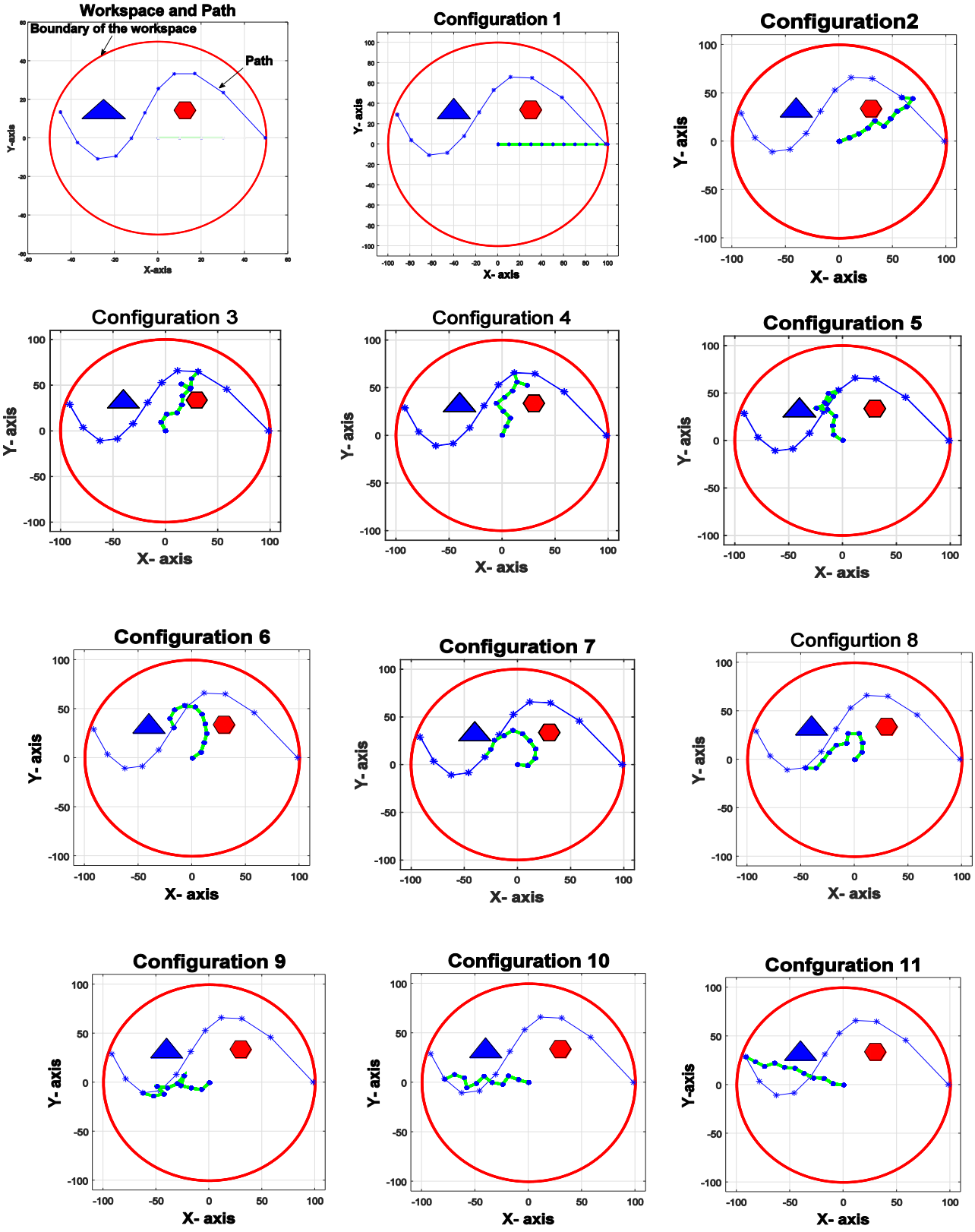
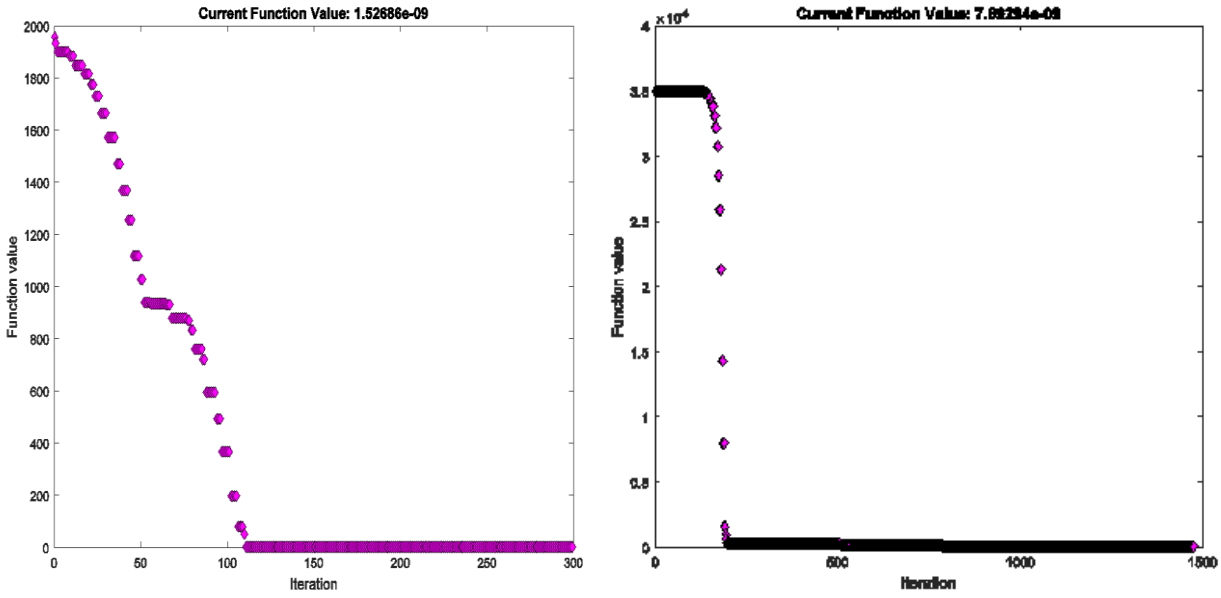


Fig. 7.6. A 10-linked robot configuration with two obstacles in workspace.

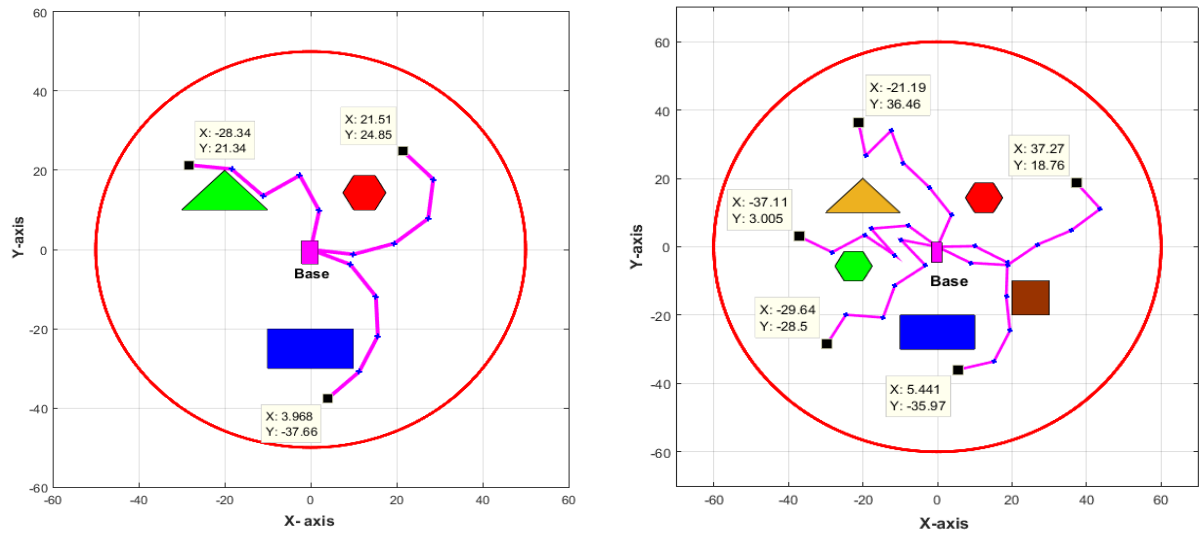


(a) (b)

Fig. 7.7. Convergence plot of IK solutions with polygonal obstacles (a) 5 DOF robot (b) 10 DOF robot.

7.1.3. IK Simulation with polygonal obstacles in the environment

Workspace with multi-shaped obstacles is modelled, which significantly reduces the free space and makes the workspace more complex. IK Simulations have been performed for planar redundant manipulator in these complex workspaces.



(a) (b)

Fig. 7.8. Robot configurations avoiding multi-shaped polygonal obstacles (a) 5 DOF manipulator (b) 6 DOF manipulator.

Fig. 7.8 (a) depicts the IK solution of 5 DOF robot at different task space locations avoiding three polygonal obstacles in the workspace. Fig. 7.8 (b) shows a 6 DOF robot avoiding five polygonal obstacles. For all the TSL's robot reached accurately without colliding obstacles.

7.1.4. IK Simulation with non-convex obstacles in the environment

Workspace with non-convex shaped obstacles has been considered for IK simulation of hyper-redundant manipulators shown in Fig. 7.9 (a-f). This case study focuses on robot configurations while avoiding E-shaped and C shaped obstacles.

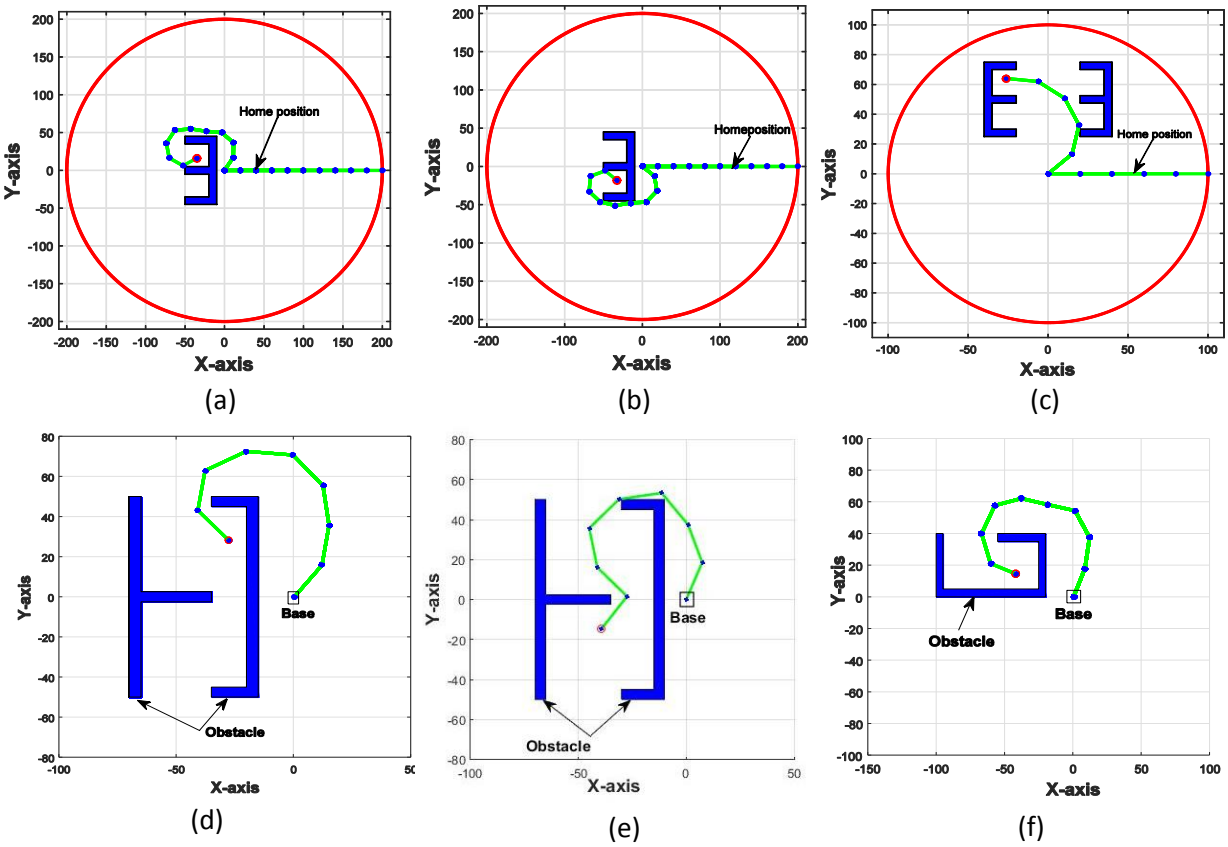


Fig. 7.9. Joint configurations of redundant robot avoiding non-convex obstacles.

This helps us to understand how accurately a robot is positioned in cluttered environments and in the narrow regions of the workspace. Fig. 7.9 (a-c) shows the robot configurations when the E-shaped obstacles are placed in different regions of the workspace. Fig. 7.9 (d-f) shows a different workspace with non-convex obstacles that are similar to a realistic cluttered environment.

These cases are difficult to solve than the previous ones because the obstacle is a non-convex object and target points are difficult to reach. It is observed that the solution may not be found always for all the target points with one initial guess itself. Hence, an automatic restart with a different initial guess is incorporated to handle such cases. Simulations have been performed for a different number of DOF robot varying from 6 to 10. It is observed in all the cases the end-effector reached the target location accurately without colliding the obstacles. The IK solutions of the redundant robot has been compared with different evolutionary based approaches. Kalra [96] proposed genetic algorithms for solving IK problem of industrial robots, in this work he reported that the number of generations needed for PUMA robot as 300 and the precision of the solution is 0.5 mm. Koker [55] proposed hybrid approach using neural networks and genetic algorithms to solve the IK solution of 6 DOF robot. This study has been made to improve the accuracy of solution. In above studies, the population size and number of generations results in high computational time for the solution. Increase in the number of DOF further increase of computational burden with these approaches. In the proposed work IK solution of hyper-redundant manipulator with 10 DOF working in complex workspace has been simulated. The computational time in cluttered workspaces is less than 120 seconds. The task of achieving multiple IK solution has also performed for spatial redundant robots, for which computational time is less than 60 seconds. Redundant robots are employed in different industrial and medical applications, where high accuracy and repeatability is required. Modern developments in manufacturing technologies such as semiconductor processing and assembly and precision material processing taken place. Micro assembly involves in joining of parts that have atleast one dimension less than 1 mm and that must be assembeled with micrometer accuracy. The accuracy of the solution with the proposed approach is 10^{-5} mm, satisfies the requirement of above applications. The accuracy of the solution is much better than the evolutionary based approaches

7.2. Redundancy resolution with joint distance minimization

Results demonstrate the redundancy resolution scheme of the serial redundant manipulator at the position level. In this case, a 5-DOF planar redundant robot of equal link lengths has been considered. Polygonal obstacles are chosen in the workspace, the robot has to trace the path in the workspace without colliding the obstacles.

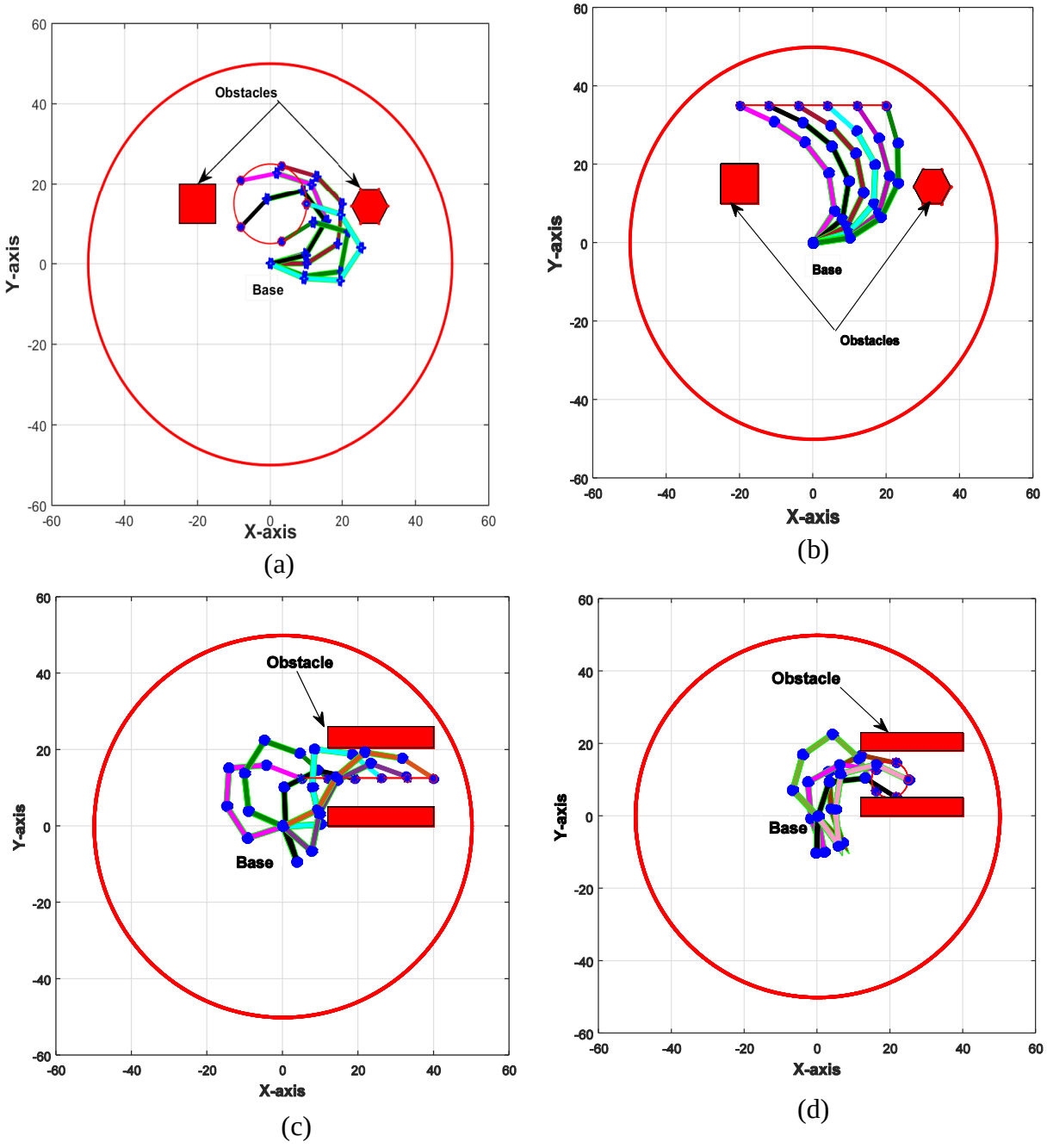


Fig. 7.10. Evaluation of joint configurations (a) Straight line path avoiding polygonal obstacles (b) Circular path avoiding polygonal obstacles (c) Straight line path in narrow passage (d) Circular path in narrow passages.

Here, the problem of redundancy resolution is carried out by minimizing the joint rotations while the robot is assumed to traverse a path in the task space. This was solved as a constrained optimization problem with the objective function, given in Eq. 4.32 and the task space constraints are shown in Eq. 4.33. The task of collision avoidance is implemented using a penalty approach, solved using Eq. 5.1. Results are reported by changing the position and shapes of the obstacles and path to be traced by the end-effector, to show the efficacy of the

proposed resolution scheme. In Fig. 7.10 (a-b), the robot is following a circular and straight-line path with polygonal obstacles in the workspace. Fig. 7.10 (c-d) shows the robot following the straight line and circular path accessing narrow passages. In each of these cases, the computational time with the proposed approach is about 30-60 seconds. Fig. 7.10 (c-d) illustrates that the optimization problem may not converge easily because the path in which the robot has to travel is very narrow and difficult to reach. This problem can be handled by restarting the optimization algorithm with different initial guesses. It is observed from all these cases that the solutions are resolved by minimizing the objective function while satisfying the constraints of reaching task space and avoiding collisions.

In the literatre [42], redundancy resolution has been performed for minimum norm solution at velocity domain, for which computation of Jacobian inverse is required. This increases the computational time with increase in number DOF of the robot and sensitive at singular positions. To reduce the computation cost associated with pseudo-inverse solution many researches proposed several methods for redundancy resolution. Kircanski [68] proposed combination of analytical and pseudo inverse solution. By adopting this approach the dimensions of Jacobian are reduced, which decreases computational complexity. Applying gradient projection method to this approach will further reduces computational burden. In the proposed approach redundancy resolution at position domain, this does not involve the computation of pseudo inverse of Jacobian. Classical optimization algorithm has been implemented for solving the redundancy resolution problem in narrow workspace. The computational time for this simulation is less than 60 seconds, which is less than Jacobian based methods and evolutionary approaches.

7.3. Singularity avoidance of planar redundant robots

Singularity avoidance is determined by choosing an objective of maximizing the manipulability measure of a robot, given in Eq. 4.35 with task space constraint (reaching the task space location), shown in Eq. 4.34. In this case, boundary singularity avoidance for the 5-DOF manipulator is considered. Fig. 7.11 shows singular configurations while the robot is tracking a straight-line path in the workspace, where the robot configurations are stretched. Fig. 7.12 shows the configurations that are away from singular points. Manipulability values are calculated for both cases to show how far the manipulator is away from singularities. Fig. 7.13 illustrates manipulability measure along with the points of the given path for both singular and non-singular configurations. The value of manipulability measure for singular

configuration is varying in the range of 580-860 on the corresponding points of the path. Whereas for the non-singular case it is about 738-1020. The percentage improvement of manipulability measure is about 27.8% for non-singular cases compared to singular configurations.

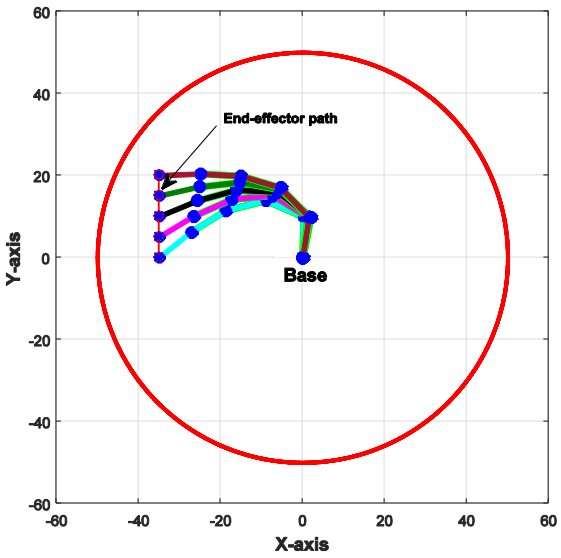


Fig.7.11. Robot configurations without singularity avoidance.

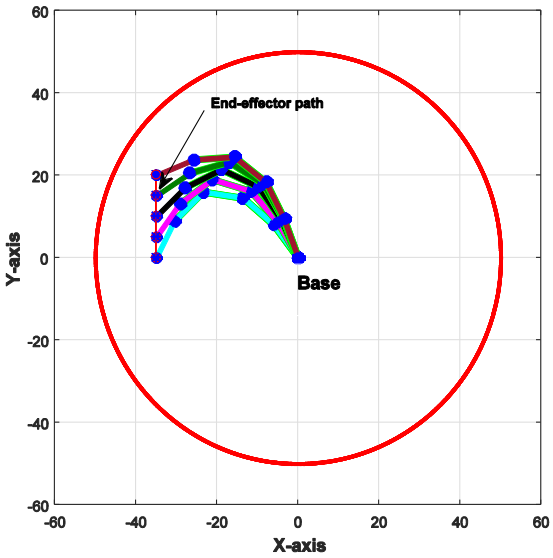


Fig. 7.12. Robot configurations with singularity avoidance.

