Performance Improvement in Robotic Arm Movement using Fuzzy and Genetic Algorithms

A

Thesis

submitted in fulfilment of the requirements
for the degree of award of

DOCTOR OF PHILOSOPHY

submitted by

Vijay Kumar Banga

(Registration No.: 9041153)

under the supervision of

Dr. Yaduvir Singh
Associate Professor
Department of Electrical and
Instrumentation Engineering

Dr. Rajesh Kumar
Associate Professor
School of Mathematics
and Computer Applications

Department of Electrical and Instrumentation Engineering

Thapar University, Patiala-147001, INDIA

May 2011

To My Family

CERTIFICATE

I hereby certify that the work which is being presented in this thesis entitled **PERFORMANCE IMPROVEMENT IN ROBOTIC ARM MOVEMENT USING FUZZY AND GENETIC ALGORITHMS**, in fulfillment of the requirements for the award of degree of **DOCTOR OF PHILOSOPHY** submitted in Department of Electrical and Instrumentation Engineering, Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Dr. Yaduvir Singh and Dr. Rajesh Kumar, and refers the work of other researchers, which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(Vijay Kumar Banga)
Registration No. 9041153

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge and belief.

09/03/2010

(Dr. Rajesh Kumar)
Associate Professor,
School of Mathematics
and Computer Applications,
Thapar University,
Patiala – 147004(INDIA)

Supervisor

(Dr. Yaduvir Singh)
Associate Professor,
Department of Electrical and
Instrumentation Engineering,
Thapar University,
Patiala – 147004(INDIA)

Supervisor

ABSTRACT

Robotics is no more restricted to defined steps of machine rather it becomes more independent and self-decision making machine with the thinking power of artificial intelligence. Robots are used in the industrial applications like assembly operations, spray painting, grinding, welding, pick-and-place operations, *etc.* Robots perform repetitive tasks with accuracy and higher efficiency. Fully autonomous robots are still under research, which can take decision on their own to perform any task and are able to perform tasks efficiently.

Industrial robots are quite prevalent in high volume manufacturing processes. In many field applications where technical support is required, manhandling is either dangerous or is not possible. In such situations, three or more arm manipulators are commonly used. They are in great demand to speed up the automation processes.

Two-link robotic arm should be able to locate any location, which is the required movement in real world situations. These are used in micro to macro scale applications, *viz.,* chip fabrications to huge mechanical actuators used in chemical processes. In these cases, the motion profile of the robot rarely changes throughout the whole operation. Therefore, searching an optimal robot arm movement is a favorable solution to those problems.

Human beings are extremely complicated systems. The researcher's aim is to develop autonomous robot, which can compete with them on any level, and to perform tasks as human being perform with accuracy and efficiently. Path planning is one of the important objectives of developing autonomous robots. The evidence of the success of the intelligent human path planner suggests that the use of some intelligent, automated motion planning would be useful to investigate. Artificial intelligence can make it possible for robots to perform complex tasks easily.

Advance computers have shown themselves to be extremely capable in many application areas, and have transformed the world in which we live. Their application has helped to enhance the quality of traditionally human tasks, and has completely automated other tasks, replacing the need for humans to carry them out. However, automated systems controlled by computers cannot beat men in all areas.

The advancement in computational techniques has followed a new branch called as Artificial Intelligence namely evolutionary computation. Techniques falling in the evolutionary computation area are also referred to as evolutionary algorithms (EA).

Evolutionary computation has become the standard umbrella for a number of evolutionary driven techniques.

This research work is restricted to path planning of robotic arm movement with the help of fuzzy and genetic algorithms in order to achieve the target of minimum energy consumption criterion. Existing techniques require much computation and are dependent upon the mathematical model accuracy. In every relevant field, it is desired to minimise the consumption of power by the appliance. This can be achieved by optimising the movement of robotic arms.

This research work has been implemented on 2 degree-of-freedom (DOF), 3 DOF and 4 DOF robotic arm manipulators, which can be generalised to more degree of freedom manipulators. Artificial techniques like fuzzy, genetic algorithms and neural networks are widely used in various fields of sciences including robotics. Here, fuzzy and genetic algorithms have been used in hybrid mode. These two tools have been successfully combined in order to maximize their individual strengths so as to achieve the target of optimisation of arm movements between an initial and a final goal positions. The increase in the publications using artificial techniques for various engineering applications acclaimed superiority over other techniques.

In this research work, new method based on fuzzy and genetic algorithms has been developed. It provides a probabilistic approach to the path planning problem. The simulation results for various degree-of-freedom have been demonstrated and also issues related to them are discussed.

This research work contributes towards the significant role of fuzzy and genetic algorithms techniques in the future path planning methods so as to develop autonomous robots. With increase in degree-of-freedom and higher dimensional work spaces, complexity increases. It become very difficult to find the solutions of path planning problems with classical or mathematical techniques. It is also very difficult to develop accurate mathematical model. Genetic algorithms and fuzzy-genetic algorithms play vital role in path planning problem. This research work is limited to few degree-of-freedom (DOF), however, it may extended to more degree-of-freedoms (DOFs). In this research work, the non-deterministic parameters like friction, settling time and movement have been compensated. These parameters have been successfully compensated and minimised for their effects on the performance of robotic arm movements in order to achieve the target effectively. There may be more number of non-deterministic parameters that may be compensated with little modifications in the future research work.

Genetic algorithms and fuzzy-genetic algorithms have carried out an important role in path planning. This research work contributes to the ultimate goal of the autonomous robots and provides motivation to the other researches in this direction**.**

This thesis is divided into seven chapters. A brief outline of each chapter is given here below.

The first chapter introduces the issues related to robotics. These robots have to work in repetitive manner in a highly structured predictable environment. The robots are supplied with inputs like position, orientation and complex work spaces in various industrial environments with required jigs and fixtures. It requires large time and incurs huge costs. Such an industrial application needs to be properly installed and accurately calibrated for the spatial specifications of these jigs and fixtures.

Attempt has been made to develop artificially intelligent robotic systems, which will sense their external environment and operate on the basis of known information for which they have been prepared before hand. Developing an automated robotic system is a challenging task. It requires various skills like programming, mechanical modeling of a robotic system, electronic devices, their interfaces and control. An integrated design based on above skills and relevant engineering areas will help in realisation of such an automated robotic systems. Fuzzy logic and genetic algorithms have been discussed. A brief outline of analytical hierarchy process and hybrid algorithm is also discussed.

In second chapter, a detailed review of the literature relating fields of fuzzy logic and genetic algorithms followed by its applications in the engineering field and especially for robotics and automation has been given.

The third chapter discusses overview of the robotics specifications based on work envelop geometries and degree-of-freedom. The robotic manipulators and their kinematics have been elaborated for direct kinematics and inverse kinematics. The detailed overview of conventional approaches of path planning has also been presented.

Chapter four describes detailed overview of arm movements of 2 DOF, 3 DOF and 4 DOF robotic arm manipulators and evaluates the mathematical model of the arm, while considering case study of these manipulators for calculating various values for joint angles using inverse kinematics.

Chapter five presents the implementation of fuzzy logic for arm movement and genetic algorithms for obtaining optimal solution from possible solutions for path optimisation problem. In this chapter, detailed steps involved in implementation of fuzzy logic and genetic algorithms have been given. Simulation results of 2 DOF, 3 DOF and 4 DOF

robotic arm manipulator are presented so as to justify the proposed technique, which is being used for path optimisation, while achieving the target of minimum energy consumption.

In chapter six, results are shown for 2 DOF, 3 DOF and 4 DOF robotic arm manipulator with genetic algorithms and fuzzy–genetic algorithms. In this chapter, comparison of results with genetic algorithms and fuzzy–genetic algorithms have been discussed.

Chapter seven presents the conclusions drawn from this research. It discusses the advantages of the techniques used in this work. Also, some suggestions are made for further research in this area.

ACKNOWLEDGEMENT

At this movement of my substantial enhancement, before we get into the thick of the things, I would like to add a few heartfelt words for the people who gave their unending support with their unfair humor and warm wishes. I am vastly indebted to many people who have helped and inspired me, in various ways, to start, to continue, and complete this work.

First and foremost my gratitude goes to my mentors and supervisors, **Dr. Yaduvir Singh**, **Associate Professor**, Department of Electrical and Instrumentation Engineering, and **Dr. Rajesh kumar**, **Associate Professor**, School of Mathematics and Computer Applications, Thapar University, Patiala, Punjab, India, for supporting me at the conception of this work and strengthens my resolve in so many direct and indirect ways.

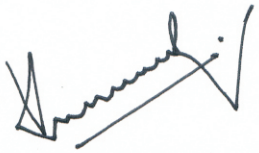
His thoughtful and valuable reviews and suggestions have helped enormously to improve the work. I cannot find words to describe the debt I owe to my Guide, for having created a stimulating atmosphere of academic excellence, the basic element of any long lasting endeavor.

I also feel very much obliged to **Prof. (Dr.) S. Ghosh**, **HEAD**, Department of Electrical and Instrumentation Engineering, Thapar University, Patiala, for his encouragement and for providing full facilities for the execution of this work.

I also wish to express my deepest gratitude to Mrs. Gagandeep Kaur, Sr. Lecturer, Department of Electrical and Instrumentation Engineering, Thapar University, Patiala, for support and help during the lab work.

I heartily acknowledge the cooperation and moral support of my wife, Mrs. Rainu along with my loveable son Arnav throughout this Ph.D. work without which it could have not possible to achieve this target so contentedly.

Last but not the least, I may thanks to my friends and Parents for their efficiency and readiness and heartfelt appreciation towards all those who helped me directly or indirectly to achieve my work.



(Vijay Kumar Banga)

Date: 09/03/2010.

Place: Thapar University, Patiala.

PUBLICATIONS FROM THE RESEARCH WORK

[A] Papers in refereed journals

Banga V. K., Singh Y., Kumar Rajesh**:** Simulation of Robotic Arm using Genetic Algorithm & AHP, International Journal of Computer Systems Science and Engineering, 2007, 2(2): 93-98.

Banga V. K., Singh Y., Kumar Rajesh**:** Uncertainty Compensation and Optimisation of 3 DOF & 4 DOF Robotics System using Genetic Algorithm, International Journal of Advances in Modeling, AMSE, France, 2008, 77 (1): 70-77.

Banga V. K., Singh Y., Kumar Rajesh**:** Robotic Arm Movement Optimisation and Trajectory Planning using Genetic Algorithms, International Journal of Soft Computing Applications (IJSCA), 2008, 3: 69-76.

Banga V. K., Kumar Rajesh, Singh Y.: Fuzzy-Genetic Optimal Control for Robotic Systems, International Journal of Physical Sciences (IJPS), Academy Press, 2011, 6(2): 204-212.

[B] Papers in conference proceedings

Banga V. K., Singh Y., Kumar Rajesh: Modeling and Simulation of Robotic Arm using Genetic Algorithm**,** International Conference on Modeling & Simulation 2006 (MS2006), 3-5 April, 2006, Kaula-Lumpur, Malaysia. (Accepted for Publication, Ref.No.UM.JKM/ CNPAM/ MS2006/ACC/Email/SEC (02-162)/MFJ Dated: 12 Dec. 2005)

Banga V. K., Singh Y., Kumar Rajesh: Movement Optimization of Robotic Arm using Genetic Algorithm & Analytical Hierarchy Process, IEEE International Conference IECON-2006, 7-10 Nov., 2006, Paris, France. (Accepted for Publication, Ref. No. PD-005746, TPC7 Mechatronics & Robotics . Dated: 29 June 2006)

Banga V. K., Kumar Rajesh, Singh Y.: Fuzzy-Genetic Optimal Control For Four Degree Of Freedom Robotic Arm Movement, WCSET 2009: World Congress on Science, Engineering and Technology, 25-27 Dec., 2009, Bangkok, Thailand, 60: 409-412.

Banga V. K., Kumar Rajesh, Singh Y.: Modeling and Simulation of Robotic arm Movement using Soft Computing, International Conference on Electrical, Computer, Electronics and Communication Engineering (ICECECE, 2011), 29-31 March, 2011, Bangkok, Thailand, 75: 617-620.

LIST OF FIGURES AND GRAPHS

LIST OF TABLES

ABBREVIATIONS

Abbreviations	Description
AHP	Analytic hierarchy process
AI	Artificial intelligence
C-space	Cartesian space
DC	Direct current
DNA	Deoxyribonucleic Acid
DOF	Degree-of-freedom
DOFs	Degree-of-freedoms
EA	Evolutionary algorithms
EL	Extremely large
ES	Extremely small
EVL	Extremely very large
EVS	Extremely very small
GAs	Genetic algorithms
GP	Genetic programming
L	Large
LISP	List Processing
M	Medium
MOGA	Multiple objective genetic algorithms
MPC	Model predictive controller
NN	Neural networks
NP	Non-deterministic polynomial
OPF	Optimal power flow
P	Prismatic
PCD	Probabilistic cell decomposition
PID	Proportional integral derivative
R	Revolute
RLFJ	Rigid link flexible joint
S	Small
SA	Simulated annealing

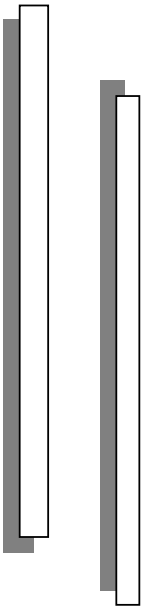
SCARA	Selective compliant assembly robot arm
TSP	Traveling salesman problem
V-grapgh	Visibility graph
VL	Very large
VS	Very small
VVL	Very, very large
VVS	Very, very small
YPR	Yaw-pitch-roll

NOTATIONS AND SYMBOLS

Notation/Symbol	Description
A_1	Movement attribute
A_2	Friction attribute
A_3	Least settling time (Min. vibration) attribute
C_1	$C_1 = \sqrt{x^2 + y^2}$
C_2	$C_2 = (z - L_1)$
D	$D = (C_1^{\ 2} + C_2^{\ 2} + L_4^{\ 2} - L^2)$
D_p	$D_p = (C_1^{\ 2} + C_2^{\ 2} + L^2 - L_4^{\ 2})$
E	$E = (D^2 - 4C_1^{\ 2}L_4^{\ 2})$
E_p	$E_p = (D_p^{\ 2} - 4C_1^{\ 2}L^2)$
F	$F = 4DC_1L_4$
F_p	$F_p = (4 \times D_p C_2 L)$
G	Goal
H	Homogeneous matrix
I	Identity matrix
J_n	Joint with number reference to the link manipulator
L	Length of a combination of link 2 and 3 in 4 DOF manipulator $L = [x_4^{\ 2} + y_4^{\ 2} + (z_4 - L_1)^2]^{1/2}$
LL	Linear joints
L_1	Length of first link in 4 DOF manipulator
L_4	Length of link between joint-4 and end effector in 4 DOF
n	Number of chromosome in a population
P	$-2l_1x'$
P	Target
P_4	Coordinates of joint J_4 (P_4 (x_4, y_4, z_4)
Q	$-2l_1y'$
R	$x'^2 + y'^2 + l_1^2 - l_2^2$

RR	Revolute joints
S	Start
T	Transnational frame
X(1), X(2),….X(M)	Fuzzifier inputs
x'	x - l_3 cos (ϕ)
Y(1), Y(2),….Y(N)	Defuzzifier output
y'	y - l_3 sin (ϕ)
$\theta_1, \theta_2, \theta_3, \alpha, \beta$	Joint angles
β	$\tan^{-1}(y/x)$
γ	$\tan^{-1}$ (Q/P)
σ	±1
f_c	Fitness function
λ	Eigen value
ψ	Pitch angle or rotation of end effector manipulator
θ	Rotation of joint J_1 around the base in 4 DOF Manipulator
ϕ	Rotation of joint J_2 in 4 DOF Manipulator
l_1, l_2, l_2	Length of manipulator links
λ	Eigen values
	Linear motion about an axis
	Rotary motion about an axis

CONTENTS



Chapter 1

Introduction

1.1 Preamble

Industrial robots are used for variety of complex, commercial applications such as palatisation, painting, welding, chip fabrication, *etc*. These robots have to work in repetitive manner in a highly structured predictable environment. The robots are supplied with inputs like position, orientation, *etc*.

Complex work spaces in various industrial environments require tailored jigs and fixtures. It requires large time and incurs huge cost. Industrial robots in such industrial applications need to be installed properly and accurately calibrated for the spatial specification of these jigs and fixtures. However, it is task specific.

Literature survey reveals that engineers and researchers worldwide are attempting to develop a closed loop control system for the robots. Such a closed loop robotic system will posses inherent capability of sensing and reacting to its external work environment. Attempt has been made to develop artificially intelligent robotic systems, which will sense their external environment and operate on the basis of known information for which they have

prepared before hand. It will eliminate the requirements for specialised work spaces, as robots in such configuration will be adaptive and tolerant for different new and unknown environments.

Creating an automated robotic system is a challenging task. It requires various skills like programming, mechanical modeling of robotic systems, electronic devices, their interfaces and control. An integrated design based on above skills and engineering areas will help in realisation of such an automated robotic systems.

An automated robotic system must plan its movement optimally. This is the subject of this research work. Following considerations have been made:

1. Robots and robotic arms are very much popular in the industrial applications like assembly operations, spray-painting, grinding, welding, pick and place operations. To conserve the energy, it is desired to get the maximum mechanical work with the lesser energy input, it calls for optimising the movement of robotic arm in the wake of finding optimal path. Also the accuracy of the movement of robotic arm's probabilistic disturbances, non linearities and uncertainties need to be compensated.
2. Uncertainty like friction in the joint is to be predicted and compensated in the robotic arm movement with the help of artificial intelligence technique like fuzzy logic.
3. Inverse kinematics is non unique and hence the state of the art optimising technique like genetic algorithms will be used for finding out an optimal solution of the robotic arm movement.

Robots with medium and large degrees of freedoms meant for commercial and industrial applications are programmed by supervision. However, unsupervised learning can also be imparted. In the learning process of robots, it is trained for a given task by specifying the desired sequences.

A learning process of an automated robotic system is specified as start, end and intermediate stages of motion. Care is required for eliminating any chance of the collision of robotic arm with objects in its vicinity. A simple motion programming of robotic arm may be "guiding". However, it fails to program robot in the complex environment. A complex industrial environment for robotic arm operation is dominated by known and unknown unpredictable end coordinates. Various sensors are used for acquiring various spatial information in on-line mode, which give robotic arm movement. An automated robotic system must possess ability to get programmed for variety of arm movement for all

type of sensor data. The robotic arm movements can be controlled by efficient algorithms using simple programming languages.

Today, industrial robots programmed for on-line decision making with advanced instrumentation and data bases. The robots required programming environment in specific programming language with graphical modeling. It also required off-line simulation and testing for new tasks of programming. Such programming requires high level programming skills and also reduces off-line timings of the robot programming.

Fig. 1.1 shows block diagram of an automated robotic system. To move robotic arm for particular task, programmer generates appropriate sequence of steps taking information of its environment and actuates on the basis of system databases. It generates trajectory to perform the particular task such that controller actuated by control signals to the robotic arm in manner of trajectory planned. To make programming simple it is required to move manipulator to the task level. The manipulator moves according to the task of the robotic arm to achieve the target. Programmer plans the task of the robotic arm and correspondingly generates necessary instruction to the manipulator to achieve the task.

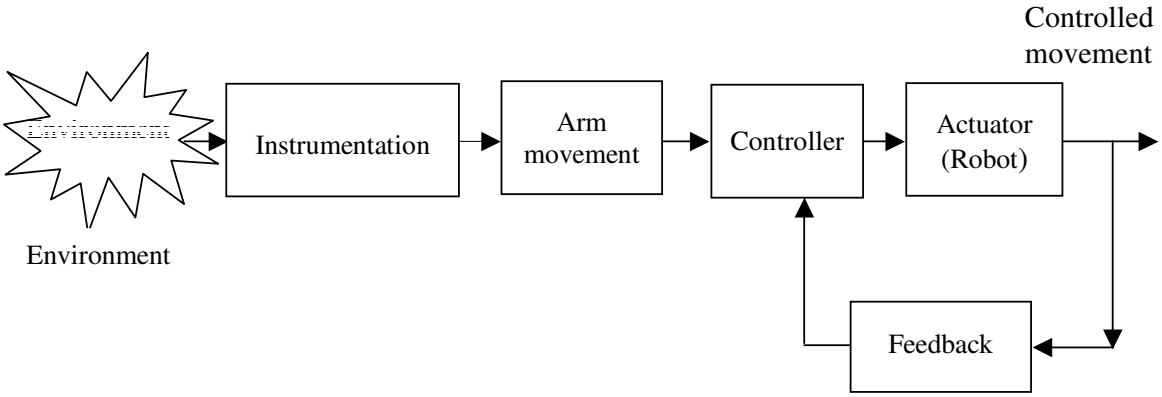


Fig. 1.1: Block diagram representation of closed loop robotic system

In robot path planning care should be taken that it moves free between start and target points without any collision with obstacles in the environment, which is a key factor of the motion planning. This is the core problem for the researcher in an automated robot to achieve collision free path planning in the wake of non-deterministic parameters like energy consumption, friction and settling time, *etc.* Then these non-deterministic parameters are dependent upon the shortest path and minimum movement. In robot path planning it requires to move manipulator from start to target (end), collision free in the work space for particular configuration of robot arm manipulator.

A robotic arm system is characterised by degree-of–freedom. Robot search space includes all possible configuration of robotic arm. Robot search space increases in its size as the size, complexity and the number of degree-of–freedom increase. Motion planning and trajectory solutions require greater efforts in the case of huge robotic search spaces. Performance of the robotic arm is also indexed in terms of "speed". Motion planners operate on-line and off-line. A closed loop automated robotic system requires on-line planners. A typical application of on-line planners is the task, which passes through plan-move-plan-move cycle. In such an application the actuator offered by controller does not vary dynamically for each plan-move duration. On-line planners are used in an automated robotic systems, which operate in "real time". A "real time" automated robotic system needs to have very high speed as the path planner iterates its planning process several times. Also, such a real time automated robotic system needs to adapt path changes within its specified work spaces in complicated trajectory. There is no time available with the controller to stop to think about its movement.

An off-line planner is normally used in non-automated robotic systems. An off-line planner requires an operator for path planning. Literature survey reveals that the recently many path planning methods have been developed. Path planning can be done very conveniently for robotic arms with lower degree-of-freedom. Some of these developed path planning methods meet the requirement of on-line mode of robot operation but there is not even a single planning method, which could be applied for real time operation of robotic systems.

Complex industrial robotic system posses higher degree-of-freedom (DOF). There are very few motion and path planning methods which could support such robotic applications and the planning method reported are only applicable in off-line mode. Search space and its representation is very crucial for path planning for the purpose of evaluation of an efficient motion and path planning method. The robot and its environment is represented in its "configuration space". An industrial robot, for example with 4 DOF may have one translational and three rotational motions. The motion of robotic arm is characterised and constrained by number of joints and their types. Every joint contributes one DOF. The concept of configuration space provides for meaningful solution for motion and path planning for the robotic arm having multiple DOF.

Basically, robotic arm is configured in terms of three parameters, which are defined over a local coordinate system. Out of three parameters, two parameters contribute local origin in terms of *x-y* positions. Third parameter gives the details about the orientation of local coordinates system. It is also known as local frame.

A robotic arm is configured in terms of joint angle vector in a given configuration space. Each coordinate gives a unique and possible robot configuration depending upon the number of DOFs. The dimensions of configuration space are proportionately defined. For example a two DOF robot can be characterised by its two dimensional configuration space. Three DOF robotic arm can be characterised by its three dimensional configuration space. Four DOF robotic arm can be characterised by its four dimensional configuration space.

In motion and path planning of robotic arm, obstacle is yet another very important issue. It is not in the scope of this research work. All the obstacles are defined in terms of spatial coordinates and are considered as "constraints" for arm movement. Where collision between robotic arm and obstacle may take place such origins in configuration space are clearly separated out and are proclaimed to be "forbidden region". Excluding such forbidden regions in the configuration space, the remaining allowable space forms "free space".

Motion and path planning for a robotic arm is the problem of trajectory planning in the given free space of configuration space. The problem of path planning for multiple DOF robotic arm can be viewed as motion and path planning of a single point in such multi-dimensional configuration space. It forms the basis of motion and path planning.

Motion and path planning is broadly characterised into three groups, *viz*., road map based, cell decomposition based and potential feed based. Road map is one dimensional curve. A path planning based on road map approach gives a trajectory, which is network of such several road maps. Star and goal configuration need to be clearly defined in road map approach. An optomiser will search such network for optimal connectivity yielding an optimal path.

Cell decomposition method is based on exact and approximate decomposition of set of free configurations into much simpler and non overlapping regions, which are commonly known as "cells". Cell decomposition method is used very widely. Connectivity graph represents such cells, which are used for path selection. As the number of DOF increases, number of cells also increases "in an exponential manner". The prerequisite of this method is all cells must be ascertained before hand and the connectivity graph to be developed. However, cell decomposition method is not recommended for large DOFs.

Potetial field method is suitable for path planning in the wake of obstacles. Two types of fields are created, *viz.,* attractive field and repulsive field. Attractive field is created around the goal where as repulsive force is created around obstacles. The gradient of combined potential is calculated which helps in finding a path towards a good and far away from

obstacles. Path planners based on potential field method usually get struck up into local minima and yield improper trajectory planning. This method is based on potential function.

1.2 Fuzzy Logic

The term "fuzzy" was first used by Dr. Lotfi Zadeh in the engineering journal, "Proceedings of the IRE," a leading engineering journal, in 1962. There are already fuzzy logic based commercial products such as self-focusing cameras, washing machines that adjust themselves according to how dirty the clothes are, automobile engine controls, anti-lock braking systems, color-film developing systems and subway control systems, trading successfully in the financial markets. The credit for fuzzy logic application to the areas of control and in engineering belongs solely to Zadeh. In 1965, he formalized fuzzy set theory (Zadeh L.A. 1965). According to Zadeh, fuzzy logic brings to control systems a "higher machine intelligence quotient" (Zadeh L.A. 1996).