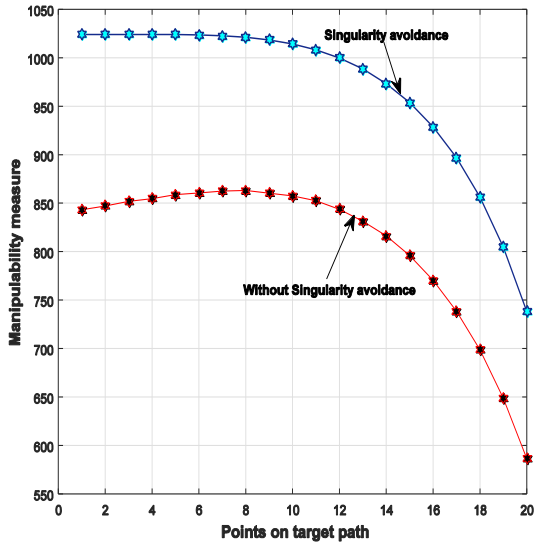


Fig. 7.13. Manipulability measure along desired path.

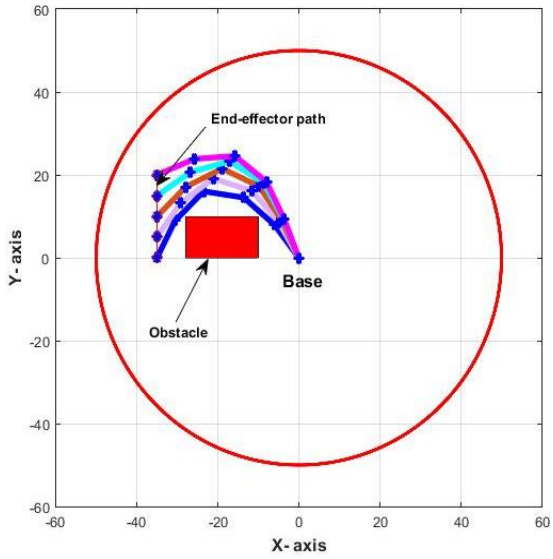


Fig.7.14. Robot configurations with obstacle and singularity avoidance.

The task of obstacle avoidance and singularity avoidance for a given path is also performed and Fig. 7.14 shows non-singular configurations of the robot while avoiding a rectangular obstacle. In the literature [12] singularity avoidance has been performed for planar

manipulators without obstacles in the environment. In the proposed approach singularity avoidance is implemented for redundant robots by considering obstacle avoidance as additional criterion. Existing methods in the literature present obstacle avoidance of redundant manipulators using pseudo-inverse techniques [64]. These methods suffer from sensitivities at the singular configuration and are known to be computationally expensive. Popularly used method for obstacle avoidance is an artificial potential method [88], this technique known to be face difficulty with local minimum. To check the computational efficiency and accuracy of IK solution, redundancy resolution of planar robots has been implemented at velocity level using pseudo-inverse of Jacobian. The computational time of IK solution using pseudo-inverse technique is high and it is twice the time when compared with IK solution performed at position level. The accuracy of the IK solution performed at joint position level is found to be better than the solution at velocity level. As shown in the results, the proposed work handled collision avoidance in a 2-Dimensional workspace using the penalty approach. This approach is computationally fast due to its simple and effective obstacle modelling.

7.4. Optimum trajectory planning of redundant manipulator

A 3-degrees of freedom planar redundant manipulator has been considered for optimal trajectory planning of robot by minimizing total power consumption at the joints. Trajectories have been evaluated for 3DOF robot for two kinds of motion such as continuous and point-to-point motion. For continuous motion, the end effector is constrained to move in a straight line path while optimizing power consumption, which is shown in Fig.7.15. A non-linear constrained optimization algorithm has been used for achieving the required objective of minimum power consumption, given in Eq. 4.39. The constraints of the optimization problem are reaching the end-effector to the task space and accelerations of joints, shown in Eq. 4.40& 4.41. For the point-to-point motion simulation, the optimal trajectories have been evaluated by posing it as a non-linear constrained optimization problem with an objective shown in Eq. 4.39. The constraints of an optimization problem are the end-conditions of the manipulator such as starting point, goal point, start velocities, and end velocities. Fig. 7.16 shows the path traced by the end-effector by minimizing the objective while satisfying the constraints. For simulation of this case, the parameters of manipulator considered are mass (m_1=*1 kg* m_2=*2 kg* m_3=*1 kg*), link lengths, inertia of links. Joint trajectories are evaluated using a quintic polynomial equation, which ensures smooth joint velocities and accelerations.

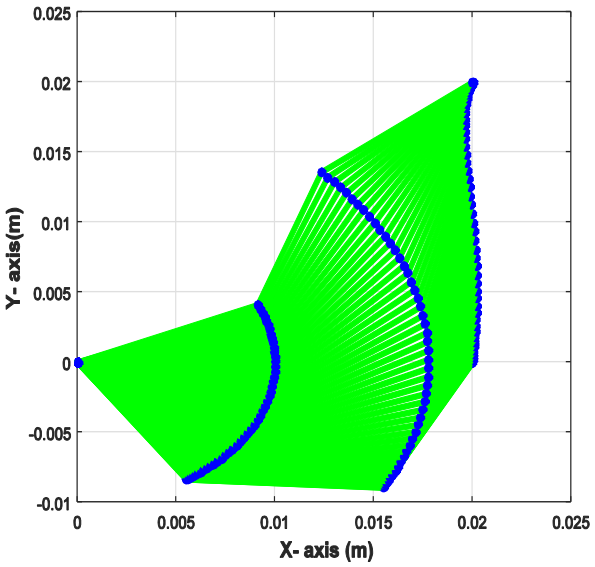


Fig.7.15. Joint configurations while traversing continuous path in Cartesian space.

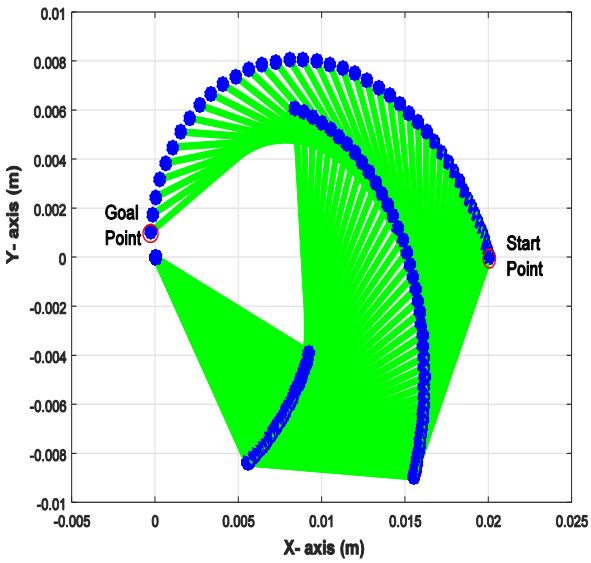
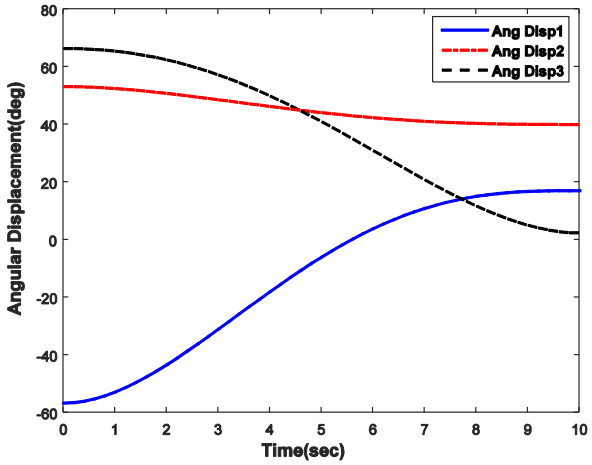
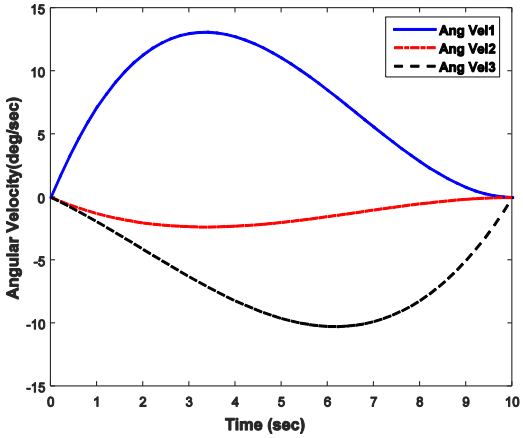


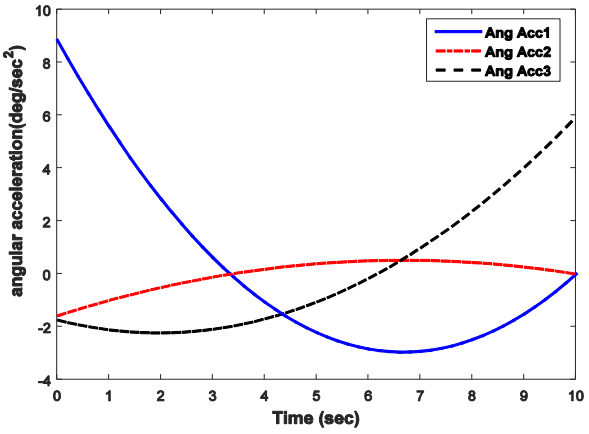
Fig. 7.16. Joint configurations while traversing point to point motion in Cartesian space.



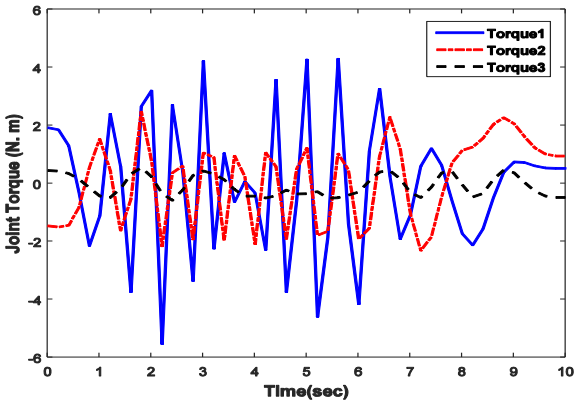
(a)



(b)



(c)



(d)

Fig. 7.17. Joint trajectories of 3DOF manipulator for continuous motion (a) Angular displacement (b) Angular velocity (c) Angular acceleration (d) Applied torques.

Fig. 7.17 (a) depicts the variation of joint displacement for a given time interval for continuous motion along a straight line. It was observed that the variation of angular displacement at joint1 is more compared to the other joints. Fig. 7.17 (b) shows the variation of angular velocity against the time, it is observed that joint velocities are satisfying end conditions and also ensuring smooth variation. Fig. 7.17 (c) depicts the variation of angular accelerations at respective joints. Fig. 7.17 (d) shows the variation of joint torques.

The joint trajectories are computed for point to point motion simulation of the redundant manipulator. Fig.7.18 shows the variation of angular displacement, velocity, accelerations, and applied torques at each joint Fig. 7.18 (a) depicts the variation of angular displacements

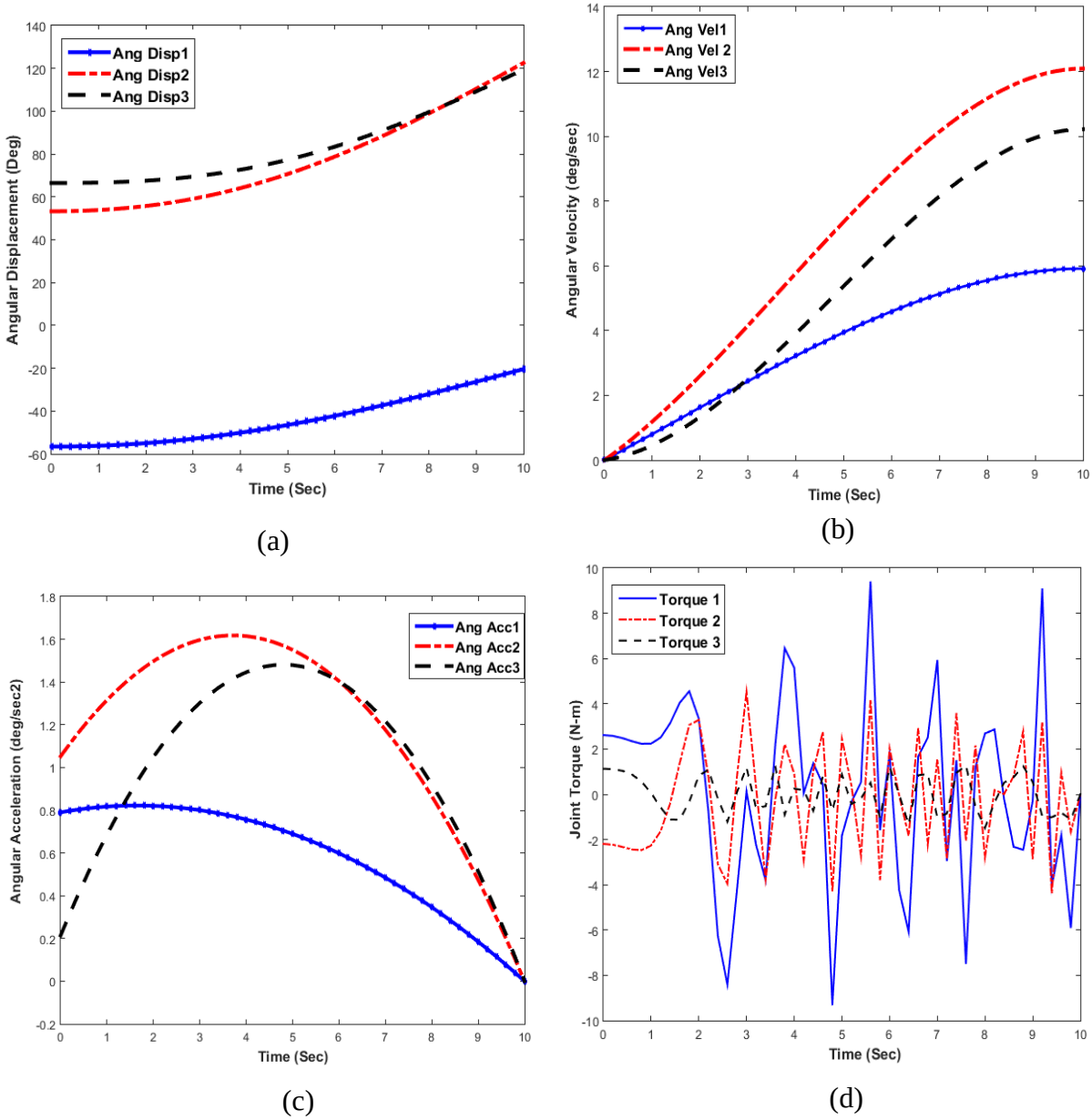


Fig. 7. 18. Joint trajectories of 3DOF manipulator for point to point motion (a) Angular displacement (b) Angular velocity (c) Angular acceleration (d) Applied torques.

at the individual joints. The variation of the joint displacement is uniform. The variation of angular velocities and accelerations are shown in Fig. 7.18 (b-c).

The variations are uniform and satisfying the end conditions of the robot. Joint torques applied at each joint are shown in Fig. 7.18(d). From Fig. 7.18 (d), the variation of torque applied at joint1 is more when compared with other joints. The maximum torque at joint1 is 9.5 N-m, which is higher than the joint torque of a continuous motion robot. In this simulation, 3 DOF planar robot has been presented. This approach can be extended for planar robots with more number of DOF.

In the above simulations redundant robots with different DOF that are suitable for a specific task has been choosen. The choice of selecting DOF for a robot depends on the following aspects

1. Geometry of the workspace
2. Geometry of the obstacles
3. Number of obstacles
4. Nature of the task to be accomplished.

The present work is not dwelling into the issue of selection of DOF for a specific task. The focus of the work is to analyze how the IK Solution methods perform while accomplishing different tasks. In this sense, different DOF of the robots that are appropriate for a particular task has been chosen to demonstrate the effectiveness of the method. Different simulations have been carried out for different tasks that are given in table 7.1.

7.5. Observations from the results

1. A general inverse kinematic method is proposed which is efficient and effective i.e., the method is capable of finding an inverse kinematic solution quickly (1-3 minutes for all the cases) for highly redundant planar robots even with obstacles in the workspace. Computational time for different performance measures are given below
   (a) Computational time for the task of redundancy resolution in complex workspace is 150 seconds.
   (b) Computational time for the task of singularity avoidance with obstacles is 120 seconds.
2. Efficiency is due to the simple and effective modeling techniques and classical optimization methods employed to solve the problem. The method can also be used

for real-time applications because of its efficiency.

3. The IK simulation has been performed in cluttered workspace. Results shows that there is no collision of joint configurations with the obstacles in all the cases and able to reach the task space precisely. Collision avoidance is tackled effectively using penalty approach. The task of collision detection is performed by modeling obstacles as polygons and robot links as lines.
4. A restart procedure with a different initial guess is included when the solution is not found in the first attempt itself. This makes sure that repeated attempts are made even if the classical optimization method fails to find the solution because of the multi-modality of the objective function.
5. Both convex and non-convex obstacles can be handled using the proposed method.
6. Redundancy resolution techniques have been implemented to determine the best joint configurations by satisfying the additional performance criterion.
7. The accuracy of the IK solution in all the cases is high, which is in the order of 5.3335×10^{-8} mm.

CHAPTER-VIII

8. Results of IK Solution of Spatial Redundant Manipulators

IK simulations of spatial 9 DOF robot have been performed in different workspaces such as un-constrained environment without obstacles and with multi-shaped 3D obstacles. A redundancy resolution scheme is also implemented with a different secondary criterion such as joint-distance minimization and singularity avoidance. Simulations of a hyper-redundant robot are performed in a realistic environment similar to plant-layout, pipe-layout inspection, and work facility layout. Different cases of IK simulation and redundancy resolution of spatial hyper-redundant robots working in various environments has been shown in the flow chart in Fig. 8.1. Results of the following simulations have been presented in subsequent sections.

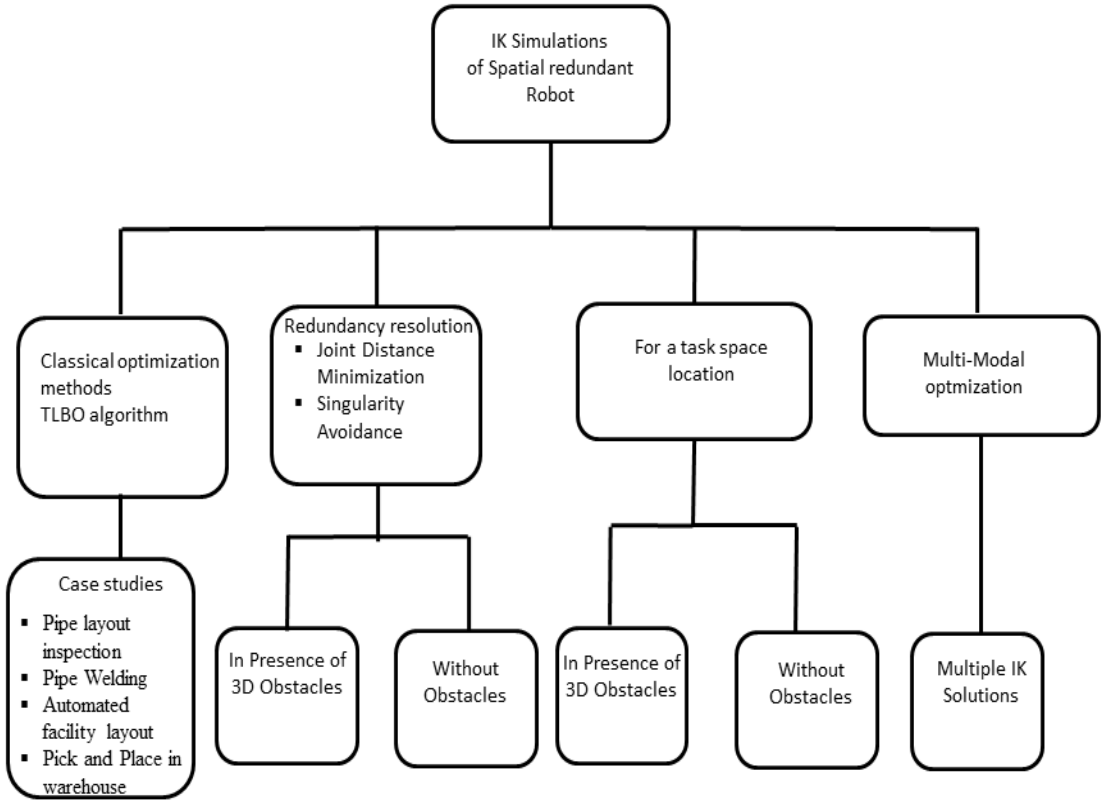


Fig.8.1. Flow chart representing the IK simulations and redundancy resolution of spatial redundant robots

8.1. IK Simulations without obstacles in the workspace

The inverse kinematics of spatial hyper-redundant manipulator is determined for a 9-DOF manipulator at different task space locations (TSL). The IK problem is posed as an optimization problem with the objective as minimization of total Euclidean distance shown in Eq. 3.20, which is a non-linear problem without constraints. The IK solution is computed using *fminsearch* function of MATLAB. This function works based on the Nelder and Mead simplex algorithm mentioned in section 6.1. Table 8.1 shows joint configurations of the 9-DOF robot for a set of six TSL's. The forward kinematic model of the end effector (given in Eq. 3.17) is a function of nine joint variables that are to be determined for each TSL specified in three-dimensional coordinates.

Table 8.1. Joint configurations corresponding to task space locations.

S. No	Task Space Location	Joint Configuration (in degrees)									Positional Error
		θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	
1	(18, 18, 20)	-23.69	-17.01	-64.86	10.12	100.68	3.53	53.67	43.87	49.54	4.23e-10
2	(28, 28, 20)	42.20	-25.64	63.96	261.04	-84.29	-40.09	-5.61	-15.03	-113.1	6.16e-11
3	(38, 35, 25)	-63.29	-10.33	39.40	89.77	77.20	43.95	4.84	-89.56	19.27	7.39e-11
4	(-18, -18,-20)	-27.16	-107.0	-4.96	80.52	-45.78	-43.66	-83.03	-114.28	-101.1	6.78e-11
5	(-28, -28, 20)	-17.57	74.0	-72.33	16.89	-20.60	64.60	30.46	-19.58	-136.5	6.22e-10
6	(-38, -35, 25)	22.13	32.43	-96.91	23.98	-33.32	38.20	-52.29	46.21	20.45	8.08e-10

Fig. 8.2 (a-d) shows the joint configurations of 9-DOF robot at TSL of (18, 18, 20), (38, 35, 25), (-28, -28, 20) and (-38, -35, 25). The positional error obtained after the convergence is about 7.39e-11. The values of positional error at different task space coordinates are reported in Table 8.1, which ensures the accuracy of the end-effector.

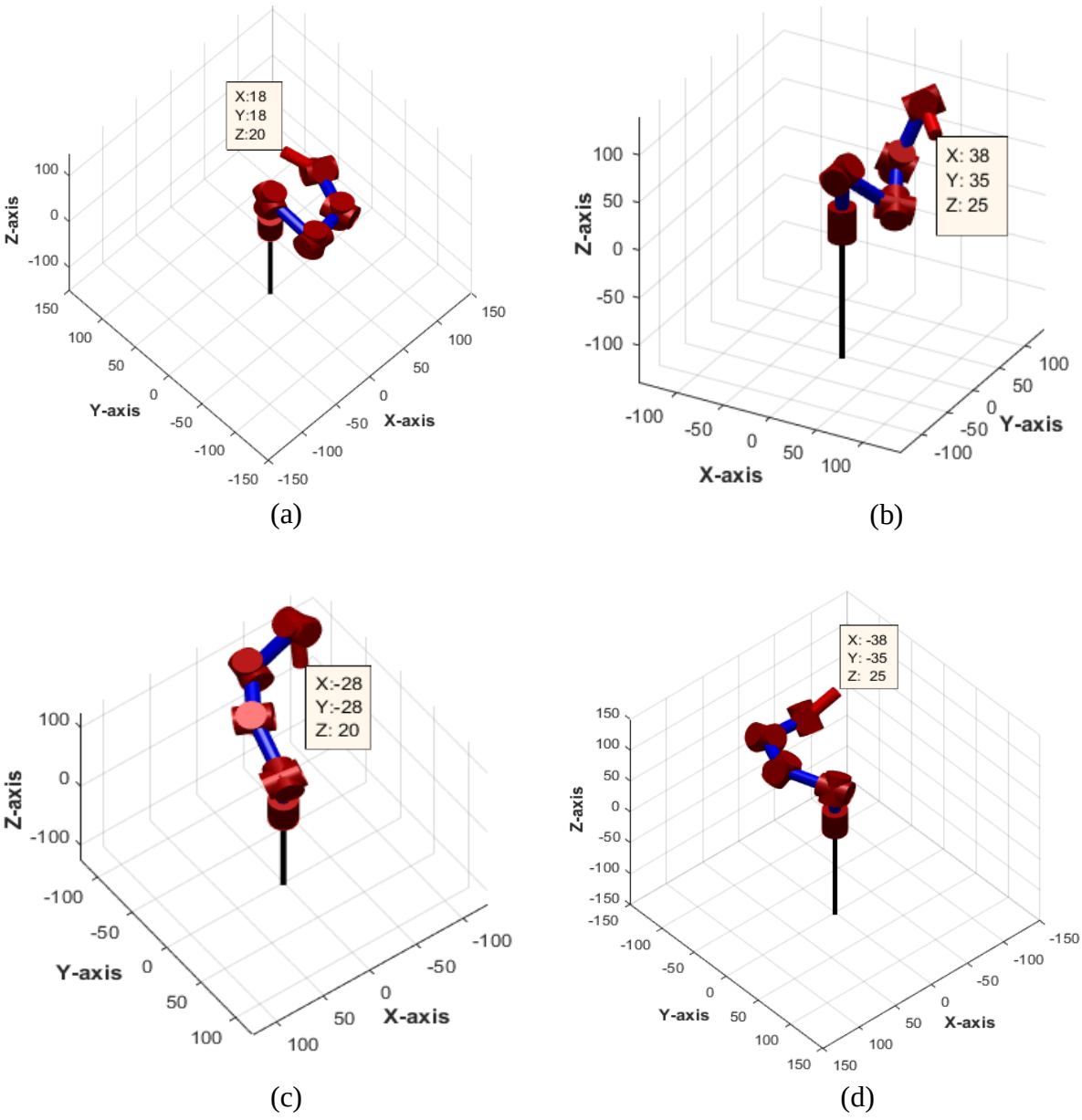


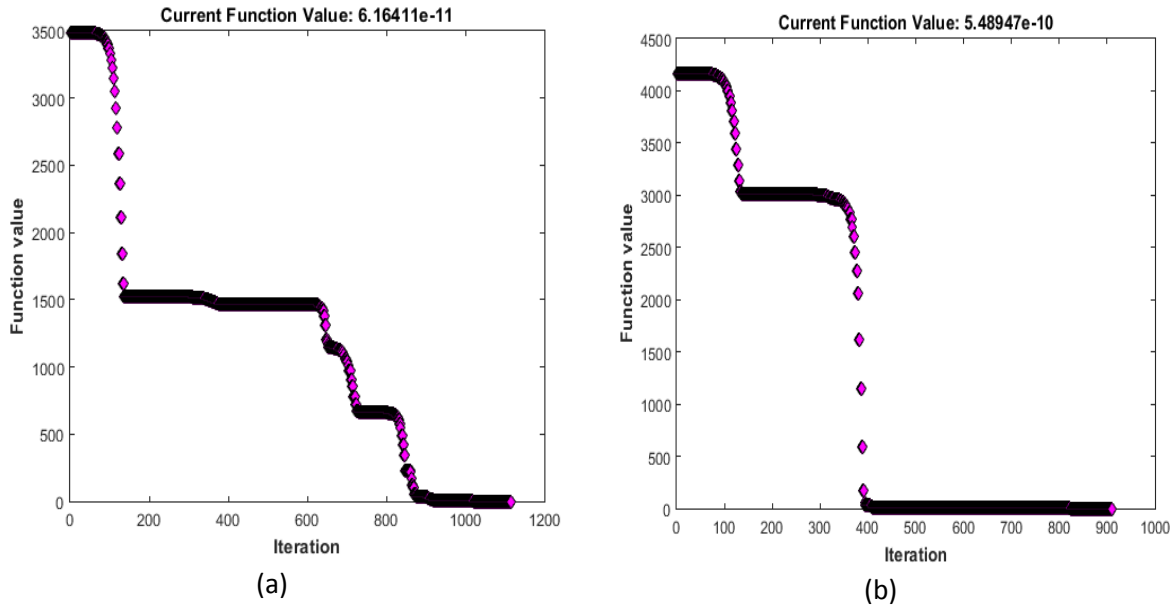
Fig. 8.2. IK Solutions of Spatial Hyper-redundant manipulator. (a) Joint configuration for target point(18, 18, 20) (b) Joint configuration for target point(20, 25, 25) (c) Joint configuration for target point(-28, -28, 20) (d) Joint configuration for target point(-38, -35, 25).

The positional error obtained has been compared with the existing literature [96, 97], and it was observed that the order of positional error is negligible (10^{-11} times) shown in Table 8.2. From the values of positional error, it was observed that the end-effector is precisely located at desired task space locations.

Table 8.2. Results of positional error comparing between a redundant robot [96, 97] and 9-DOF spatial hyper redundant robot.

Type of the Robot	SCARA[88]	7 DOF spatial robot [89]	9-DOF spatial hyper redundant robot.
Positional Error (mm)	0.69	0.002	4.23e-10
	0.50	0.003	6.16e-11
	0.45	0.004	7.39 x10^{-11}

The convergence of function value during the optimization process is shown in Fig. 8.3. The number of iterations required to attain the minimum function value for a task coordinate of (18, 18, 20) mm is at 900, which is shown in Fig. 8.3(a). From Fig. 8.3 (b) it is observed that the convergence of function is achieved after 400 iterations for a task coordinate (38, 35, 25). From the rate of convergence, it is inferred that the computational time for this simulation is less, and it is about 10 seconds.



(a) (b)

Fig. 8.3. Function plots. (a) Convergence plot for target point(18, 18, 20) (b) Convergence plot for target point (38, 35, 25).

8.2. Multiple IK solutions using multi-modal optimization

Multiple IK solutions of spatial redundant robots for a given TSL are computed using a global search optimization algorithm, presented in section 6.2. The objective function for this simulation is similar to the function chosen for a general IK procedure. The multi-start framework has been implemented to determine the local optimum solutions for the problem. The optimization procedure starts with the generation of start points in search space, and a non-linear optimization solver was executed at each start point to compute optimal solutions. The start points and basin of attractions are visualized through basins of attraction shown in Fig. 8.4. Start points at which the optimization algorithm executes are represented as dots, and basins of attractions are depicted as stars in Fig. 8.4. The dots that are closer to the basins of attraction represent the required local optimal solutions. The global optimization process has been implemented by considering ten start points initially. Fig. 8.5 (a-f) shows six different kinematic configurations of the same TSL (18, 18, 20) for ten optimal solutions. From Fig. 8.5 (a-f), it was observed that all kinematic configurations are distinct, and these can be a suitable candidate solution for reconfigurable spatial redundant manipulators.

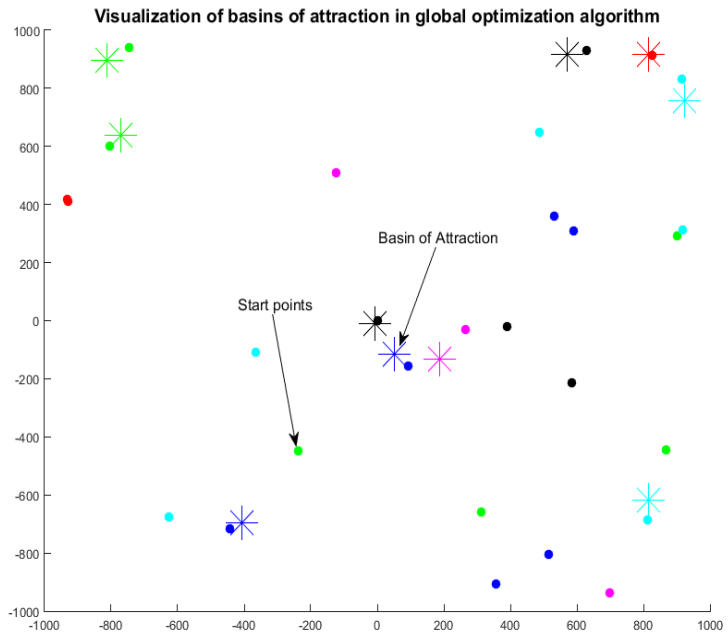
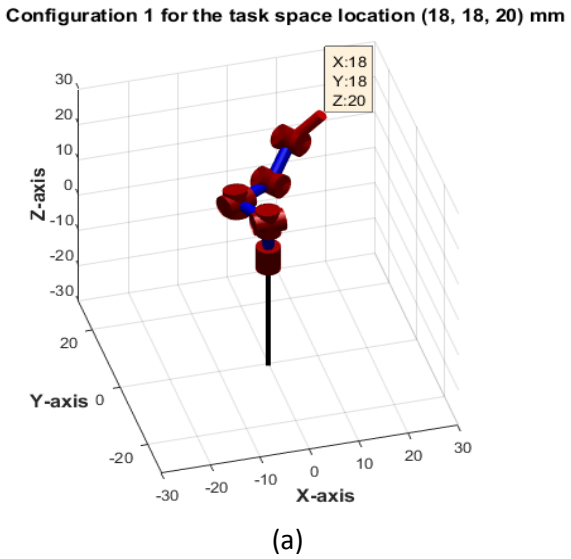
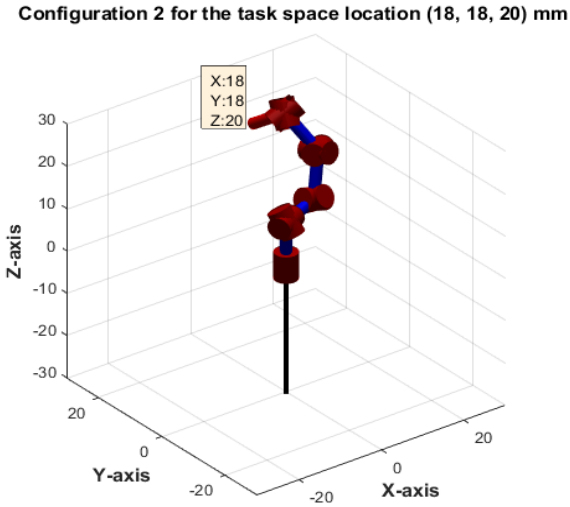


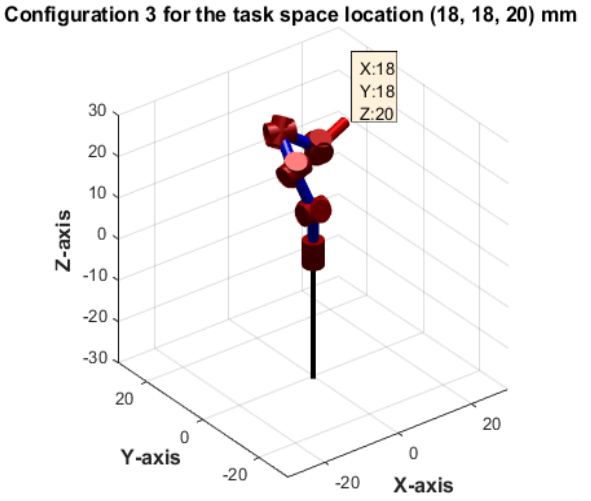
Fig. 8.4. Visualization of basins of attraction and multiple solutions for a task location (18, 18, 20).



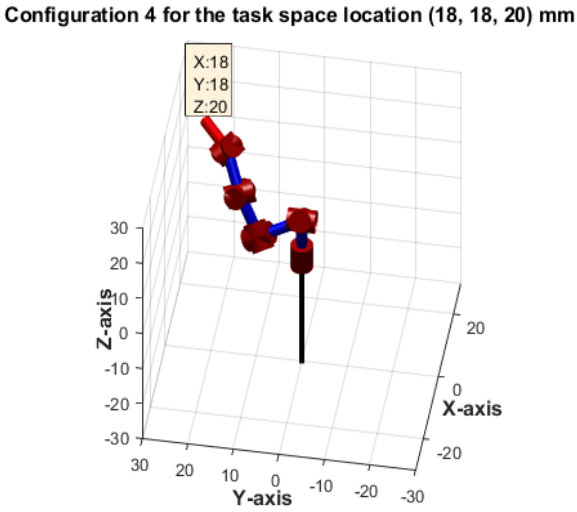
(a)



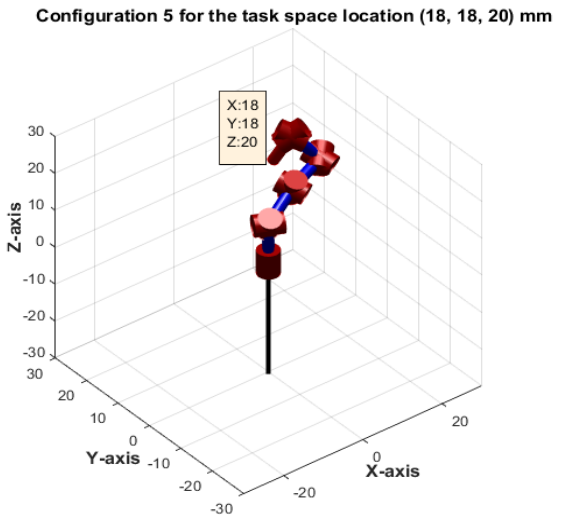
(b)



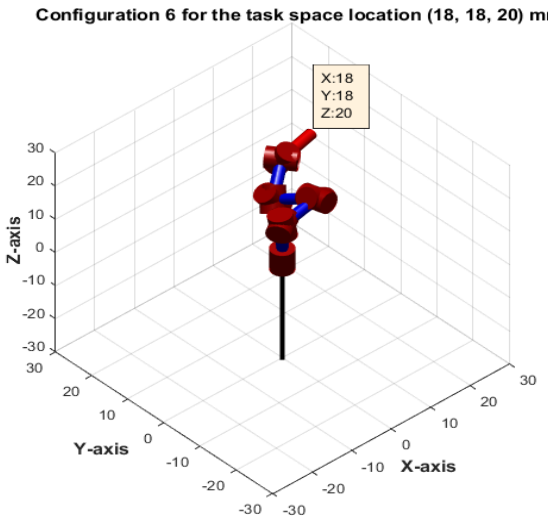
(c)



(d)



(e)



(f)

Fig. 8.5 (a-f). Multiple kinematic configurations of 9- DOF robot at end-effector position (18, 18, 20).

Here, ten feasible kinematic configurations are obtained as optimal solutions. The computational time for attaining multiple IK solutions, i.e., ten joint configurations of a redundant manipulator, is about 45 seconds.

To show the efficacy of the approach, this algorithm has been executed with a different number of start points. Fig. 8.6 (a-d) show the minimized objective function values of multiple solutions

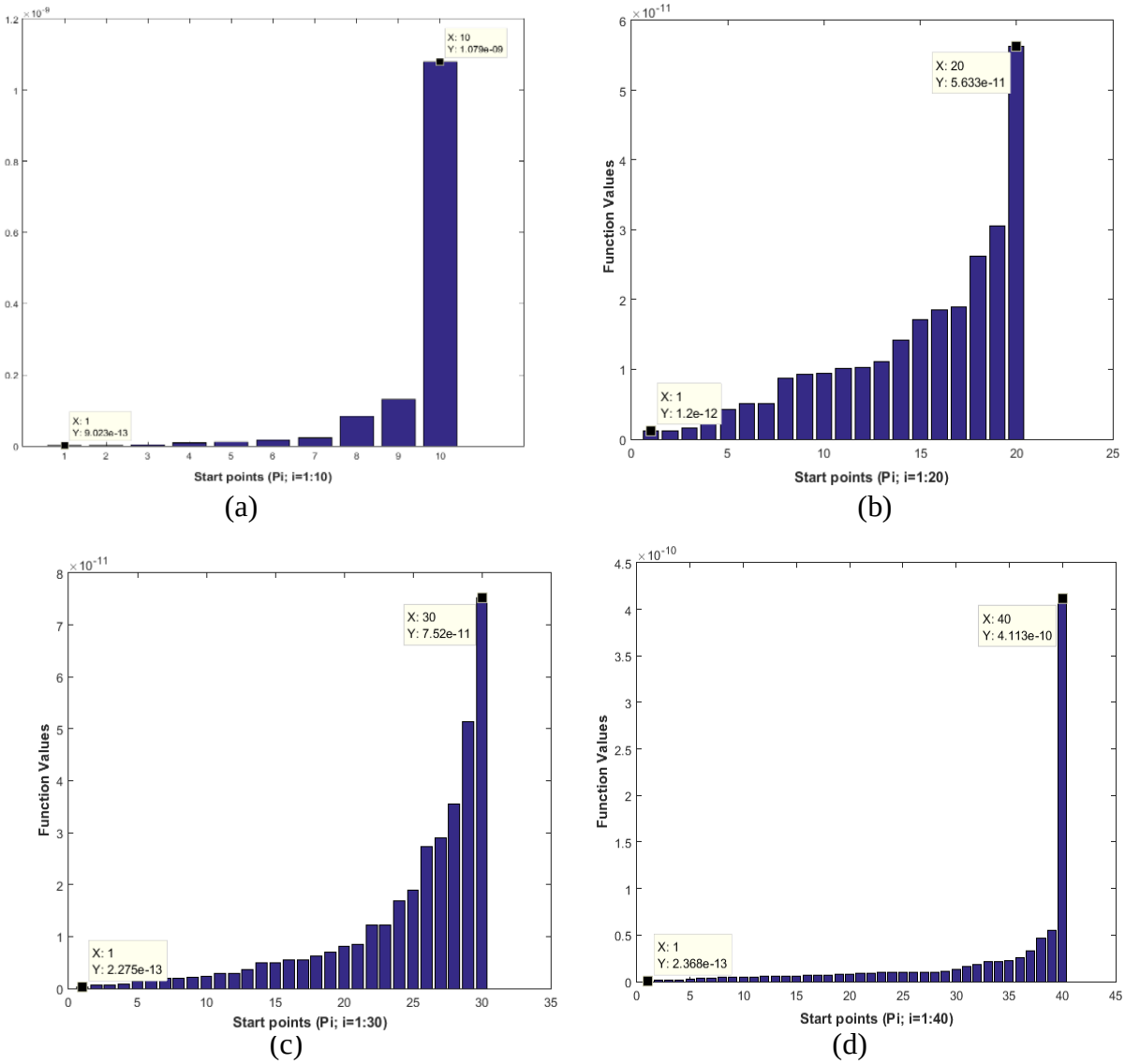


Fig. 8.6 (a-d). Plot showing converged function values with different number of start points.

for a range of 10- 40 start points. It was observed that the least positional error is achieved with 40 start points, and it is in the order of 2.36e-13. From Fig. 8.6 (a-d), it was also

observed that the magnitude of positional error was less at different start points range. The solutions are consistent even for less number of start points.

In the existing literature [96, 97], multiple IK solutions have been computed using niching based approaches, they require more algorithmic control parameters. For the same problem, using the proposed approach solutions have converged quickly even with ten start points. The global optimization techniques [97] require a large population size to obtain multiple joint configurations. In the multi-modal approach of industrial robots [96], a maximum of four different kinematic configurations have been obtained with a population size of 50. From the size of the population and the number of iterations required to converge a solution, it is known that the evolutionary algorithms are computationally expensive. The proposed global optimization algorithm can achieve ten distinct kinematic configurations for a 9 DOF spatial robot in less time. As shown in Fig. 8.6, the number of distinct configurations can be increased by increasing the number of start points.

Table 8.3. Multiple kinematic configurations corresponding to the task space coordinate (18, 18, 20).

S. No	Task Space Coordinate (in mm)	Joint Configuration (in degrees)									Positional Error (in mm)
		θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	
1	(18, 18, 20)	-268.04	-5.78	740.7	-546.1	-100.4	-403.1	-3.06	360.02	-310.1	9.023e-13
2	(18, 18, 20)	301.13	8.85	104.97	-680.9	-756.1	-391.9	113.14	-211.6	296.41	1.55e-12
3	(18, 18, 20)	476.55	-31.048	622.50	-116.6	-159.6	166.92	577.33	81.96	858.93	2.462e-12
4	(18, 18, 20)	-807.8	-67.03	192.53	-404.8	-747.6	480.14	-827.6	-566.3	575.05	9.37e-12
5	(18, 18, 20)	544.99	678.83	260.47	-571.4	-481.76	-128.7	-526.0	-248.6	-730.6	1.03e-11
6	(18, 18, 20)	-383.3	558.38	-274.0	-377.9	807.81	-479.6	-490.4	84.97	781.19	1.74e-11
7	(18, 18, 20)	359.32	-127.1	-296.4	649.42	799.88	-549.4	878.67	-24.502	-765.2	2.16e-11
8	(18, 18, 20)	-834.7	690.71	139.19	-191.3	-173.47	84.70	-136.5	963.91	421.78	8.27e-11
9	(18, 18, 20)	-721.8	-415.0	832.9	-628.4	1020.7	-792.9	444.98	238.99	-488.9	1.30e-10
10	(18, 18, 20)	122.74	890.86	294.89	-590.8	-429.16	181.93	-815.3	910.6	-925.4	1.07e-09

Table 8.3 shows ten distinct joint configurations corresponding to the TSL (18, 18, 20). The positional error is also reported in Table 8.3, which is almost zero and ensures that the end-effector reaches the desired target precisely. From the IK solutions in Table 8.3, it was

observed that the solutions were not repetitive. The function value of the local optimum IK solution has been compared with the global optimum solution to check if the function value of a local solution is close to the global solution. The global solution can be obtained using a global search algorithm described in section 6.2.3. This was implemented for the redundant robot at different TSL.

Start points in the global search algorithm were generated using a scatter search mechanism. Local optimization solver runs based on the score of the start points, and this feature enables the algorithm to arrive at the global optimal solution without trapping at the local optimal point. Table 8.4 shows the joint configurations and corresponding global minimum function values at different task space coordinates. From the values of positional error in Table 8.4, it was observed that the magnitude of the converged objective function is almost the same as the positional error magnitudes reported in Tables 8.1& 8.3.

Table 8.4: Joint configurations corresponding to a global minimum at different task space locations.