On a mathematical level, fuzzy logic abandons the strict bivalent logic of TRUE and FALSE, ONE and ZERO, ON and OFF. Fuzzy logic allows for half-truths. On an engineering level, fuzzy logic provides a platform for easily encoding human knowledge into the control of a system. It has been used in an increasing number of applications, especially in Japan. The Sendai railway in Japan is controlled by fuzzy logic controllers. Applications have been developed in tracking problems, tuning, interpolation, classification, handwriting, voice recognition, image stabilisation in video cameras, washing machines, vacuum cleaners, air conditioners, electric fans, hot plates, and automatic transmissions, *etc.* Fuzzy logic is a method of characterising knowledge in terms of fuzzy sets and a rule base. A fuzzy system has one or more inputs that are fuzzified. A rule base is evaluated according to the inputs. One or more outputs are defuzzified into "crisp" values. A fuzzy system structure is illustrated in Fig. 1.2. Bringing fuzzy logic to control problems is a way to use a human expert's knowledge about an analog process in a digital computer. Fuzzy logic is not always the best way to solve a control problem, but it offers several advantages.

Fuzzy sets are values to which a variable can belong. Fuzzy control involves describing the control procedure in terms of subjective descriptions like "very low", "low", "just right", "high", and "very high". Suppose a human were to describe the operation of an elevator. If the elevator did not stop exactly on the desired floor, he might describe the elevator's error in terms of a fuzzy variable like the one above. If the elevator stopped three feet too high, the position of the elevator would definitely be "very high". Six inches too high might just be "high". The question is, "At what height did the elevator go from just being 'high' to

being 'very high'?". Fuzzy logic avoids this problem by its multivalent nature. A fuzzy variable can have a certain degree of membership to "high" and a degree of membership to "very high". These fuzzy sets are graphed as shown in Fig. 1.3.

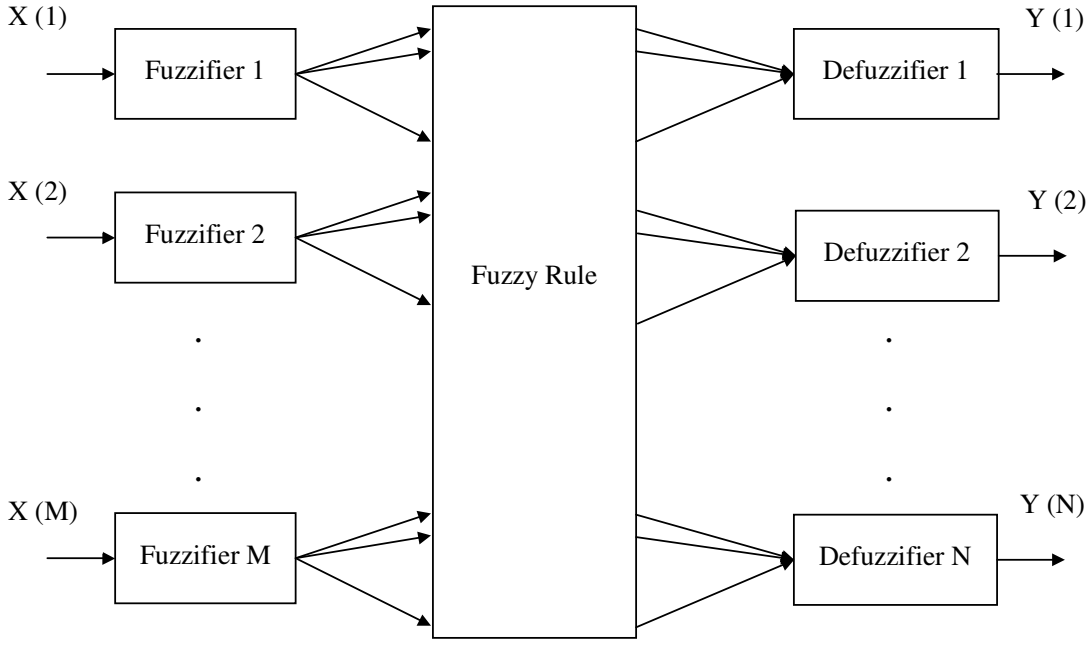


Fig. 1.2: Fuzzy system

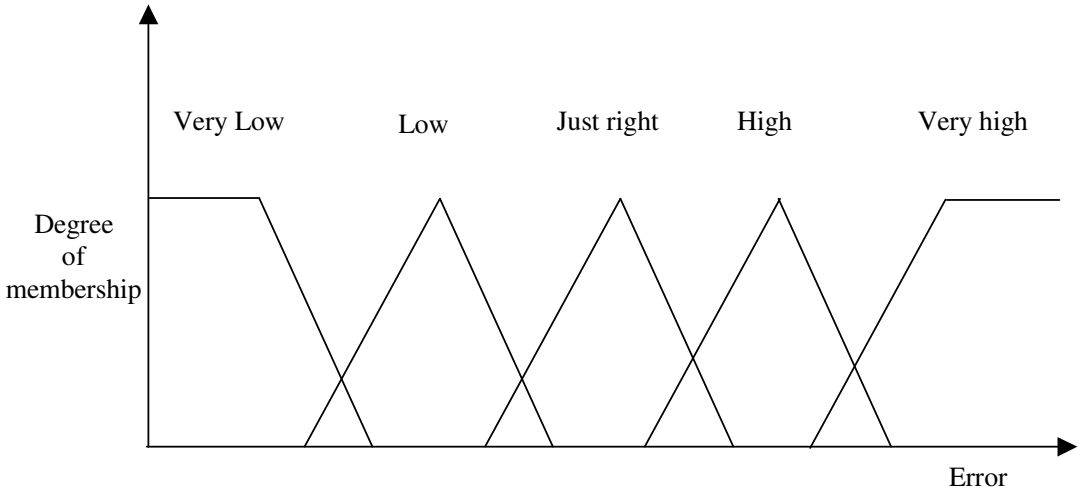


Fig. 1.3: Fuzzy set

The control rules are defined with these fuzzy sets. These control rules are usually in the form of IF/THEN statements. "If error is very high, then voltage is negative large." or "If error is very low and load is very high, then motor power is positive large". Combining several if-conditions necessitates the use of fuzzy logic operations. Zadeh defines these basic operations in Eqs. (1.1), (1.2), and (1.3).

$$A \cap B = \min(A, B) \quad \text{----(1.1)}$$

$$A \cup B = \max(A, B) \quad \text{----(1.2)}$$

$$A' = \min(A, B) \quad \text{----(1.3)}$$

Suppose a rule in the rule base stated, "If error is very high and load is zero, motor power is negative small". Consider for an example that the elevator was located a distance of 1.5 feet above the desired floor. The distance of 1.5 feet might have a 0.4 degree of membership in error's fuzzy value very high. The weight of a child in the elevator might cause a 0.5 degree of membership in load's fuzzy value zero. According to Eq. (1.1), the rule's antecedent "If ... and ..." would evaluate as min (0.4, 0.5) = 0.4. Therefore, in determining the control, the output variable would have a 0.4 degree of membership in motor power's fuzzy value of negative small. Usually, an output of a fuzzy system is determined by more than one rule and the total output is calculated according to the centeroid of the output membership functions.

Sometimes a fuzzy variable has a degree of membership to a fuzzy value of 0.5. "The elevator is not quite low enough to qualify as 'high', but it is not 'very high' either". For the special case of A = 0.5, Eq. (1.3) evaluates as (not A)=A. This contradicts traditional, bivalent logic. Traditional zero-or-one logic is a special case of fuzzy logic. If the state space is considered to be a hyperspace cube, whose axes correspond to individual fuzzy variables, traditional logic holds true for the corners of the cube. Fuzzy logic applies to all points in the state space.

Fuzzy logic is helpful in situations where the control variables are continuous. Fuzzy rules often take the place of a mathematical model. Therefore, fuzzy logic is useful if a mathematical model of a process does not exist, is too difficult to encode, is too complex to be evaluated in real-time, or requires too much memory. Other situations that may make fuzzy control advantageous are when there are high ambient noise levels, it is important to use inexpensive sensors, or it is important to use low precision microcontrollers. They are easier to prototype and implement and simpler to describe and verify. They can be maintained and extended with greater accuracy in less time.

Control of some systems cannot be easily specified in terms of an IF/THEN rule base. An example of such a system would be a robotic arm operating in the presence of an obstacle. Also, sometimes the 'experts' providing the rule base disagree among themselves. This was demonstrated at the Kawasaki Steel Corporation in Japan. A fuzzy logic control system was installed to help operators make decisions regarding control of a blast furnace. The researchers summarised, "There is a slight difference in knowledge between a multiple number of experts". In the case of expert systems which process ill structured problems, an 100% success could not be possible. Their data suggests that, in practice, the operators ignore the expert system's suggestion over 15% of the time.

1.2.1 Fuzzy Rules

Fuzzy rule base is based upon truth table of logic. Rule base is a collection of rules related to the fuzzy sets, the input variables and output variables. These rules make the system to decide what to do. These rules relate the input and output variables. Consider the air conditioning system in which temperature and humidity are the two input variables and power setting is output variable. Fuzzy rule may be written as

If temperature is hot AND humidity is humid then power is high.

However, the logic of the same rule may be written as

If temperature is hot OR humidity is humid then power is high**.**

As these two rules differ by AND/OR operator their behaviour is different. For "AND" operator both condition must be satisfied for output and for "OR" operator either condition is satisfied for output. The result of an "AND" operation is the minimum of the two variable values. The result of an "OR" operation is the maximum of the two values. For given problem, logic can check all the rules for the given inputs and generate corresponding outputs. It is called as fuzzy inference engine. The total number of rules in a rule base is equal to the product of the number of sets of each input variables.

1.2.2 Defuzzification

Defuzzification is an important operation in the theory of fuzzy sets. It transforms a fuzzy set information into a numeric data information. This operation along with the operation of fuzzification is critical to the design of fuzzy systems as both of these operations provide nexus between the fuzzy set domain and the real valued scalar domain. One needs the synergy of both of these domains to solve many of our ill-posed problems effectively.

1.3 Genetic Algorithms

Genetic Algorithms(GAs) are a family of computational models inspired by evolution. These algorithms encode a potential solution to a specific problem on a simple chromosome-like data structure and apply recombination operators to these structures so as to preserve critical information. Genetic algorithms are often viewed as function optimisers. Although the range of problems to which genetic algorithms have been applied is quite broad.

An implementation of a genetic algorithm begins with a population of (typically random) chromosomes. One then evaluates these structures and allocates reproductive opportunities in such a way that those chromosomes which represent a better solution to the target problem are given more chances to reproduce than those chromosomes which are poorer solutions.

The “goodness” of a solution is typically defined with respect to the current population. This particular description of a genetic algorithm is intentionally abstract because in some sense, the term genetic algorithm has two meanings. In a strict interpretation, the genetic algorithm refers to a model introduced and investigated by John Holland (1975) and their students.

In a broader usage of the term, genetic algorithms is any population-based model that uses selection and recombination operators to generate new sample points in a search space. Many genetic algorithm models have been introduced by researchers largely working from an experimental perspective. Many of these researchers are application oriented and are typically interested in genetic algorithms as optimisation tools.

Genetic algorithms are a part of evolutionary computing, which is a rapidly growing area of artificial intelligence. Genetic algorithms are inspired by Darwin's theory of evolution. Simply said, problems are solved by an evolutionary process resulting in a best (fittest) solution (survivor) - in other words, the solution is evolved.

In the mid 1800s, a British naturalist named Charles Darwin published a book that would change the way humans view the world. In this book, *The Origin of Species*, Darwin proposed that humans and in fact all creatures, were not put on this planet by God and made unchanging, but rather that they evolved from other creatures. At the time, the idea sounded preposterous. Advances in technology have made it possible for us to read our DNA and that of other creatures. Over time, creatures change to adapt to their environment to survive and thrive.

Evolutionary computing was introduced in the 1960s by I. Rechenberg in his work "*Evolution strategies*" (*Evolutionsstrategie* in original). His idea was then implemented by other researchers. Genetic Algorithms were invented by John Holland and developed by him and his students and colleagues. This lead to Holland's book "*Adaption in Natural and Artificial Systems*" published in 1975.

In 1992 John Koza has used genetic algorithm to evolve programs to perform certain tasks. He called his method "Genetic Programming" (GP). LISP programs were used, because programs in this language can easily be expressed in the form of a "parse tree", which is the object the genetic algorithms work.

Chromosome

All living organisms consist of cells. In each cell there is the same set of chromosomes. Chromosomes are strings of DNA and serve as a model for the whole organism. A chromosome consists of genes, blocks of DNA. Each gene encodes a particular protein. Basically, it can be said that each gene encodes a trait, for example color of eyes. Possible settings for a trait (*e.g.,* blue, brown) are called alleles. Each gene has its own position in the chromosome. This position is called locus.

Complete set of genetic material (all chromosomes) is called genome. Particular set of genes in genome is called genotype. The genotype is with later development after birth base for the organism's phenotype, its physical and mental characteristics, such as eye color, intelligence, *etc*.

Reproduction

During reproduction, recombination (or crossover) first occurs. Genes from parents combine to form a whole new chromosome. The newly created offspring can then be mutated. Mutation means that the elements of DNA are a bit changed. These changes are mainly caused by errors in copying genes from parents.

The fitness of an organism is measured by success of the organism in its life (survival).

Search Space

If we are solving a problem, we are usually looking for some solution, which will be the best among others. The space of all feasible solutions (the set of solutions among which the desired solution resides) is called search space (also state space). Each possible solution can be "marked" by its value (or fitness) for the problem. With genetic algorithm, we look for the best solution among a number of possible solutions - represented by one point in the search space.

Looking for a solution is then equal to looking for some extreme value (minimum or maximum) in the search space. At times the search space may be well defined, but usually we know only a few points in the search space. In the process of using GAs, the process of finding solutions generates other points (possible solutions) as evolution proceeds.

The problem is that the search can be very complicated. One may not know where to look for a solution or where to start. There are many methods one can use for finding a suitable solution, but these methods do not necessarily provide the best solution. Some of these methods are hill climbing, tabu search, simulated annealing and the genetic algorithms. The solutions found by these methods are often considered as good solutions, because it is not often possible to prove what the optimum is.

NP-Hard Problems

One example of a class of problems, which cannot be solved in the "traditional" way, are NP problems. There are many tasks for which we may apply fast (polynomial) algorithms. There are also some problems that cannot be solved algorithmically.

There are many important problems in which it is very difficult to find a solution, but once we have it, it is easy to check the solution. This fact led to NP-complete problems. NP stands for nondeterministic polynomial and it means that it is possible to "guess" the solution (by some nondeterministic algorithm) and then check it. If we had a guessing machine, we might be able to find a solution in some reasonable time.

Studying of NP-complete problems is, for simplicity, restricted to the problems where the answer can be yes or no. Because there are tasks with complicated outputs, a class of problems called NP-hard problems has been introduced. This class is not as limited as class of NP-complete problems.

A characteristic of NP-problems is that a simple algorithm, perhaps obvious at a first sight, can be used to find usable solutions. But this approach generally provides many possible solutions - just trying all possible solutions is very slow process. For even slightly bigger instances of these types of problems, this approach is not usable at all.

Today, nobody knows if some faster algorithms exist to provide exact answers to NP-problems. The discovery of such algorithms remains a big task for researchers. Today, many people think that such algorithms do not exist and so they are looking for an alternative method. An example of an alternative method is the genetic algorithms.

Examples of the NP problems are satisfiability problem, traveling salesman problem (TSP) or knapsack problem, *etc.*

Basics of Genetic Algorithms

Genetic algorithms are inspired by Darwin's theory of evolution. Solution to a problem solved by genetic algorithms uses an evolutionary process (it is evolved). The most common type of genetic algorithms works like this.

A population is created with a group of individuals created randomly. The individuals in the population are then evaluated. The evaluation function is provided by the programmer and gives the individuals a score based on how well they perform at the given task. Two individuals are then selected based on their fitness. The higher the fitness, the higher the chance of being selected. These individuals then "reproduce" to create one or more offspring, after which the offspring are mutated randomly. This continues until a suitable solution has been found or a certain number of generations have passed, depending on the needs of the programmer.

Outline of the Basic Genetic Algorithms

[Start]: Generate random population of *n* chromosomes (suitable solutions for the problem).

[Fitness]: Evaluate the fitness *f(x)* of each chromosome *x* in the population.

[New population]: Create a new population by repeating following steps until the new population is complete.

a. Encoding: The encoding mainly depends on the solved problem. Various encodings are used depending upon the problem like value, binary, integer or real number, *etc*.

b. Selection: Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected).

c. Crossover: With a crossover probability, cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.

d. Mutation: With a mutation probability, mutate new offspring at each locus (position in chromosome).

e. Accepting: Place new offspring in the new population.

[Replace]: Use new generated population for a further run of the algorithm.

[Test]: If the end condition is satisfied, stop, and return the best solution in current population.

[Loop]: Go to step [Fitness].

Selection of parameters for crossover is done in many ways. The main idea is to select the better parents (best survivors) in the hope that the better parents will produce better

offspring. You may think that generating populations only from two parents may cause to loose the best chromosome from the last population. This is true, and so elitism is often used. This means, that at least one of a generation's best solution is copied without changes to a new population, so the best solution can survive to the succeeding generation.

1.3.1 Operators of Genetic Algorithms

The crossover and mutation are the most important parts of the genetic algorithm. The performance is influenced mainly by these two operators.

Encoding

A chromosome contains information about solution that it represents. The most used way of encoding is a binary string as shown in Fig. 1.4.

Chromosome 1	1101100100110110
Chromosome 2	1101111000011110

Fig. 1.4: Encoding of chromosome

Each chromosome is represented by a binary string. Each bit in the string represents some characteristics of the solution. Another possibility is that the whole string can represent a number. There are many other ways of encoding. The encoding depends mainly on the solved problem. For example, one can encode directly integer or real numbers; sometimes it is useful to encode some permutations and so on.

Crossover

After encoding, one can proceed to crossover operation. Crossover operates on selected genes from parent chromosomes and creates new offspring. The simplest way how to do that is to choose randomly some crossover point and copy everything before this point from the first parent and then copy everything after the crossover point from the other parent. Crossover is illustrated as shown in Fig. 1.5 (| is the crossover point).

There are other ways to make crossover, for example we can choose more crossover points. Crossover can be quite complicated and depends mainly on the encoding of chromosomes. Specific crossover made for a specific problem can improve performance of the genetic algorithms.

Chromosome 1	11011 | 00100110110
Chromosome 2	11001 | 11000011110
Offspring 1	11011 | 11000011110
Offspring 2	11001 | 00100110110

Fig. 1.5: Crossovers

Mutation

After a crossover is performed, mutation takes place. Mutation is intended to prevent falling of all solutions in the population into a local optimum of the solved problem. Mutation operation randomly changes the offspring resulted from crossover. In case of binary encoding, we can switch a few randomly chosen bits from 1 to 0 or from 0 to 1. Mutation can be then illustrated as shown in Fig. 1.6.

Original offspring 1	110**1**1110000011110
Original offspring 2	1100100100110110
Mutated offspring 1	110**0**1110000011110
Mutated offspring 2	1100101100110110

Fig. 1.6: Mutation

The technique of mutation (as well as crossover) depends mainly on the encoding of chromosomes.

1.3.2 Parameters of Genetic Algorithms

Following are the various parameters of genetic algorithms.

Crossover Probability

Crossover probability is defined as how often crossover will be performed. If there is no crossover, offspring are exact copies of parents. If there is crossover, offspring are made from parts of both parent's chromosome. If crossover probability is 100%, then all offspring are made by crossover. If it is 0%, whole new generation is made from exact copies of

chromosomes from old population (but this does not mean that the new generation is the same).

Crossover is made in hope that new chromosomes will contain good parts of old chromosomes and therefore the new chromosomes will be better. However, it is good to leave some part of old population survives to next generation.

Mutation Probability

Mutation probability is defined as how often parts of chromosome will be mutated. If there is no mutation, offspring are generated immediately after crossover (or directly copied) without any change. If mutation is performed, one or more parts of a chromosome are changed. If mutation probability is 100%, whole chromosome is changed, if it is 0%, nothing is changed. Mutation generally prevents the GAs from falling into local extremes. Mutation should not occur very often, because then GAs will in fact change to random search.

Population Size

Population size is defined as how many chromosomes are in population (in one generation). If there are too few chromosomes, GAs has few possibilities to perform crossover and only a small part of search space is explored. On the other hand, if there are too many chromosomes, GAs slows down. Research shows that after some limit (which depends mainly on encoding and the problem), the use of very large population does not solve the problem faster than moderate sized population.

1.3.3 Selection of Chromosomes

For the purpose of crossover two best chromosomes are selected from the population, which are called as parents. The problem is how to select these chromosomes. According to Darwin's theory of evolution the best ones survive for creating new offspring.

There are many methods in selecting the best chromosomes such as Roulette wheel selection, Rank selection, Steady state selection, Elitism, *etc.*

Roulette Wheel Selection

Parents are selected according to their fitness. The better the chromosomes are, the more chances to be selected they have.

Imagine a Roulette wheel where all the chromosomes in the population are placed. The size of the section in the Roulette wheel is proportional to the value of the fitness function of every chromosome - the bigger the value is, the larger the section is. Roulette wheel selection is shown in Fig. 1.7.

A marble is thrown in the Roulette wheel and the chromosome where it stops is selected. Clearly, the chromosomes with bigger fitness value will be selected more times.

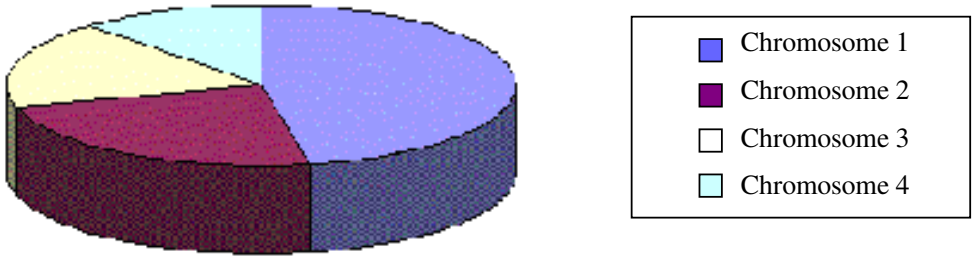


Fig. 1.7: Roulette wheel selection

Rank Selection

The previous type of selection will have problems when they have big differences between the fitness values. For example, if the best chromosome fitness is 90% of the sum of all fitnesses then the other chromosomes will have very few chances to be selected.

Rank selection ranks the population and every chromosome receives fitness value determined by this ranking. The best will have the fitness 1, the second best 2, *etc.,* and the worst will have fitness *N* (number of chromosomes in population).

Fig. 1.8 and 1.9 depicts the situation changes after changing fitness to the numbers determined by the ranking.

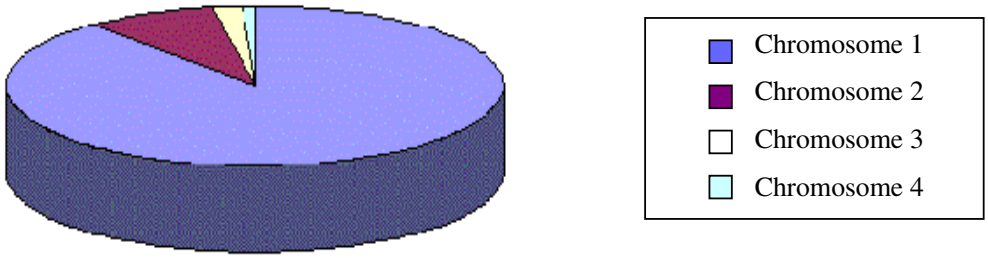


Fig. 1.8: Situation before ranking

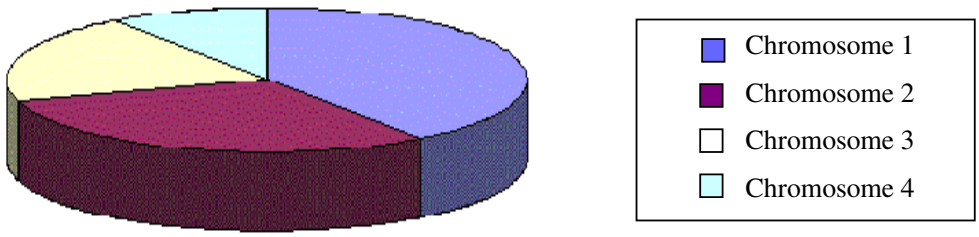


Fig. 1.9: Situation after ranking

Now all the chromosomes have a chance to be selected. However this method can lead to slower convergence, because the best chromosomes do not differ so much from other ones.

Steady State Selection

This is not a particular method of selecting parents. The main idea of this type of selecting to the new population is that a big part of chromosomes can survive to next generation. In every generation a few good (with higher fitness) chromosomes are selected for creating new offspring. Then some bad (with lower fitness) chromosomes are removed and the new offspring is placed in their place. The rest of population survives to new generation.

Elitism

When creating a new population by crossover and mutation, we have a big chance, that we will loose the best chromosome. Elitism is the name of the method that first copies the best chromosome (or few best chromosomes) to the new population. The rest of the population is constructed in similar manner. Elitism can rapidly increase the performance of GAs, because it prevents a loss of the best found solution.

1.3.4 Encoding

Encoding of chromosomes is the first question to ask when starting to solve a problem with GAs. Encoding depends on the problem heavily.

Binary Encoding

Binary encoding is the most common as it is used by most of the researchers of GAs because of its relative simplicity.

In binary encoding, every chromosome is a string of bits - 0 or 1 as shown in Fig. 1.10.

Chromosome A	10110010110010101110010 1
Chromosome B	11111110000011000001111 1

Fig. 1.10: Chromosomes with binary encoding

Binary encoding gives many possible chromosomes even with a small number of alleles. On the other hand, this encoding is often not natural for many problems and sometimes corrections must be made after crossover and/or mutation.

Permutation Encoding

Permutation encoding can be used in ordering problems such as traveling salesman problem or task ordering problem. In permutation encoding, every chromosome is a string of numbers that represent a position in a sequence as shown in Fig. 1.11.

Chromosome A	1 5 3 2 6 4 7 9 8
Chromosome B	8 5 6 7 2 3 1 4 9

Fig. 1.11: Chromosomes with permutation encoding

Permutation encoding is useful for ordering problems. For some types of crossover and mutation, corrections must be made so that chromosomes remain consistent.

Value Encoding

Direct value encoding can be used in problems where some more complicated values such as real numbers are used. Use of binary encoding for this type of problems would be difficult.

In the value encoding, every chromosome is a sequence of some values. Values can be anything connected to the problem, such as (real) numbers, characters or any objects, as shown in Fig. 1.12.

Chromosome A	1.2644 5.4243 6.3456 0.2341 2.1342
Chromosome B	ABCFE BGDFR AGFER YTRFG DEFSE
Chromosome C	(forward), (right), (left), (black), (red)

Fig. 1.12: Chromosomes with value encoding

Value encoding is a good choice for some special problems. However, for this encoding it is often necessary to develop some new crossover and mutation specific for the problem.

Tree Encoding

Tree encoding is used mainly for evolving programs or expressions, *i.e.,* for genetic programming. In the tree encoding every chromosome is a tree of some objects, such as functions or commands in programming language as shown in Fig. 1.13.

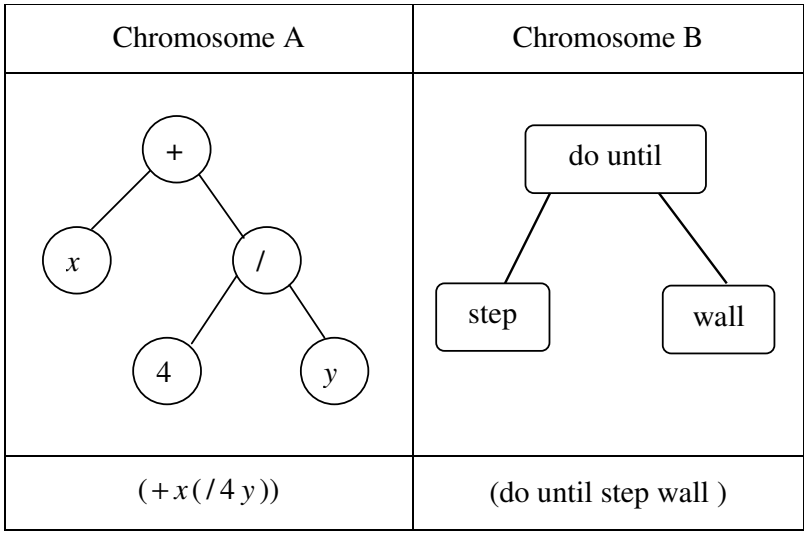


Fig. 1.13: Chromosomes with tree encoding

Tree encoding is useful for evolving programs or any other structures that can be encoded in trees. Programming language LISP is often used for this purpose, since programs in LISP are represented directly in the form of tree and can be easily parsed as a tree, so the crossover and mutation can be done relatively easily.

1.3.5 Crossover and Mutation

Crossover and mutation are two basic operators of GAs and performance of GAs depends very much on these operators. The type and implementation of operators depends on the encoding and also on the problem.

There are many ways to perform crossover and mutation.

Crossover with Binary Encoding

Single Point Crossover: One crossover point is selected, binary string from the beginning of the chromosome to the crossover point is copied from the first parent, and the rest is copied from the other parent (Fig. 1.14)**.**

(Parent A)		(Parent B)		(Offspring)
11001011	+	11011**111**	=	**11001111**

Fig. 1.14: Single point crossover

Two Point Crossover: Two crossover points are selected, binary string from the beginning of the chromosome to the first crossover point is copied from the first parent, the part from the first to the second crossover point is copied from the other parent and the rest is copied from the first parent again (Fig. 1.15).

(Parent A)		**(Parent B)**		**(Offspring)**
11001**011**	+	1**1011111**	=	**11011111**

Fig. 1.15: Two point crossover

Uniform Crossover: Bits are randomly copied from the first or from the second parent (Fig.1.16).

(Parent A)		(Parent B)		(Offspring)
1**10**01**011**	+	**110111**01	=	11011111

Fig. 1.16: Uniform point crossover

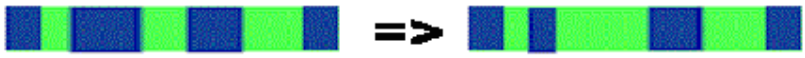
Arithmetic Crossover: Some arithmetic operation is performed to make a new offspring (Fig.1.17).

(Parent A)		(Parent B)		(Offspring)
11001011	+	11011111	=	11001001 (**AND**)

Fig. 1.17: Arithmetic crossover

Mutation with Binary Encoding

Bit Inversion: Selected bits are inverted (Fig. 1.18).



(After crossover) (After muatation)

1**1**001001 ⇨ 1**0**001001

Fig. 1.18: Bit Inversion

Crossover with Permutation Encoding

Single Point Crossover: One crossover point is selected, the permutation is copied from the first parent till the crossover point, then the other parent is scanned and if the number is not yet in the offspring, it is added. There are many ways to produce the rest after crossover point.

(**1 2 3 4 5** 6 7 8 9) + (**4 5 3 6 8** 9 7 2 1) = (**1 2 3 4 5** 6 8 9 7)

Mutation with Permutation Encoding

Order Changing: Two numbers are selected and exchanged.

(1 **2** 3 4 5 6 **8** 9 7) => (1 **8** 3 4 5 6 **2** 9 7)

Crossover with Value Encoding

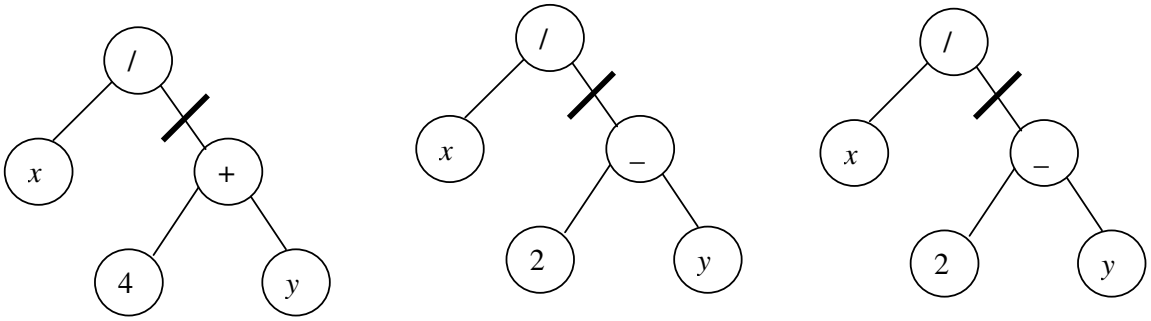
Mutation with Permutation Encoding

Adding a Small Number: A small number is added to (or subtracted from) selected values.

(1.29 5.68 2.86 4.11 5.55) => (1.29 5.68 **2.73** **4.22** 5.55)

Crossover with Tree Encoding

Tree Crossover: One crossover point is selected in both parents, parents are divided in that point and the parts below crossover points are exchanged to produce new offspring as shown in Fig. 1.19.



(Parent A) (Parent B) (Offspring)

Fig. 1.19: Crossover with tree encoding

Mutation with Tree Encoding

Changing Operator, Number: Selected nodes are changed.

Crossover Rate

Crossover rate should be high generally, about 80%-95%. However, some results show that for some problems crossover rate about 60% is the best.

Mutation Rate

On the other side, mutation rate should be very low. Best rates seem to be about 0.5% -1%.

Population Size

It may be surprising, that very big population size usually does not improve performance of GAs (in the sense of speed of finding solution). Good population size is about 20-30, however sometimes sizes 50-100 are reported as the best. Some research also shows that the best population size depends on the size of encoded string (chromosomes). It means that if we have chromosomes with 32 bits, the population should be higher than for chromosomes with 16 bits.

Selection

Basic Roulette wheel selection method can be used, but sometimes rank selection can be better. There are also some more sophisticated methods that change parameters of selection during the run of GAs. Basically, these behave similarly to simulated annealing. Elitism should be used for sure if you do not use other method for saving the best found solution. Steady state selection method can also be used.

Encoding

Encoding depends on the problem and also on the size of instance of the problem.

Crossover and Mutation Type

Operators depend on the chosen encoding and on the problem.

1.4. Analytic Hierarchy Process

Analytic hierarchy process (AHP) is to structure complexity in gradual steps from the large to the small, or the general to the particular, so that we can relate them with greater accuracy according to our understanding. Because experience is too vast to lay it out in a single network structure, we are satisfied with piecemeal decompositions and with occasional linkages of them. AHP is considered effective approach to deal with complex decision-making. It allows a better, easier and more efficient identification of selection criteria, their weighing and analysis. The purpose is to improve our awareness by richer synthesis of our knowledge and intuition. The AHP is a learning tool. It is not a means to discover the TRUTH because truth is relative and changing.

The reciprocal property is the most fundamental aspect for creating a scale. A hierarchy is an efficient way to organise complex system, and functionally, for controlling and passing information down the system. Unstructured problems are best grappled within the systematic framework of the hierarchy or a feedback network.

1.4.1 Steps of the Analytical Hierarchy Process

The steps of analytical hierarchy process are given below.

a. Decomposing: The first objective is to structure the problem into manageable sub-problem form in the manner that the sub-problem modules will become sub hierarchies. Initially, the problem is unstructured and may be in non-manageable form. As the problem is structured from top to bottom, AHP structures into systematic branches and nodes, evaluation parameters and alternative ratings, to measure the adequate solution for the specified criterion. As the number of criterion increases, the importance of each criterion is thus diluted, weight assigned to each criterion represents importance.

b. Weighing: This step assigns the relative weight to each criterion, as accordance with their importance within the module. The total sum of all the criteria belongs to a common module in the same hierarchy level must be equal to 1 or 100%. A global priority is evaluated which indicates the relative importance of a criterion within the overall structure problem model.

c. Evaluating: In AHP, relative score for each alternative is assigned within the hierarchy and compare each one with others. By comparing all alternatives, an overall score is evaluated for selection.

d. Selecting: The entire possible alternative compares and selects among them the best for objective of the problem.

AHP in some cases, when the possibility of different hierarchy being applied to similar problems, there is possibility of major changes in results if there is minor change in hierarchy. Still AHP is used widely as a decision-making method and works well in practice.

1.5 Hybrid Systems

Hybrid systems use more than one technology to solve the problems. Hybridisation of various technologies helps to solve the problem. These technologies could not find solutions individually for the same problem. The objective of hybridisation has been to overcome the weaknesses in one technology during its application with the strengths of the others, by integrating them appropriately. Each of technologies, in their own merits, have provided efficient solution to a wide range of problems belonging to different domains.

Hybrid technology can be implemented with care, as it is double or multi-edge technology. This technology should only be practised for the objective of solving the problem for better solution. If proper care has not be taken, it may result into worst solution of the problem. These technologies may only be used for getting better solution by properly integrating the different technologies using appropriate method.

1.5.1 Fuzzy Genetic Hybrid Algorithms

Fuzzy logic system addresses the imprecision or vagueness in input output description of system. Fuzzy sets have no crisp boundaries and provide a gradual transition between membership and non-membership of elements. Genetic algorithms (GAs) inspired by biological process of evolution, are adaptive search and optimisation algorithms.

In this research work, fuzzy systems have been integrating with genetic algorithms. Fuzzy system is involved with namely input/output variables and the membership functions that define the fuzzy systems have been optimised using GAs. Hybrid genetic algorithms have received significant interest in recent years and are being increasingly used to solve real-world problems (Tarek A. El-Mihoub *et. al.*, 2006). Genetic algorithms can be incorporated

with other techniques within its framework so as to produce a hybrid that reaps the best from the combination.

The ability of a genetic hybrid to solve hard problems depends on the way of utilising local search information and the mechanism of balancing genetic and local search. There is a trend towards adapting some of the hybrid design choices through adapting the control parameters associated with these choices. Different adaptation techniques have been used to adapt the selection of a local search method, the selection of individuals for a local search, the duration of local search, the learning strategy, and other design aspects.

1.6 Overview of the Thesis

Chapter 1 focuses on issues related to industrial robots used for variety of complex commercial applications such as palatisation, painting, welding, chip fabrication, *etc.* These robots have to work in repetitive manner in a highly structured predictable environment. The robots are supplied with inputs like position, orientation, *etc.* Complex work spaces in various industrial environment require tailored jigs and fixtures. It requires large time and incurs huge cost. Such an industrial application needs to be properly installed and accurately calibrated for the spatial specifications of these jigs and fixtures. However, it is task specific.

Literature survey reveals that engineers and researchers worldwide are attempting to develop a closed loop control system for the robot. Such a closed loop robotic system will possess inherent capability of sensing and reacting to its external work environment. Attempt has been made to develop artificially intelligent robotic systems, which will sense their external environment and operate on the basis of known information for which they have prepared before hand. It will eliminate the requirements for specialised work spaces, as robots in such configuration will be adaptive and tolerant for different new and unknown environments.

Creating an automated robotic system is a challenging task. It requires various skills like programming, mechanical modeling of a robotic system, electronic devices and their interfaces and control. An integrated design based on above skills and engineering areas will help in realisation of such an automated robotic system.

Fuzzy logic and genetic algorithms with special emphasis on robotics applications is presented in this chapter. A brief outline of analytical hierarchy process (AHP) and hybrid algorithm is also discussed.

Chapter 2 contains the detailed review of the literature relating to the fields of fuzzy logic and genetic algorithms in contrast to various applications in the engineering field and especially for robotics and automation.

Chapter 3 discusses overview of the robotics based upon work envelop geometries, specifications and degree-of-freedom. The robotic manipulators and their kinematics have been elaborated with respect to direct and inverse kinematics. Robotic arm movement refers to the problem of geometrically specifying a sequence of robot positions and orientations that moves the robot between the start and goal positions. It deals only with the kinematics of the robot. Various conventional approaches of path planning have also been discussed.

Chapter 4 describes arm movements of 2 DOF, 3 DOF and 4 DOF robotic arm manipulators and evaluates the mathematical model of the arm, while considering case study of these manipulators for calculating various values for joint angles using inverse kinematics.

Chapter 5 elaborates that how fuzzy logic is used for arm movement and how genetic algorithms helps in finding optimal solution from possible solutions of given path problem, while achieving the target of minimum energy consumption criterion. The chapter also presents various steps involved in implementation of fuzzy logic and genetic algorithms. Simulation results of 2 DOF, 3 DOF and 4 DOF robotic arm manipulator are compared and generalised genetic algorithms based optimal arm movement method is evolved. A description of the robotic arm movement optimisation using hybrid systems has also been given in this chapter.

Chapter 6 emphases on the results of this research while discussing 2 DOF, 3 DOF and 4 DOF robotic arm manipulators. Potential of these manipulators and their implications in robot systems have also been discussed.

Chapter 7 concludes the work carried out in this thesis while showing advantages of the techniques used in this work. For future work in this direction, suggestions have also been given.

Chapter 2

Literature Survey

2.1 Preamble

The literature review, in this thesis, has been divided into two major parts. The first part describes innovations in the field of robotics and path planning and the second part elaborates the applications of artificial intelligence techniques especially in the area robotics.

2.2 Robotics and Path Planning

Automation has become an extremely fast growing phenomenon, impacting all engineering applications. The robots and robotic arms have become the major part of this trend. Autonomous navigating robots have become increasingly important. Motion planning is one of the important tasks in intelligent control of an autonomous mobile Robot (Qin C., *et al.,* 1995).

Robots and robotic arms in the present shape are reprogramable devices, which have been designed to manipulate and transport parts, tools or specialised manufacturing implements

through programs. In robotics, in order to program tool motion, the designer first formulates the relationship between the joint variables, position and orientation of the tool. This is called kinematics transformation. The kinematics can be done in the two ways, *viz.,* direct kinematics and inverse kinematics. The direct kinematics transforms joint coordinates to end-effectors coordinates. Contrary to this, in inverse kinematics end-effectors coordinates are transformed to joint coordinates. The inverse kinematics offers multiple solutions. Because of this inverse kinematics has always attracted researchers and practising engineers to select the best solution out of multiple available solutions (Olson C. F., Pasadena C., 2000).

Gillespie R. B., *et al.,* (2001) presented a general framework for the design and analysis of robot controllers. While a human applies forces and moments, the controller guides motion by tuning the robot's set of continuously variable transmissions. The need for open model-based robot controllers becomes more and more important with the increasing quality of autonomous joint motion planners and applications. A new method for determining parameters of a dynamical robot model based on experiments has also been proposed (Olsen M. M. and Petersen H.G., 2001).

Sirouspour M. R. and Salcudean S. E., (2001) addresses the control problem of hydraulic robot manipulators. The tracking errors are shown to be exponentially stable under the proposed control. The controller is further augmented with adaptation laws in order to compensate for parametric uncertainties in the system dynamics.

Hemami A., (1986) worked on control of two-arm robots by determining the kinematic relationships between the two end-effectors. The elements of the orientation-position matrix are determined by solving the individual kinematic equations. Various kinematic configurations of planar arms with revolute and sliding joints have been analysed (Lumelsky V. J., 1987).

Ostrowski J. P., (1999) proposed a reduced equations for robotic systems with constraints and symmetries. These types of systems frequently occur in robotics and are generally found in robotic locomotion, wheeled mobile robots, and satellites or underwater vehicles. Ploen S. R. and Park F. C. (1999) have proposed a technique for coordinate-invariant formulation of the dynamics of open kinematic chains. Algebraic manipulations have been used to recast the resulting algorithm into a set of closed-form dynamic equations. An efficient recursive algorithm for forward dynamics is also extracted from the inverse factorisation. The resulting equations lead to a succinct high-level description of robot dynamics in both joint and operational space coordinates which minimises symbolic

complexity without sacrificing computational efficiency (Doyle A. B., 1995; Antonelli G., *et al.,* 2003).

Fahimi F., *et al.,* (2002) presented an efficient kinematics position and velocity solution scheme for spatial hyper-redundant manipulators. A simple and efficient recursive fitting method is introduced to avoid complications involved with solving systems of non-linear algebraic equations. This method also guarantees the existence of solutions for the inverse kinematic problems at the velocity levels. Velocity properties of the backbone curves were investigated and the inverse velocity propagation was solved for the spatial hyper-redundant arm. The velocity propagation scheme is recursive and is efficiently applicable to any number of links.

Chung W. K., *et al.,* (2004) developed a straightforward inverse kinematic algorithm for multilink flexible robots so as to improve the control performance. The inclusion of a dynamic constraint maximizes the performance of feedback controllers in high-speed applications. To obtain a numerically feasible solution, the singular perturbation approach is employed, which decomposes the inverse kinematics into an averaged part (slow part) and a parasitic part (fast part). The solution of the averaged part is considered, while the parasitic part is intentionally removed. The parameter expansion is carried out so as to obtain the solutions sequentially. The implicit expansion method is a refined version of the expansion method. It reduces computing time considerably (Tokhi M.O. and Azad A. K. M., 1997).

Chen Jigien and Chao Lih-Ming, (1987) worked on industrial robots to accomplish different tasks with program sequences that were executed in digital computers and the errors that contributed to these differences for robots with rotary joints were examined. Information on positions and orientations of the end-effectors was extracted by computing these as functions of the joint variables was also extracted.

Liao D. X. *et al.,* (1987) worked on flexible robotic manipulators with optimal arm geometries that was fabricated from composite laminates having optimal material properties. The results of the finite element simulations clearly demonstrated the superiority of the proposed optimal design. The fabrication of lightweight robot arms in composite materials and the ability of robotic devices to dissipate residual vibrations at the termination of a maneuver have been highlighted.

An algorithm for path planning of a mobile robot in an unknown environment is presented. The robot maps the environment only to the extent necessary to achieve the goal. Mapping is achieved using tactile sensing, while the robot is executing a path to the specified goal. Paths are generated by treating unknown regions in the environment as free space. As

obstacles are encountered en route to a goal, the model of the environment is updated and a new path to the goal is planned and executed. Initially, the paths to the goal generated by this algorithm were negotiable paths. However, as the robot acquires more knowledge about the environment, the lengths of the planned paths have been optimised. The optimisation criteria has been modified so as to either favour or avoid unexplored regions in the environment (Brooks R., 1983; Zelinsky A., 1992; Isto P., 1996; Fukao T. *et al.,* 2000; Ge S.S. and Cui Y. J., 2000; Kuffner Jr. J. J., *et al.,* 2000; Sugar T.G. and Kumar V., 2002).

A computationally efficient algorithm is developed for finding a near-optimal path with a weighted distance-safety criterion by using a variational calculus and dynamic programming (VCDP) method. An approach to robot-path planning is developed by considering both the travelling distance and the safety of the robot (Suh S.H. and Kang G. S., 1988).

A new approach has been proposed for robot path planning that consists of building and searching a graph connecting the local minima of a potential function which defined over the robot's configuration space. A planar based on this approach has been implemented. This planar is considerably faster than previous path planars and solves problems for robots with many more degrees-of-freedom (DOFs). The power of the planar derives both from the "good" properties of the potential function and from the efficiency of the techniques used to escape the local minima of this function. The most powerful of these techniques is a Monte Carlo technique that escapes local minima by executing Brownian motions. The overall approach is made possible by the systematic use of distributed representations (bitmaps) for the robot's work space and configuration space (Barraquand J. and Latombe J.C., 1991; Ahuactzin J. M. and Portilla. A., 2000; Helguera C. and Zegloul S., 2000; Strandberg M., 2004).

Hemami A. and Cheng R. M. H., (1992) dealt with kinematics of two collaborating robot arms handling an object. Such a task is much more difficult, both kinematically and dynamically, than when only one robot arm is involved. If a task becomes impossible to continue because of the inaccessibility of the required position and orientation by the end-effector, it might be possible to overcome the difficulty by performing the necessary orientation adjustments at each instant during motion. As a preliminary step toward this goal, this problem is studied for one robot arm (Rostami S. et al., 2001).

Tafazoli S., *et al.,* (1999) have given a novel approach for experimental determination of the link (mass and inertia-related) parameters and friction coefficients for a typical excavator arm. The parameters are needed for indirect measurement of the external forces. Treating

the machine arm as an open kinematic chain, its dynamic equations are presented symbolically. A new method for decoupled estimation of the gravitational parameters from static is also presented.

Yuan. J. and Yu S. L., (1999) presented a low-cost device for non-contact measurement of robot position and orientation. It consists of an optical sensor and three laser scanners. The sensor is attached to the end-effector, while the laser scanners cast light planes to trace the sensor movements. The device measures equations of the light planes from which end-effector position and orientation are obtained.

Packaging products such as telephones and two-way radios after assembly is a common manufacturing task. Carton folding is a packaging operation typically performed by human operators or with fixed automation. Lu L. and Akella S. (2000) presented a flexible method to fold cardboard cartons using fixtures. A carton blank is folded by moving it through a fixture with a robot. Method uses interchangeable fixtures so as to enable rapid changeovers between product models. A motion planning algorithm that generates all folding sequences for a carton by modeling it kinematically high degree-of-freedom robot manipulator with revolute joints and branching links has being deveoped (Sundaram S., *et al.,* 2001).

Devendra P. Garga *et al.,* (2002) presented the formulation and application of a strategy for the determination of an optimal trajectory for a multiple robotic configuration. Genetic Algorithms (GAs) have been used as the optimisation techniques and results obtained from them have been compared. The initial and final positions of the end-effectors are specified.

Trinkle J. C. *et al.,* (2002) described path planning for closed kinematic chains with spherical joints. The path planning problem for closed kinematics chains with 'n' links connected by spherical joints in space or revolute joints in the plane is considered. The configuration space of such systems is a real algebraic variety whose structure are fully determined using techniques from algebraic geometry and differential topology. This structure is then exploited to design a complete path planning algorithm that produces a sequence of compliant moves, each of which monotonically increases the number of links in their goal configurations.

Oya M., *et al.,* (2003) have presented the position/force tracking control of Lagrangian mechanical systems with classical non-holonomic constraints. They proposed control strategy at the dynamic levels, which can deal with model uncertainties in the mechanical systems and ensure desired trajectory tracking of the configuration state of the closed-loop system.

Han Y. (2004) presented an algorithm for kinematic tracking of a trajectory in the presence of static and moving obstacles. The algorithm considered the physical abilities of the robot system for tracking applications. Design goals have been formulated. The problem is considered as a constrained least-squares problem subjected to time-dependent state and control constraints. The state constraints have been used to describe both the obstacle constraints and the work space constraints of the robot. The control constraint is used to describe the speed constraint of the robot (Green A. and Sasiadek J. Z., 2004).

Lamiraux F., *et al.,* (2004) worked on motion planning for complex dynamic systems. The systems for which no steering method is known, the existing algorithms consist of an exploration tree in the configuration space which explore the input space of the system. The main drawback of this method is that they never reach exactly the goal but stops the search, when a small neighborhood of the goal has been reached.

Lingelbach. F. (2004) in his paper presented a new approach for path planning in high-dimensional static configuration spaces. The concept of cell decomposition is combined with probabilistic sampling to obtain a method called probabilistic cell decomposition (PCD). The use of lazy evaluation techniques and supervised sampling in important areas led to a very competitive path planning method. It is shown that PCD is probabilistic complete. PCD is easily scalable and applicable to many different kinds of problems. Experimental results show that PCD performs well under various conditions. Rigid body movements, maze like problems as well as path planning problems for chain-like robotic platforms have been successfully implemented (Bohlin R. and Kavraki. L. E., 2000).

Solteiro Pires E. J. *et al.,* (2007) presented a multi-objective genetic algorithms based technique for trajectory problem. Generating manipulator trajectories considering multiple objectives and obstacle avoidance is a non-trivial optimisation problem. Multiple criteria are optimised while considering up to five simultaneous objectives. Simulation results are presented for robots with two and three degrees-of-freedom, considering two and five objectives optimisation.

A method for the inverse kinematic problem of planar manipulators with multiple degrees of redundancy is proposed. This method starts by decomposing a redundant arm into a series of the local arms, which are either two-link or three-link planar manipulator modules. A displacement distribution scheme has been developed (Chung W. J. *et al.,* 1991; Chang P. H. *et al.,* 2000; Galicki M., 2000; Oriolo G. *et al.*, 2002).

Kinematic robot calibration is the key requirement for the successful application of off-line programming of industrial robots. To compensate for inaccurate robot tool positioning, off-

line generated poses need to be corrected using a calibrated kinematic model, leading the robot to the desired poses. An alternative approach to conventional kinematic robot calibration has been developed and also they develops a new inverse static kinematic calibration method based on the recent genetic programming paradigm. In their method the process of robot calibration has been fully automated by applying symbolic model regression to model synthesis (structure and parameters) without involving iterative numerical methods for parameter identification, thus avoiding their drawbacks such as local convergence, numerical instability and parameter discontinuities (Dolinsky Jens Uwe, 2001).