S. No	Task Space Location	Joint Configuration (in degrees)									Positional Error
		θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	
1	(18, 18, 20)	-239.05	0	-239.05	0	183.73	0	386.28	0	260.72	1.529e-13
2	(28, 28, 20)	-18.333	0	-18.333	0	48.447	0	66.641	0	47.152	1.801e-13
3	(38, 35,25)	-13.515	-0.321	-10.035	0.626	39.576	10.441	53.674	11.184	36.676	1.950e-13
4	(-18, -18, 20)	-47.67	-121.9	-11.07	-114.45	-72.44	-108.7	-71.70	-52.96	-40.06	6.711e-13
5	(-28, -28, 20)	-24.74	0	-24.74	0	-53.32	0	-65.87	0	-43.40	8.975e-14
6	(-38, -35, 25)	-72.46	49.945	-57.04	59.662	-157.07	58.02	-180.36	-3.564	-31.59	3.188e-12

8.3. IK Solutions in a 3D cluttered environment

This section presents the results of different IK solutions of the robot operating in the cluttered environment. A wide variety of 3D working environments are considered to simulate the robots working in real-time applications. For this simulation, a redundant robot with five links and 9 DOF is considered. Each joint of the robot is modelled with 2 DOF (universal joints), which allows the robot to move easily in a cluttered environment. The kinematic modelling of this robot is described in section 3.4.2. All the links are of uniform length and are taken as 20 units. The IK solutions of the robot are computed using

optimization techniques discussed in section 6.1. The collision detection scheme is implemented using the bounding box technique, discussed in section 5.2. Collision avoidance of robots with different types of 3D obstacles was implemented using the penalty approach. The modified equation of objective function with a penalty is given by Eq. 5.2. To determine the collision of a specific link with any of the obstacle, the links of the robot has been modelled as line segments, which are discretized as a series of points. A set of points are chosen uniformly along with the link so that all portions on the link are considered for collision avoidance. Fig. 8.7 (a) shows the joint configuration of the robot for a TSL (14, 14, 14) with three spherical obstacles in the workspace. While Fig. 8.7 (b) depicts the IK solution of the robot with four spherical obstacles. The end-effector of the robot reached the task space location accurately, and the positional error is negligible.

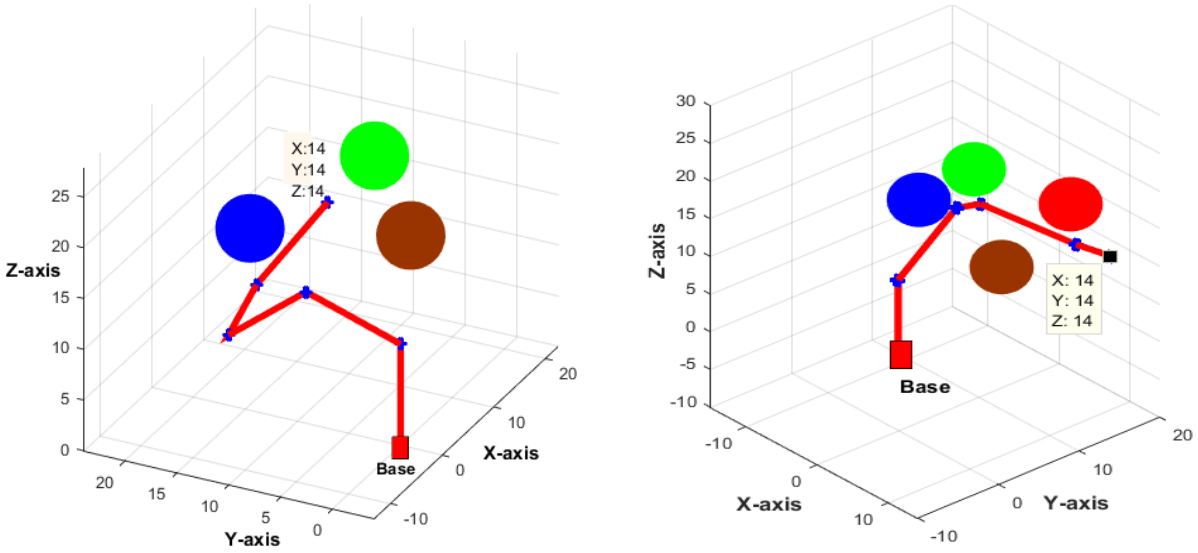
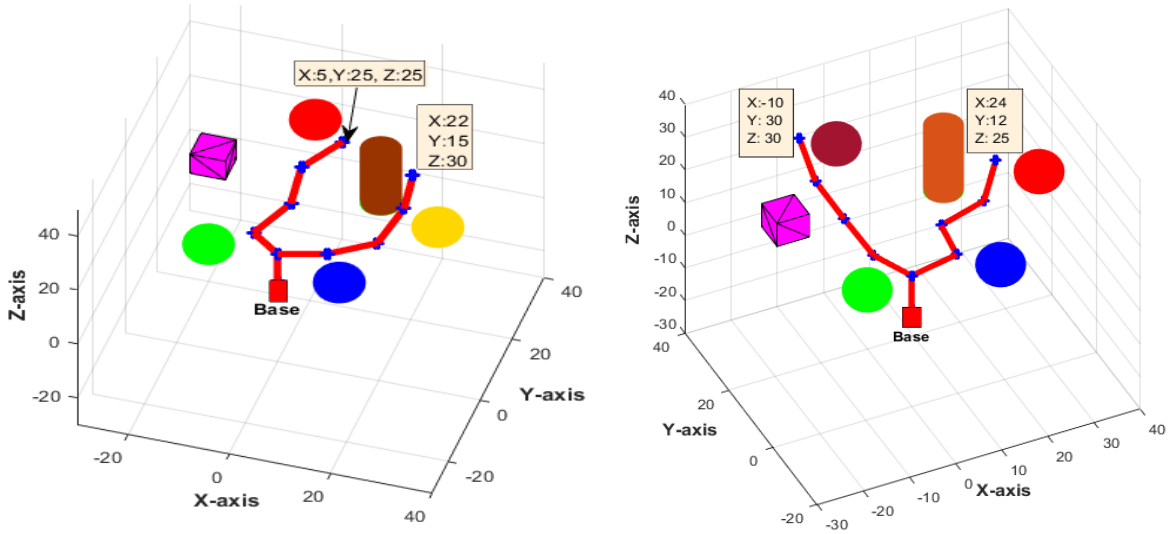


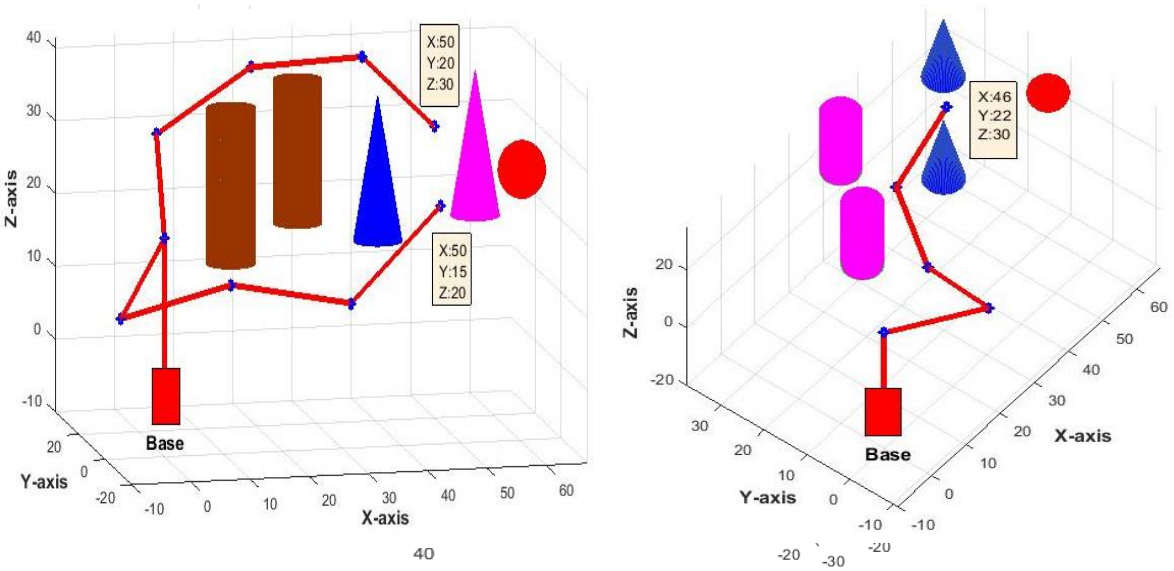
Fig. 8.7. IK Solution of spatial redundant robot with spherical obstacles (a) 3 spherical obstacles (b) 4 spherical obstacles.

Robot configurations at different TSL while avoiding 3D obstacles of different shapes in the workspace are shown in Fig. 8.8. IK solution of robot at TSL (5, 25, 25) and (22, 15, 30) shown in Fig. 8.8 (a), while Fig. 8.8 (b) shows IK solution at task space locations (-10, 30, 30) and (24, 12, 25). A workspace with cylindrical and conical obstacles has been considered, and kinematic configurations of the robot are shown while avoiding these obstacles. Fig. 8.9 (a) shows the joint configurations of the robot at task space locations (50, 20, 30) and (50, 15, 20), while Fig. 8.9 (b) shows the configuration at the TSL (46, 22, 30).



(a) (b)

Fig. 8.8. IK solution of spatial redundant robot avoiding multi shaped obstacles (a) At task space location (5, 5, 25), (22, 15, 30) (b) At task space location (-10, 30, 30), (24, 12, 25).

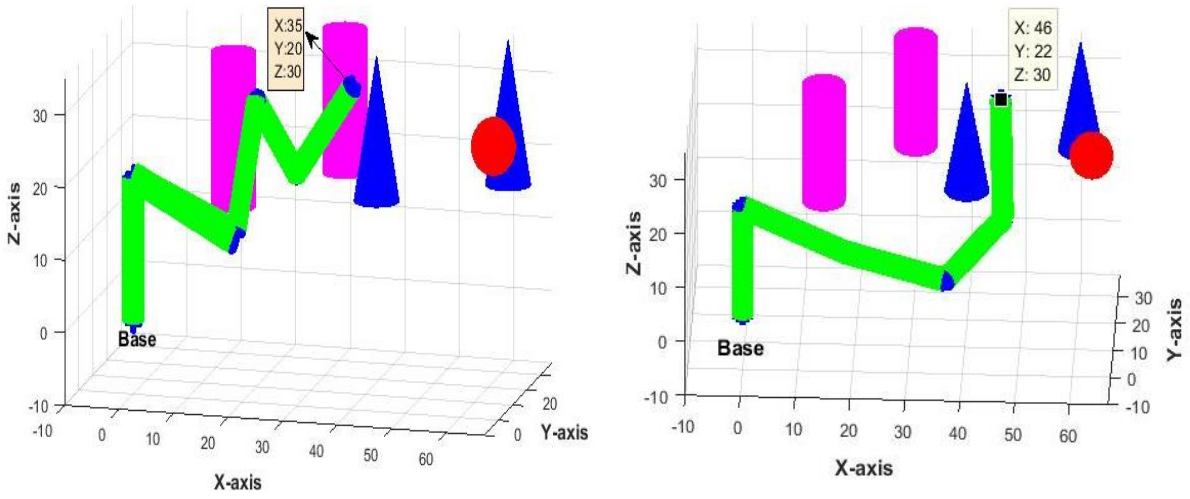


(a) (b)

Fig. 8.9. IK Solution of spatial redundant robot avoiding cylindrical and conical obstacles (a) At task space location (50, 20, 30), (50, 15, 20) (b) At task space location (46, 22, 30).

To show the realistic robot model, the links of the robot are considered as cylinders. The collision detection scheme in section 5.2 has been extended for cylindrical links. The axis of the cylinder is considered along the length of the link. The link length is discretized intopoints and a series of circles are generated along the axis by choosing the points on the axis as the center of the circles. The coordinates on the circle are represented as points on the surface of the cylindrical link. Collision detection has been performed by checking whether

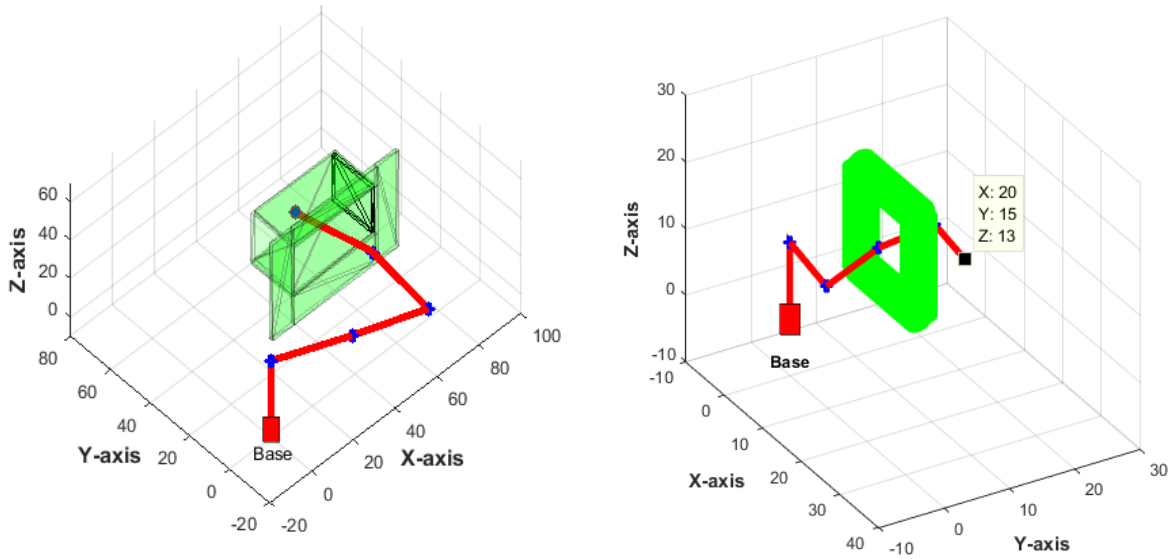
the points on the surface of the cylinder lie within the bounding box, which is implemented using an algorithm in Table 5.1. Fig. 8.10 shows the joint configurations of the robot, whose links were modelled as cylinders at task space locations (35, 20, 30) and (46, 22, 30).



(a) (b)

Fig. 8.10. IK Solution of spatial redundant robot avoiding cylindrical and conical obstacles links modelled as cylinders (a) At task space location (35, 20, 30) (b) At task space location (46, 22, 30).

Narrow passages similar to ducts and frame cut-outs were modelled in the workspace, and the task coordinates are chosen in a way that robot enters through those passages.



(a) (b)

Fig. 8.11. IK Solution of spatial redundant robot avoiding obstacles (a) With a duct shaped object as obstacle (b) Rectangular frame as obstacle at task space location at task space location (20 15 13).

Fig. 8.11 (a) shows the robot configuration while passing through the duct-like passage and reached the target location inside the duct without colliding with the boundary of the duct. Fig. 8.11 (b) shows the IK solution of the robot at the task location of (20, 15, 13) entering through a frame. The computational time for a single IK solution with multiple obstacles is less than 120 seconds.

The collision avoidance schemes proposed in the literature [24, 94] are implemented for different types of obstacles, but the obstacle modelling and collision detection techniques are quite complex. Collision detection in the 3D workspace has been implemented by enclosing the obstacles by bounding boxes. Due to the simplicity of the collision detection technique, this approach can be implemented for any shape of obstacles. As shown in the results, this approach can be easily adopted in real-time working environments. From the results of obstacle avoidance, in all the cases the manipulator able to reach the required TSL by avoiding obstacles with less computational time.

8.4. Results of redundancy resolution using SQP without obstacles

Simulations present the redundancy resolution scheme of hyper-redundant manipulators. Here, the problem is to evaluate the best IK solution among the multiple solutions by considering the performance criterion of joint rotation minimization from the previous position, given in Eq. 4.32, and the corresponding task constraint is given in Eq. 4.34

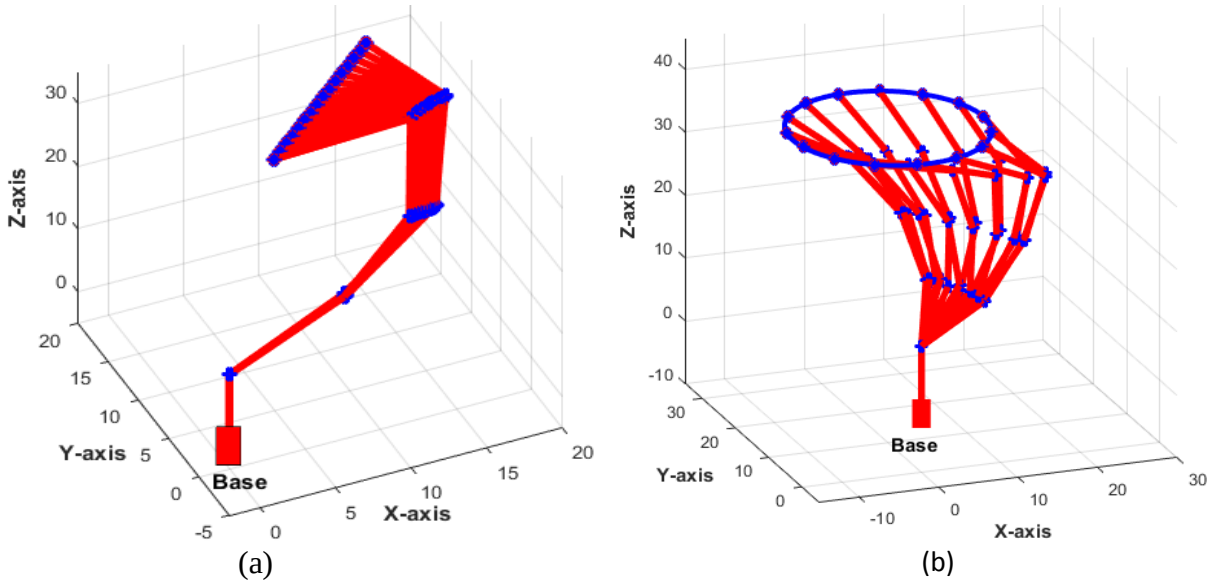


Fig. 8.12. Joint configurations of a 9-DOF robot. (a) Joint configurations of robot for a straight-line path (b) Joint configurations of robot for a circular path.

Here, 9-DOF redundant manipulator was considered to trace two different paths, such as a straight line and circular path in task space. The path in the task space is taken as a series of points, and the optimization algorithm is implemented at every point to determine the kinematic configuration at the corresponding point.

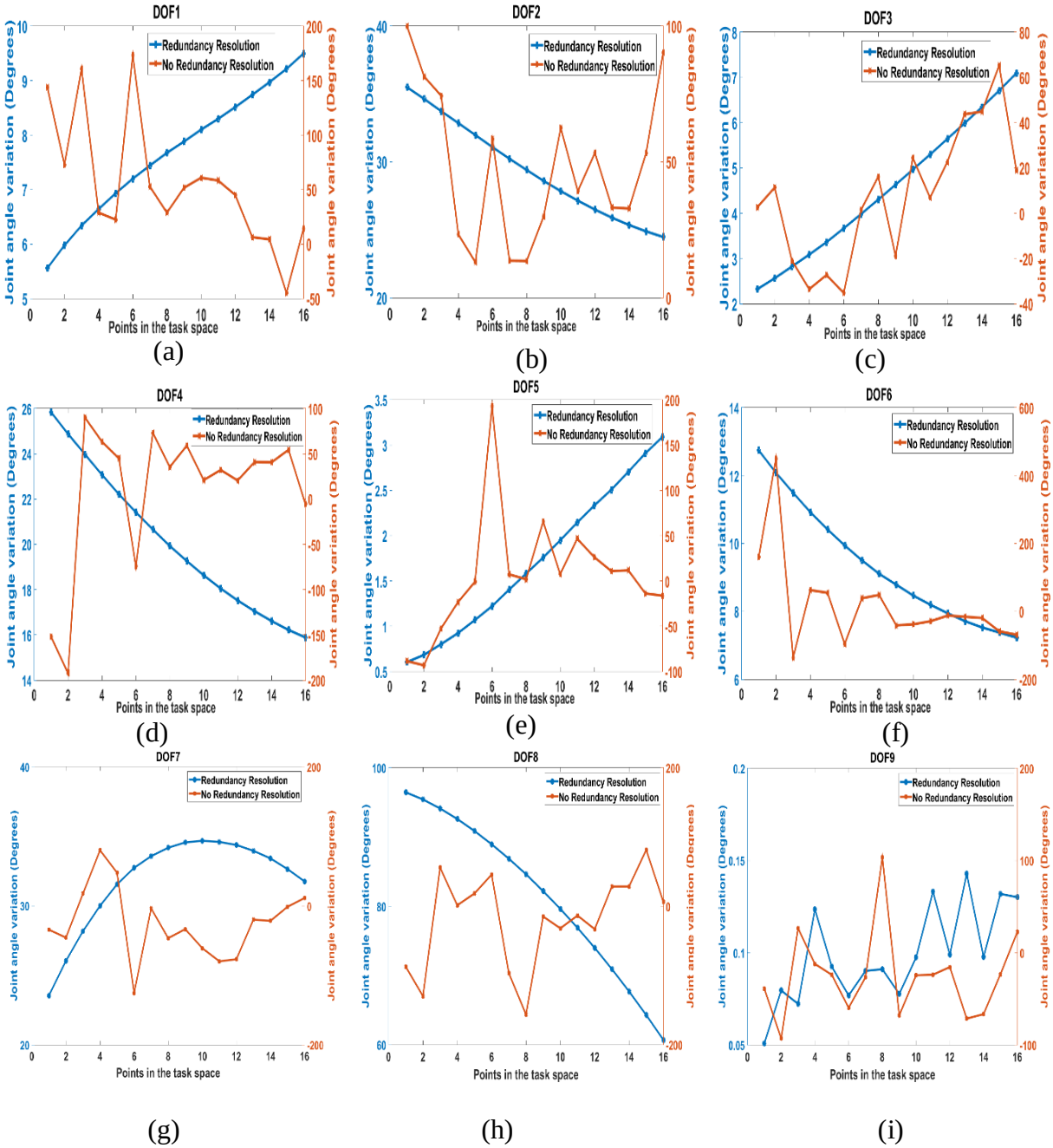


Fig. 8.13. Angle variation of a 9-DOF robot at task locations while tracing a straight line path.

Fig. 8.12(a) shows the kinematic configurations of the robot while tracing a straight-line path, whereas Fig. 8.12(b) shows kinematic configurations while tracing a circular path. In

both cases, the accuracy in reaching the task space is ensured. The computational time for the simulation of the robot while tracing each path is about 90 seconds. The angular displacement at each DOF, while the manipulator is tracing a straight-line path in the task space, is shown in Fig. 8.13. As the objective formulated is joint distance minimization, which provides the least movement of joints from the previous configurations.