Paredis C. J. *et al.,* (1991) have successfully implemented the problem of mapping kinematic task specifications into a kinematic manipulator configuration on 2 degrees-of-freedom (2- DOF) planar manipulators. The reconfigurable modular manipulator system (RMMS) consists of modular links and joints assembled into many manipulator configurations.

Ahuactzin J. M. and Gupta K. K. (1999) proposed a novel and global approach for solving the point-to-point inverse kinematics problem for highly redundant manipulators. Initial configuration of the robot, finds a reachable configuration that corresponds to a desired position and orientation of the end-effector. Groom K. N., *et al.,* (1999) considered a real-time fault-tolerant control of kinematically redundant manipulators. The goal is to continuously follow this trajectory with the manipulator in configurations that maximize the fault-tolerance measure.

Danner T. and Kavraki L. E. (2000) worked on randomised planning for short inspection paths. An algorithm has been presented for planar work spaces, which operates in two steps: selecting art gallery-style guards and then connecting them to form an inspection path. In inspection problem, a known work space and a robot with vision capabilities compute a short path path for the robot such that each point on boundary of the work space is visible from some point on the path. Autonomous inspection, such as by a flying camera, or a virtual reality architectural walkthrough, is guided by a solution to the inspection problem. Visibility constraints on both maximum viewing distance and maximum angle of incidence are considered to better model real sensors.

Yuefa Fang and Lung-Wen Tsai (2002) presented a systematic approach for the structural synthesis of a class of 4 DOF and 5 DOF parallel manipulators with identical serial limbs. The theory of screws and reciprocal screws has been implemented for the analysis of the geometric conditions for limbs of parallel manipulators. Limb structures are used for

constructing 4 DOF or 5 DOF parallel manipulators according to the reciprocity of the twist and wrench systems.

Feliu V. *et al.,* (2003) described a new control scheme, which has been designed for a 3 DOF flexible arm. This arm has been built with light links which has most of its mass concentrated at the tip, and it uses a special mechanical configuration to approximately decouple tip motions in spherical coordinates. A consequence of this simple dynamic is that minimum sensing effort is required (only direct motor and tip measurements), and the use of complex observers is avoided because the state of the system can be very easily obtained from these measurements.

Manipulator control is one of the main research areas in robotics. Using the Denavit-Hartenberg methodology forward and inverse kinematics models for a 5 DOF robot arm has been developed (Rosales E. M. and Gan J. Q., 2003).

Feddema J. T. *et al.,* (2002) developed a decentralised control for multiple cooperative robotic vehicles. Models relating input/output reachability, structural observability, and controllability of the entire system have been developed.

Lozano-P´erez T. (1983) has given algorithms for computing constraints on the position of an object due to the presence of other objects. The approach presented here is based on characterising the position and orientation of an object as a single point in a configuration space, in which each coordinate represents a degree-of-freedom in the position or orientation of the object. The author has successfully implemented algorithms for computing these configuration space obstacles, when the objects are polygons or polyhedra.

Khatib O., (1986) presented a unique real-time obstacle avoidance approach for manipulators and mobile robots based on the artificial potential field concept. The artificial potential field approach has been extended to collision avoidance for all manipulator links. In addition, a joint space artificial potential field is used to satisfy the manipulator internal joint constraints. Real-time collision avoidance has been successfully demonstrated on moving obstacles with the aid of visual sensing.

A simple and efficient algorithm has been presented which uses configuration space, for planning collision-free motions for general manipulators (Lozano P. and Erez T., 1987). An implementation of the algorithm for manipulators made up of revolute joints is also presented. This obstacle representation leads to an efficient approximation of the free space outside of the configuration-space obstacles (Wang D., *et al.,* 1992; Mclean A. W. *et al.,* 1996; Cameron S., 1998; Sugie T. *et al.,* 2003).

Global planning involves potential functions and direct kinematic and dynamic equations of the manipulator (Galicki M., 1992). This planning is based on using the necessary conditions of minimum for an integral type criterion. A closed systems of boundary dependences fully specifying differential equations arising from the necessary conditions of extreme are determined. The system renders it possible to reduce the collision-free trajectory planning problem. (Bicchi A. *et al.,* 1995; Brunn P., 1996; Reggiani M. *et al.,* 2002). A generalised pattern search method is designed for a collision-free trajectory planning in the case of planar redundant manipulators (Ata A. A. *et al.,* 2006).

Sugie T. *et al.,* (2003) presented a new control method, which achieves autonomous obstacle avoidance for manipulators with rate constraints. To achieve the autonomous obstacle avoidance, the freedom of the coordinate transformation for feedback linearisation of non-linear control systems has been considered (Hsu D. *et al.,* 1999).

Motion planning is one of the most important areas of robotics research. The complexity of the motion planning problem has hindered the development of practical algorithms. The work on gross-motion planning, including motion planners for point robots, rigid robots, and manipulators in stationary, time-varying, constrained, and movable-object environments has been reported (Hwang Y. K. and Ahuja N., 1992). Seraji H. *et al.,* (1999) presented a new approach for real-time collision avoidance for position-controll of conventional arms. The collision avoidance problem is formulated and solved as a position-based force control problem. Virtual forces representing the intrusion of the arm into the obstacle safety zone are computed in real-time using a spring-damper model.

In order to control gymnastic and jumping robots, a high diving motion is simulated. The complete analytical solution to the posture control problem of a two-link free flying object with initial angular momentum was presnted (Mita T. *et al.,* 2001). Lee Sung-Hee *et al.,* (2005) described Newton and quasi-Newton optimisation algorithms for dynamics-based robot movement generation. The robot is modelled as rigid multi-body systems containing multiple closed loops, active and passive joints, and redundant actuators and sensors.

A robot system is capable of locating a part in an unstructured pile of objects, choosing a grasp on the part, planning a motion to reach the part safely, and planning a motion to place the part at a commanded position has been developed. The system requires an input of polyhedral world model including models of the part to be manipulated. The system needs to build a depth map, using structured light of the area, where the part is to be found initially (Lozano-P´erez T. *et al.,* 1987).

Planning a grasp by robot is fundamental problem that requires the object to be firmly held by robot gripper. Jones L., *et al.,* (1990) have presented their work on planning of two-fingered grasps for pick-and-place operations. Li Z., *et al.,* (1988) have discussed the problem of optimal grasping to an object by a multi-fingered robot hand. A multi-fingered grasp firmly holding an object while resisting external loads and/or disturbances is complex task. For planar grasps and frictionless three-dimensional (3-D) grasps, the quantitative measure is computed efficiently by solving a set of linear programs, while for frictional 3-D grasps, it can be computed by solving non-linear programs without linearisation of the friction (Ferrari C., *et al.,*1992 ; Zhu X., *et al.,* 2003). The approach has potential application in grasp planning with multiple optimality criteria.

Robots and intelligent machines are becoming common in the developed countries during the next decade. Fred E. Sistler, (1987) have discussed intelligent machines and robotics in agriculture. Later, they have discussed the issues for the future role of robotics as an intelligent machine in agriculture.

Muller-Karger C. M., *et al.,* (2000) have proposed trajectories of hyperbolic type technique for pick and place applications. The method requires defining the inverse kinematics in these positions. This process is performed with the help of a normalised hyperbolic trajectory, which may be a symmetric curve for the simpler case. The final function is such that the velocity, acceleration, jerk, and higher derivatives are all zero at the extreme points. The simulator was able to avoid a specific obstacle.

Jakopec M., *et al.,* (2003) have worked on robotic system for total knee replacement surgery. This technique has been successfully used in seven clinical trials with encouraging results. A computer tomography-based preoperative planning software is used to accurately plan the procedure. Intraoperatively, the surgeon guides a small special-purpose robot, called Acrobot, which is mounted on a gross positioning device.

Aarno D., *et al.,* (2004) have implemented artificial potential biased probabilistic roadmap methods (PRMs) to solve path planning problems. The developed sampling scheme increases the probability of finding paths, through narrow passages. A biased sampling scheme is used to increase the distribution of nodes in narrow regions of the free space.

Measurement of velocities of robot control system by using tachometers, increases costs, and the signals delivered set contaminated with noise. The use of encoders allows reading joint position accurately. Sometimes, it is desirable to estimate joint velocities through an observer. Arteaga M.A. *et al.,* (2004) have presented a robust control scheme designed in conjunction with a linear observer. The performance of the new control-observer law is

better in comparison with well-known algorithms reported erlier. Rodr´ıguez M., *et al.,* (2007) have proposed robot techniques for home-care or domestic applications. They had successfully implemented genetic algorithms to teach a robot to perform a task when only the specification of the main restrictions of the desired behaviour were provided.

Industrial robots perform complex tasks in the minimum possible cycle time in order to obtain high productivity. The multiple solutions of the inverse kinematics problem are taken into consideration for determining the optimum sequence of task points visited by the tip of the end-effector of robot. Also, it is applied to any non-redundant manipulator.

2.3 Artificial Intelligence Applications in Robotics

In this section of literature review, we have laid emphasis on artificial techniques applications in robotics, which is further divided into genetic algorithms, fuzzy logic and hybrid techniques for robotic applications.

2.3.1 Genetic Algorithms Applications in Robotics

Genetic algorithms are based on a biological metaphor. They view learning as a competition among a population of evolving candidate problem solutions. A “fitness” function evaluates each solution to decide whether it will contribute to the next generation of solutions or not. Then, through operations analogous to gene transfer in sexual reproduction, the algorithm creates a new population of candidate solutions. The idea of applying the biological principle of natural evolution to artificial systems, introduced more than three decades ago, has seen impressive growth in the past few years. Usually, grouped under the term evolutionary algorithms or evolutionary computation, we find the domains of genetic algorithms, evolution strategies, evolutionary programming, and genetic programming.

Alexei Zakharov and Sindor Halhsz (1999) have developed method for inverse dynamics of a robot arm based on genetic algorithms (GAs). It is shown that the GAs are able to search robot parameters effectively and accurately, even if, the robot has low resolution position. It is possible because the method requires only position feedback and there is no need to find out the speed and acceleration of the links that usually can be done only through finite difference calculations that cause dramatic errors during identification. The effectiveness of the algorithm is demonstrated with an example of parameter identification of the PUMA 560 robot (for second and third links).

Yuval Davidor (1990) has given a GAs model, which is capable of processing robot trajectories. The performance of genetic algorithms shows characteristic improvements

when compared with that of a hill-climbing and random search algorithms. Lewis M. *et al.* (1994) have implemented genetic algorithms for synthesis of a hexapod robot, which describes the staged evolution of a complex motor pattern generation for leg movements of walking robot.

Yoshida E. *et al.,* (2003) have proposed a method using genetic algorithms for self-reconfigurable modular robot to realise various robotic motions. The behaviour of the robots are described using a motion sequence. The motion sequences specify simultaneous motor actuations and self-reconfiguration by connection/disconnection. These simple descriptions have been successfully implemented by encoded genetic representations.

The search for the minimum-time path of a robotic manipulator through the joint space movement involves heavy computational burden. Genetic algorithms (GAs) have been used to tackle this problem and have reduced the computational search time. The work presented provides a practical GAs motion control for possible real-time implementations. It has been found that the quickest motion is not necessarily a straight line in joint space. (Wang Q. and Zalzala A.M.S., 1996; Ahuactzin, J. *et al.,* 1992; Craig Eldershaw and Stephen Cameron, 2000; Tsuji T. *et al.,* 1998). Although, extensive work about robot path planning has been carried out, but still there is a need for fast, practical and general-purpose motion planars. In these situations, it becomes necessary to cope with variable and complex environments and direct human interactions. The approach using genetic algorithms has been successfully tested over several selected experiments thus obtaining fast planning even for complex situations (Hernando M. and Gambao E., 2002; Shibata T. *et al.*, 1997).

The path planning problem for a mobile manipulator system has been demonstrated to perform a sequence of tasks specified by locations. It finds an optimal sequence of base positions and manipulator configurations for performing sequence of tasks for given specifications. Genetic algorithms applied to such problems appear to be very promising, while traditional optimisation methods demonstrated difficulties (Zhao Min., *et al.,* 1994; Handley S. G., 1993; Messom Chris, 2002). A genetic algorithms (GAs) based path planning software for mobile robot systems focusing on energy consumption has been developed (Gemeinder M., Gerke M., 2003). The criterion of minimum energy consumption has been implementation while taking into the consideration changing textures and changing energy requirements, when moving within given work space. For each obstacle within the work space, path circumventing is computed in an initial phase. Problem specific genetic operators have also been proposed, and especially the handling of exceptional situations is discussed.

The trajectory generation problem for a robot in a work space with obstacles is really a challenging task. To generate the robot's trajectories, genetic algorithms searches valid solutions in the configuration space. The evolutionary search process allows the user to solve the trajectory problem in an n-dimensional space where the 'curse of dimensionality' inevitably stalls conventional methods (Pack D. *et al.,* 1996; Monteiro D. C. and Madrid M. K., 1999; Tian L., *et al.,* 2004). A trajectory planning method for an underwater manipulator has been discussed whose criterion of path generation is the energy consumption during operation. The minimum energy trajectory is obtained as the solution of the two-point boundary-value problem. From the results of a series of numerical calculations, moving the manipulator (e.g. folding its arm) is effective to reduce energy consumption during motion. To calculate the trajectory without the complexity of formulation and computation, a simple trajectory planning method has been proposed, which uses a Genetic Algorithms (Shintaku E., 1999).

Wilson Lucas A., *et al.,* (2004) worked on parallel genetic algorithms so as to generate a best path for a robot arm to move from a starting position to a goal in three-dimensional space. The algorithm uses multiple optimisation criteria, independent cross-pollinating populations, and handles multiple hard constraints. Fonseca C. M. and Fleming P. J. (1993) describes a rank-based fitness assignment method for multiple objective genetic algorithms (MOGAs). The MOGA is generalised for the genetic algorithms as the optimising element of a multi-objective optimisation.

Kwok D. P. and Sheng F. (1994) describes the use of genetic algorithms (GAs) and simulated annealing (SA) for optimising the parameters of PID controllers for a robot arm. GAs and a SA are designed to optimal-tune the parameters of the PID controller of each joint for a single step response and for the tracking of other specified trajectories. GAs and the SA are required to optimise evaluation functions related to the combinations of different performance indices. Simulations are carried out on a PUMA 560 arm model which is being controlled by PID controllers with their parameters optimised using the proposed GAs and SA.

Kubilay K. Aydin and Erol Kocaoglan (1995) presented a genetic algorithms based approach for redundancy resolution of robot manipulators. The genetic algorithms works under joint limits and produces end-effector positions with negligible error. Genetic algorithms has been employed to globally optimise six relevant redundant degrees of the multiple robotic systems for welding applications (Wu L., et al., 1999). Movement of a multi-joint robot arm using genetic algorithms has been presented (Yano F and Tooda Y.,

1999). The joint angles of the arm were generated by inverse kinematics. The problem is complex as there are nine axes involved and a number of permutations are possible, which achieve the same movements of the weld torch. The system is redundant and the robot has singular configurations. The six-axis robot is constrained to move the weld torch along the weld trajectory. Robot coordination is achieved by placing the positioning table in a good manoeuvrability position, *i.e.,* far from its singular configurations and far from the motion limits of the six-axis arm and the motion limits of the track. The path planning for traditional polishing robot is teach-play-back method. That is, after adjusting offset errors, the polishing robot repeats the same fixed program in a pre-determined path. The path planning schemes based on genetic algorithms have been implemented and simulations show effectiveness of this technique (Guo Tong-ying *et al.,* 2004; Zacharia P. Th. and Aspragathos N. A., 2005).

2.3.2 Fuzzy Logic Applications in Robotics

Fuzzy Logic is well accepted technique for non-linear control of random processes. It transforms linguistic knowledge into a mathematical mode. Zadeh L. A., 1965 introduced fuzzy set theory in his paper. A fuzzy set is class of objects with a continuum of grades of membership. The researchers have tried to provide learning capability in the fuzzy logic models. Fuzzy models have shown superior results in the terms of their performance, stability and their robustness (Johansen T. A., 1994). Fuzzy logic has been applied to nearly all the fields of industrial applications. They are best suited for signal processing and automatic control systems including robotics, which can not be accurately executed using classical control techniques like PID controller, model predictive controller (MPC)(Bagachi A. *et al*., 1990; Jou C. C. *et al*., 1993; Naffenger C., 1983; Moudgal V. G. *et al*. 1995; Saffiotti A., 1997).

Zadeh L.A. (1996) has developed fuzzy logic models. He has concluded that it is a best suiting technology as it computes with the words. Especially, in data forecasting and uncertainty compensation, the fuzzy logic models provide a unique alternative approach.

Singh Chanan *et al.* (2002) has presented a conceptual possibilistic approach using fuzzy set theory to manage the uncertainties in the reliability input data of real power systems. Kim E. *et al.* (2003) developed discrete time fuzzy disturbance observer and successfully implemented it in various control applications. Seraji H. *et al.* (2002) developed fuzzy logic approach based on behaviour analysis of the robot and its navigational control on challenging terrain.

As on date fuzzy logic models have emerged out as a mathematically proven technology (Rovatti R. *et al.,* 1996; Euntai Kim *et al.,* 1997). Numerous efficient fuzzy logic models, including the intelligent modules like, fuzzy observers have been developed and have been reported to be running successfully (Ma X. J. *et al.,* 1998).

Wang Li-Xin *et al.* (1992) successfully developed fuzzy rules from numerical data. This method was capable of approximating any real continuous function on a compact set to arbitrary accuracy. Applications to truck backer-upper control and time series prediction problems are presented. Tang Y. *et al.* (2003) proposed fuzzy logic system (FLS) to approximate the unknown dynamics of the system. Based on the apriori information, the premise part of the FLS as well as a nominal weight matrix are designed first and are fixed. To further reduce the tracking error due to the function reconstruction error, a second compensation signal is also synthesised. By running two estimators on-line for weight matrix error bound and function reconstruction error bound, the implementation of the proposed controller needs no a priori information on these bounds. Johnson J. A. *et al.* (1995) has reported the fuzzy implication to determine product operator. Advantages of this method over the "standard" methods include elimination of the defuzzification step, direct control of the shape of the input-to-output mapping surface, and an analytic formulation that can be easily implemented in software or hardware. Conventional controllers are shown to be a special case of the method. Simulation results during large on-line variations in system parameters derived from a typical example of an agricultural robot show the effectiveness of the proposed controller. The controller stability is verified by using the so-called cell-to-cell mapping algorithm (Christophe Collewet *et al.,* 1998). The Fuzzy supervisory controller is used to tune a set of lower level distributed fuzzy control modules that reduces work-in-process and synchronises the production system's operation. The overall control objective is to keep the work-in-process and cycle time as low as possible, while maintaining quality of service by keeping backlog to acceptable levels (Ioannidis S. *et al.,* 2004).

Fuzzy logic plays an important role in the design of reactive robot behaviours. A learning approach to the development of a fuzzy logic controller, based on the delayed rewards from the real world has been presented. The delayed rewards are apportioned to the individual fuzzy rules by using reinforcement Q-learning. The efficient exploration of a solution space is one of the key issues in the reinforcement learning. The proposed approach is evaluated on some reactive behaviour of the football-playing robots. (Dongbing Gu *et al.,* 2003). One important application of mobile robots is in searching a geographical region to locate the

specific position. Fuzzy logic algorithm is an efficient approach for the collective robots to locate the target source (Zhang Nian *et al.,* 2003). The FLC strategy is a viable option for precise and robust tracking control of a two-link flexible robot manipulator (Green A. *et al.* 2001). Edward T. Lee, (1995) has proposed fuzzy languages and fuzzy instructions to perform robot navigation. This approach arms at trade precision with speed. It also can be applied to object motion tracking.

Khoury G. M. *et al.,* 2004 applied fuzzy logic control on five degrees-of-freedom (DOF) robot arm. The elaboration of the fuzzy control laws was based on two structures of coupled rules fuzzy PID controllers. The fuzzy PID controllers were numerically simulated and the simulation results confirmed the success of the fuzzy PID control in trajectory tracking problems. Seeking a performance optimisation, a systematic study of the choice of tuning parameters of the controllers has also been done. Barai Ranjit Kumar *et al.,* 2007 worked on locomotion control of legged robots. This was a very challenging task because it needed very accurate foot trajectory tracking control for stable walking.

2.3.3 Hybrid Techniques Applications in Robotics

Hybrid genetic algorithms have played significant role in recent years. GAs are being used to solve real-world problems (Tarek A.*, et al.,* 2006). Kent S., (1999) in his research work explored the potential of evolutionary techniques successfully. He implemented it in robot path planning. Kelly-III W. E., (1994, 1996) presented in his work neuro-fuzzy control of a robotic arm. The combinations of two technologies *i.e.,* neuro-fuzzy has been successfully combined so as to maximize their individual strengths and compensate for individual shortcomings. (Fonseca C.M. and Fleming P. J., 1995; Fogel D. B., 1997; Hocaoglu C. *et al.,* 1998).

Researchers claimed the superiority of soft computing tools like fuzzy systems, neural networks and genetic algorithms which can efficiently solve difficult problems, especially non-linear control problems. The soft computing based nonlinear control algorithms have been successfully applied for the control of a rigid link flexible joint (RLFJ) 4 DOF SCARA robot in order to prove the effectiveness of the proposed methods (Szilveszter Pletl, *et al.,* 2001; Belarbi K., *et al.,* 2000). Chiang C-K *et al.* (1997) worked on self-learning fuzzy logic controllers. Genetic algorithms reinforcement have been developed and various efficient new approaches in fuzzy

modeling have been evolved. Intelligent robot has been developed using adaptive fuzzy hybrid position and force control capabilities (Fu F-YHL-C 2000).

Sidhu T. S. *et al.* (1997) had presented an artificial neural network for the directional comparison protection of power transmission lines. Training patterns were generated using voltage and current samples for faults at various locations along a transmission line. The faults were simulated using an electromagnetic transient program and a sample three-phase power system. Singh Chanan *et al.*, (2000) proposed a neural network solution methodology for the problem of real power transfer capability calculations based on the optimal power flow formulation. The inputs for the neural networks are generator status, line status and load status and the output is the transfer capability. This new method is highly useful for reliability assessment in the new utility environment.

Singh S. N. *et al.,* (2000) have proposed a cascade neural network based approach for fast voltage contingency screening and ranking. The developed cascade neural network is a combination of a filter module and a ranking module. Neural network gives fast and accurate screening and ranking for unknown patterns and will be suitable for on-line applications (Srivastva L. *et al.,* 2000).

Bansal R. C. *et al.*, (2003) have explored the expert systems, fuzzy systems, artificial neural networks and evolutionary computing for power/voltage control in power systems. They successfully implemented the control by providing appropriate placement of compensation devices so as to ensure a satisfactory voltage profile while minimising the cost of compensation.

Leu Y-G *et al.,* (1999) have developed robust adaptive fuzzy neural controllers for non-linearities compensation in robots and other automatic systems.

Galantucci L. M. *et al*., (2004) proposed the implementation of hybrid fuzzy logic-genetic algorithms (FL-GA) methodology in order to plan the automatic assembly and disassembly sequence of products. The hybridisation model consists of the fuzzy controller for the parameters of an assembly or disassembly planar based on GAs.

Saxena A. K. *et al.* (2008) have reported genetic algorithms (GAs) based solution for optimal power flow (OPF), one of the non-linear problems of electric power system. The results obtained for GA-fuzzy OPF on various power systems related problems have shown faster convergence and lesser generation costs as compared to other approaches.

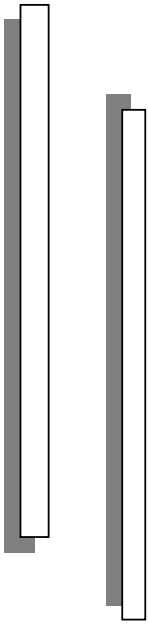
Wong S. V. and Hamouda A. M. S. (2004) have developed a genetic algorithms based fuzzy rules design for metal cutting data selection.

Homaifar A. and McCormick E. (1995) found applicability of genetic algorithms (GAs) in the simultaneous design of membership functions and rule sets for fuzzy logic controllers. Previous work using genetic algorithms had focused on the development of rule sets or high performance membership functions. However, the inter-dependence between these two components suggests a simultaneous design procedure. GAs are fully capable of creating complete fuzzy controllers for the given equations of motion. It eliminates the need for human input in the design loop. Rosales E. M. *et al.,* (2003) presents a hybrid approach based on inverse kinematics modelling. A genetic-fuzzy approach for optimal path-planning of a robotic manipulator among static obstacles has been developed (Nasser Sadati and Javid Taheri, 2002; Roy S. S. and Pratihar D. K., 2003; Sun Y. L. and Er M. J., 2004).

The inverse kinematics problem is a non-linear problem. In some cases it cannot be solved in closed form. Several iterative and neural network approaches are studied in solving the inverse kinematics problems. Deflection of the manipulator arms due to flexibility and mass load causes positioning error. The magnitude of the error depends on the amount of mass load and arm positions and also the stiffness characteristics of arms. Genetic algorithms are used to solve the inverse kinematics of three degrees-of-freedom of a log crane. Neural networks are used to solve the correction values for deflection compensation (Rouvinen A. *et al.,* 1997).

Machine learning techniques such as evolutionary algorithms or artificial neural networks are used for implementation of behaviours in mobile robotics. Fuzzy controller using genetic algorithms for the implementation of behaviour in a mobile robot have been discussed (Mucientes M. *et al.,* 2007). Genetic algorithms tuned fuzzy logic controller were successfully implemented for a robot arm movement. Many recent contributions on flexible link and elastic joint robotic arms focus on how to solve path tracking and vibration damping problems both in slow and fast mode control. As a result, system performances are often tiresome and intractable. The system with the new controller is simulated and its behaviour is compared with that provided by conventional and expert-designed fuzzy logic controllers (Nguyen V. B. *et al.,* 2007; Alam M. S. *et al.,* 2008). Integrating fuzzy logic (FL) and genetic algorithms (GAs) in order to solve the simultaneous localisation and mapping (SLAM) problem of mobile robots. The core of the proposed SLAM algorithm is based on an island model GA (IGA) that searches for the most probable map(s), which provide robot with the best localisation information. Prior knowledge about the problem domain is transferred to GA in order to speed up the convergence. Fuzzy logic is employed

to serve this purpose and allows the IGA to conduct the search starting from a potential region of the pose space (Momotaz Begum *et al.,* 2008).