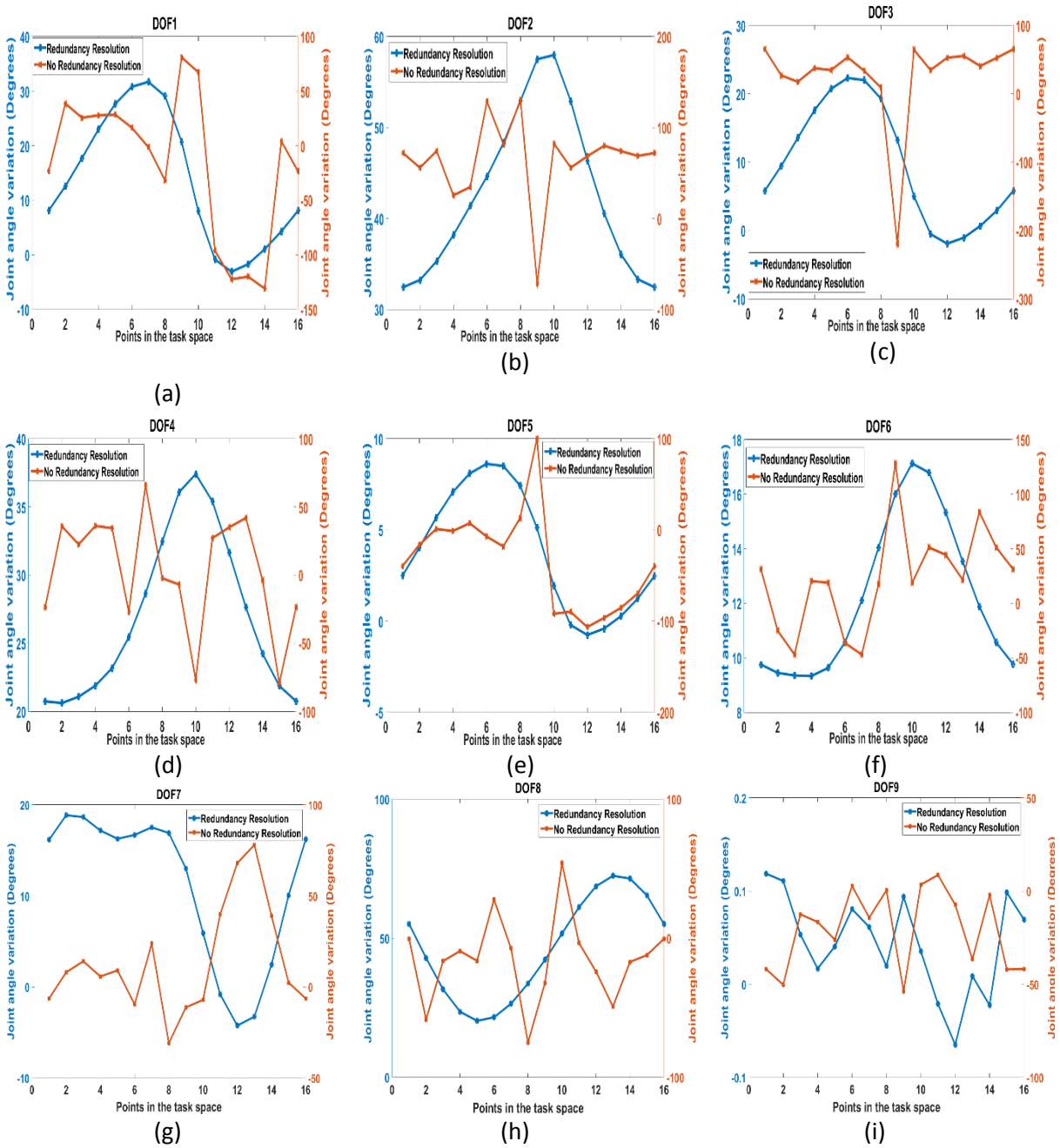


Fig. 8.14. Angle variation of a 9 -DOF robot at task locations while tracing a circular path.

The variation of joint rotation at each DOF corresponding to task space coordinates is shown in Fig. 8.13 (a)-(f), and these variations are shown for both redundancy resolution and no redundancy resolution. While Fig. 8.14 (a)-(f) shows the angular displacement at each DOF while the robot is tracing a circular path in task space, with and without redundancy resolution. In both cases, it is evident that the joint rotations were minimized with redundancy resolution along the defined path when compared with no redundancy resolution.

8.4.1. Redundancy resolution of the spatial robot with obstacles

A redundancy resolution scheme has been implemented for the spatial redundant robot while tracing a path in the 3D cluttered environment. Fig. 8.15 shows a case with a circular path around the spherical obstacle. In this case, the robot configurations corresponding to the target points along the path are evaluated by applying the redundancy resolution scheme discussed in section 4.3.1. The obstacle avoidance has been carried out by adding penalties to the optimization criteria given in Eq. 5.2. From Fig. 8.15, it was observed that the joint displacements are reduced while tracing a path in the workspace.

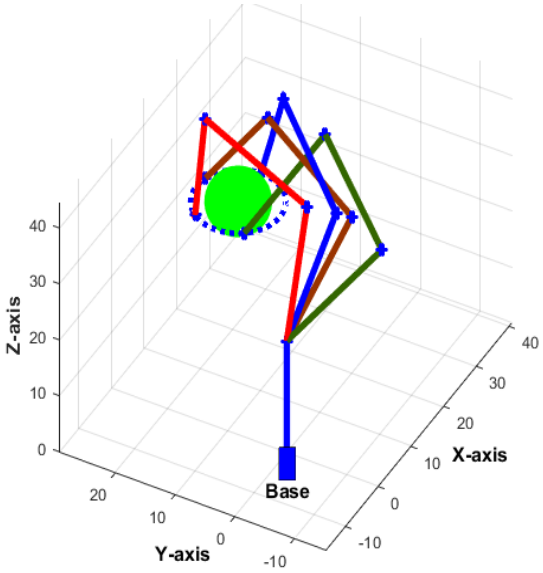


Fig. 8.15. IK solution of spatial redundant robot while tracing a circular path around the spherical obstacle.

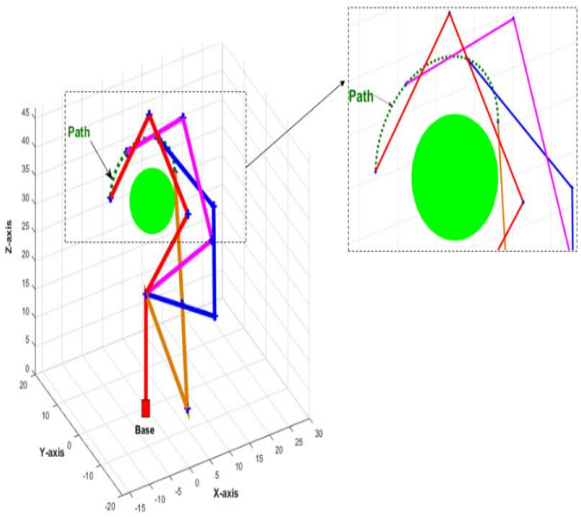


Fig. 8.16. IK solution of spatial redundant robot while tracing a semi-circular path around the spherical obstacle.

The path chosen was very close to the obstacle, and the IK solution in the figure ensures no collision with the sphere. Fig. 8.16 shows the robot configurations while tracing a semi-circular path behind the spherical obstacle. For better visibility of the path in the workspace, a magnified view has been shown in Fig. 8.16, which depicts the robot configurations along the path without colliding with the obstacle. Another case of IK simulation of a robot has been

performed in the 3D environment similar to a closed chamber with a spherical obstacle resting on a support member. Here, the path is chosen in a confined area so that the robot should trace the path without colliding the walls of the chamber and sphere.

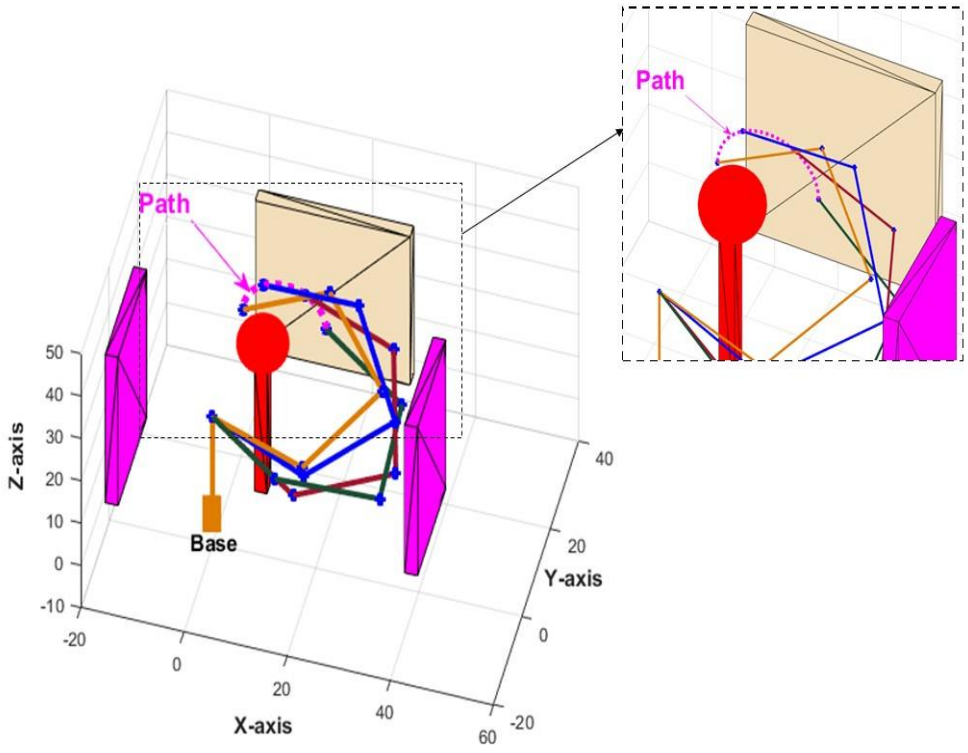


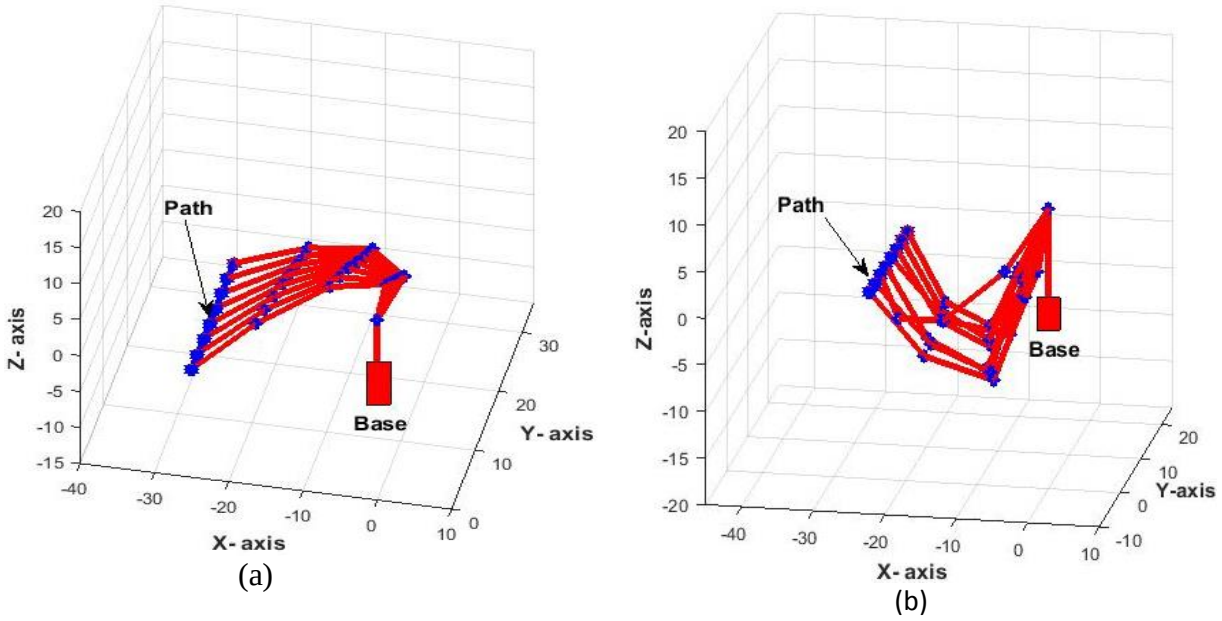
Fig. 8.17. IK Solution of spatial redundant robot while tracing a semi-circular path in a closed and cluttered environment.

The working environment and the path traced are similar to robots working in real-time applications like welding and painting. Fig. 8.17 shows the IK solution of the robot while tracing a semi-circular path in the 3D working environment. It was observed that all the robot configurations reaching the given task locations in the path are accurately reached without colliding the obstacles.

8.5. Singularity avoidance of spatial redundant robots

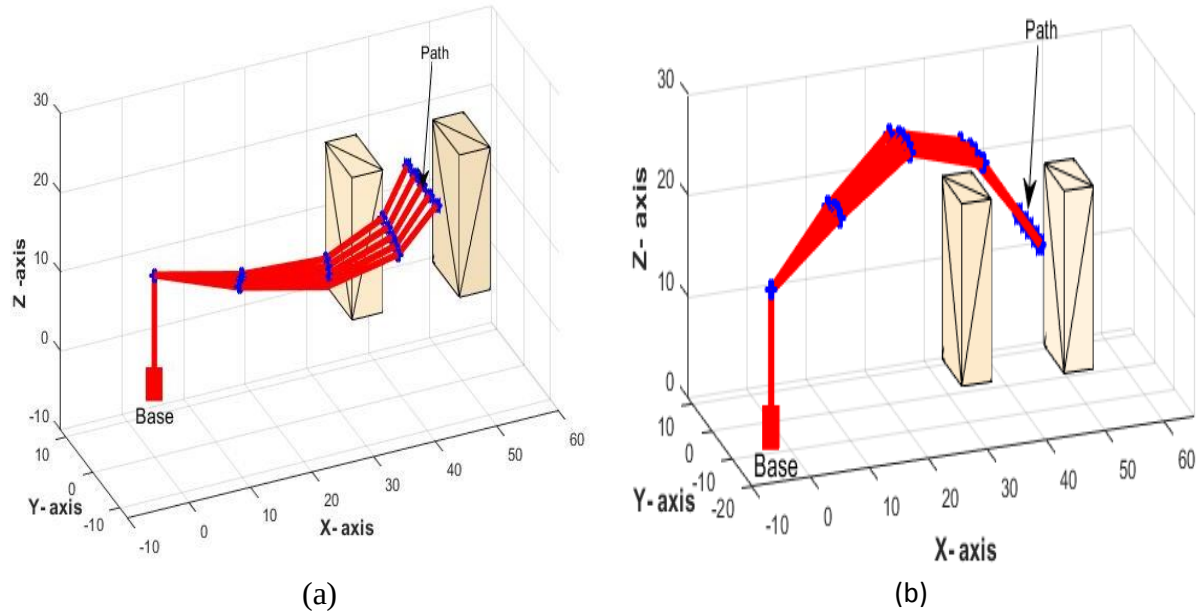
Singularity avoidance is chosen as performance criteria to improve the manipulability measure of robots at singular configurations in workspace, which is given in Eq. 4.35. Fig. 8.18 (a) shows the joint configurations of a robot while traversing a path without singularity avoidance. While Fig. 8.18 (b) shows the joint configurations with singularity avoidance.

Simulation of singularity avoidance along with obstacle avoidance has also been implemented.



(a) (b)

Fig. 8.18. Joint configurations while traversing a straight line path (a) Without singularity avoidance (b) With singularity avoidance.



(a) (b)

Fig. 8.19. Joint configurations while traversing a straight line path avoiding obstacles (a) Without singularity avoidance (b) With singularity avoidance.

Fig. 8.19 (a) shows robot avoiding obstacles with singular configurations, whereas Fig. 8.19 (b) illustrates robot avoiding both obstacles and singularities. Manipulability values are also calculated for both cases to show how far the manipulator is away from singularities. The value of manipulability measure for singular configuration is varying in the range of 1550-1700 on the corresponding points of the Path, whereas for the non-singular case, it is about 2200-2450. The percentage improvement of manipulability measure is about 41.21% for non-singular configurations compared to singular configurations.

The joint trajectories are evaluated by approximating a cubic polynomial equation for a given path with via points by considering both the cases i.e. with and without singularity avoidance. Fig. 8.20 shows the variation of the joint velocities. It was observed that there is a uniform velocity for non-singular configurations when compared with singular configurations.

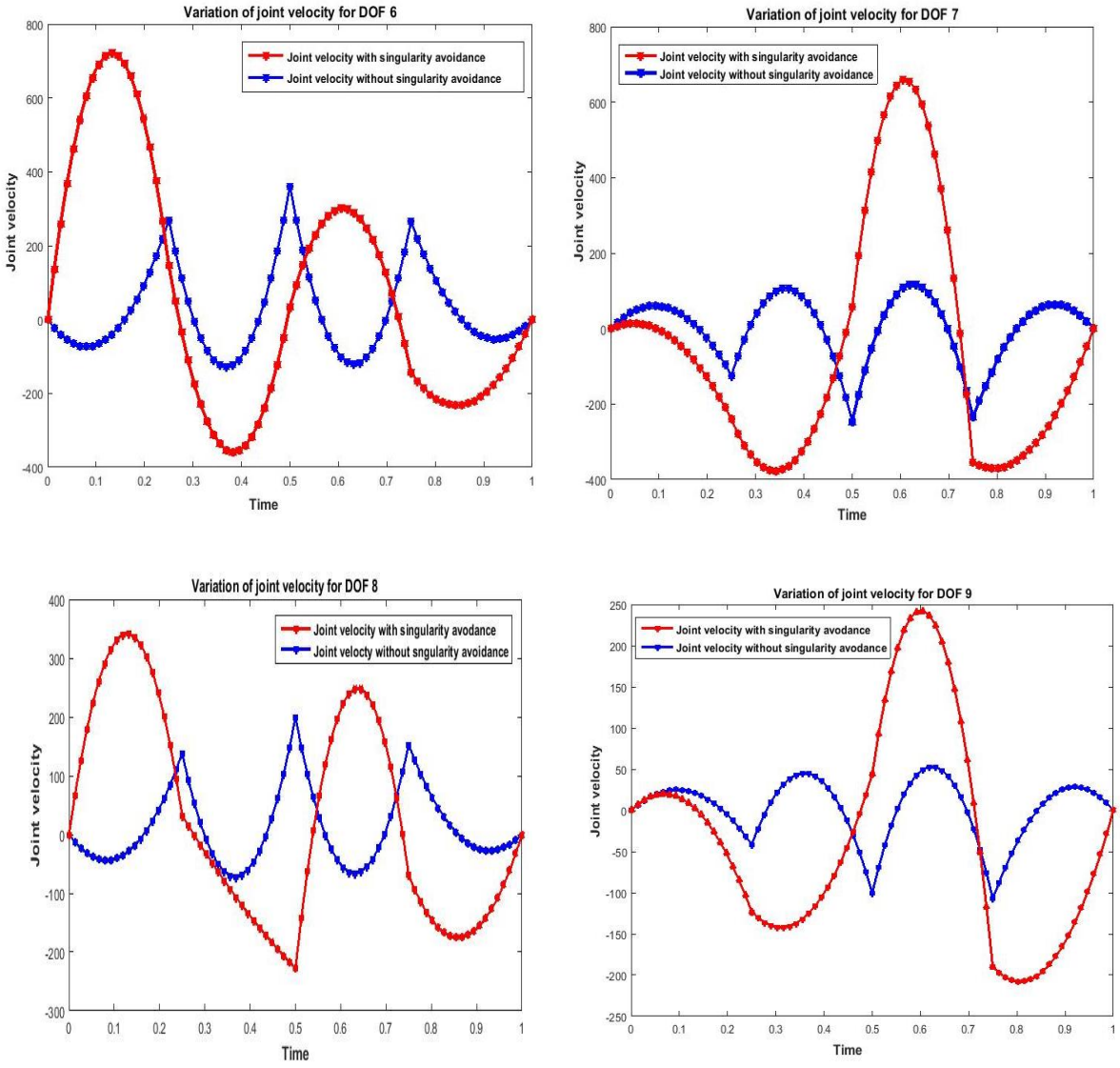


Fig. 8.20. Joint velocities of 9 DOF robot while traversing a path corresponding to singular and non-singular configurations.

The joint displacements were shown in Fig. 8.21. From Fig. 8.21, it is observed that the joint displacements are more for the configurations avoiding singularities, whereas the joint displacements are less for singular configurations. But, singular configurations require high joint velocities to move in a specified path. Thus, from Fig. 8.20 it is known that the joint velocities are uniform and not varying suddenly, which is observed for singular configurations.

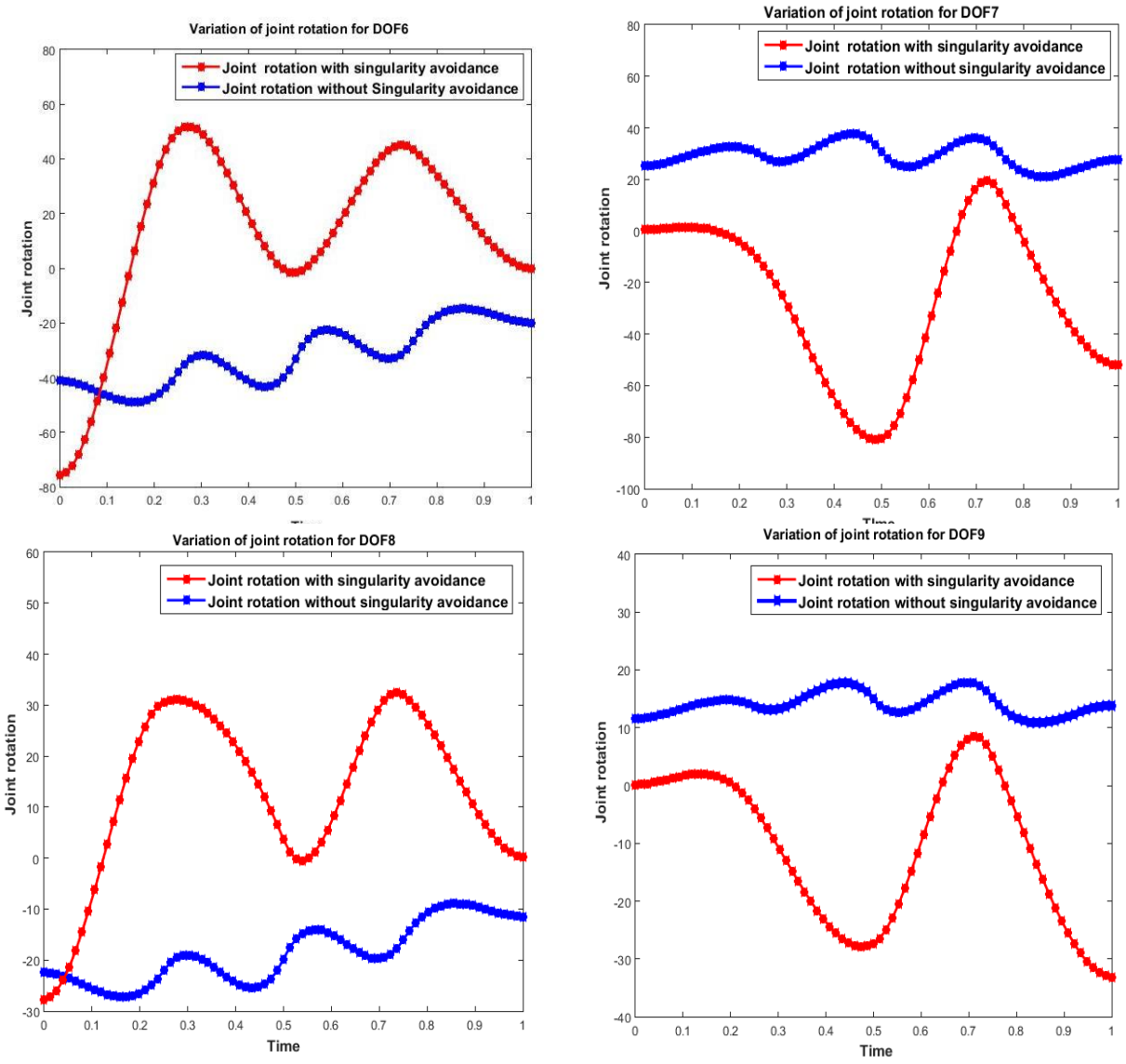


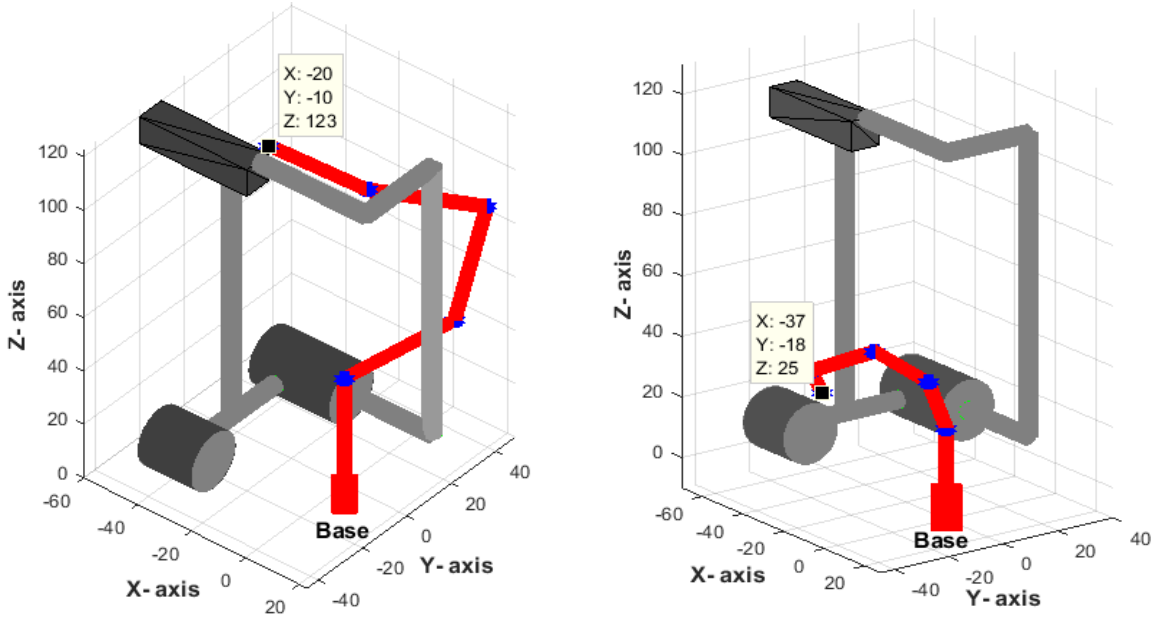
Fig. 8.21. Joint displacements of 9 DOF robot while traversing a path corresponding to singular and non-singular configurations.