Chapter 3

Robotics and Path Planning

3.1 Preamble

Robotics is the science of designing and building robots suitable for real-life applications in automated manufacturing and other non-manufacturing environments. The first industrial modern robots were the "Unimate" developed by George Devol and Joe Engelberger in 1956. In 1962, the Unimation company was formed and Engelberger became a president of the company and was the first to market robots. The first patents were by Devol for parts transfer machines. As a result, Engelberger has been called the "father of robotics". The first installation of a Unimate robot was at the Ford Motor Company for unloading a die-casting machines.

Modern industrial rbotic arms have better in capability and performance through controller and language development, improved mechanisms, sensing, and drive systems. In the early to mid 80's, the robot industry grew very fast primarily due to large investments by the automotive industry. The quick leap into the factory of the future turned into a plunge when the integration and economic viability of these efforts proved disastrous. The robot industry

has only recently recovered to mid-80's revenue levels. In the meantime there has been an enormous shakeout in the robot industry. In the unites sates of America (US), for example, only one US Company, Adept, remains in industrial robot arm business. Most of the rest went under, consolidation, or were sold out to the European and Japanese companies.

Automation has become an extremely fast growing phenomenon, impacting all engineering applications. The robots and robotic arms have become the major parts of this trend. Autonomous navigating robots have become increasingly important.

Robotic arms are extremely used in industries in many hazardous areas. In many field applications, technical support is required where manhandling is either dangerous or is not possible. In such situations robotic arm manipulators are used. These manipulators are in great demand to speed up the automation process (Craig J. J., 2004).

An Industrial robot is a general purpose, programmable machine that possesses certain anthropomorphic, or humanlike characteristics. The most typical human like characteristics of present day robots are their arms. These arms can be programmed to perform some useful task, through sequence of motions in order to accomplish the task. The main advantage of such a robotic arm is that it can repeat the task over and over until reprogrammed to perform some other task. The programming features allow robots to be used for a variety of different industrial operations.

Some definitions of robotics are

"A reprogrammable, multifunctional manipulator designed to move material, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks".

-Robot Institute of America

"A mechanical system which has flexible motion functions analogous to living organisms or combination of such motion functions with intellectual functions that may involve recognition, adaptation, learning or judgment or any combination thereof".

-Japanese robot association

"A reprogrammable device designed to both manipulate and transport parts, tools or specialized manufacturing implements through variable programmed motions for the performance of specific manufacturing tasks".

-British robot association

3.2 Laws of Robotics

The word "robotics" was first used in *Runaround*, a short story published in 1942. *I, Robot*, a collection of several of these stories, was published in 1950. Asimov (Clarke Roger, 1993) proposed his three "Laws of Robotics", and he later added a zeroth law.

Law Zero: A robot may not injure humanity, or, through inaction, allow humanity to come to harm.

Law One: A robot may not injure a human being, or, through inaction, allow a human being to come to harm, unless this would violate a higher order law.

Law Two: A robot must obey orders given it by human beings, except where such orders would conflict with a higher order law.

Law Three: A robot must protect its own existence as long as such protection does not conflict with a higher order law.

3.3 Classification of Robots

In order to refine the general notion of robotic manipulators, it is helpful to classify manipulators according to various criteria such as:

- Drive technologies
- Work envelope geometries
- Motion control methods

3.3.1 Drive Technologies

It is one of the most fundamental classification schemes. It is based upon the source of power used to drive the joints of the robot. The two most popular drive technologies are electric**,** and hydraulic. Most robotic manipulators today use electric drives in the form of either DC servomotors or DC stepper motors. However, when high-speed manipulation of substantial loads is required, such as in molten steel handling or auto body part handling, hydraulic-drive robots are preferred. One serious drawback of hydraulic-drive robots lies in their lack of cleanliness, a characteristic that is important for many assembly applications.

Both electric-drive robots and hydraulic-drive robots often use pneumatic tools or end-effectors, particularly when the only gripping action required is a simple open-close type of operation. An important characteristic of air-activated tools is that they exhibit built-in compliance in grasping objects, since air is a compressible fluid. This is in contrast to

sensor less rigid mechanical grippers, which can easily damage a delicate object by squeezing too hard.

3.3.2 Work Envelope Geometries

The end-effectors, or tool, of a robotic manipulator is typically mounted on a flange or plate secured to the wrist of the robot. The gross work envelope of a robot is defined as the locus of points in three-dimensional space that can be reached by the wrist. We have refered to the axes of the first three joints of a robot as the major axes throughout this work.
Major axes determine the position of the wrist. The axes of the remaining joints, the minor axes, are used to establish the orientation of the tool. As a consequence, the geometry of the work envelope is determined by the sequence of joints used for the first three axes. Six types of robot joints are possible.

However, only two basic types are commonly used in industrial robots, and they are listed in Table 3.1.

Table 3.1: Types of robot joints

Type	Notation	Symbol	Description
Revolute	R		Rotary motion about an axis
Prismatic	P		Linear motion about an axis

Revolute joints (R) exhibit rotary motion about an axis. They are the most common type of joints. The next most common type is a prismatic joint (P), which exhibits sliding or linear motion along an axis. The particular combination of revolute and prismatic joints for the three major axes determines the geometry of the work envelope, as summarised in Table 3.2.

Table 3.2: Robot work envelopes based on major axes

Robot	Axis 1	Axis 2	Axis 3	Total revolute
Cartesian	Prismatic	Prismatic	Prismatic	0
Cylindrical	Revolute	Prismatic	Prismatic	1
Spherical	Revolute	Revolute	Prismatic	2
SCARA	Revolute	Revolute	Prismatic	2
Articulated	Revolute	Revolute	Revolute	3

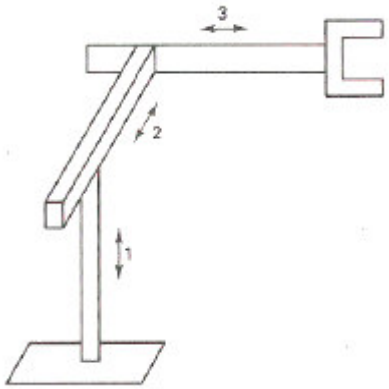


Fig. 3.1: Cartesian robot

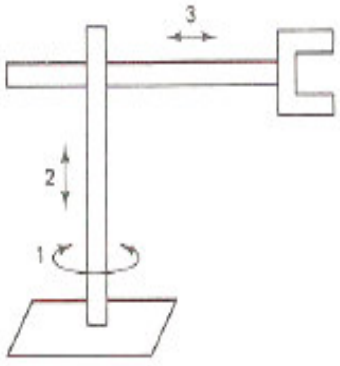


Fig. 3.2: Cylindrical robot

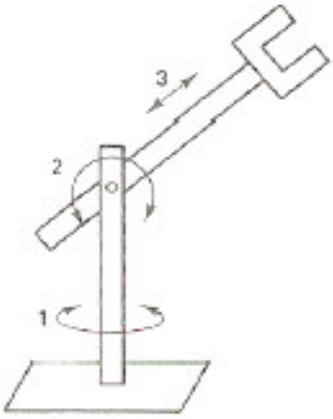


Fig. 3.3: Spherical robot

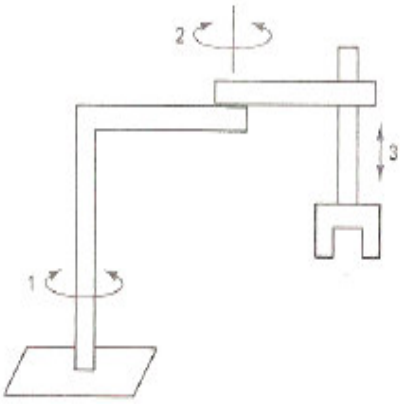


Fig. 3.4: SCARA robot

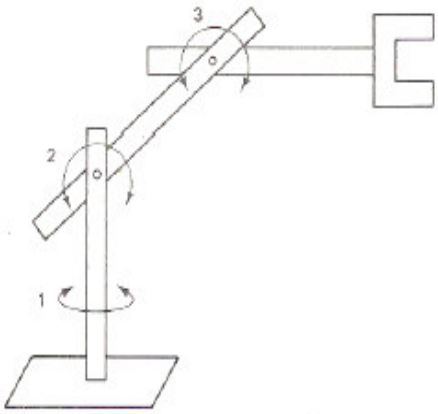


Fig. 3.5: Articulated robot

The list given in Table 3.2 is not exhaustive, since there are many possibilities, but it is representative of the vast majority of commercially available robots. As far as analysis of the motion of the arm is concerned, prismatic joints tend to be simpler than revolute joints.

3.3.3 Motion Control Methods

Another fundamental classification criterion is the method used to control the movement of the end-effector or tool. The two basic types of movements are listed in Table 3.3.

Table 3.3: Types of robot motion control

Control method	Applications
Point to point	Spot welding, pick-and-place loading and unloading
Continuous path	Spray painting, arc welding, gluing

The first type is *point-to-point* motion, where the tool moves to a sequence of discrete points in the work space. The path between the points is not explicitly controlled by the user. Point-to-point motion is useful for operations which are discrete in nature. For example, spot welding is an application for which point to point motion of the tool is required.

The other type of motion is *continuous path motion*, sometimes also called as *controlled path motion*. Here, the end-effector must follow a prescribed path in three dimensional space, and the speed of motion along the path may vary. This clearly presents a more challenging control problem. Examples of applications for robots with continuous-path motion control include paint spraying, arc welding, and the application of glue or sealant.

3.4 Robot Specifications

While the drive technologies, work-envelope geometries, and motion control methods provide convenient ways to broadly classify robots, yet there are a number of additional characteristics that allow the user to further specify robotic manipulators. Some of the more common characteristics are listed in Table 3.4.

Table 3.4: Robot characteristics

Characteristic	Units
Number of axis	---
Load carrying capacity	kg
Maximum speed, cycle time	mm/sec
Reach and stroke	mm
Tool orientation	deg
Repeatability	mm
Precision and accuracy	mm

3.4.1 Number of Axis

Each robotic manipulator has a number of axes about which its links rotate or along which its links translate. Usually, the first three axes, or major axes, are used to establish the position of the wrist, while the remaining axes are used to establish the orientation of the tool or gripper, as shown in Table 3.5.

Table 3.5: Axis of a robotic manipulator

Axes	Type	Function
1-3	Major	Position of wrist
4 -6	Minor	Orient the tool
7-n	Redundant	Avoid obstacles

Since robotic manipulation is done in three-dimensional space, a six axes robot is a general manipulator in the sense that it can move its tool or hand to both an arbitrary position and an arbitrary orientation within its work space. The mechanism for opening and closing the fingers or otherwise activating the tool is not regarded as an independent axis, because it does not contribute to either the position or the orientation of the tool. Practical industrial robots typically have four to six axes. Of course, it is possible to have manipulators with

more than six axes. The redundant axes can be useful for such things as reaching around obstacles in the work space or avoiding undesirable geometrical configurations of the manipulator.

3.4.2 Capacity and Speed

Load-carrying capacity varies greatly between robots. For example, the Minimover 5 Microbot, an educational table-top robot, has a load-carrying capacity of 2.2 kg. At the other end of the spectrum, the GCA-XR6 Extended Reach industrial robot has a load-carrying capacity of 4928 kg. The maximum tool-tip speed can also vary substantially between manipulators. The Westinghouse Series 4000 robot has a tool-tip speed of 9000 mm/sec. A more meaningful measure of robot speed may be the cycle time, the time required to perform a periodic motion similar to a simple pick-and-place operation. The Adept One SCARA robot carrying a 2.2-kg payload along a 700 mm path consists of six straight-line segments having a cycle time of 0.9 sec. Thus the average speed over a cycle is 778 mm/sec, which is considerably less than the 9000 mm/sec maximum tool-tip speed. Although, the load-carrying capacities and maximum operating speeds of robots vary by several orders of magnitude, it is, of course, the mix of characteristics that is important when selecting a robot for a particular application. In some cases, a large load-carrying capacity may not be necessary, while in other cases accuracy may be more important than speed. Clearly, there is no point in paying for additional characteristics that are not relevant to the class of applications for which the robot is intended.

3.4.3 Reach and Stroke

The reach and the stroke of a robotic manipulator are rough measures of the size of the work envelope (Fu K.S. *et al.*, 1987). The horizontal reach is defined as the maximum radial distance the wrist mounting flange can be positioned from the vertical axis about which the robot rotates. The horizontal stroke represents the total radial distance the wrist can travel. Thus, the horizontal reach minus the horizontal stroke represents the minimum radial distance the wrist can be positioned from the base axis. Since this distance is non-negative, therefore

$$\text{Stroke} \leq \text{Reach}$$

For example, the horizontal reach of a cylindrical coordinate robot is the radius of the outer cylinder of the work envelope, while the horizontal stroke is the difference between the

radii of the concentric outer and the inner cylinders, as shown in Fig. 3.6. The vertical reach of a robot is the maximum elevation above the work surface that the wrist mounting flange can reach. The vertical stroke is the total vertical distance that the mounting flange can reach.

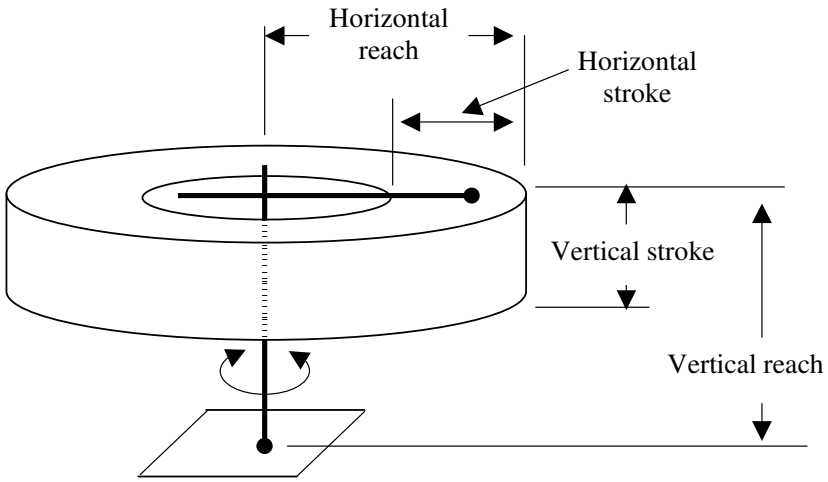


Fig. 3.6: Reach and stroke

Also, the vertical stroke is the total vertical distance that the wrist can travel. Again, the vertical stroke is less than or equal to the vertical reach. For example, the vertical reach of a cylindrical robot will be larger than the vertical stroke if the range of travel of the second axis does not allow the wrist to reach the work surface, as shown in Fig. 3.6. One of the useful characteristics of articulated robots lies in the fact that they often have full work envelopes, in the sense that the stroke equals the reach. However, this feature gives rise to a need for programming safeguards, because an articulated robot can be programmed to collide with itself or the work surface.

3.4.4 Tool Orientation

While the three major axes of a robot determine the shape of the work envelope, the remaining axes determine the kinds of orientation that the tool or hand can assume. If three independent minor axes are available, then arbitrary orientations in a three-dimensional work space can be obtained. A number of conventions are used in the robotics literature to specify tool orientation. The tool orientation convention that will be used here is the yaw-pitch-roll (YPR) system. Yaw, pitch, and roll angles have long been used in the aeronautics

industry to specify the orientation of aircraft. They can also be used to specify tool orientation, as shown in Fig. 3.7.

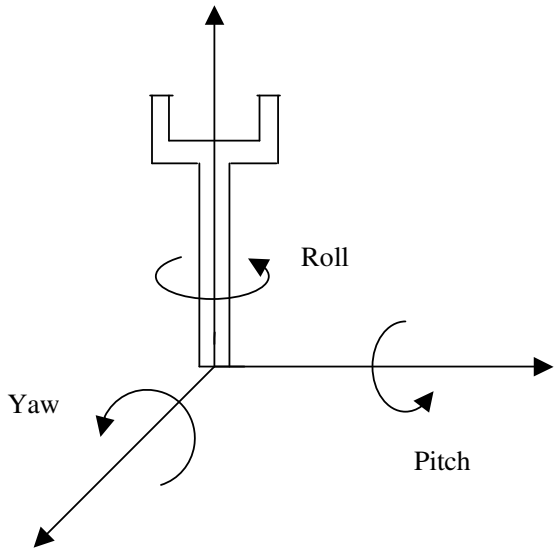


Fig. 3.7: Tool orientation

3.4.5 Repeatability

Repeatability is the measure of accuracy of reaching the same target if the motion is repeated many times. It is concerned with that the robot may not reach the same target every time due to some parameters, which may effect the accuracy of the system.

3.4.6 Precision and Accuracy

Precision is concerned with which accuracy target point can be achieved. It is function of the resolution of actuator. Accuracy refers to a robots ability to achieve target within the work volume.

3.5 Degrees-of-freedom of System

In order to locate a point in space, one needs to be specify three coordinates information, *i.e.*, x, y and z coordinates along the cartesian axes. Three coordinates are necessary and sufficient information to define the location of the point. Similarly, for a three-dimensional device with three degree-of-freedom, within the work space of the device, one can place any point at any desired location. The robots movement can be divided into two general categories, *viz.,* arm or body motions, and wrist motions. The individual joint motions

associated with these two categories are referred as “degree-of-freedom” (Groover M.P. *et al.,* 1986).

In robotic system, the movement of end effector is never being considered as a degree-of-freedom of the system. In some cases the movement of joints are limited or not fully controlled, so such movement is assigned as ½ degree-of-freedom to the joint. Consider linear joint actuated by pneumatic system, where robotic arm is fully extended or fully retraced is the example of ½ degree-of-freedom movement of the joint. For revolute joint, where movement is up to limited angle is also the example of ½ degree-of-freedom. The robots with 2, 3, 3.5, 4 and 5 degree-of-freedom are very commonly used in industrial applications.

Robots have different type of joints for feasible movements of robot manipulator to perform desired task or the connections between links. These joints may be linear, rotary (revolute), sliding or spherical a nature. Spherical joints are not common in robotics as they possess multiple degrees of freedom and are difficult to control. In industry, most of the robots are made up of either a linear (prismatic) or a rotary (revolute) joint. Prismatic joints are linear in nature so that displacement is linear and there is no rotation involved in it. Revolute joints are rotary in nature so that displacement is rotational or angular displacement.

3.5.1 Kinematics Transformations

A robotic manipulator can be modeled as a chain of rigid bodies called links. The links are inter-connected to one another by joints. One end of the chain of the links is fixed to the base, while the other end is free to move (Deb S. R., 2002). The mobile ends have a flange, or faceplate, with a tool or end effector, attached to it. There are typically two types of joints, which interconnect the links, *viz.,* revolute joints and prismatic joints.

The objective is to control both the position and the orientation of the tool in three-dimensional space. The tool, or end effector, can then be programmed to follow a planner trajectory so as to manipulate objects in the work space. In order to program the tool motion, one must first formulate the relationship between the joint variables and the position and the orientation of the tool. This is called the *kinematics transformation*.

Kinematics can be done in two ways as given below

- Direct kinematics
- Inverse kinematics

3.5.2 Direct Kinematics

For direct kinematics, if one must have all the link lengths and joint angles of the robot system. By using this information, calculating the position and orientation of the hand of the robot is called forward kinematics. In direct kinematics, by substituting the values of joint and link variables in the set of equations that defines the particular configuration of the robotic system, one can calculate the position and orientation of the robot.

3.5.3 Inverse Kinematics

For inverse kinematics, one has to place the robot manipulator at a desired location and orientation, then to achieve this desired location, calculating the values of joint and link variables of the hand of the robot is called inverse kinematics. In inverse kinematics, by substituting the position and orientation in the set of equations that defines the particular configuration of the robotic system, one can calculate the values of joint and link variables of the robot.

3.6 Path Planning: Various Conventional Approaches

Path planning is a fundamental problem in robotics, which has been reported by many researchers across the world. Path planning refers to the problem of geometrically specifying a sequence of steps that robot moves between start and goal positions. Path planning is a subset of the trajectory problem and concerns with the kinematics of the robot. Path planning differs from trajectory planning as it includes planning for the linear and angular velocities and acceleration of the robot.

Path planning should be free from any collision with obstacle between the paths and is usually optimised in the terms of displacement/movement or energy consumption, *etc*. As the number of degree-of-freedom increases, the problem of path planning becomes more complicated to find optimal path amongst the possible paths.

Motion planning problem is a combination of path and trajectory planning problems. It is more concerned with the collision checking and obstacle avoidance problem which is quite critical for development of intelligent robots.

The aim of research in robotics is to create autonomous robot, which is capable of analysing and solving general problems in real life. The design and construction of such a robot is out of scope of this thesis. In this research work, our work is limited to path planning. If an autonomous robot is able to plan its path route by itself, it will be a contribution towards the development of autonomous robot.

Configuration Space

The configuration space is an alternative way for considering robot environment was proposed by Lozano P´erez (1983). It specifies instant of a robot or manipulator within its environment. It describes different configuration space or C-space. It reduces the robot to a single point, while expanding the obstacle in the work space to take into account the shape of the robot.

The work space is the physical space in which the robot or manipulator moves. A configuration space is a set of independent parameters that specify the position of robot or manipulator. In the process of constructing C-space, data structure containing obstacles are difficult and mathematics involved is quite complex (Kent S., 1999).

The C-space ability to reduce a planning problem to a single point is utilised by evolutionary path planning which helps in reducing execution time for speeding up the process of path planning.

Classical Approaches to Path Planning

For path planning problem, most of the methods used are the classical approaches. These rely on the conventional problem solving approach. They have limited understanding in order to find the solutions. The classical approaches for path planning problem are discussed as under.

3.6.1 Roadmap Methods

The Roadmap approach reduces C-space to a network of one dimensional curves. Roadmap methods basically involve constructing a graph of the work space in C-space. If start and goal configurations are linked to this map then path planning becomes a graph searching problem. The key issue is the method used to construct the roadmap for a given problem. The search is carried out using standard graph searching techniques, usually a heuristic method that can efficiently locate the shortest path.

3.6.2 Sub-Goal Method

In this method, reachable points are generated from the start point. While joining points with each other to the next point path is generated with local operators. A direct straight-line path is created with local operators having no collision occurring with any obstacle. The local operator is used to check whether the path is feasible without any collision or not? If there is any collision, then some intermediate path is generated. These are called sub-goals.

These sub-goals may be selected at random or using some heuristics. The main objective of this process is to reach the goal without any collision with various sub-goals. This method depends upon local operator to generate sub-goals as shown in Fig. 3.8.

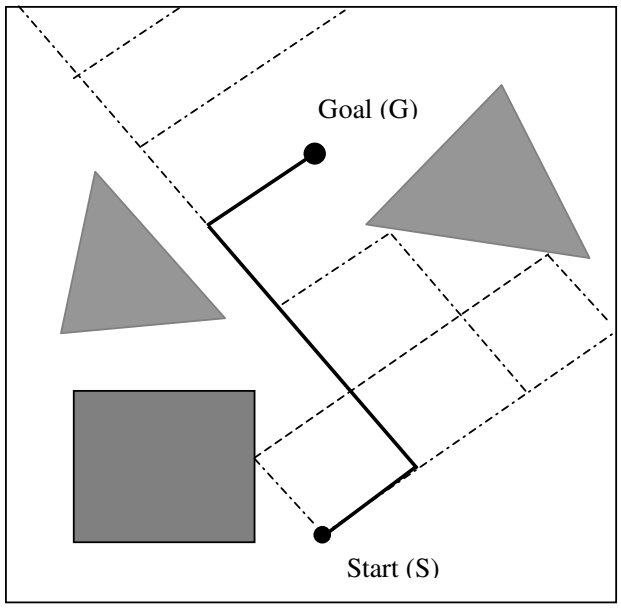


Fig. 3.8: Sub-goal method

3.6.3 Visibility Graph

In Visibility graph (V-graph) method, all the vertices of the obstacles are considered. The start and goal points are called as nodes. These nodes may be connected through segments without intersection with any obstacle and only vertices of obstacles may in contact with segments as shown in Fig. 3.9.

The V-graph relies upon the obstacles being polygons though proposes an extension for non-polygons where obstacle boundaries are comprised of straight-line segments and arcs of a circle. This technique can be improved by removing all the lines that are not tangents or boundaries of the obstacles, resulting in the reduced visibility or tangent (T-graph). This graph can be significantly smaller than the V-graph and therefore quicker to search.

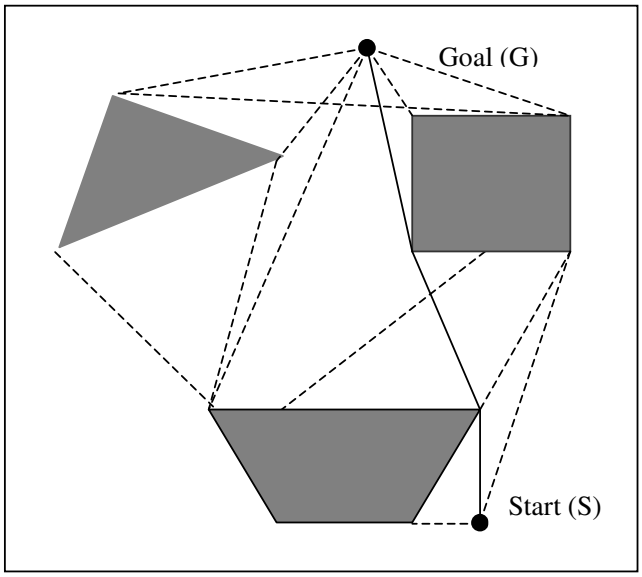


Fig. 3.9: Visibility graph with three obstacles

3.6.4 Voronoi Diagram

A Voronoi diagram is a graph, which follows a line that is equidistant from two or more obstacles. The space is partitioned into regions each containing one obstacle and any point in a region which is closer to the object in that region than to any other obstacle. Fig.3.10 shows a Voronoi diagram, where the obstacles themselves are taken as features. The result is a roadmap, where the edges stay away from the obstacles, often a desirable feature in path planning. A path is formed from S to G by linking S and G as shown in the Fig. 3.10.

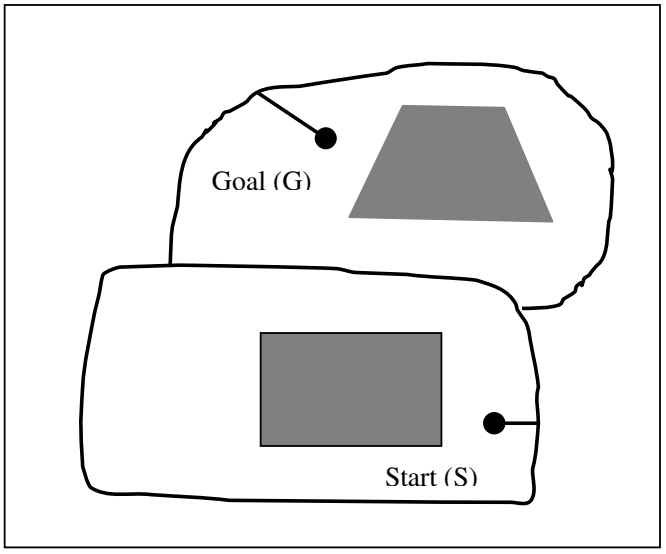


Fig. 3.10: Vornoi diagram

3.6.5 Silhouette Method

In this method, one can construct roadmaps in arbitrary dimensions in a manner, that higher dimensional spaces are projected into a lower dimensional space and the boundary curves of the projection are traced to create silhouette (Canny J., 1987). Silhouettes are projected recursively into lower dimensional spaces until they are reduced to one-dimensional lines. This results in a network of curves, which are linked by new curves that are introduced where silhouettes appear or disappear. The complete network forms the roadmap to which the start and goal are connected and which is searched for the solution path as shown in Fig. 3.11. This method tends to generate paths that trace around the edges of obstacles, not a desirable trait for a path planner. This method rarely used for practical path planning (Hwang and Ahuja, 1992).

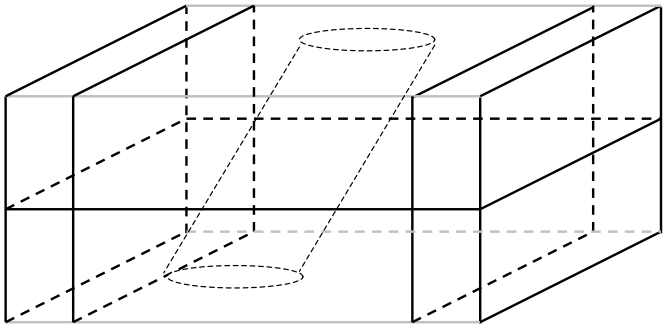


Fig. 3.11: Silhouettes curves

3.6.6 Randomised Roadmaps

It has been shown that it is possible to construct useful roadmaps on a random basis. This method generates a network of randomly, but properly, selected nodes in working area. Pairs of neighboring nodes are linked using a simple local planner, producing a roadmap covering the whole of space. Once the roadmaps are generated, the start and goal are added and normal search algorithms are used to find a solution path. These methods are suited to static environments where the roadmap generation can be done as a preprocessing step.

3.6.7 Cell Decomposition

In this approach, work space is decomposed into a set of cells and the occupancy. Adjacency of cells is computed for free space as well as for obstacles. A path is found by locating the cells containing the start and goal configurations and searching for a sequence

of cells that form a channel through which a path can be passed. These methods can be classified as either exact or approximate cell decomposition.

Exact Cell Decomposition

In exact cell decomposition, free space is divided into a set of non-overlapping cells whose union is exactly free. The adjacency of the cells is represented as a connectivity graph, which is then searched to find a channel from start to goal. The method of decomposition must ensure that each cell has a simple geometry, thus, it is straightforward to compute a path between any two configurations in the cell. Also, it is simple to test the adjacency of cells and to compute a path across the boundary of adjacent cells. Fig. 3.12 and Fig. 3.13 shows examples of two decomposition schemes. In Fig. 3.12 cells are drawn into convex polygons format, whose vertices are formed from the obstacles.

A series of cells connecting graph is constructed from starting and goal point. The cells are convex so once this channel is located, a path is defined through the cells by connecting the midpoints of the portions of the boundaries shared by each successive pair of cells in the channel.

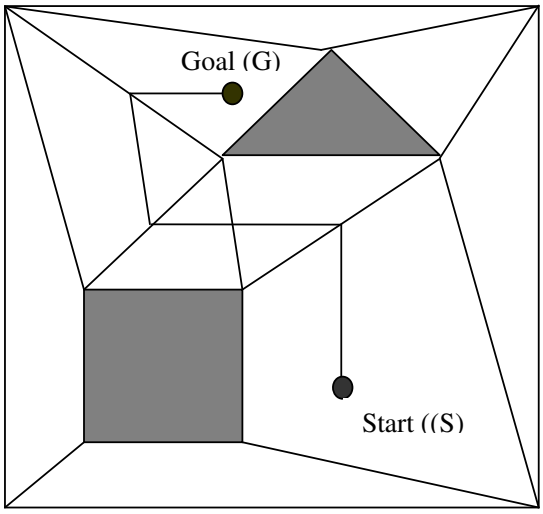


Fig. 3.12: Exact cell decomposition (convex polygon)

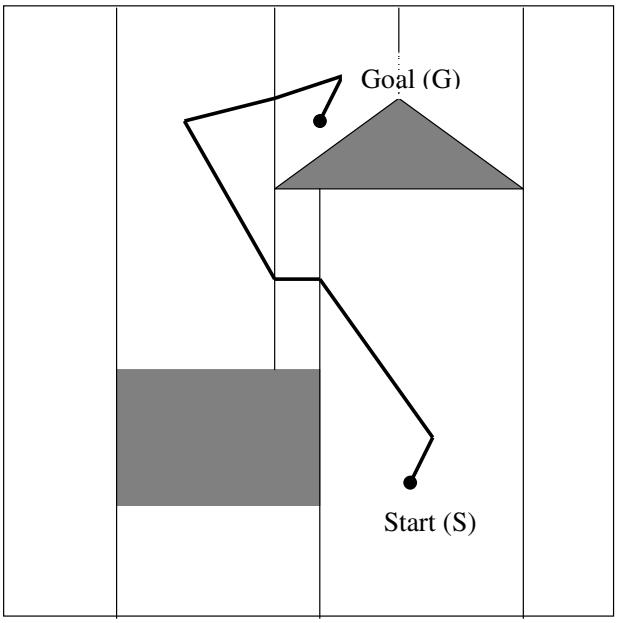


Fig. 3.13: Exact cell decomposition (trapezoidal)

Approximate Cell Decomposition

Exact cell decomposition approach decomposes the space into a set of cells and a path is found through a channel of cells. But here in the decomposition method all the cells must have a simple pre-specified shape. This approach is attractive because cells are generated by iterating the same simple computation. However, because the cells are only an approximation of free space, the planner may fail to find a free path where one exists. The size of the cells can be adapted to the geometry of the obstacle regions. Often, cells are generated in a hierarchical fashion, starting with a coarse decomposition until a path is found or a minimum cell size is reached. Each cell can be classified as fully occupied, partially occupied or free space depends upon the occupancy of obstacles within that cell. The planner will define a path through a channel of connected free cells.

Fig. 3.14 uses a simple regular grid superimposed over the search space. A channel of free cells is located and the path is defined by connecting the centre points of successive cells in the channel.

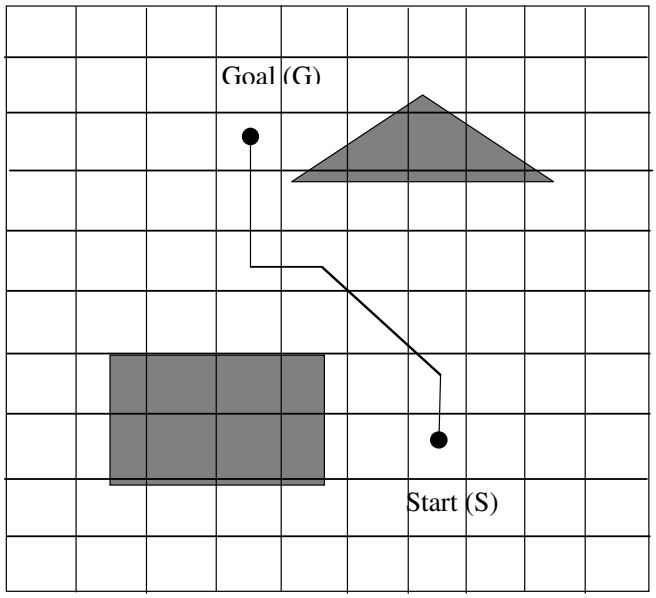


Fig. 3.14: Approximate cell decomposition (regular grid cells)

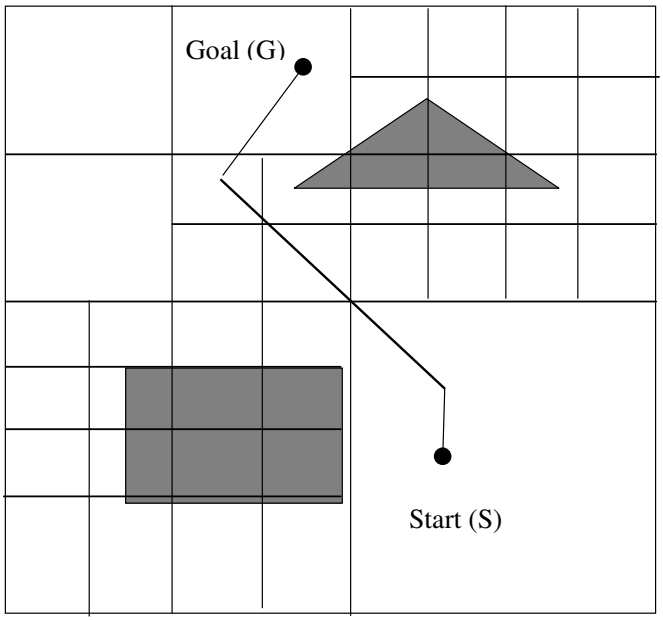


Fig. 3.15: Approximate cell decomposition (quad tree decomposition space)

Fig. 3.15 uses a quad tree decomposition of the space. The space is divided into four quarters and each obstacle occupying quarter is recursively divided into quarters until a pre-

specified minimum cell size is reached. A path is defined by connecting the center points of successive free cells.

3.6.8 Potential Field

Potential field planning methods use principles of physics of electrical potentials as a heuristic to guide the search for a path. The robot is represented as a point that moves in a work space having obstacles. This point is considered to be a particle under the influence of an artificial potential field. Variations in the artificial potential reflect the structure of the space and any obstacle it contains.

Path planning and obstacle avoidance can be handled by classical approaches, based on the conventional techniques. Such a solution depends upon the information available for the particular problem. These conventional methods provide perfect answers. The amount of computation required to provide the solutions may be very large.

With the adoption of artificial intelligence techniques, it can be expected that the robots will perform very complex tasks. The researchers around the world have already reported the successful implementation of the intelligent human path planning using artificial techniques.

3.7 Trajectory Planning

Path and trajectory planning refers to the movement of robot from one location to another in a controlled manner. One can determine the position of a robot, by knowing how much joint variables can be moved to achieve the desired position or target. A path is defined as a sequence of robot motion in a particular order without consideration of time. In trajectory planning, path movement must be attained with specific timings depending upon velocity and accelerations.

3.7.1 Basics of Trajectory Planning

To understand the trajectory in joint-space and cartesian-space, we have considered the two degree-of–freedom robot manipulator. In this case, we desired to move the robot from point A to point B, then the robot joint angles' movement is equal to θ_1 and θ_2 as shown in Fig. 3.16. In this kind of movement path is irregular and distance traversed by each link is also not equal.

Table 3.6: Joint angle movement

θ_1 (in degrees)	30	40	50	50	50
θ_2 (in degrees)	40	50	60	70	80

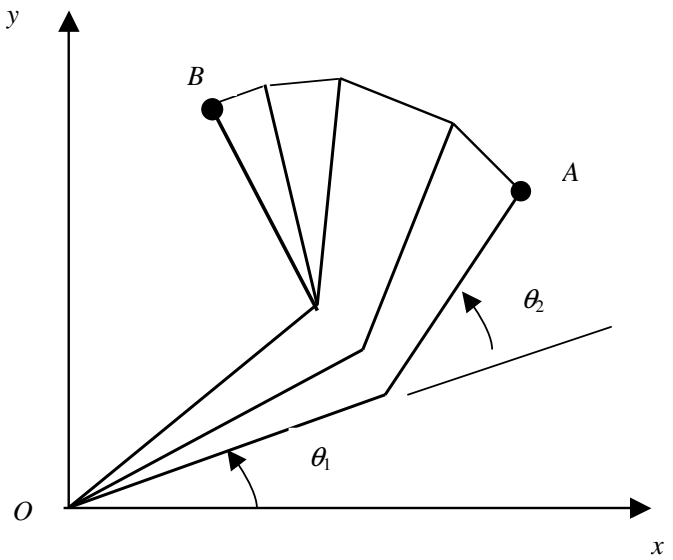


Fig. 3.16: Joint-space movement of a robot with 2 DOF

To move both the joints with equal displacement, there is need of normalising the joint movements by some common factor. In this case both the joints move with different speed, so that they start and stop at the same time as shown in Table 3.7.

Table 3.7: Joint angle normalised movement

θ_1 (in degree)	30	35	40	45	50
θ_2 (in degree)	40	50	60	70	80

In this case, the solution is divided into the straight lines of equal lengths and parts. In this case four parts have been considered as shown in Fig. 3.17. The values for each joint corresponding to each part is called inverse kinematics. In this case, the joint angle movement is not uniform.

This kind of motion is called *trajectory*. To improve accuracy, more parts are

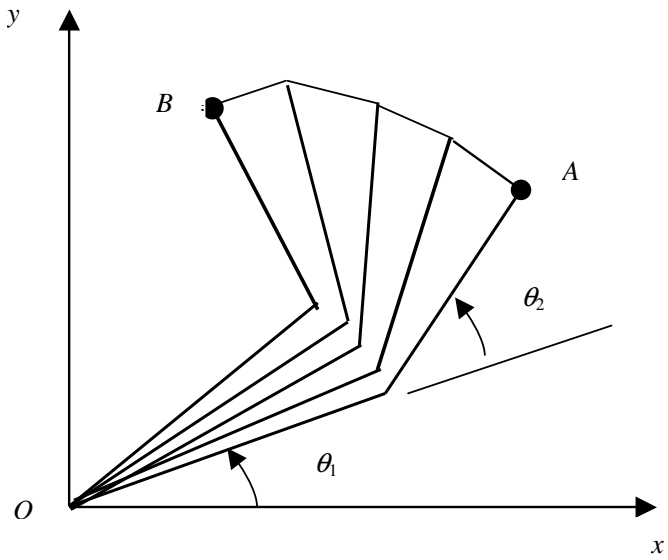


Fig. 3.17: Joint-space movement of a robot with 2 DOF

considered between point A and point B. This trajectory is expressed in cartesian-space, as the movement is calculated on the basis of cartesian-space information as shown in Table 3.8 and Fig. 3.18.

Table 3.8: Cartesian space movement for 2 DOF

θ_1 (in degree)	30	26	28	38	50
θ_2 (in degree)	40	58	68	78	80

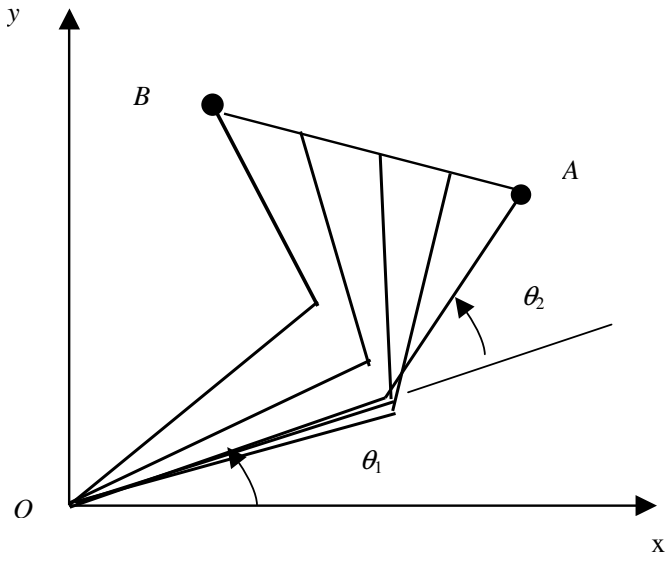


Fig. 3.18: Cartesian space movement of a robot with 2 DOF

3.7.2 Robot Arm Kinematics Representation

Robot kinematics problem can be solved by several techniques. Most commonly used techniques are algebraic, iterative, matrix and geometric approaches. A geometric approach is based on the coordinate system with respect to different links of the manipulator and then obtaining the solution of joint angles movement with rotary joints. Matrix approach uses 4 x 4 homogeneous matrices for obtaining joint solutions for different manipulators. Matrix approach uses systematic and generalized ways to describe and represent the position of robotic arm manipulator link with respect to reference frame. Robotic manipulator link can rotate or translate with respect to a reference frame. A generalised 3 x 3 rotation matrix is used to describe the rotational operations of the link coordinate frame with respect to the reference frame. The homogeneous coordinate frames are used to define the position of manipulator in three dimensional space. The 4 x 4 homogeneous matrix includes translational operations of the adjacent coordinate frames. The homogeneous transformation is a general method for solving the kinematics of robotic arm manipulator.

To make square matrix, if we represent both orientation and position in the same matrix, with the inclusion of scaling factor, it becomes 4 x 4 matrix. Matrices of this form are called homogeneous matrices as given below.

$$H = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.7.3 Representation of Transformations

A transformation is defined as movement in the space as a frame attached with joint moves in the space with respect to reference frame. Transformation may be in the form of pure translation, pure rotational, and combination of rotational or translation.

For pure translation, frame attached with link moves in space without any change in its orientation, such a transformation is called pure translation.

$$T = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where d_x, d_y and d_z are three components of pure translation vector d relative to the reference frame.

Rotation about an x-axis can be written as result of pure rotational movement as

$$Rot(x,\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

Similarly rotation of the frame about the y-axis is

$$Rot(y,\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

Similarly rotation of the frame about the z-axis is

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

Combined transformations consist of translations and rotations about the fixed reference frame. Any transformation can be resolved into a set of translations and rotation motion in particular order. In case of combined transformations, order is very important, and if the order of successive transformations are changed then it gives completely different results.

3.7.4 Combined Transformations

In combined transformations, number of successive translations and rotation motions are involved. These transformations may be about the reference frame or the moving current frame. For robotic arm to reach particular target, it undergoes various transformations and the order of these transformations are fixed. If order is changed, the result will be totally different. These property of combined transformation is used extensively in robotics for forward and inverse kinematics.

3.8 Summary

This chapter has provided an overview of robotics and classical techniques of path planning. Initially, robot and their specifications have been discussed. Then discussion on robotics is restricted to the area of robotic arm as in most of the applications robotic arms are used widely. Different types of robots have been described for which path planning in the next

chapter has been designed. As the robots' degree-of-freedom increase, the problem of path planning also increase in size, resolution, complexity and dimension.

The various classical methods of path planning have been discussed such as obstacle avoidance and task planning. These methods are important for robot obstacle avoidance and task planning and also contribute towards robots solutions. However, these methods are not the focus of this research work. Robotic trajectory planning has been discussed in context to the transformations, which was used in forward and inverse kinematics evaluation of robotic arm movements.

Chapter 4

Robotic Arm Kinematics for Different Degree-of-freedom

4.1 Preamble

Robotic arm manipulators are used for performing useful work, which consists of various joints. To achieve the target, it is required to control the path that manipulator or robotic arm follows through movement of various joint positions. We have analysed the robot arm movement control. The mathematics involved to find the position of target while varying the values of various joints of manipulators has been dealt with. In order to analyse the control of movement of a manipulator, it is required to represent the position of a manipulator at various instants. Thus, for this we have considered two parameters for manipulators, *i.e.,* links and joints. Each joint represents one degree-of-freedom for the robotic arm or manipulator. Joints are labeled J_n where 'n' represents the number of joint with respect to the link of the manipulator. Joint labeled as J_1 where 'n = 1' represents base of the manipulator. Links are labeled as l_n. Here, 'n = 1' is the link attached with the base. In Fig. 4.1 shows the RR (revolute joints) for the manipulator. Fig. 4.2 shows the LL (linear joints) for manipulator.

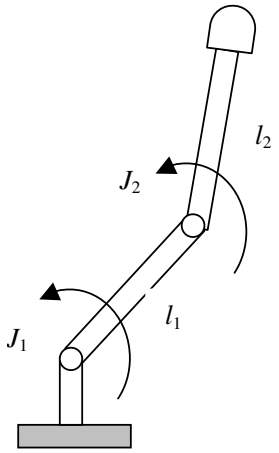


Fig. 4.1: Two revolute joints (RR)

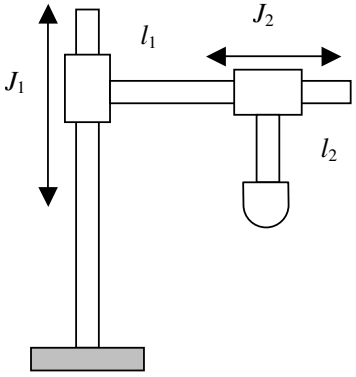


Fig. 4.2: Two linear joints (LL)

4.2 Path Planning for 2 DOF Manipulator

In industrial applications, robotic arms are used for performing various activities. Two DOF robotic arm manipulator is the basic arm model with no complexity. We have considered a planar robot with 2 DOF as shown in Fig. 4.3. The arm moves in *xy*-plane with joint J_1 by link l_1 and joint J_2 by link l_2. The length of link '1' and '2' are l_1 and l_2, respectively. The angles for link '1' and link '2' are θ_1 and θ_2, respectively. The end of arm position or target is at point P(*x,y*).

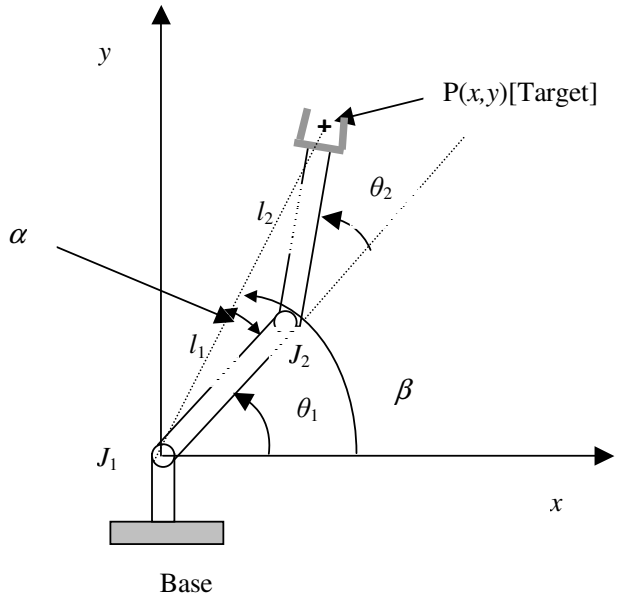


Fig. 4.3: 2 DOF manipulator

4.2.1 Forward Kinematics for 2 DOF Manipulator

In forward kinematics, we have determined the position of the arm manipulator so as to achieve the target with the vector for link '1' and link '2'.

The end-effector coordinates x and y are expressed in terms of the input joint angles θ_1 and θ_2, and link lengths l_1 and l_2 as

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \quad \text{---- (4.1)}$$

and

$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \quad \text{---- (4.2)}$$

4.2.2 Inverse Kinematics for 2 DOF Manipulator

In inverse kinematics, joint angle movements can be found so as to achieve target. Rather, inverse kinematics is more important in order to drive joint angle movement for given target position for a robotic arm manipulator. For the two link manipulator, there are two possible configurations for reaching the target position defined as P (x, y) as shown in Fig. 4.4. In Fig. 4.4 thick dashed line shows the alternate path to reach the target and respective angle θ_1 of link l_1 and $-\theta_2$ of link l_2 (- indicates opposite direction of the angle movement)

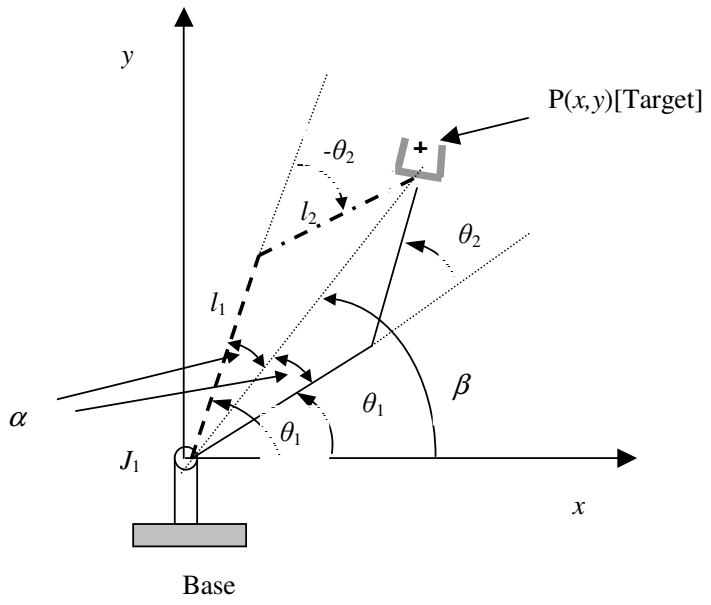


Fig. 4.4: 2 DOF manipulator showing alternate path

Using inverse kinematics, the values of θ_1 and θ_2 are obtained.

Rewriting Eqs. (4.1) and (4.2) can be written as

$$x = l_1 \cos\theta_1 + l_2 \cos\theta_1 \cos\theta_2 - l_2 \sin\theta_1 \sin\theta_2 \qquad \text{----(4.3)}$$

$$y = l_1 \sin\theta_1 + l_2 \sin\theta_1 \cos\theta_2 + l_2 \cos\theta_1 \sin\theta_2 \qquad \text{----(4.4)}$$

Squaring and adding Eqs. (4.3) and (4.4), we get

$$x^2 + y^2 = (l_1 \cos\theta_1 + l_2 \cos\theta_1 \cos\theta_2 - l_2 \sin\theta_1 \sin\theta_2)^2 + (l_1 \sin\theta_1 + l_2 \sin\theta_1 \cos\theta_2 + l_2 \cos\theta_1 \sin\theta_2)^2$$

$$\theta_2 = \cos^{-1}\left(\frac{(x^2 + y^2) - (l_1^2 + l_2^2)}{2 l_1 \times l_2}\right) \qquad \text{---- (4.5)}$$

Defining α and β from Fig. 4.3 or 4.4.

$$\tan\alpha = \frac{l_2 \sin\theta_2}{l_2 \cos\theta_2 + l_2}$$

$$\tan\beta = \frac{y}{x}$$

Where $\tan(\beta-\alpha)=\dfrac{\tan\beta-\tan\alpha}{1+\tan\alpha\tan\beta}$

or

$$\tan(\beta-\alpha)=\frac{\dfrac{y}{x}-\dfrac{l_2\sin\theta_2}{l_2\cos\theta_2+l_2}}{1+\dfrac{l_2\sin\theta_2}{l_2\cos\theta_2+l_2}\times\dfrac{y}{x}}$$

Here, $\theta_1=\beta-\alpha$

We get

$$\theta_1=\tan^{-1}\left(\frac{y(l_1+l_2\cos\theta_2)-xl_2\sin\theta_2}{x(l_1+l_2\cos\theta_2)-yl_2\sin\theta_2}\right) \quad \text{---- (4.6)}$$

Knowing the arm link lengths l_1 and l_2 for position (x, y), the values of joint angles θ_1 and θ_2 can be calculated.