8.6. IK Simulation of the spatial redundant robot in a realistic environment

Different domains of working environments have been modelled for implementation and illustration of inverse kinematics of spatial redundant manipulators reaching a specific task location and traversing a path while satisfying a secondary criterion such as joint distance minimization and maximization of manipulability measure. Simulation results are presented for a redundant robot operating in different realistic working environments such as pipe layout model, pipe-line welding, work cell, and warehouse environment. Six such case studies are presented in this section.

8.6.1. Case study 1

A spatial redundant manipulator employed at pipe layout resembling a nuclear power plant or air-conditioning applications is considered. In these application areas, the environment is cluttered and hazardous for human exposure. Redundant robots are employed for checking leakages and welding of pipes at the junctions. Joint configurations have been evaluated for a spatial redundant manipulator with 9 DOF.



(a) (b)

Fig. 8.22. IK solution of 9 DOF robot deployed in pipe layout application (a) At task space location (-20,-10,123), (b) At task space location (37,-18, 25).

IK solutions are determined by posing it as an optimization problem with an objective of Euclidean distance minimization and obstacles in the environment have been avoided by using the penalty approach shown in Eq. 5.2. The link lengths are uniform in size and they are considered as 40 units. The manipulator has to reach the TSL (-20, -10, 123) shown in

Fig. 8.22(a). Fig. 8.22(b) shows the joint configuration of the robot at TSL (37, -18, 25). The task of redundancy resolution has been implemented with an objective of joint distance minimization shown in Eq. 4.32. A spatial redundant robot is commanded to traverse a path around the pipe joint suitable for welding application. The joint configurations along the path with minimized joint rotation are shown in Fig.8.23.

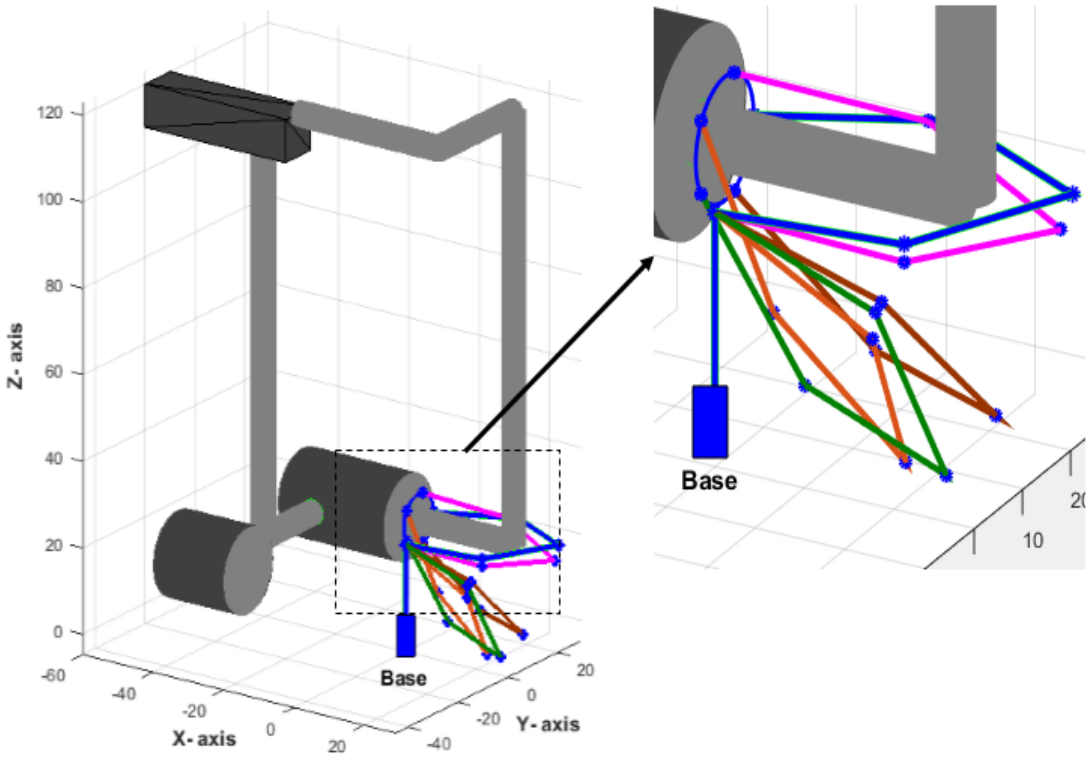
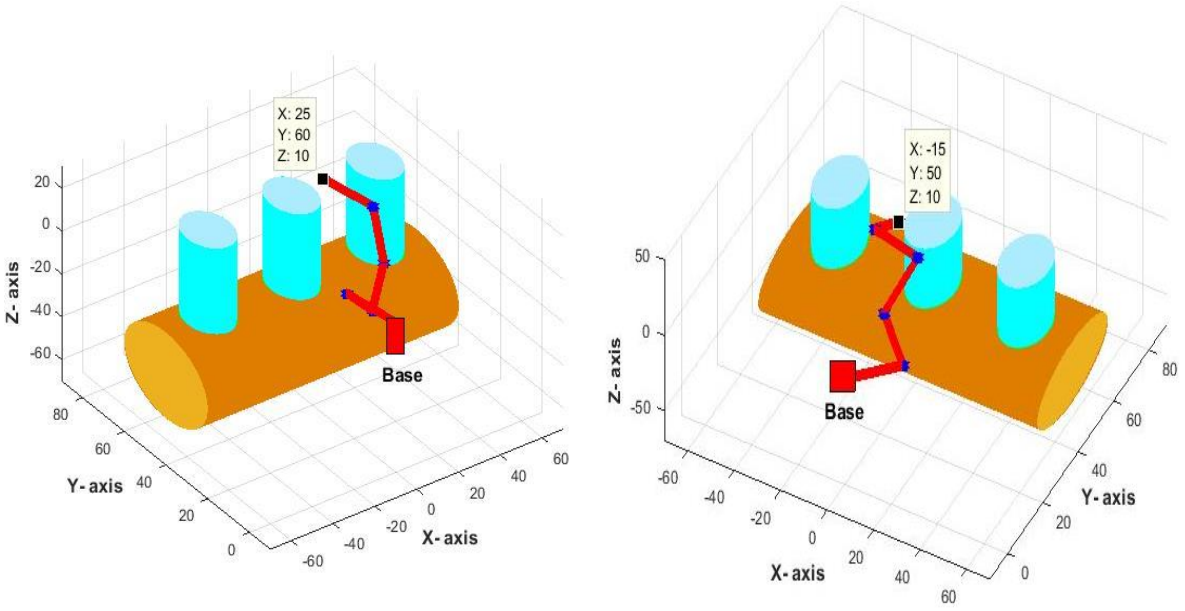


Fig. 8.23. Redundancy resolution scheme while end-effector is traversing a circular path at pipe joint of 9 DOF robot deployed in pipe layout application.

The magnified portion of the Fig. 8.23 shows the joint configurations of the spatial robot traversing a path at the junction of the pipe. In this case, the joints of the robot travelled with minimum joint rotation without colliding obstacles.

8.6.2. Case Study 2

A robot employed for pipeline inspection and welding at pipeline joint has been simulated. IK solution of the robot is shown at TSL (25, 60, 10) shown in Fig. 8.24 (a). Fig. 8.24 (b) shows the joint configuration of the robot at TSL (-15, 50, 10). A circular path has been chosen around the pipe near the junction. A redundancy resolution scheme has been implemented for evaluating joint configurations of the robot along the path. Fig. 8.25 shows the IK solution of a robot for a prescribed path.



(a) (b)

Fig. 8.24. IK solution of 9 DOF robot deployed in pipe line application (a) at task space location (25, 60, 10) (b) at task space location (-15, 50, 10)

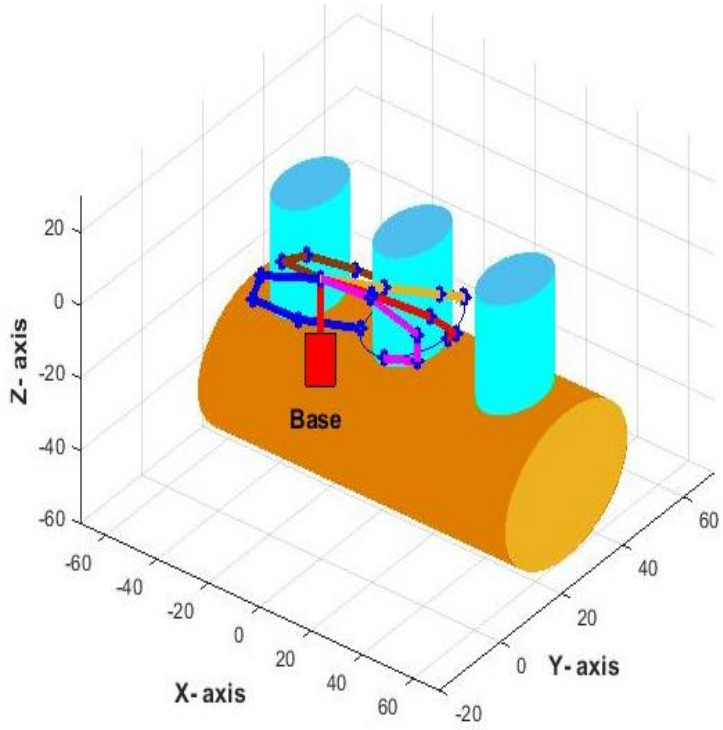


Fig. 8.25. Redundancy resolution scheme while end-effector is traversing a circular path at the pipe-joint of 9 DOF robot deployed in pipe line application.

8.6.3. Case Study 3

This case represents a robot that has been deployed in a work cell environment in industry. The spatial redundant robot is simulated for pick and place operation in work cell from the conveyer to workbench avoiding a cylindrical obstacle. IK solution of the robot is shown in Fig. 8.26 for corresponding task space locations (-10, 70, 62).

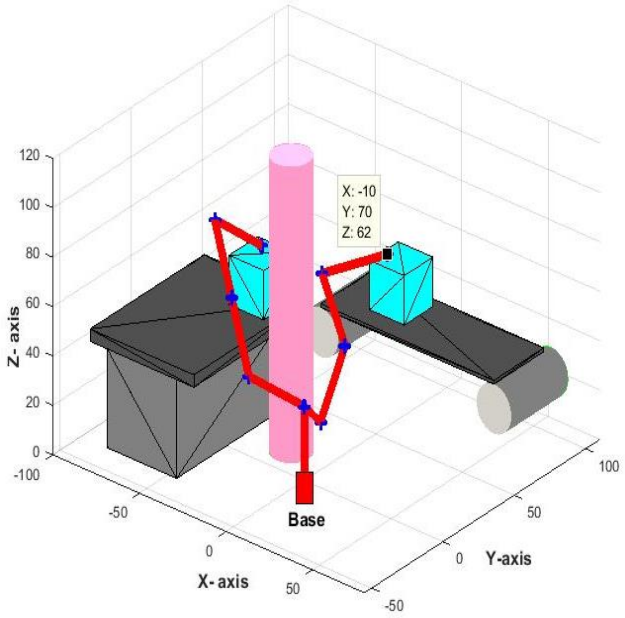


Fig. 8.26. IK solution of 9 DOF robot deployed in work facility for pick and place application.

IK solutions of spatial redundant robots employed in real-time applications have been proposed in the literature [42], but the solution techniques to obtain the joint configurations are complex. The proposed approach adopts classical optimization techniques to solve the IK problem. The problem of the multi-modal optimization has also been addressed by using a global optimization approach, by which multiple IK solutions can be obtained. To show the efficacy, the proposed approach has been applied to spatial redundant robot deployed in different working environments.

8.6.4. Case Study 4

In this case, the workspace is cluttered and narrow to reach a specific TSL. Minimization of geometric distance is chosen as an objective for determining the IK solution of the robot. The workspace in this cases have considered in such a way that classical optimization approaches fail to give solution in a single attempt, rather they require multiple restart procedure with different initial guesses to arrive the desired solution. The working environment chosen in

these simulations is similar to the realistic complex environments, where the workspace is cluttered and hazardous. Computing IK solution using classical approaches remains challenging for these cases. Hence, a population-based TLBO approach, has been implemented for solving the IK problem by minimizing the objective function shown in Eq. 3.20. IK solutions of spatial redundant robots have been depicted for pick and place operations in warehouse applications.

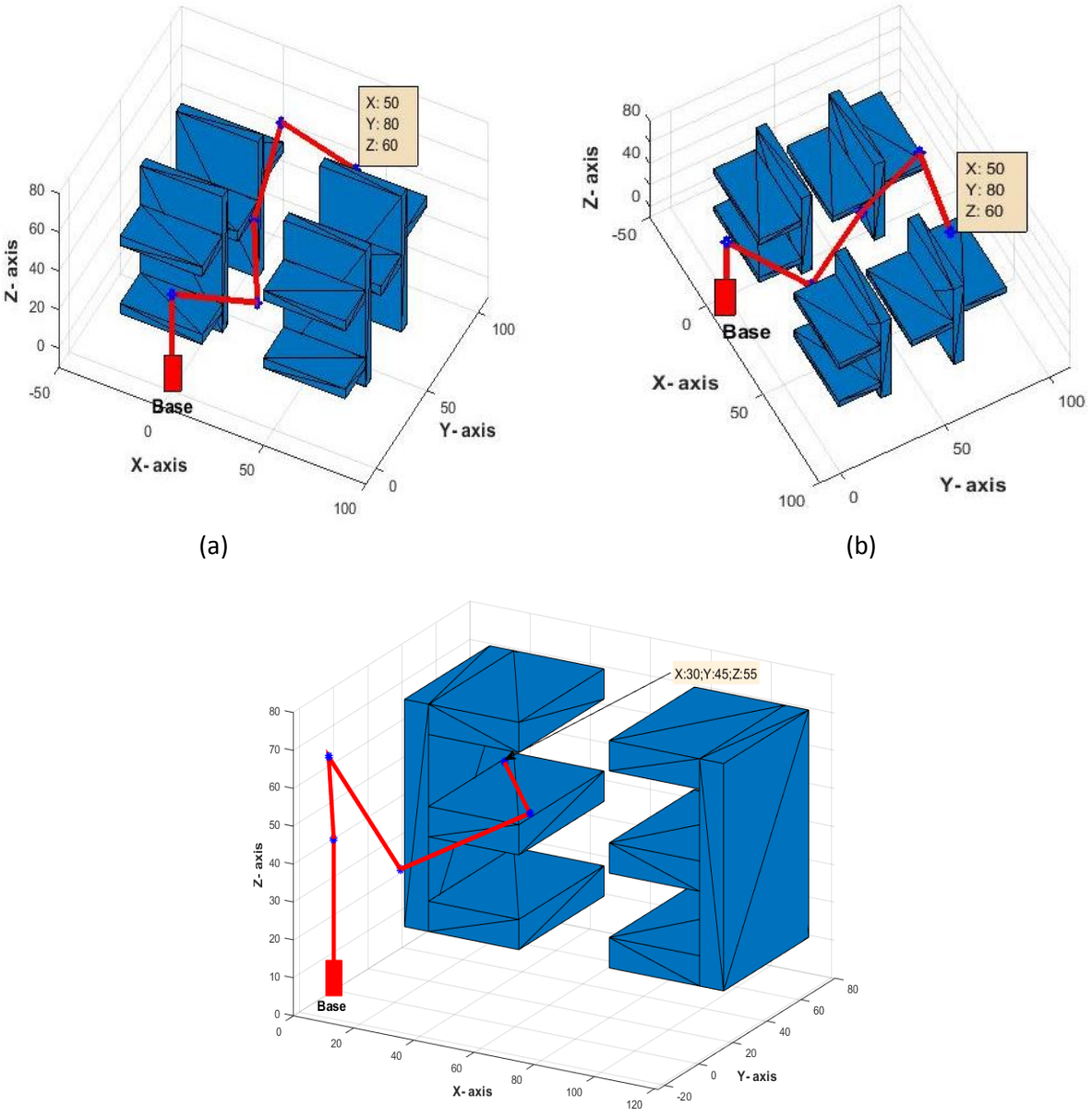


Fig. 8.27. IK solution of 9 DOF robot deployed in ware house environment for pick and place application at TSL (50, 80, 60) (a) Front view (b) Side view (c) At TSL (30,45,55).

Fig. 8.27 shows the robot deployed in a warehouse model of an environment suitable for pick and place operations. IK solution of the robot at a TSL of (50, 80, 60) mm is shown in

Fig. 8.27 (a). Joint configuration in Fig. 8.27 (b) are for the same TSL, but it has shown in two different views. Fig. 8.27 (c) depicts the IK solution of redundant robots working in the different workspace at TSL (30, 45, 55) mm, and the robot can be used for the same pick and place operation.

8.6.5. Case Study 5

A spatial 9 DOF robot when it is employed for an application in servicing and inspection of heat exchanger models have been investigated in this case.

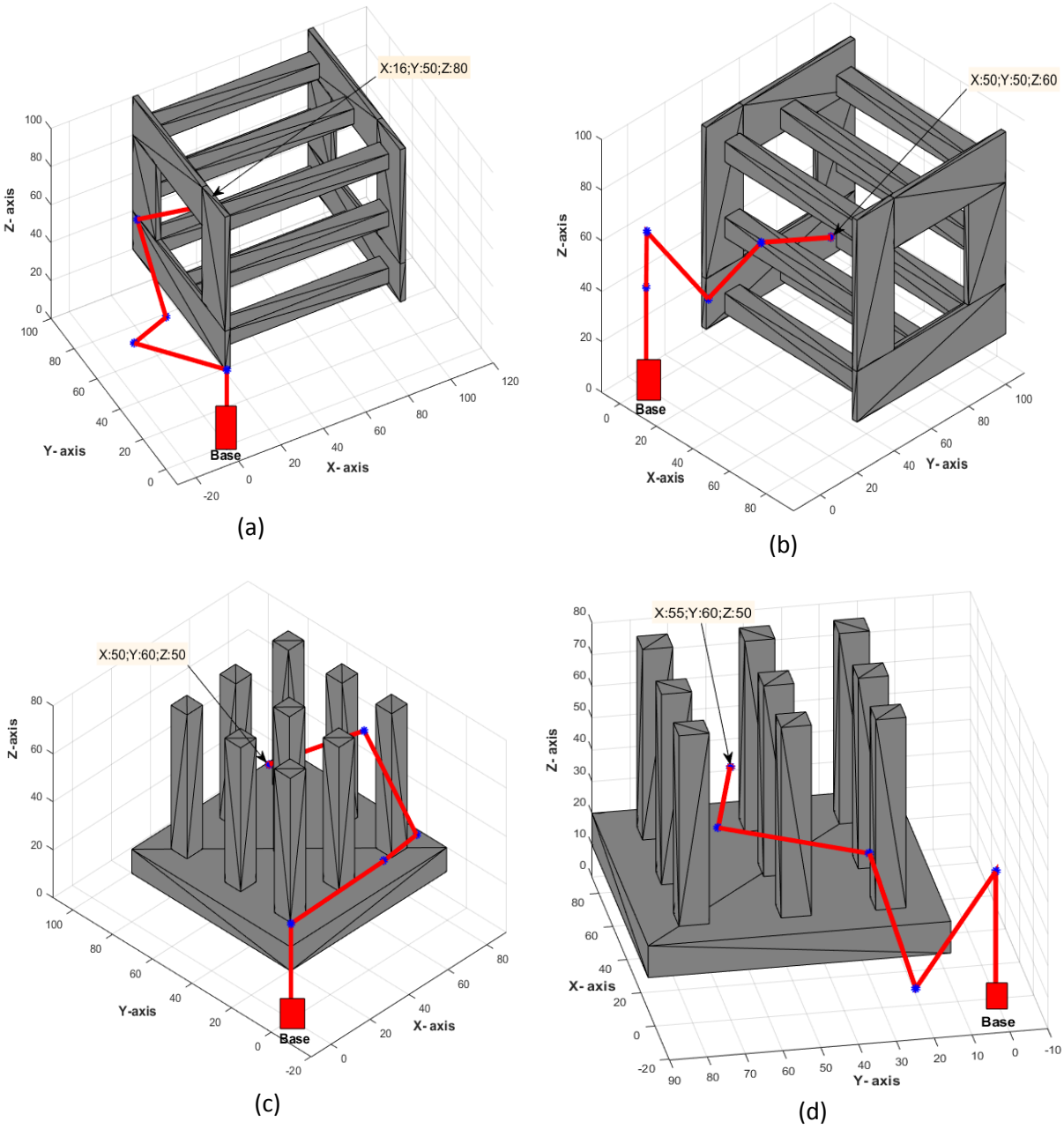
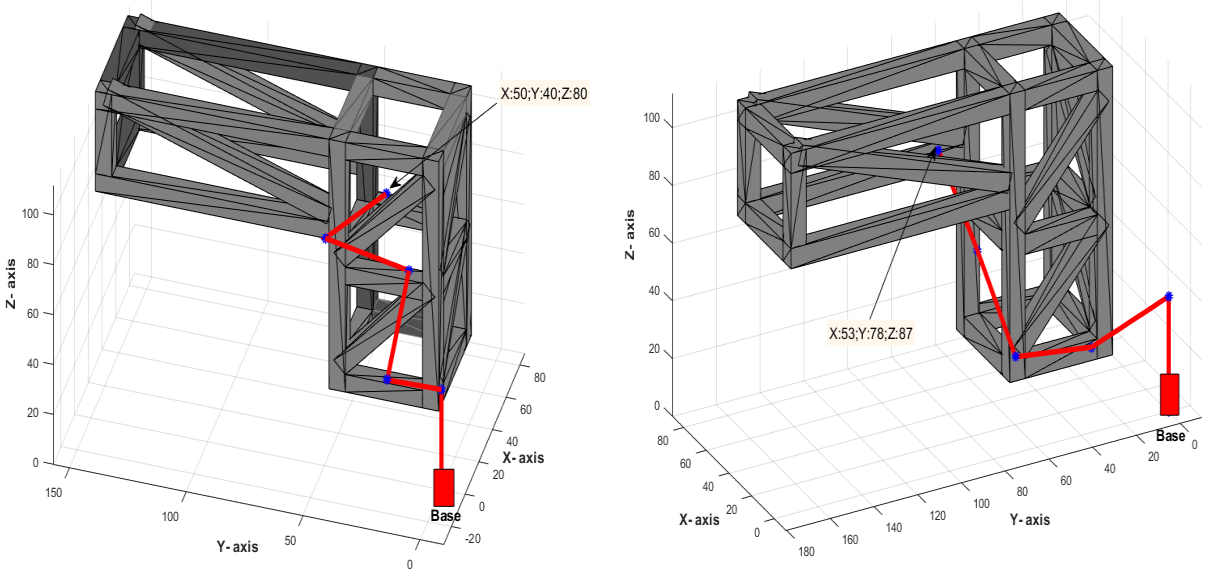


Fig. 8.28. IK solution of 9 DOF robot deployed in narrow environment at task space locations (a) (16, 50, 80) (b) (50, 50, 60) (c) (50, 60, 50) (d) (55, 60, 50).