4.2.3 Case Study: 2 DOF Manipulator

We have considered the following configurations and specification for 2 DOF robotic arm manipulator in order to reach the target at coordinates (339**,** 320)**.** The manipulator base is placed at origin (0, 0).

Maximum reach of the robot arm: 559 mm

Length of first link (l_1): 305 mm

Length of second link (l_2): 254 mm

Coordinates of target or destination point (P): (x, y)

Here x = 399 mm and y = 320 mm

From Eq. (4.5)

$$\theta_2=\cos^{-1}\left(\frac{(x^2+y^2)-(l_1^2+l_2^2)}{2l_1\times l_2}\right)$$

$$=\cos^{-1}\left(\frac{((399)^2+(320)^2)-((305)^2+(254)^2)}{2(305)(254)}\right)$$

$$=\cos^{-1}\left(\frac{(159201+102400)-(93025+64516)}{154940}\right)$$

$$= \cos^{-1}\left(\frac{261601 - 157541}{154940}\right)$$

$$= \cos^{-1}\left(\frac{104060}{154940}\right)$$

$$= \cos^{-1}(0.6716)$$

$$\theta_2 = 47.81^\circ \text{ or } -47.81^\circ \qquad \text{----(4.7)}$$

For $\theta_2 = 47.81^\circ$

Now $\sin\theta_2 = \sin(47.81^\circ)$

$$\sin\theta_2 = 0.7409$$

From Eq. (4.6)

$$\theta_1 = \tan^{-1}\left(\frac{y(l_1 + l_2\cos\theta_2) - xl_2\sin\theta_2}{x(l_1 + l_2\cos\theta_2) - yl_2\sin\theta_2}\right)$$

$$\theta_1 = \tan^{-1}\left(\frac{320(305 + 254\times 0.6716) - (399\times 254\times 0.7409)}{399(305 + 254\times 0.6716) - (320\times 254\times 0.7409)}\right)$$

$$= \tan^{-1}\left(\frac{152187.648 - 75087.251}{399(475.5864) - 60220.35}\right)$$

$$= \tan^{-1}\left(\frac{152187.648 - 75087.251}{189758.97 - 60220.35}\right)$$

$$= \tan^{-1}\left(\frac{77100.397}{129538.62}\right)$$

$$= \tan^{-1}(0.5952)$$

$$\theta_1 = 30.76^\circ \qquad \text{----(4.8)}$$

$$\tan\alpha = \frac{l_2\sin\theta_2}{l^2\cos\theta_2 + l_1} \quad \text{and} \quad \tan\beta = \frac{y}{x}$$

$$\tan\alpha = \frac{254\times 0.7409}{254\times 0.6716+305}$$

$$= \frac{188.188}{170.586+305}$$

$$= \frac{188.188}{475.586}$$

$$\tan\alpha = 0.3957$$

$$\therefore \alpha = 21.59^{\circ}$$

$$\tan\beta = \frac{y}{x}$$

$$\tan\beta = \frac{320}{399}$$

$$\tan\beta = 0.802$$

$$\therefore \beta = 38.729°$$

For $\theta_2 = -47.81^{\circ}$

$$\theta_1 = \theta_1 + \alpha + \alpha \text{ (From Fig. 4.4)}$$

$$\theta_1 = 73.94^{\circ} \qquad \text{----(4.9)}$$

Hence, we get the following values for the angles of the links from Eqs. (4.7), (4.8) and (4.9).

$\theta_1 = 30.76^{\circ}$, $\theta_2 = 47.81^{\circ}$

$\theta_1 = 73.94^{\circ}$, $\theta_2 = -47.81^{\circ}$

The results indicate that for each joint, there are two solutions given by Eqs. (4.7), (4.8) and (4.9) using inverse kinematics. Values for joint angles with respect to the different positions of target for 2 DOF manipulator have been obtained through computer program.

4.3 Path Planning for 3 DOF Manipulator

Three link manipulators are the fundamental robotic arms, which are used in micro to macro scale applications, *viz.*, chip fabrications to huge mechanical actuators used in chemical processes. A vertical articulated robotic arm with 3 links is shown in Fig. 4.5. It is considered as an addition to one more degree to the 2 DOF manipulator. It adds to the capability of orientation as well as positioning an end-effector or a tool attached. This third degree-of-freedom represents wrist joint of human arm. The length of links '1', '2' and '3' are l_1, l_2, and l_3 respectively and are joints angles θ_1, θ_2, and θ_3 for joints '1', '2' and '3'

respectively. Also J_1, J_2, and J_3 are the joints of links '1', '2' and '3' respectively. The target point P(x,y) is to be achieved while moving manipulator.

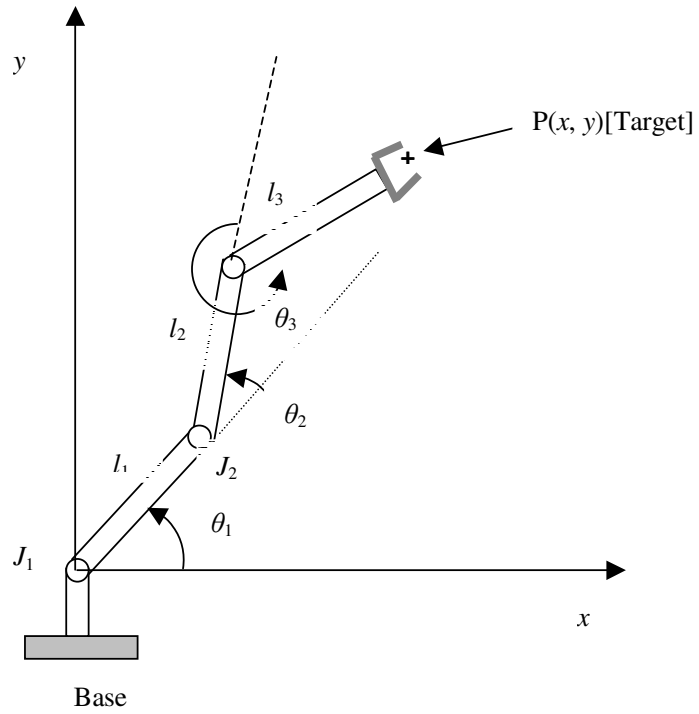


Fig. 4.5: 3 DOF manipulator

4.3.1 Forward Kinematics for 3 DOF Manipulator

In forward kinematics, we have determined the position of the arm manipulator so as to achieve the target with the vector for link '1', link '2' and link '3'.

The end-effector coordinates x and y are expressed in terms of the input joint angles θ_1, θ_2 and θ_3 and link lengths l_1, l_2 and l_3, respectively. These are given as.

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \quad \text{---- (4.10)}$$

$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3) \quad \text{---- (4.11)}$$

and $$\phi = (\theta_1 + \theta_2 + \theta_3) \quad \text{---- (4.12)}$$

4.3.2 Inverse Kinematics for 3 DOF Manipulator

In inverse kinematics, one can find joint angle movements so as to achieve the target. For the three link manipulator, there are many possible configurations for reaching the target position defined as P(x, y). Here, it results multiple solutions for joint angles.

The values of θ_1, θ_2 and θ_3 are obtained using inverse kinematics.

Rewriting Eqs. (4.10) and (4.11) using Eq. (4.12).

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\phi) \quad \text{----(4.13)}$$

and $$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\phi) \quad \text{----(4.14)}$$

These, Eqs. can be rearranged as

$$x - l_3 \cos\phi = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \quad \text{----(4.15)}$$

and $$y - l_3 \sin\phi = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \quad \text{----(4.16)}$$

Renaming the left hand sides, we have

$$x' = x - l_3 \cos\phi \quad \text{----(4.17)}$$

Where $\phi = \tan^{-1}\left(\frac{y}{x}\right)$

and $$y' = y - l_3 \sin\phi \quad \text{----(4.18)}$$

Eq. (4.13) and Eq. (4.14) are modified using Eq. (4.17) and Eq. (4.18) as

$$x' - l_1 \cos\theta_1 = l_2 \cos(\theta_1 + \theta_2) \quad \text{----(4.19)}$$

and $$y' - l_1 \sin\theta_1 = l_2 \sin(\theta_1 + \theta_2) \quad \text{----(4.20)}$$

Squaring and adding the Eqs. (4.19) and (4.20), we get

$$(-2l_1 x')\cos\theta_1 + (-2l_1 y')\sin\theta_1 + \left(x'^2 + y'^2 + l_1^2 - l_2^2\right) = 0 \quad \text{----(4.21)}$$

Eq. (4.21) is a single Eq. with one unknown θ_1, we can write this as

$$P\cos\alpha + Q\sin\alpha + R = 0 \quad \text{----(4.22)}$$

Were $P = -2l_1 x'$, $Q = -2l_1 y'$, $\alpha = \theta_1$ and $R = x'^2 + y'^2 + l_1^2 - l_2^2$

Defining another variable 'γ' such that

$$\gamma = \tan^{-1}\left(\frac{Q/\sqrt{P^2 + Q^2}}{P/\sqrt{P^2 + Q^2}}\right) \quad \text{----(4.23)}$$

Rewriting Eq. (4.21), using Eqs. (4.22) and (4.23), we get

$$\cos\gamma\cos\alpha + \sin\gamma\sin\alpha + \frac{R}{\sqrt{P^2 + Q^2}} = 0 \quad \text{----(4.24)}$$

or

$$\cos(\alpha - \gamma) = \frac{-R}{\sqrt{P^2 + Q^2}} \quad \text{----(4.25)}$$

Eq. (4.28) has two solutions, *viz.*,

$$\alpha = \gamma + \sigma \cos^{-1}\left(\frac{-R}{\sqrt{P^2 + Q^2}}\right) \qquad \text{where } \sigma = \pm 1 \qquad \text{----(4.26)}$$

Obtaining θ_1, θ_2 and θ_3 as

$$\theta_1 = \gamma + \sigma \cos^{-1}\left(\frac{-(x'^2 + y'^2 + l_1^{\,2} - l_2^{\,2})}{2l_1\sqrt{x'^2 + y'^2}}\right) \qquad \text{----(4.27)}$$

Substituting the value of θ_1 in Eqs. (4.19), and (4.20), we get

$$\theta_2 = \tan^{-1}\left(\frac{y' - l_1 \sin\theta_1 / l_2}{x' - l_1 \cos\theta_1 / l_2}\right) - \theta_1 \qquad \text{----(4.28)}$$

Here $x' = x - l_3 \cos(\phi)$ and $y' = y - l_3 \sin(\phi)$

and θ_3 will be calculated as:

$$\theta_3 = (\phi - \theta_1 - \theta_2) \qquad \text{----(4.29)}$$

There are two solutions to the inverse kinematics of a three link manipulator.

4.3.3 Case Study: 3 DOF manipulator

For 3 DOF manipulator, we have considered the manipulator having the lengths of the three links as 330 mm (l_1), 320 mm (l_2) and 265 mm (l_3) respectively. End-effector is moved from reference point (0, 0, 0) to the destination point i.e. x = 50, y = 25. These specifications are fed to the following three fundamental inverse kinematics Eqs. (4.10), (4.11) and (4.12).

$$\phi = \tan^{-1}\left(\frac{25}{50}\right)$$

$$\phi = \tan^{-1}(0.5)$$

$$\phi = 26.565^\circ \qquad \text{----(4.30)}$$

Substituting values of x, y, l_1, l_2, l_3 and ϕ we get

$$50 = 330\cos\theta_1 + 320\cos(\theta_1 + \theta_2) + 265\cos(26.565^\circ) \qquad \text{----(4.31)}$$

$$25 = 330\sin\theta_1 + 320\sin(\theta_1 + \theta_2) + 265\sin(26.565^\circ) \qquad \text{----(4.32)}$$

and $$26.565^\circ = (\theta_1 + \theta_2 + \theta_3) \qquad \text{----(4.33)}$$

Rewriting Eq. (4.31)

$$50 = 330\cos\theta_1 + 320\cos(\theta_1 + \theta_2) + 265\cos(26.565^\circ)$$

$$50 - 237.023 = 330\cos\theta_1 + 320\cos(\theta_1 + \theta_2)$$

$$-187.023 - 330\cos\theta_1 = 320\cos(\theta_1 + \theta_2) \qquad \text{----(4.34)}$$

Rewriting Eq. (4.32) again

$$25 = 330\sin\theta_1 + 320\sin(\theta_1 + \theta_2) + 265\sin(26.565^\circ)$$

$$-93.511 - 330\sin\theta_1 = 320\sin(\theta_1 + \theta_2) \qquad \text{----(4.35)}$$

Squaring both sides of Eq. (4.34) and Eq. (4.35), we get

$$(-187.023 - 330\cos\theta_1)^2 = 320^2\cos^2(\theta_1 + \theta_2)$$

$$(34977.603 + 108900\cos^2\theta_1 + 123435.18\cos\theta_1) = 102400\cos^2(\theta_1 + \theta_2) \qquad \text{----(4.36)}$$

$$(-93.511 - 330\sin\theta_1)^2 = 320^2\sin^2(\theta_1 + \theta_2)$$

$$(152621.91 + 108900\sin^2\theta_1 + 61717.26\sin\theta_1) = 102400\sin^2(\theta_1 + \theta_2) \qquad \text{----(4.37)}$$

Adding Eqs. (4.36) and Eq. (4.37), we get

$$(152621.91 + 123435.18\cos\theta_1 + 61717.26\sin\theta_1) = 102400$$

or

$$123435.18\cos\theta_1 + 61717.26\sin\theta_1 + 50221.91 = 0 \qquad \text{----(4.38)}$$

Eq. (4.38), can now be written as

$$P\cos\alpha + Q\sin\alpha + R = 0$$

Here P = 123435.18, Q = 61717.26, R = 50221.91 and $\alpha = \theta_1$

From Eq. (4.23)

$$\gamma = \tan^{-1}\left(\frac{Q}{P}\right)$$

$$= \tan^{-1}\left(\frac{61717.26}{123435.18}\right)$$

Then $\gamma = 26.565^\circ$ ----(4.39)

Eq. (4.22) can be written in the form

$$\cos\gamma\cos\alpha + \sin\gamma\sin\alpha + \frac{R}{(P^2 + Q^2)} = 0 \qquad \text{----(4.40)}$$

or

$$\cos(\alpha - \gamma) = -0.364$$

$$\cos(\alpha - 26.565^\circ) = -0.364$$

$$\alpha - 26.565^\circ = \pm\cos^{-1}(-0.364)$$

$$\alpha = 137.911^\circ,\ -84.781^\circ$$

or

$$\theta_1 = 137.911^\circ,\ -84.781^\circ \qquad \text{----(4.41)}$$

For $\theta_1 = 137.911^0$**,** Eq. (4.34) and (4.35) can be rewritten as

$$-187.023 - 330\cos(137.911^\circ) = 320\cos(137.911^\circ + \theta_2)$$

and $$-93.511 - 330\sin(137.911^\circ) = 320\sin(137.911^\circ + \theta_2)$$

From above Eqs., we get

$$-57.872 = 320\cos(137.911^\circ + \theta_2) \quad \text{----(4.42)}$$

$$-314.705 = 320\sin(137.911^\circ + \theta_2) \quad \text{----(4.43)}$$

Dividing Eq. (4.43) by (4.42) we get as below.

$$\tan(\theta_2 + 137.911^\circ) = -5.438$$

$$\theta_2 = -217.49^\circ \quad \text{----(4.44)}$$

For $\theta_1 = -84.781^\circ$**,** Eq. (4.34) and (4.35) can be rewritten as

$$-187.023 - 330\cos(-84.781^\circ) = 320\cos(-84.781 + \theta_2)$$

and $$-93.511 - 330\sin(-84.781) = 320\sin(-84.781 + \theta_2)$$

From above Eqs., we get

$$-217.041 = 320\cos(-84.781 + \theta_2) \quad \text{----(4.45)}$$

and $$235.121 = 320\sin(-84.781 + \theta_2) \quad \text{----(4.46)}$$

Dividing Eqs. (4.46) by (4.45), we get

$$\tan(\theta_2 - 84.781^\circ) = -1.083$$

$$\theta_2 - 84.781^\circ = \tan^{-1}(-1.083)$$

$$\theta_2 = 37.491^\circ \quad \text{----(4.47)}$$

Now, finding value of θ_3 from Eq. (4.13), we get

$$\phi = (\theta_1 + \theta_2 + \theta_3)$$

$$26.565^\circ = (\theta_1 + \theta_2 + \theta_3)$$

For $\theta_1 = 137.911^\circ$, $\theta_2 = -217.49^\circ$

$$26.565^\circ = (137.911^\circ - 217.49^\circ + \theta_3)$$

$$\theta_3 = 106.144^\circ \quad \text{----(4.48)}$$

For $\theta_1 = -84.781^\circ$, $\theta_2 = 37.491^\circ$

$$26.565^\circ = (-84.781^\circ + 37.491^\circ + \theta_3)$$

$$\theta_3 = 73.855^\circ \quad \text{----(4.49)}$$

Hence,

$$\theta_3 = 106.144^\circ,\ \theta_3 = 73.855^\circ$$

Hence, we get the following values for the angles of the links from Eqs. (4.41), (4.44), (4.47), (4.48) and (4.49).

$\theta_1 = 137.911^\circ,\ \theta_2 = -217.49^\circ,\ \theta_3 = 106.144^\circ$

and $\theta_1 = -84.781^\circ,\ \theta_2 = 37.491^\circ,\ \theta_3 = 73.855^\circ$

From the above results, we find that each joint have two solutions using inverse kinematics. Values for joint angles with respect to the different positions of target for 3 DOF manipulator have been obtained through program.

4.4 Path Planning for 4 DOF Manipulator

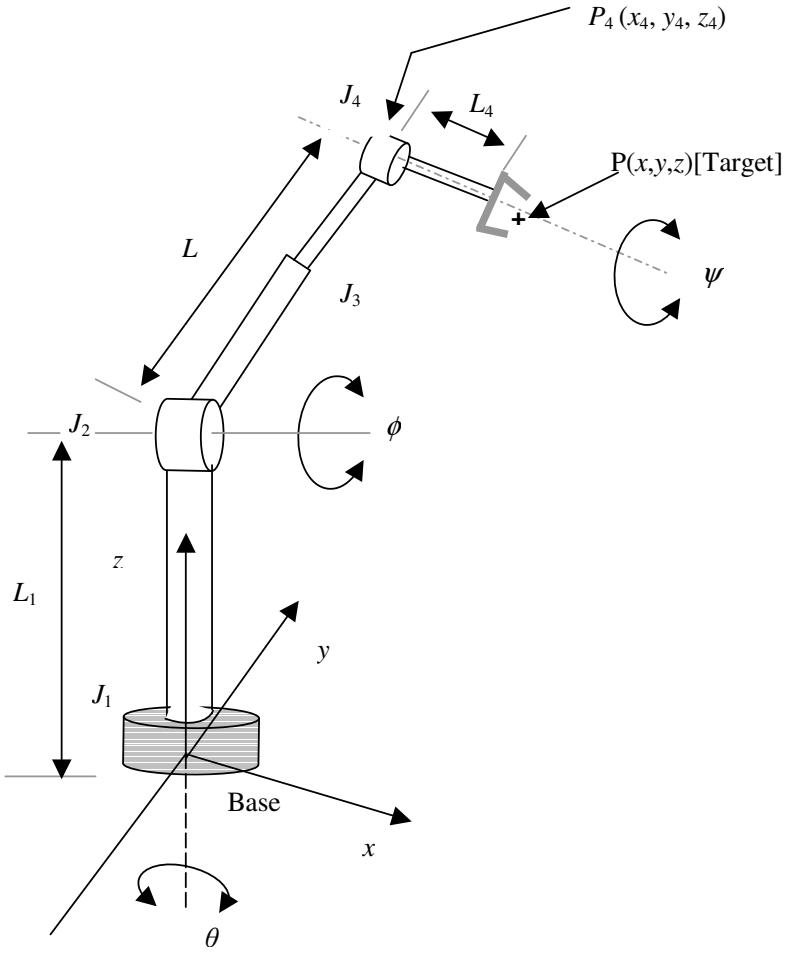


Fig. 4.6: 4 DOF manipulator

The manipulator has 4 degree-of-freedom as shown in Fig. 4.6. Joint '1' (J_1) allows rotation about the z-axis. Joint '2' (J_2) allows rotation about an axis that is perpendicular to the z-axis. Joint '3' (J_3) is a linear joint, which is capable of sliding over a certain range. Joint '4' (J_4) allows rotation about an axis that is parallel to the axis along which joint '2' (J_2) moves. Angle θ is the rotation along joint (J_1). Angle ϕ is the rotation along joint (J_2) called as elevation angle. The length of linear extension L is the distance between joint (J_2) and joint (J_4). Angle ψ is the angle which joint (J_4) subtends with x-y plane called as pitch angle. Here L_1 is the length of link between joint (J_1) and joint (J_2) and L_4 is the length of link between joint (J_4) and end-effector.

4.4.1. Forward kinematics for 4 DOF Manipulator

In forward kinematics, the arm manipulator tries to achieve the target. Let it be given as P(x, y, z). The end-effector coordinates x, y and z are expressed in terms of the input joint angles θ, ϕ and ψ and link lengths L_1, L and L_4 as

$$x = \cos\theta(L\cos\phi + L_4\cos\psi) \quad \text{----(4.50)}$$

$$y = \sin\theta(L\cos\phi + L_4\cos\psi) \quad \text{----(4.51)}$$

and $$z = L_1 + L\sin\phi + L_4\sin\psi) \quad \text{----(4.52)}$$

where $P_4(x_4, y_4, z_4)$, coordinates of joint J_4 are

$$x_4 = L_4(\cos\psi\cos\theta) \quad \text{----(4.53)}$$

$$y_4 = L_4(\cos\psi\sin\theta) \quad \text{----(4.54)}$$

and $$z_4 = L_4\sin\psi \quad \text{----(4.55)}$$

4.4.2 Inverse Kinematics for 4 DOF Manipulator

For the four link manipulator, there are multiple possible configurations for reaching the target position defined as P(x, y, z). It results in multiple solutions for joint angles.

Using inverse kinematics, the values of θ, ϕ and ψ are obtained.

By shifting the frame of reference to the joint J_2. The axis of new frame becomes parallel to respective axis of other frame as such, we have

$$x = x_1 \quad \text{----(4.56)}$$

$$y = y_1 \quad \text{----(4.57)}$$

and $$z = z_1 + L_1 \quad \text{----(4.58)}$$

where (x_1, y_1, z_1) are associated with next frame attached to joint J_2.