Fig. 8.28 (a) shows the IK solution of the robot at TSL (16, 50, 80) by avoiding obstacles. Fig. 8.28 (b) shows the IK solution of the robot at task location (50, 50, 60). Fig. 8.28 (c-d) shows another IK simulation of robot deployed in servicing of fin models in cooling applications of server rooms and heat exchangers in a power plant. IK solution of the robot at TSL (50, 60, 50) shown in Fig. 8.28 (c). Fig. 8.28 (d) shows the joint configuration of the robot at TSL (55, 60, 50).

8.6.6. Case Study 6

IK simulation of the robot is performed in an environment with large structures like trusses. The working environment is similar to on-orbit servicing and the large structural environment in the construction of plants, where the robot is deployed for servicing by avoiding collisions with the structural elements in the environment. Fig. 8.29 (a) shows the IK solution of the robot at TSL (50, 40, 80). Joint configuration of the robot at TSL (53, 78, 87) shown in Fig. 8.29 (b). From Fig. 8.29 it is shown the end-effector of the robot is accessing the narrow regions of the workspace without colliding the obstacles.



(a) (b)

Fig. 8.29. IK solution of 9 DOF robot deployed at large structures with task space location (a) (50, 40, 80) (b) (53, 78, 87).

The environment modelled for this simulation is complex and cluttered. The obstacles have been modelled with different shapes that are similar to the real working environment. From the simulation results, it is shown that spatial redundant robot can reach the confined regions in the workspace without colliding obstacles.

8.7. Observation from the results

1. IK simulation of spatial redundant robots have been performed with two DOF at each joint
2. Multi-modal optimization is performed to achieve multiple IK solutions and these are utilized for switching from one configuration to another when robots are employed for diverse tasks
3. The computational time for the IK solution is less due to the use of classical optimization algorithms and simple collision modelling techniques.
4. An effective collision detection and avoidance technique has been implemented, which make the robot work in a cluttered environment and avoid obstacles of different shapes.
5. Simulations are performed on 9 DOF robot working in different real-time environments such as pipe joint inspection and welding, pick and place operation in plant layout and warehouse models, etc.
6. The spatial redundant robot successfully reached the confined zones by avoiding obstacles with an accuracy of 8.185×10^{-10} mm.

CHAPTER-IX

9. Summary and Conclusions

Inverse kinematics of redundant manipulators has been solved from long time. The issues related to IK solution techniques and increasing use of redundant manipulators in the fields of engineering and science makes this study challenging. A lot of research has been carried out in computation of IK problem of redundant manipulators with additional capability such as obstacle avoidance, singularity avoidance, joint-distance minimization and joint torque minimization. Some of the issues with these techniques are computational cost, sensitive at singular configurations, complex collision avoidance techniques etc. The proposed approach addresses these issues.

In the present thesis, IK solution and redundancy resolution of hyper-redundant manipulators have been presented. The simulations of hyper-redundant robots are performed while they were working in planar and spatial environments. The proposed IK solution technique is efficient and effective, the method is capable of finding an IK solution quickly (computational time of 1-3 minutes for planar robots in all the cases) for highly redundant robots with obstacles in the workspace. The efficiency of the proposed approach lies in the use of classical optimization methods and effective collision detection techniques. Collisions in the workspace are handled using penalty approach by modelling the obstacles as polygons in 2D environments and bounding boxes in 3D workspace. A restart procedure with a different initial guess is included when the solution is not found in first attempt itself. This ensures that repeated attempts are made if the classical methods fails to give solution because of multi-modality of the objective function. The proposed approach is used for avoiding convex and non-convex obstacles. The task of redundancy resolution has been implemented to determine the best joint configuration satisfying the secondary criterion such as joint-distance minimization, minimization of power consumption and singularity avoidance. For all the cases, IK solution of redundant robots are achieved while satisfying task space constraints and performance metrics with good accuracy (in the order of 10^{-11}mm).

IK simulation of spatial redundant robots is performed with 2 DOF at each joint, this type of robots able to access narrow regions of workspace. Since the classical optimization methods suffer from local optima, a multi-start frame work has been implemented to determine multiple IK solutions. The multiple IK solutions can be used to switch the configurations of the reconfigurable robots when they are employed in cluttered environments. Collision

avoidance of redundant robot is performed exclusively in 3D workspace by considering different shapes of obstacles and in cluttered environment. Collision detection and avoidance of 3D obstacles is performed using bounding box approach. To show the efficacy of the approach, realistic working environments are modelled and simulations have been performed on spatial redundant robot to reach the target location without collisions in the workspace. Redundancy resolution of spatial robot is performed with a performance criteria of minimizing joint rotation and singularity avoidance. The singularity avoidance of the spatial robots while traversing a straight line path with obstacles in the environment has been performed. Manipulability measure of the robot is chosen as performance criterion for singularity avoidance. Manipulability measure of robot with non-singular configurations is found to be improved and it is increased by 41.21% when compared with singular configurations.

The multi-modal nature of the objective function leads to furnish a local optimum solution in optimization process. This solution is not feasible in the cases of robot working in confined environments. Thus, a population based TLBO algorithm is used. This algorithm is global optimization approach, which requires less algorithmic control parameters when compared with other evolutionary approach. The computational time to solve the IK problem is about (5-10 minutes) in complex working environments. By using this approach simulations of robot in the complex working environments has been performed. As shown in the results, the robot able to reach the target location accurately by avoiding obstacles.

For the future scope, the proposed approach can be implemented for real time kinematic control of hyper-redundant robots suitable for industrial applications. The multi-start global optimization approach can be adopted for reconfigurable robots used in diverse working environments. Different performance criterion of the robot can be chosen as multi-objective optimization problem and compute the solution that satisfy several objectives simultaneously. An experimental model can be developed for the proposed robot, which is suitable to work in different environments. The flexibility of hyper-redundant robots, increased their application in various fields. Hence a detailed investigation of inverse kinematics and control of spatial redundant robots with different configurations suitable for different applications can be performed.

Appendices

Appendix I. Optimization work flow to find global or multiple local solutions

Steps to run global optimization frame work in MATLAB

1. Create problem structure
2. Create solver object
3. Set start points for multi-start
4. Run the solver

Inputs for problem structure

S. NO	Required Inputs of the problem
1	Local Solver (fmincon MATLAB function)
2	Objective Function
3	Start point x_0
4	Constraint functions
5	Local option structure

Create problem structure

To use global search or multi-start, create a problem structure. The problem structure can be created by create *createOptimProblem* Function

Steps to create a problem structure using the "*createOptimProblem*" function

I. Define your objective function as a file or anonymous function. If the solver is lsqcurvefit or lsqnonlin, ensure the objective function returns a vector, not scalar.

II. Create constraints, such as bounds and nonlinear constraint functions.

III. Create a start point. For example, to create a three-dimensional random start point xstart:

```
xstart = randn(3,1);
```

IV. Create an options structure using optimoptions.
For example, options = optimoptions(@fmincon,'Algorithm','interior-point');

V. Enter problem = createOptimProblem(solver, where solver is the name of your local solver)

- For GlobalSearch: 'fmincon'
- For MultiStart the choices are:
  - 'fmincon'
  - 'fminunc'
  - 'lsqcurvefit'
  - 'lsqnonlin'

VI. Set an initial point using the 'x0' parameter. If your initial point is xstart, and your solver is fmincon

VII. Include the function handle for your objective function in objective
problem = createOptimProblem('fmincon','x0',xstart, ... 'objective',@objfun)

VIII. Set bounds and other constraints as applicable.

Constraint	Name
lower bounds	`'lb'`
upper bounds	`'ub'`
matrix `Aineq` for linear inequalities `Aineq x ≤ bineq`	`'Aineq'`
vector `bineq` for linear inequalities `Aineq x ≤ bineq`	`'bineq'`
matrix `Aeq` for linear equalities `Aeq x = beq`	`'Aeq'`
vector `beq` for linear equalities `Aeq x = beq`	`'beq'`
nonlinear constraint function	`'nonlcon'`

IX. If using the lsqcurvefit local solver, include vectors of input data and response data, named 'xdata' and 'ydata' respectively.

X. validate the problem structure by running your solver on the structure.
For example, if the solver is fmincon: [x fval eflag output] = fmincon(problem);

Appendix II. DH Algorithm and Forward Kinematics of Serial Robots

The definition of a manipulator with four joint-link parameters for each link and a systematic procedure for assigning right-handed orthonormal coordinate frames, one to each link in an open kinematic chain, was proposed by Denavit and Hartenberg (1955) and is known as Denavit-Hartenberg (DH) notation.

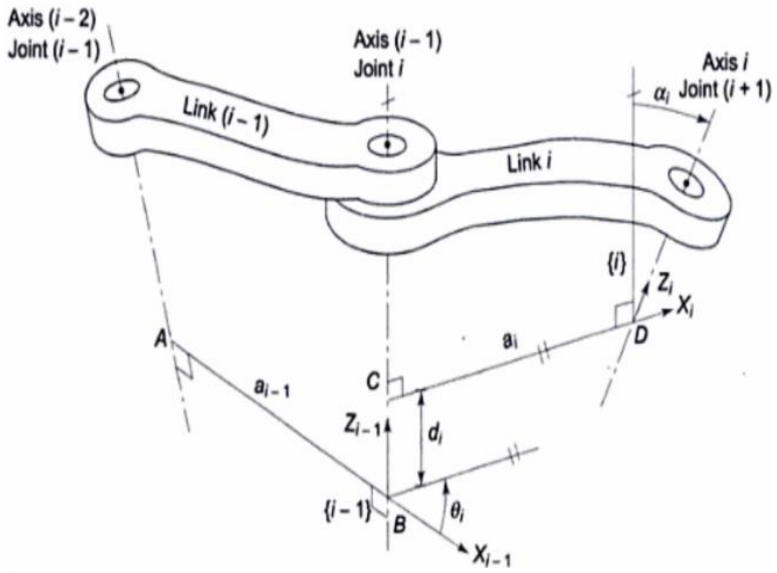


Fig. AII.1. Standard DH representation of a revolute joint.

A frame $\{\mathbf{i}\}$ is rigidly attached to distal end of link i and it moves with link i. An *n* DOF manipulator will have *(n + 1)* frames with the frame {0} or base frame acting as the reference inertial frame and frame {n} being the "tool-frame".

Figure 3.8 shows a pair of adjacent links, link *(i-1)* and link *i*, their associated joints, joints *(i-1)*, *i* and *(i+1)*, and axes *(i-2)*, *(i-1)*. and *i*, respectively. Line AB in the figure, is the common normal to *(i-2)* and *(i-1)* axes and line CD is the common normal to (i-1) and i-axes. A frame {i} is assigned to link i as follows:

(i) The z_i-axis is aligned with axis *i*, its direction being arbitrary. The choice of direction defines the positive sense of joint variable θ_i

(ii) The x_i-axis is perpendicular to axis z_{i-1} and z_i, and points away from axis z_{i-1} that is, x_i axis is directed along the common normal CD.

(iii) The origin of the i^{th} coordinate frame, frame *{i}*, is located at the intersection of axis of joint *(i+1)*, that is, axis *i*, and the common normal between axes *(i-1)* and *i* (common normal is CD), as shown in the figure.

(iv) Finally, y-axis completes the right-hand orthonormal coordinate frame {i}.

Note that the frame *{i}* for link *i* is at the distal end of link i and moves with the link. With respect to frame *{i-1}* and frame *{i)*, the four DH-parameters two link parameters ($\mathbf{a_i}, \alpha_i$) and two joint parameters ($\mathbf{d_i}, \theta_i$) are defined as:

(a) Link Length ($\mathbf{a_i}$) distance measured along x_i axis from the point of intersection of x_i axis with z_{i-1} axis (point C) to the origin of frame {i}, that is, distance CD.

(b) Link twist (α_i) angle between z_{i-1} and z_i axes measured about x_i axis in the right-hand sense.

(c) Joint distance ($\mathbf{d_i}$) distance measured along z_{i-1} axis from the origin of frame *{i-1}* (point B) to the intersection of x_i axis with z_{i-1} axis (point C), that is, distance BC.

(d) Joint angle (θ_i) angle between x_{i-1} and x_i axes measured about the z_{i-1} axis in the right-hand sense.

The convention outlined above does not result in a unique attachment of frames to links because alternative choices are available. For example, joint axis *i* has two choices of direction to point z_i axis, one pointing upward (as in Fig. AII. 1) and other pointing downward. To minimize such options and get a consistent set of frames.

Once the frames are assigned to each link, the joint-link parameters ($\theta_i, \mathbf{d_i}, \alpha_i, \mathbf{a_i}$) can be easily identified for each link, using which, the direct kinematic model is developed.

In fixing the frames. It is desirable to make as many of the joint-link parameters Zero as possible because the amount of computations necessary in later analysis is dependent on these. Hence, whenever there is a choice in frame assignment, emphasis is on making a choice, which results in as many zero parameters as possible

KINEMATIC RELATIONSHIP BETWEEN ADJACENT LINKS

To find the transformation matrix relating two frames attached to the adjacent links, consider frame *{i-1}* and frame *{i}* as shown in Fig. 3.9. These two frame are associated with link *(i-1)* and *i* but for clarity the links are not shown in the figure. The kinematic joint-link parameters involved (($\theta_i, \mathbf{d_i}, \alpha_i, \mathbf{a_i}$) are shown therein. Points B, C, D and frame *{i-1}* and *{i}* are the same as in Fig. 3.8. The transformation of frame *{i-1}* to frame *{i}* consists of four basic transformations as shown in Fig. 3.9.

(a) A rotation about z_{i-1} axis by an angle θ_i ;

(b) Translation along z_{i-1} axis by distance ($\mathbf{d_i}$):($\theta_i, \mathbf{d_i}, \alpha_i, \mathbf{a_i}$)

(c) Translation by distance α_i along x_i axis, and

(d) Rotation by an angle α_i about x_i axis

Using the spatial coordinate transformations, the composite transformation matrix, which describes frame *{i}* with respect frame *{i-1}.* is obtained using Equation given below.

$$^{i-1}\mathbf{T}_i = \mathbf{T_z}(\theta_i)\mathbf{T_z}(\mathbf{d_i})\mathbf{T_x}(\mathbf{a_i})\mathbf{T_X}(\alpha_i) \qquad \text{(AII. 1)}$$

$$^{i-1}\mathbf{T}_i = \begin{bmatrix} \mathbf{C}\theta_i & -\mathbf{S}\theta_i & 0 & 0 \\ \mathbf{S}\theta_i & \mathbf{C}\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \mathbf{d}_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & \mathbf{a}_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \mathbf{C}\alpha_i & -\mathbf{S}\alpha_i & 0 \\ 0 & \mathbf{S}\alpha_i & \mathbf{C}\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(AII. 2)}$$

$$^{i-1}\mathbf{T}_i = \begin{bmatrix} \mathbf{C}\theta_i & -\mathbf{S}\theta_i\mathbf{C}\alpha_i & \mathbf{S}\theta_i\mathbf{S}\alpha_i & \mathbf{a}_i\mathbf{C}\theta_i \\ \mathbf{S}\theta_i & \mathbf{C}\theta_i\mathbf{C}\alpha_i & -\mathbf{C}\theta_i\mathbf{S}\alpha_i & \mathbf{a}_i\mathbf{S}\theta_i \\ 0 & \mathbf{S}\alpha_i & \mathbf{C}\alpha_i & \mathbf{d}_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(AII. 3)}$$

The homogeneous transformation matrix $^{i-1}\mathbf{T}_i$ describes the position and orientation of frame *{i}* relative to frame *{i-1}* and completely specifies the geometric relationship between these links in terms of four DH-parameters ($\theta_i, \mathbf{d_i}, \alpha_i, \mathbf{a_i}$) . These four parameters, only one is a variable for link *i*, the displacement variable θ_i and other three are constant. The matrix $^{i-1}\mathbf{T}_i$ is known as link *i* transformation matrix. As shown before, the 3 x 3 upper left corner sub-matrix of Eq. AII. 3 gives the orientation of coordinate axes of frame {i}, while the 3 xl upper right corner sub-matrix represents the position of the origin of frame {i}.

Appendix III. Robotic toolbox commands of redundant manipulator

To illustrate this create a PUMA robot.

mdl_puma560

Creating a manipulator arm

L(2)=Link('d', 0, 'a', 0.302, 'alpha', -pi/2)

SerialLink(L, 'name', 'Puma')

Rotation and Translation

*T=transl(x, y, z)*trotx(theta)*

T= trans(x, y, z)

T= trotx(theta)

Forward and Inverse Kinematics and Plotting

T= p560.fkine(q)

q= p560.ikine(T)

q= p560.ikine6s(T)

p560.plot(q)

Trajectory Planning

Preparing a time vector for the trajectory:

t = [0:0.05:2]'

Converting the initial pose and final pose from Cartesian Space to Joint Space:

qi = p560.ikine6s(Ti)

qf= p560.ikine6s (Tf)

Generating a trajectory vector in joint space:

q = jtraj(qi, qf, t)

Ploting the angles of the joints against time:

qplot(t,q)

The following command extracts the vectors for the position, rotational velocity and rotational acceleration from the joint space trajectory planning:

[q, qd, qdd]=jtraj(qi, qf, t)

Plotting the angular velocity of the joints against time:

qplot(t,qd)

Plotting the angular acceleration of the joints against time:

qplot(t,qdd)

Plotting each joint's kinematics separately:

plot(t, q(:,1), t, qd(:,1), t, qdd(:,1))

plot(t, q(:,2), t, qd(:,2), t, qdd(:,2))

plot(t, q(:,3), t, qd(:,3), t, qdd(:,3))

plot(t, q(:,4), t, qd(:,4), t, qdd(:,4))

Cartesian space trajectory planning:

Generating a series of poses in Cartesian space (in a straight line)

T=ctraj(T1, T2, length(t))

Plotting a straight line between T1 and T2:

p1=transl(T1)

p2=transl(T2)

x=[p1(1),p2(1)]

y=[p1(2),p2(2)]

z=[p1(3),p2(3)]

plot3(x,y,z)

The Jacobian and manipulability

Finding the manipulability at a certain pose:

m=p560.maniplty(q)

Finding the Jacobean at a certain pose and checking its properties:

Jn=p560.jacob0(qn)

det(Jn)

rank(Jn)

jsingu(Jn)

Calculating the Jacobean elements manually:

dTdq1=(Tp1-T0)/dq

dRdq1=dTdq1(1:3,1:3)

R=T0(1:3,1:3)

*S1=dRdq1*R'*

vex(S1)

Manipulator dynamics

Finding the joint torques required for a certain trajectory:

Q=p560.rne(q, dq, ddq)

Finding the dynamic properties of a link of manipulator arm:

p560.links(1).dyn

Finding the torques caused by gravity

p560.gravload(qn)

Changing the payload

p560.payload(2.5, [0, 0, 0.1])

Changing the gravity settings:

p560.gravity=p560.gravity/6

Appendix IV. TLBO algorithm code

```
%Population initialization
pp= input('enter the population size');
DV= input('enter the number of the design variables');
iter= input('enter the maximum number of iterations');

lb=input('enter the lower bounds of the design variables');
ub= input('enter the higher bounds of the design variables');

for i=1:DV
 a=lb(i)+(ub(i)-lb(i))*rand(pp,1);
 x(:,i)=a;
end

%%Teacher phase
for kk=1:iter
for i=1:pp
  x(i,DV+1)=myobjj(x(i,1:DV));
end
temp=x;
teachr=find(trial1(:,DV+1)==min(trial1(:,DV+1)));
   if size(teachr,1)>1
       teachr=teachr(1);
    end
for i=1:DV
meantrial1(i)=mean(trial1(:,i));
r=rand(1,1);
meandiff_trial1(i)=r*(x(teachr,i)-1*meantrial1(i));
```

```
 trial1(:,i)=meandiff_trial1(i)+trial1(:,i);

for ii=1:pp
    if trial1(ii,i) >ub(i)
        rr=rand(1,1);
   trial1(ii,i)=lb(i)+(ub(i)-lb(i))*rr;
    end
       if trial1(ii,i) <lb(i)
        rr=rand(1,1);
   trial1(ii,i)=lb(i)+(ub(i)-lb(i))*rr;
      end
    end
end

for i=1:pp
  trial1(i,DV+1)=myobjj(trial1(i,1:DV) ); %% Objective function %
end

for i=1:pp
     if(trial1(i,DV+1)>x(i,DV+1))
         trial1(i,:)=x(i,:);
     end
 end
%%End of teacher phase%

%% Start of learner phase
for i=1:pp
    k=1;
trial2(1,:)=trial1(i,:);
```

```
end
            end
            trial2(k,DV+1)=myobjj(trial2(k,1:DV) );
          end
        end
    end

 I=find(trial2(:,DV+1)==min(trial2(:,DV+1)));

 if size(I,1)>1
       I=I(1);
    end
    x(i,:)=trial2(I,:);

End
%% End of learner phase

% fprintf('At the end of %d iteration',icount);
% disp(x)
I=find(x(:,DV+1)==min(x(:,DV+1)));
   if size(I,1)>1
       I=I(1);
    end
best(kk,:)=x(I,:);
% fprintf('The best solution in the iteration\n');
% disp(best)
```

REFERENCES

[1] Craig, John J. Introduction to robotics: mechanics and control. Vol. 3. Upper Saddle River: Pearson Prentice Hall, (2005).

[2] Fabrizio Caccavale, Ciro Natale, Bruno Siciliano, and Luigi Villani. Resolved-acceleration control of robot manipulators: A critical review with experiments. *Robotica*, 16:565573, 1998.

[3] Alessandro De Luca and Bruno Siciliano. Inversion-based nonlinear control of robot arms with flexible links. *Journal of Guidance, Control, and Dynamics*, 16(6):1169–1176, December 1993.

[4] Wang, Xuguang. "A behavior-based inverse kinematics algorithm to predict arm prehension postures for computer-aided ergonomic evaluation." *Journal of biomechanics* 32.5 (1999): 453-460.

[5] Grochow, Keith, et al. "Style-based inverse kinematics." *ACM SIGGRAPH 2004 Papers*. 2004. 522-531.

[6] Oveisi, Atta, et al. "Genetic Algorithm and Adaptive Model Reference Controller in Tracking Problem of PUMA 560 Arm Robot." *Proceeding of the RSI/ISM International Conference on Robotics and Mechatronics A* 2.3: 3, 2013

[7] http://www9.cs.tum.edu/praktika/robotarm.SS08/lectures/lecture_01.pdf

[8] https://www.machinedesign.com/markets/robotics/article/21835000/whats-the-difference-between-industrial-robots

[9] Siciliano, Bruno, et al. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[10] http://www.brainkart.com/article/Robot-Anatomy-and-Related-Attributes_6409/

[11] Chirikjian, Gregory S., and Joel W. Burdick. "An obstacle avoidance algorithm for hyper-redundant manipulators." *Robotics and Automation. Proceedings., IEEE International Conference on*. IEEE, 1990.

[12] Yahya, Samer, Mahmoud Moghavvemi, and Haider AF Mohamed. "Singularity avoidance of a six degree of freedom three dimensional redundant planar manipulator." *Computers & Mathematics with Applications* 64.5 (2012): 856-868

[13] Iossifidis, Ioannis, and GregorSchoner. "Dynamical systems approach for the autonomous avoidance of obstacles and joint-limits for a redundant robot arm." *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE, 2006

[14] Hollerbach, J. O. H. N. M., and Ki Suh. "Redundancy resolution of manipulators through torque optimization." *IEEE Journal on Robotics and Automation* 3.4 (1987): 308-316.

[15] Tringali, Alessandro, and Silvio Cocuzza. "Globally optimal inverse kinematics method for a redundant robot manipulator with linear and nonlinear constraints." *Robotics* 9.3 (2020): 61.

[16] Mahdavian, M., Yousefi-Koma, A., Shariat-Panahi, M., Khadiv, M., & Ghasemi Toudeshki, A. Optimal Trajectory Generation for Energy Consumption Minimization and Moving Obstacle Avoidance of SURENA III Robot's Arm. *International Journal of Robotics, Theory and Applications*, *4*(3), 1-9, 2015.

[17] http://aimlab.wpi.edu/education/RBE580-ME5205-BME595_C_2014

[18] M. Shan, Jian Guo, E. Gill, Review and comparison of active space debris capturing and removal methods

[19] A.Flores-Abad, O. Ma, K. Pham, S. Ulrich, A review of space robotics technologies for on-orbit servicing, Prog. Aerosp. Sci. 68 (2014) 1–26

[20] P. Huang, M. Wang, Z. Meng, F. Zhang, Z. Liu, Attitude takeover control for post-capture of target spacecraft using space robot, Aerosp. Sci. Technol. 51 (2016) 171–180.

[21] Beschi, Manuel, et al. "Optimal Robot Motion Planning of Redundant Robots in Machining and Additive Manufacturing Applications." *Electronics* 8.12 (2019): 1437.

[22] A.Wolf, H.B. Brown, R. Casciola, A. Costa, M. Schwerin, E. Shamas, H. Choset, "A mobile hyper redundant mechanism for search and rescue tasks ", Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems, Las Vegas

[23] https://ecommons.udayton.edu/cgi/viewcontent.cgi?article=1788&context=stander_posters.

[24] Xu, W., Mu, Z., Liu, T. and Liang, B. A modified modal method for solving the mission-oriented inverse kinematics of hyper-redundant space manipulators for on-orbit servicing, *Acta Astronautica*. 2017,*139*, 54-66

[25] Li, Wei-Jie, et al. "On-orbit service (OOS) of spacecraft: A review of engineering developments." *Progress in Aerospace Sciences* 108 (2019): 32-120.

[26] Yesiloglu, S. Murat, and Hakan Temeltas. "Dynamical modeling of kinematically deficient cooperating manipulators." *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006.

[27] http://www.dpaonthenet.net/article/30991/OC-Robotics-launches-the-Explorer-range-of-snake-arm-robots.aspx

[28] Wolf, Alon, et al. "Design and control of a mobile hyper-redundant urban search and rescue robot." *Advanced Robotics* 19.3 (2005): 221-248.

[29] Kelasidi, Eleni, et al. "Modeling of underwater snake robots." *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014.

[30] https://mycuriositybot.wordpress.com/2019/02/18/application-of-robotics-in-the-field-of-biomedical-science/

[31] Clark, James, et al. "A novel flexible hyper-redundant surgical robot: prototype evaluation using a single incision flexible access pelvic application as a clinical exemplar." *Surgical endoscopy* 29.3 (2015): 658-667.

[32] Hartenberg, Richard S., and Jacques Denavit. "A kinematic notation for lower pair mechanisms based on matrices." *Journal of applied mechanics* 77.2 (1955): 215-221.

[33] Gupta, Krishna C. "Kinematic analysis of manipulators using the zero reference position description." *The International Journal of Robotics Research* 5.2 (1986): 5-13.

[34] Sheth, Pradip N., and J. J. Uicker Jr. "A generalized symbolic notation for mechanisms." (1971): 102-112.

[35] Peiper, Donald L. The kinematics of manipulators under computer control. Stanford university, dept of computer science, 1968.

[36] Featherstone, Roy. "Position and velocity transformations between robot end-effector coordinates and joint angles." *The International Journal of Robotics Research* 2.2 (1983): 35-45.

[37] Hollerbach, J., and Gideon Sahar. "Wrist-partitioned inverse kinematic accelerations and manipulator dynamics." *Proceedings. 1984 IEEE International Conference on Robotics and Automation*. Vol. 1. IEEE, 1984.

[38] Paul, R. P., & Shimano, B. Kinematic control equations for simple manipulators. *IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes* (pp. 1398-1406). IEEE, *1978*.

[39] Mihelj, Matjaz. "Human arm kinematics for robot based rehabilitation." *Robotica* 24.3 (2006): 377-383.

[40] Shimizu, Masayuki, Hiromu Kakuya, Woo-Keun Yoon, Kosei Kitagaki, and Kazuhiro Kosuge. "Analytical inverse kinematic computation for 7-DOF redundant manipulators with joint limits and its application to redundancy resolution." *IEEE Transactions on Robotics* 24, no. 5 (2008): 1131-1142.

[41] Traub, Joseph Frederick. Iterative methods for the solution of equations. Vol. 312. American Mathematical Soc., 1982.

[42] Whitney, D. "Optimum step size control for Newton-Raphson solution of nonlinear vector equations." *IEEE Transactions on Automatic Control* 14, no. 5 (1969): 572-574.

[43] Park, Frank C. "Computational aspects of the product-of-exponentials formula for robot kinematics." *IEEE Transactions on Automatic Control* 39, no. 3 (1994): 643-647.

[44] Wolovich, William A., and H. Elliott. "A computational technique for inverse kinematics." In *The 23rd IEEE Conference on Decision and Control*, pp. 1359-1363. IEEE, 1984.

[45] Buss, Samuel R. "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods." *IEEE Journal of Robotics and Automation* 17, no. 1-19 (2004): 16.

[46] Luenberger, David G., and Yinyu Ye. Linear and nonlinear programming. Vol. 2. Reading, MA: Addison-wesley, 1984.

[47] Wang, L.C. and Chen, C.C. A combined optimization method for solving the inverse kinematics problems of mechanical manipulators, *IEEE Transactions on Robotics and Automation*. 1991, 7(4), 489-499.

[48] Lee, Hong-You, and Chong-Gao Liang. "Displacement analysis of the spatial 7-link 6R-P linkages." *Mechanism and Machine Theory* 22, no. 1 (1987): 1-11.

[49] Ananthanarayanan, H. and Ordóñez, R. Real-time Inverse Kinematics of (2n+ 1) DOF hyper-redundant manipulator arm via a combined numerical and analytical approach, *Mechanism and Machine Theory*. 2015, 91, 209-226.

[50] Chirikjian, G.S. and Burdick, J.W. The kinematics of hyper-redundant robot locomotion, *IEEE transactions on robotics and automation*.1995, 11(6), 781-793

[51] Menon, M.S., Ananthasuresh, G.K. and Ghosal, A. Natural motion of one-dimensional flexible objects using minimization approaches, *Mechanism and Machine Theory*. 2013, 67, 64-76.

[52] Sardana, L., Sutar, M.K. and Pathak, P.M. A geometric approach for inverse kinematics of a 4-link redundant In-Vivo robot for biopsy, *Robotics and Autonomous Systems*. 2013 61(12), 1306-1313.

[53] Yahya, S., Moghavvemi, M. and Mohamed, H.A. Geometrical approach of planar hyper-redundant manipulators: Inverse kinematics, path planning and workspace, *Simulation Modelling Practice and Theory*. 2011, 19(1), 406-422.

[54] Parker, J. K., Khoogar, A. R., & Goldberg, D. E. Inverse kinematics of redundant robots using genetic algorithms, *Proceedings of International Conference on Robotics and Automation*. 1989, 271-276, IEEE.

[55] KöKer, R. A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization, *Information Sciences*. 2013, 222, 528-543.

[56] Bingul, Z., Ertunc, H.M. and Oysu, C., Applying neural network to inverse kinematic problem for 6R robot manipulator with offset wrist, *Adaptive and Natural Computing Algorithms*, Springer, Vienna. 2005, 112-115.

[57] Nearchou, A.C.: Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm. Mechanism and machine theory 33 (3), 273–292 (1998).

[58] Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Systems 9(3), 1-15 (1994).

[59] Rao, R. V., Savsani, V. J. & Vakharia, D. P. (2011). Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer Aided Design*, 43, 303–315. (2011).

[60] Rao, R. V., Savsani, V. J., & Vakharia, D. P. Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. *Information sciences*, *183*(1), 1-15 (2012).

[61] Aouf, A., Boussaid, L., & Sakly, A. (2018). TLBO-based adaptive neurofuzzy controller for mobile robot navigation in a strange environment. *Computational intelligence and neuroscience*, *2018*.

[62] Savsani, P., Jhala, R. L., & Savsani, V. J. Comparative study of different metaheuristics for the trajectory planning of a robotic arm. *IEEE Systems Journal*, *10*(2), 697-708(2014).

[63] Klein, C. A., & Huang, C. H. Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, (2), 245-250 (1983).

[64] Maciejewski, Anthony A., and Charles A. Klein. "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments." *The international journal of robotics research* 4.3 (1985): 109-117

[65] Klein, C. A., Chu-Jenq, C., & Ahmed, S. A new formulation of the extended Jacobian method and its use in mapping algorithmic singularities for kinematically redundant manipulators. *IEEE Transactions on Robotics and Automation, 11*(1), 50-55 (1995).

[66] Moradi, H., & Lee, S. Joint limit analysis and elbow movement minimization for redundant manipulators using closed form method. In *International Conference on Intelligent Computing* (pp. 423-432). Springer, Berlin, Heidelberg (2005).

[67] Baillieul, John. "Avoiding obstacles and resolving kinematic redundancy." *Robotics and Automation. Proceedings. IEEE International Conference on*. Vol. 3. IEEE, 1986.

[68] Kircanski, Manja V., and Tatjana M. Petrovic. "Inverse kinematic solution for a 7 DOF robot with minimal computational complexity and singularity avoidance." *Robotics and Automation, 1991. Proceedings., IEEE International Conference on*. IEEE, 1991.

[69] Liegeois, A. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE transactions on systems, man, and cybernetics*, *7*(12), 868-871 (1977).

[70] Chiaverini, S. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, *13*(3), 398-410 (1997).

[71] Huo, L. and Baron, L. The joint-limits and singularity avoidance in robotic welding, *Industrial Robot: An International Journal*. 2008, 35(5), 456-464.

[72] Singla, Ashish, Sandeep Kumar, and Bhaskar Dasgupta. "High-index norm redundancy resolution scheme for kinematically redundant serial manipulators." *International Journal of Computational Science (IJCS)* 1.4 (2007): 351-370.

[73] Seraji, H. Configuration control of redundant manipulators: Theory and implementation. *IEEE Transactions on Robotics and Automation*, *5*(4), 472-490 (1989).

[74] Badler, N. I., Phillips, C. B., & Webber, B. L. *Simulating humans: computer graphics animation and control*. Oxford University Press (1993).

[75] Zhang, Yunong, et al. "Equivalence of position-level and velocity-level redundancy-resolution schemes." *Mechatronics and Automation (ICMA), International Conference on*. IEEE, 2012.

[76] Chembuly, V. S., & Voruganti, H. K. An optimization based inverse kinematics of redundant robots avoiding obstacles and singularities. In *Proceedings of the Advances in Robotics* (pp. 1-6) (2017.

[77] Menasri, Riad, et al. "A trajectory planning of redundant manipulators based on bilevel

optimization." *Applied Mathematics and Computation* 250 (2015): 934-947.

[78] Liu, Huashan, Xiaobo Lai, and Wenxiang Wu. "Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints." *Robotics and Computer-Integrated Manufacturing* 29.2 (2013): 309-317.

[79] Gasparetto, A., and V. Zanotto. "A new method for smooth trajectory planning of robot manipulators." *Mechanism and machine theory* 42.4 (2007): 455-471.

[80] Bobrow, James E. "Optimal robot plant planning using the minimum-time criterion." *IEEE Journal on Robotics and Automation* 4.4 (1988): 443-450

[81] Sakamoto, K. "Trajectory generation for redundant robot manipulator by variational approach." *Advanced Motion Control-San Francisco* (1994): 378-385.

[82] Xidias, Elias K. "Time-optimal trajectory planning for hyper-redundant manipulators in 3D workspaces." *Robotics and Computer-Integrated Manufacturing* 50 (2018): 286-298.

[83] Mahdavian, Mohammad, et al. "Optimal trajectory generation for energy consumption minimization and moving obstacle avoidance of a 4DOF robot arm." *Robotics and Mechatronics (ICROM), 2015 3rd RSI International Conference on*. IEEE, 2015.

[84] Garg, Devendra P., and Manish Kumar. "Optimization techniques applied to multiple manipulators for path planning and torque minimization." *Engineering applications of artificial intelligence* 15.3-4 (2002): 241-252.

[85] Hirakawa, Andre R., and Atsuo Kawamura. "Trajectory planning of redundant manipulators for minimum energy consumption without matrix inversion." *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*. Vol. 3. IEEE, 1997.

[86] Saramago, S. F. P., and V. Steffen Jr. "Optimization of the trajectory planning of robot manipulators taking into account the dynamics of the system." *Mechanism and machine theory*33.7 (1998): 883-894.

[87] Zhang, Yunong, Shuzhi Sam Ge, and Tong Heng Lee. "A unified quadratic-programming-based dynamical system approach to joint torque optimization of physically constrained redundant manipulators." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 34.5 (2004): 2126-2132.

[88] Khatib, Oussama. "Real-time obstacle avoidance for manipulators and mobile robots." *Autonomous robot vehicles*. Springer New York, 1986. 396-404.

[89] Nakamura, Yoshihiko, Hideo Hanafusa, and Tsuneo Yoshikawa. "Task-priority based redundancy control of robot manipulators." *The International Journal of Robotics Research* 6.2 (1987): 3-15.

[90] Chirikjian, Gregory S., and Joel W. Burdick. "A geometric approach to hyper-redundant manipulator obstacle avoidance." (1992): 580-585.

[91] Choset, Howie, and Wade Henning. "A follow-the-leader approach to serpentine robot motion planning." *Journal of Aerospace Engineering* 12.2, (1999): 65-73.

[92] Jiménez, Pablo, Federico Thomas, and Carme Torras. "3D collision detection: a survey." *Computers & Graphics* 25.2, (2001): 269-285.

[93] Hubbard, Philip M. "Approximating polyhedra with spheres for time-critical collision detection." *ACM Transactions on Graphics* (TOG) 15.3 1996: 179-210.

[94] Van Henten, E. J., et al. "Collision-free inverse kinematics of the redundant seven-link manipulator used in a cucumber picking robot." *Biosystems engineering* 106.2 (2010): 112-124.

[95] Chen, I.M. and Yang, G. Inverse kinematics for modular reconfigurable robots, *Proceedings of IEEE International Conference on Robotics and Automation.* 1998, 2, 1647-1652.

[96] Kalra, P., Mahapatra, P.B. and Aggarwal, D.K. An evolutionary approach for solving the multimodal inverse kinematics problem of industrial robots, *Mechanism and machine theory*. 2006, 41(10), 1213-1229.

[97] Tabandeh, S., Melek, W.W. and Clark, C.M. An adaptive niching genetic algorithm approach for generating multiple solutions of serial manipulator inverse kinematics with applications to modular robots, *Robotica*. 2010, 28(4), 493-507.

[98] Espinoza, M. S., Gonçalves, J., Leitao, P., Sánchez, J. L. G., & Herreros, A. (2012, October). Inverse kinematics of a 10 dof modular hyper-redundant robot resorting to exhaustive and error-optimization methods: A comparative study. In *2012 Brazilian Robotics Symposium and Latin American Robotics Symposium* (pp. 125-130). IEEE.

[99] Das, H., Slotine, J. E., & Sheridan, T. B. (1988, April). Inverse kinematic algorithms for redundant systems. In *Proceedings. 1988 IEEE International Conference on Robotics and Automation* (pp. 43-48). IEEE.

[100] Spong, M. W., Hutchinson, S., & Vidyasagar, M. (2006). *Robot modeling and control*.

[101] Greville, T. N. E. The pseudo-inverse of a rectangular or singular matrix and its application to the solution of systems of linear equations. *SIAM review*, *1*(1), 38-43, (1959).

[102] Baillieul, J. Kinematic programming alternatives for redundant manipulators. In *Proceedings. IEEE International Conference on Robotics and Automation* (Vol. 2, pp. 722-728), 1985 IEEE.

[103] Caccavale, F., Chiaverini, S., & Siciliano, B. Second-order kinematic control of robot manipulators with Jacobian damped least-squares inverse: Theory and experiments. *IEEE/ASME Transactions on Mechatronics*, *2*(3), 188-194, 1997.

[104] Buss, S. R., & Kim, J. S. (2005). Selectively damped least squares for inverse kinematics. *Journal of Graphics tools*, *10*(3), 37-49.

[105] Toth, C. D., O'Rourke, J., & Goodman, J. E. (Eds.). (2017). *Handbook of discrete and computational geometry*. CRC press.

[106] Haines, E. (1994). Point in Polygon Strategies. *Graphics Gems*, *4*, 24-46.

[107] http://geomalgorithms.com/a03-_inclusion.html

[108] http://philliplemons.com/posts/ray-casting-algorithm

[109] Weiler, M., Kraus, M., Merz, M., & Ertl, T. (2003, October). Hardware-based ray casting for tetrahedral meshes. In *IEEE Visualization, 2003. VIS 2003.* (pp. 333-340). IEEE.

[110] Rahman, K. M., Alam, T., & Chowdhury, M. (2012, October). Location based early disaster warning and evacuation system on mobile phones using OpenStreetMap. In *2012 IEEE conference on open systems* (pp. 1-6). IEEE.

[111] Biswas, J., & Veloso, M. (2012, May). Depth camera based indoor mobile robot localization and navigation. In *2012 IEEE International Conference on Robotics and Automation* (pp. 1697-1702). IEEE.

[112] Ochilbek, R. (2018, November). A new approach (extra vertex) and generalization of Shoelace Algorithm usage in convex polygon (Point-in-Polygon). In *2018 14th International Conference on Electronics Computer and Computation (ICECCO)* (pp. 206-212). IEEE.

[113] Huang, C. W., & Shih, T. Y. (1997). On the complexity of point-in-polygon algorithms. *Computers & Geosciences*, *23*(1), 109-118.

[114] o'Rourke, J. (1998). *Computational geometry in C*. Cambridge university press.

[115] Deb, K. *Optimization for engineering design: Algorithms and examples*. PHI Learning Pvt. Ltd, (2012).

[116] Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. *The computer journal*, *7*(4), 308-313.

[117] Takahama, T., Sakai, S., & Iwane, N. (2005, December). Constrained optimization by the ε constrained hybrid algorithm of particle swarm optimization and genetic algorithm. In *Australasian Joint Conference on Artificial Intelligence* (pp. 389-400). Springer, Berlin, Heidelberg.

[118] Mehta, V. K., & Dasgupta, B. (2012). A constrained optimization algorithm based on the simplex search method. *Engineering Optimization*, *44*(5), 537-550.

[119] Dasgupta, B., Divya, K., Mehta, V. K., & Deb, K. (2013). RePAMO: recursive perturbation approach for multimodal optimization. *Engineering Optimization*, *45*(9), 1073-1090.

[120] Toolbox, G. O. (2011). User's Guide (r2011b). *The MathWorks Inc*.

[121] Martí, R., Lozano, J. A., Mendiburu, A., & Hernando, L. Multi-start methods. Handbook of Heuristics, 2016. 1-21.

[122] Ugray, Zsolt, et al. "Scatter search and local NLP solvers: A multistart framework for global optimization." INFORMS Journal on Computing 19.3 2007: 328-340

[123] Glover, Fred. "A template for scatter search and path relinking." Lecture notes in computer science 1363 1998: 13-54.

[124] Boggs, Paul T., and Jon W. Tolle. "Sequential quadratic programming." *Acta numerica* 4 1995: 1-51.

[125] Rao, S. S. (2019). *Engineering optimization: theory and practice*. John Wiley & Sons.

List of Publications

Journals:

1. V. V. M. J. Satish Chembuly & Hari K. Voruganti. (2020). An efficient approach for inverse kinematics and redundancy resolution of spatial redundant robots for cluttered environment. *SN Applied Sciences*, *2*, 1-20. (Indexed in ESCI)

2. V. V. M. J. Satish Chembuly & Hari K. Voruganti (2018). Trajectory planning of redundant manipulators moving along constrained path and avoiding obstacles. *Procedia Comput. Sci*, *133*, 627-634. (Indexed in Scopus)

3. V. V. M. J. Satish Chembuly & Hari K. Voruganti. An optimization-based approach for redundancy resolution of spatial robots in realistic working environment. *(Communicated to IJIRA)*

Scopus-indexed proceedings:

1. V. V. M. J. Satish Chembuly & Hari K. Voruganti. (2017). An optimization based inverse kinematics of redundant robots avoiding obstacles and singularities. In *Proceedings of the Advances in Robotics* (pp. 1-6)., ACM digital library.

2. V. V. M. J. Satish Chembuly & Hari K. Voruganti (2018, April). An efficient approach for inverse kinematics and redundancy resolution scheme of hyper-redundant manipulators. In *AIP Proceedings* (Vol. 1943, No. 1, p. 020019). AIP Publishing LLC.

Book chapters:

1. V. V. M. J. Satish Chembuly & Hari K. Voruganti. (2020). Inverse kinematics and trajectory planning of planar redundant manipulators in cluttered workspace. In *Emerging Trends in Mechanical Engineering* (pp. 461-469). Springer, Singapore.

2. Dhoke, A., V. V. M. J. Satish Chembuly & Hari K. Voruganti (2020). Kinematic Analysis for Optimum Manipulability and Trajectory Planning of Human Leg. In *Advances in Applied Mechanical Engineering* (pp. 633-642). Springer, Singapore.

Conference proceedings:

1. V. V. M. J. Satish Chembuly, Hari K. Voruganti (2016). Inverse Kinematics and Trajectory Planning of Redundant Manipulators, National Symposium of Mechanical Engineering Research Scholars (NSMERS) – 2016, NIT Warangal.

2. Mayank Singh Parihar, V. V. M. J. Satish Chembuly, Hari K. Voruganti (2017). Real-time Robot Path Planning by Mapping of the Workspace, Proceedings of iNaCoMM 2017, BAARC Mumbai.