Length of link '*L*', with joint J_4 are spherically polar with respect to frame attached to J_2. Hence (*x*, *y z*) are coordinates of the frame attached with the joint J_2, thus we have

$$x_1 = L\cos\phi\cos\theta \quad \text{----(4.59)}$$

$$y_1 = L\cos\phi\sin\theta \quad \text{----(4.60)}$$

and $$z_1 = L\sin\phi \quad \text{----(4.61)}$$

shifting the frame of reference from J_2 to J_4, we get

$$x_1 = x_4 + L\cos\phi\cos\theta \quad \text{----(4.62)}$$

$$y_1 = y_4 + L\cos\phi\sin\theta \quad \text{----(4.63)}$$

and $$z_1 = z_4 + L\sin\phi \quad \text{----(4.64)}$$

In above Eqs., substituting the (x_4, y_4, z_4) from Eqs. (4.53), (4.54) and (4.55), we get

$$x_1 = L_4\cos\psi\cos\theta + L\cos\phi\cos\theta \quad \text{----(4.65)}$$

$$y_1 = L_4\cos\psi\sin\theta + L\cos\phi\sin\theta \quad \text{----(4.66)}$$

and $$z_1 = L_4\sin\psi + L\sin\phi \quad \text{----(4.67)}$$

From Eqs. (4.56), (4.57) and (4.58), we get

$$x = \cos\theta(L_4\cos\psi + L\cos\phi) \quad \text{----(4.68)}$$

$$y = \sin\theta(L_4\cos\psi + L\cos\phi) \quad \text{----(4.69)}$$

and $$z = L_1 + L_4\sin\psi + L\sin\phi \quad \text{----(4.70)}$$

Hence, the coordinates of Joint J_4 with respect to Joint J_1 can be written as

$$x_4 = x - \cos\theta(L_4\cos\psi) \quad \text{----(4.71)}$$

$$y_4 = y - \sin\theta(L_4\cos\psi) \quad \text{----(4.72)}$$

and $$z_4 = z - L_4\sin\psi \quad \text{----(4.73)}$$

Replacing the value of *x*, *y*, *z* from Eqs. (4.68), (4.69) and (4.70) in Eqs. (4.71), (4.72) and (4.73), we get

$$x_4 = \cos\theta(L\cos\phi + L_4\cos\psi) - \cos\theta(L_4\cos\psi)$$

$$y_4 = \sin\theta(L\cos\phi + L_4\cos\psi) - \sin\theta(L_4\cos\psi)$$

and $$z_4 = L_1 + L\sin\phi + L_4\sin\psi - L_4\sin\psi$$

which implies

$$x_4 = L\cos\phi\cos\theta \quad \text{----(4.74)}$$

$$y_4 = L\sin\theta\cos\phi \quad \text{----(4.75)}$$

and $z_4 = L_1 + L\sin\phi$ ----(4.76)

From above Eqs., we get

$$x_4^2 + y_4^2 + (z_4 - L_1)^2 = L^2\cos^2\phi\cos^2\theta + L^2\sin^2\theta\cos^2\phi + L^2\sin^2\phi$$

$$= L^2\cos^2\phi[\cos^2\theta + \sin^2\theta] + L^2\sin^2\phi$$

$$= L^2\cos^2\phi + L^2\sin^2\phi$$

$$x_4^2 + y_4^2 + (z_4 - L_1)^2 = L^2 \quad \text{----(4.77)}$$

From Eqs. (4.68) and Eq. (4.69), we get

$$\frac{y}{x} = \frac{\sin\theta(L_4\cos\psi + L\cos\phi)}{\cos\theta(L_4\cos\psi + L\cos\phi)}$$

therefore

$$\theta = \tan^{-1}\left(\frac{y}{x}\right) \quad \text{----(4.78)}$$

From Eqs. (4.74) and (4.75), we get

$$\frac{y_4}{x_4} = \frac{L\sin\theta\cos\phi}{L\cos\theta\cos\phi}$$

$$\frac{y_4}{x_4} = \tan\theta$$

therefore

$$\theta = \tan^{-1}\left(\frac{y_4}{x_4}\right) \quad \text{-----(4.79)}$$

Using Eq. (4.74) and Eq. (4.75), we find value of $(x_4^2 + y_4^2)$ as

$$(x_4^2 + y_4^2) = L^2\cos^2\phi\cos^2\theta + L^2\sin^2\theta\cos^2\phi$$

$$= L^2\cos^2\phi(\cos^2\theta + \sin^2\theta)$$

Therefore, $(x_4^2 + y_4^2) = L^2\cos^2\phi$ ----(4.80)

Substituting the value of $(x_4^2 + y_4^2)$ from Eq. (4.77) in Eq. (4.80), we get

$$L^2 - (z_4 - L_1)^2 = L^2\cos^2\phi$$

$$L^2 - L^2\cos^2\phi = (z_4 - L_1)^2$$

$$L^2(1 - \cos^2\phi) = (z_4 - L_1)^2$$

$$L^2\sin^2\phi = (z_4 - L_1)^2$$

$$\sin^2\phi = \frac{(Z_4 - L_1)^2}{L^2}$$

Hence, there are two possible values of ϕ, *viz.,*

$$\phi = \sin^{-1}(\pm\frac{Z_4 - L_1}{L}) \qquad \text{----(4.81)}$$

Squaring x_4, y_4 and then adding using Eqs. (4.74) and (4.75), we get

$$x_4^2 + y_4^2 = L_4^2\cos^2\psi\cos^2\theta + L_4^2\cos^2\psi\sin^2\theta$$

$$= L_4^2\cos^2\psi(\cos^2\theta + \sin^2\theta)$$

$$= L_4^2\cos^2\psi$$

therefore,

$$\cos^2\psi = \frac{x_4^2 + y_4^2}{L_4^2}$$

$$\psi = \cos^{-1}\pm\left(\frac{x_4^2 + y_4^2}{L_4^2}\right)^{1/2}$$

or from Eq.(4.77)

$$\psi = \cos^{-1}\pm\left(\frac{L^2 - (z_4 - L_1)^2}{L_4^2}\right)^{1/2} \qquad \text{----(4.82)}$$

From Eqs. (4.78), (4.79), (4.81) and (4.82), we can calculate the values of joint angles θ, ϕ and ψ.

4.4.3 Case Study: 4 DOF Manipulator

We have considered the four degree-of-freedom robotic arm manipulator having following specifications:

Maximum reach of the robot arm: 485 mm

Length of first link (L_1): 305 mm

Length of second link (L): 434 mm

Length of third link (L_4): 51 mm

Coordinate of origin or reference point: (0, 0, 0)

Coordinate of target (P) or destination point: (*x, y, z*)

Here x = 406 mm, y = 127 mm and z = 533 mm

A three dimensional 4 degree-of-freedom manipulator has three links L_1, L and L_4. We move the end-effector from reference point (0,0,0) to destination position (x, y, z) with pitch angle ψ. We have assumed the fixed length of extendable link L.

These specifications are fed to the three fundamental inverse kinematics Eqs. (4.50), (4.51) and (4.52).

From Eq. (4.78), we get

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

$$\theta = \tan^{-1}\left(\frac{127}{406}\right)$$

$$\theta = \tan^{-1}(0.3128)$$

or

$$\theta = 17.36° \qquad \text{----(4.83)}$$

For angle ψ,

Let us assume, $C_1 = \sqrt{x^2 + y^2}$

$$C_1 = \sqrt{(406)^2 + (127)^2}$$

$$C_1 = 425.4 \qquad \text{----(4.84)}$$

Let us assume, $C_2 = (z - L_1)$

$$C_2 = 533 - 305$$

$$C_2 = 228 \qquad \text{----(4.85)}$$

Let us assume, $D = ({C_1}^2 + {C_2}^2 + {L_4}^2 - L^2)$

$$D = (425.4)^2 + (228)^2 + (51)^2 - (434)^2$$

$$D = 47194 \qquad \text{----(4.86)}$$

Let us assume, $E = (D^2 - 4{C_1}^2{L_4}^2)$

$$E = (47194)^2 - 4(425.4)^2(51)^2$$

$$E = 3.44514 \times 10^8 \qquad \text{----(4.87)}$$

Let us assume, $F = 4DC_1L_4$

$$F = 4 \times (47194)(425.4)(51)$$

$$F = 4.09557 \times 10^9 \qquad \text{----(4.88)}$$

From Eq. (4.82)

Thus, $$\sin\psi = \frac{(F + \sqrt{(F^2 - 4(8 \times {C_1}^2{L_4}^2)E)})}{2 \times 8{C_1}^2{L_4}^2} \qquad \text{----(4.89)}$$

$$\sin\psi = \frac{(4.09557\times10^{9} + \sqrt{((4.09557\times10^{9})^{2} - 4(8\times(425.4)^{2}(51)^{2})\times3.44514\times10^{8}))}}{2\times8(425.4)^{2}(51)^{2}}$$

$$\sin\psi = \frac{7.49918\times10^{9}}{7.53104\times10^{9}}$$

$$\sin\psi = 0.9957$$

$$\psi = \sin^{-1}(0.9957)$$

$$\psi = 84.694^{\circ} \qquad \text{----(4.90)}$$

From Eq. (4.82)

Thus, $$\sin\psi = \frac{(F - \sqrt{(F^{2} - 4(8\times C_{1}^{2}L_{4}^{2})E)})}{2\times8C_{1}^{2}L_{4}^{2}} \qquad \text{----(4.91)}$$

$$\sin\psi = \frac{(4.09557\times10^{9} - \sqrt{((4.09557\times10^{9})^{2} - 4(8\times(425.4)^{2}(51)^{2})\times3.44514\times10^{8}))}}{2\times8(425.4)^{2}(51)^{2}}$$

$$\sin\psi = \frac{6.91955\times10^{8}}{7.53104\times10^{8}}$$

$$\sin\psi = 0.0918$$

$$\psi = \sin^{-1}(0.0918)$$

$$\psi = 5.2696^{\circ} \qquad \text{----(4.92)}$$

For angle ϕ,

Let us assume, $D_p = (C_1^{2} + C_2^{2} + L^{2} - L_4^{2})$ ----(4.93)

$$D_p = (425.4)^{2} + (228)^{2} + (434)^{2} - (51)^{2}$$

Let us assume, $E_p = (D_p^{2} - 4C_1^{2}L^{2})$ ----(4.94)

$$E_p = D_p^{2} - 4C_1^{2}L^{2}$$

$$E_p = 3.89697\times10^{10} \qquad \text{----(4.95)}$$

Let us assume, $F_p = (4\times D_p C_2 L)$ ----(4.96)

$$F_p = 4\times D_p C_2 L$$

$$F_p = 1.65726\times10^{11} \qquad \text{----(4.97)}$$

From Eq. (4.81)

Thus, $$\sin\phi = \frac{(F_p + \sqrt{(F_p^{2} - 4(4\times(C_1^{2} + C_2^{2})L^{2})E_p))}}{2\times4\times(C_1^{2} + C_2^{2})L^{2}} \qquad \text{----(4.98)}$$

$$\sin\phi = \frac{(F_p + \sqrt{(F_p{}^2 - 4(4\times(C_1{}^2 + C_2{}^2)L^2)E_p))}}{2\times 4\times(C_1{}^2 + C_2{}^2)L^2}$$

$$\sin\phi = \frac{1.76074\times 10^{11}}{3.51018\times 10^{11}}$$

$$\sin\phi = 0.5016$$

$$\phi = \sin^{-1}(0.5016)$$

$$\phi = 30.09^{\circ} \qquad \text{----(4.99)}$$

From Eq. (4.81)

Thus, $$\sin\phi = \frac{(F_p - \sqrt{(F_p{}^2 - 4(4\times(C_1{}^2 + C_2{}^2)L^2)E_p))}}{2\times 4\times(C_1{}^2 + C_2{}^2)L^2} \qquad \text{----(4.100)}$$

$$\sin\phi = \frac{(F_p - \sqrt{(F_p{}^2 - 4(4\times(C_1{}^2 + C_2{}^2)L^2)E_p))}}{2\times 4\times(C_1{}^2 + C_2{}^2)L^2}$$

$$\sin\phi = \frac{1.55379\times 10^{11}}{3.51019\times 10^{11}}$$

$$\sin\phi = 0.4426$$

$$\phi = \sin^{-1}(0.4426)$$

$$\phi = 26.26^{\circ} \qquad \text{----(4.101)}$$

The following values of θ, ϕ and ψ have been obtained for 4 degree-of-freedom manipulator from Eqs. (4.83), (4.90), (4.92), (4.99) and (4.101).

$\theta = 17.36°$, $\phi = 30.09^{\circ}$, $\psi = 84.694^{\circ}$

and $\theta = 17.36°$, $\phi = 26.26^{\circ}$, $\psi = 5.2696^{\circ}$

From the above results, we have obtained two solutions using inverse kinematics. Values for joint angles with respect to the different positions of target for 4 DOF manipulator have been obtained through computer program.

4.5 Summary

In this chapter, path planning of the robotic arm manipulator has been discussed. Path planning for forward kinematics as well as for inverse kinematics have been discussed. Case study of 2 DOF, 3 DOF and 4 DOF manipulators have been considered. Using inverse kinematics, solutions for case studies have been evaluated. These solutions are mathematically determined and have been used in next chapter for optimisation process

using only GAs and fuzzy GAs. For robotic path optimisation inverse kinematics has been preferred over direct kinematics as it is easy to define the problem. If we know the target, we can correspondingly determine the values of various joint movements so as to achieve that target. In robotics application in most of the time, we know the target and correspondingly have to evaluate joint angle movement to achieve the target.

Chapter 5

Movement Optimisation of Different Degree-of-freedom using Fuzzy Logic and Genetic Algorithms†

5.1 Preamble

Path planning problems are widely exercised using traditional methods. This chapter describes the use of fuzzy logic and genetic algorithms to the path planning problems in robotic arm movement. The capabilities of artificial intelligence techniques are explored in order to find the optimal solution without using much complex mathematics while using this technique.

The genetic optimisation replaces the tedious process of trial and error for a better combination of joint angles. It is valid as per inverse kinematics for robotic arm movement. The fitness function in genetic algorithms as implemented in this case is augmented by three attributes, *viz.*, joint movement, friction and settling time. At any time, the values of these three attributes are found with the help of fuzzy logic. In a given case of fitness function, the weightages for these three attributes are determined through fuzzy reasoning. Fuzzy logic models have been developed for the above said three attributes as its inputs and the

† **The major findings of this chapter have been published in International Journal of Computer Systems Science and Engineering, vol. 2, issue 2, 93-98, 2007.**

outputs. The developed fuzzy genetic optimal control architectures have been implemented on 2, 3 and 4 degree-of-freedom robotic arm system. The results for optimised joint angles are presented and have been duly discussed. A new paradigm of fuzzy-GA control architecture has been contemplated. The methods proposed are robust. It allows optimal robotic arm movement and adaptation to the dynamic conditions in the environment.

5.2 Role of Fuzzy Logic

The fitness function in genetic algorithms as implemented in this case is augmented by three attributes, *viz.,* joint movement, friction and settling time. At any time, the values of these three attributes are found with the help of fuzzy logic. In a given case of fitness function the weightages for these three attributes are determined through fuzzy reasoning. Fuzzy logic models have been developed for the above said three attributes as its input and the weightages as required for these three attributes in the fitness function as three outputs. The developed fuzzy genetic optimal control architectures have been implemented on 2, 3 and 4 degree-of-freedom robotic systems.

5.3 Role of Genetic Algorithms

Intelligent methods can also be used in optimisation of movement and trajectory planning of robotic arm manipulators. These methods can be used for solving redundancy resolution problems. Genetic algorithms are viewed as function optimisers. The range of problems to which genetic algorithms can be applied is quite broad. GAs are tools on probabilistic and causality, not necessarily they will have the same type of evolution when applied to the same problem.

Inverse kinematics solutions obtained are considered as search space. The solution is evolved from the search space. By selecting the inverse kinematic solutions and evaluating the value of fitness function is obtained. Selection of fitness function value is based on rank selection. Lowest fitness value solutions are considered for next run. The values of all angles are considered as positive because the negative sign is only due to the right hand side or left hand side movement of the links.

5.4 Role of Analytical Hierarchy Process

The analytical hierarchy process (AHP) is to structure complexity in gradual steps from the large to the small, or the general to the particular, so that one could relate them with greater accuracy according to our understanding. Because experience is too vast to lay it out in a

single network structure, piecemeal decompositions is suitable along with occasional linkages.

The purpose is to improve our awareness by richer synthesis of knowledge and intuition. AHP is a learning tool. It is not a means to discover the truth because truth is relative and changing.

In the AHP, next is to setting up a structure to represent a problem, the reciprocal property is the most fundamental aspect for creating a scale. A hierarchy is an efficient way to organise complex system for controlling and passing information down the system. Unstructured problems are best grappled with in the systematic framework of the hierarchy or a feedback network.

The fitness function in genetic algorithms as implemented in this case is augmented by three attributes, *viz.,* joint movement, friction and settling time. At any time the values of these three attributes is determined with the help of AHP. The fitness function is used in genetic algorithms and the weightages for these three attributes are determined through AHP.

5.5 Implementation of Analytical Hierarchical Process

A hierarchy is an efficient way to organise complex system and functionally for controlling and passing information down the system. Unstructured problems are best grappled with in the systematic framework of the hierarchy or a feedback network. In this work, fitness of each chromosome depends upon many factors. We have considered three factors on which the fitness function has been calculated by applying analytical hierarchical process. These three main factors are: Movement (A_1), Friction (A_2), Settling time (minimum vibration) (A_3).

We have decided the importance and value of these three attributes for the each angle separately for different degree-of-freedom robotic arm manipulators.

5.5.1 Case I: 2 DOF Manipulator

Firstly, we decide the importance and value of these three attributes for the each angle separately.

Table 5.1 is for angle θ_1 moved by link '1' and Table 5.2 is for angle θ_2 moved by link '2'.

Table 5.1: Importance and value of the three attributes for θ_1

Attribute	Importance	Value (total 1)
A_1	High	0.5
A_2	Medium	0.3
A_3	Low	0.2

Table 5.2: Importance and value of the three attributes for θ_2

Attribute	Importance	Value (total 1)
A_1	Medium	0.3
A_2	Medium	0.3
A_3	High	0.4

The general form of comparison matrix is given in Table 5.3.

Table 5.3: General form of comparison matrix

	A_1	A_2	A_3
A_1	A_1/A_1	A_1/A_2	A_1/A_3
A_2	A_2/A_1	A_2/A_2	A_2/A_3
A_3	A_3/A_1	A_3/A_2	A_3/A_3

For angle θ_1**,** which is moved by link '1', the comparison matrix is given in Table 5.4.

For angle θ_2**,** moved by link '2', the comparison matrix is given by Table 5.5.

Table 5.4: Comparison matrix for angle θ_1

	A_1 (0.5)	A_2 (0.3)	A_3 (0.2)
A_1 (0.5)	1	0.6	1.5
A_2 (0.3)	1.7	1	2.5
A_3 (0.2)	0.7	0.4	1

Table 5.5: Comparison matrix for angle θ_2

	A_1 (0.3)	A_2 (0.3)	A_3 (0.4)
A_1 (0.3)	1	1	0.75
A_2 (0.3)	1	1	0.75
A_3 (0.4)	1.3	1.3	1

Using Table 5.4, the eigen vector for θ_1 is given by

$$[A - \lambda I] = 0 \qquad \text{----(5.1)}$$

and the eigen values for θ_1, can be found as

$$\begin{vmatrix} 1-\lambda & 0.6 & 1.5 \\ 1.7 & 1-\lambda & 2.5 \\ 0.7 & 0.4 & 1-\lambda \end{vmatrix} = 0$$

$$\Rightarrow \quad [1-\lambda][1-\lambda^2-2\lambda-1]-0.6[1.7-1.7\lambda-1.75]+1.5[0.68-0.7+0.7\lambda]=0 \qquad \text{----(5.2)}$$

$$\Rightarrow \quad [\lambda^3-3\lambda^2-0.02\lambda]=0 \qquad \text{----(5.3)}$$

Solving the above Eq. (5.3), we get (λs) for θ_1

$$\lambda = 2.99, 0.01, 0 \qquad \text{----(5.4)}$$

Using Table 5.5, the eigen vector for θ_2 is given by

$$[A - \lambda I] = 0$$

Therefore, eigen values for θ_2 can be calculated as

$$\begin{vmatrix} 1-\lambda & 1 & 0.75 \\ 1 & 1-\lambda & 0.75 \\ 1.3 & 1.3 & 1-\lambda \end{vmatrix} = 0$$

$$\Rightarrow [1-\lambda][1-\lambda^2-2\lambda-1.3\times 0.75]-[1-\lambda-1.3\times 0.75]+0.75[1.3-1.3(1-\lambda)]=0 \quad \text{----(5.5)}$$

$$\Rightarrow \quad [\lambda^3 - 3\lambda^2 + 0.04\lambda] = 0 \quad \text{----(5.6)}$$

Solving the above Eq. (5.6), we get (λs) for θ_2

$$\lambda = 2.98, 0.02, 0 \quad \text{----(5.7)}$$

Obtaining eigen values from these comparison matrices and then taking their maximum values as λ for θ_1 (say λ_1) = 2.99, λ for θ_2 (say λ_2) = 2.98 for the purpose of finding the value of fitness function, used for ranking chromosomes (the set of joint angles) for execution of genetic algorithms, the corresponding fitness function (f_c) is given by

$$f_c = (\theta_1 \times \lambda_1 + \theta_2 \times \lambda_2) \quad \text{----(5.8)}$$

5.5.2 Case II: 3 DOF Manipulator

The importance and values of these three attributes for the each angle separately are obtained for 3 DOF manipulator. Table 5.6 is for angle θ_1 which is the angle moved by link '1' and Table 5.7 is for angle θ_2, which is the angle moved by link '2'.

Table 5.6: Importance and value of the three attributes for θ_1

Attribute	Importance	Value (total 1)
A_1	High	0.5
A_2	Medium	0.3
A_3	Low	0.2

Table 5.7: Importance and value of the three attributes for θ_2

Attribute	Importance	Value (total 1)
A_1	Medium	0.3
A_2	Medium	0.3
A_3	High	0.4

Table 5.8 is for angle θ_3**,** which is the angle, moved by link '3' and the general form of comparison matrix is given in Table 5.9.

Table 5.8: Importance and value of the three attributes for θ_3

Attribute	Importance	Value (total 1)
A_1	High	0.5
A_2	Medium	0.2
A_3	Low	0.3

Table 5.9: General form of comparison matrix

	A_1	A_2	A_3
A_1	A_1/A_1	A_1/A_2	A_1/A_3
A_2	A_2/A_1	A_2/A_2	A_2/A_3
A_3	A_3/A_1	A_3/A_2	A_3/A_3

For angle θ_1**,** which is moved by link '1', the comparison matrix is given in Table 5.10.

For angle θ_2**,** moved by link '2', the comparison matrix is given by Table 5.11.

Table 5.10: Comparison matrix for angle θ_1

	A_1 (0.5)	A_2 (0.3)	A_3 (0.2)
A_1 (0.5)	1	0.6	1.5
A_2 (0.3)	1.7	1	2.5
A_3 (0.2)	0.7	0.4	1

Table 5.11: Comparison matrix for angle θ_2

	A_1 (0.3)	A_2 (0.3)	A_3 (0.4)
A_1 (0.3)	1	1	0.75
A_2 (0.3)	1	1	0.75
A_3 (0.4)	1.3	1.3	1

For angle θ_3, moved by link-3, the comparison matrix is given in Table 5.12.

Table 5.12: Comparison matrix for angle θ_3

	A_1 (0.5)	A_2 (0.2)	A_3 (0.3)
A_1 (0.5)	1	2.5	1.7
A_2 (0.2)	0.4	1	0.7
A_3 (0.3)	0.6	1.5	1

Using Table 5.10, the eigen vector for angle θ_1 is given by

$$[A - \lambda I] = 0$$

The eigen values for angle θ_1 is determined as

$$\Rightarrow \quad \begin{vmatrix} 1-\lambda & 0.6 & 1.5 \\ 1.7 & 1-\lambda & 2.5 \\ 0.7 & 0.4 & 1-\lambda \end{vmatrix} = 0$$

$$\Rightarrow \quad [1-\lambda][1-\lambda^2-2\lambda-1]-0.6[1.7-1.7\lambda-1.75]+1.5[0.68-0.7+0.7\lambda]=0 \qquad \text{----(5.9)}$$

$$\Rightarrow \quad [\lambda^3-3\lambda^2-0.02\lambda]=0 \qquad \text{----(5.10)}$$

Solving the above Eq. (5.10), we get (λs) for θ_1

$$\lambda = 2.99, 0.01, 0 \qquad \text{----(5.11)}$$

Using Table 5.11, the eigen vector for angle θ_2 is given by

$$[A - \lambda I] = 0$$

The eigen values for angle θ_2 can be obtained as

$$\begin{vmatrix} 1-\lambda & 1 & 0.75 \\ 1 & 1-\lambda & 0.75 \\ 1.3 & 1.3 & 1-\lambda \end{vmatrix} = 0$$

$$\Rightarrow [1-\lambda]\left[1-\lambda^2-2\lambda-1.3\times 0.75\right]-\left[1-\lambda-1.3\times 0.75\right]+0.75\left[1.3-1.3(1-\lambda)\right]=0 \quad \text{----(5.12)}$$

$$\Rightarrow \left[\lambda^3-3\lambda^2+0.04\lambda\right]=0 \quad \text{----(5.13)}$$

Solving the above Eq. (5.13), we get (λs) for θ_2

$$\lambda = 2.98, 0.02, 0 \quad \text{----(5.14)}$$

Using Table 5.12, the eigen vector for θ_3 is given by

$$[A-\lambda I]=0$$

The eigen values for θ_3 can be calculates as

$$\begin{vmatrix} 1-\lambda & 2.5 & 1.7 \\ 0.4 & 1-\lambda & 0.7 \\ 0.6 & 1.5 & 1-\lambda \end{vmatrix}$$

$$\Rightarrow \quad [1-\lambda]\left[1-\lambda^2-2\lambda-1.05\right]-2.5\left[0.4-0.4\lambda-0.42\right]+1.7\left[0.6-0.6(1-\lambda)\right]=0 \quad \text{----(5.15)}$$

$$\Rightarrow \quad \left[\lambda^3-3\lambda^2-0.03\lambda\right]=0 \quad \text{----(5.16)}$$

Solving the above Eq. (5.16), we get (λs) for θ_3

$$\lambda = 2.99, 0.01, 0 \quad \text{----(5.17)}$$

Taking the higher values of all the eigen values from Eq. (5.11), Eq. (5.14) and Eq. (5.17), we have λ for θ_1 (say λ_1) = 2.99, λ for θ_2 (say λ_2) = 2.98 and λ for θ_3 (say λ_3) = 2.99. Obtaining eigen values from these comparison matrices and then taking their maximum values as $\lambda_1 = 2.99$, $\lambda_2 = 2.98$ and $\lambda_3 = 2.99$ for the purpose of finding the value of fitness function used for ranking chromosomes (the set of joint angles) for execution of genetic algorithms, the corresponding fitness function (f_c) is given by

$$f_c = (\theta_1 \times \lambda_1 + \theta_2 \times \lambda_2 + \theta_3 \times \lambda_3) \quad \text{----(5.18)}$$

5.5.3 Case III: 4 DOF Manipulator

The importance and value of these three attributes for the each angle has been decided separately. Table 5.13 is for angle θ, which is the angle moved by joint '1' and Table 5.14 is for angle ϕ, which is angle moved by joint '2'. Table 5.15 is for angle ψ, which is the angle moved by joint '4', where as Table 5.16 gives a general form of comparison matrix.

Table 5.13: Importance and value of the three attribute for θ

Attribute	Importance	Value (total 1)
A_1	High	0.5
A_2	Medium	0.3
A_3	Low	0.2

Table 5.14: Importance and value of the three attribute for ϕ

Attribute	Importance	Value (total 1)
A_1	Medium	0.3
A_2	Medium	0.3
A_3	High	0.4

Table 5.15: Importance and value of the three attributes for ψ

Attribute	Importance	Value (total 1)
A_1	High	0.5
A_2	Medium	0.2
A_3	Low	0.3

Table 5.16: General form of comparison matrix

	A_1	A_2	A_3
A_1	A_1/A_1	A_1/A_2	A_1/A_3
A_2	A_2/A_1	A_2/A_2	A_2/A_3
A_3	A_3/A_1	A_3/A_2	A_3/A_3

For angle θ, which is the angle moved by joint '1', the comparison matrix is given in Table 5.17. For angle ϕ, which is the angle moved by joint '2', the comparison matrix is given by Table 5.18.

Table 5.17 Comparison matrix for angle θ

	A_1 (0.5)	A_2 (0.3)	A_3 (0.2)
A_1 (0.5)	1	0.6	1.5
A_2 (0.3)	1.7	1	2.5
A_3 (0.2)	0.7	0.4	1

Table 5.18 Comparison matrix for angle ϕ

	A_1 (0.3)	A_2 (0.3)	A_3 (0.4)
A_1 (0.3)	1	1	0.75
A_2 (0.3)	1	1	0.75
A_3 (0.4)	1.3	1.3	1

For angle ψ, which is the angle, moved by Joint '4' the comparison matrix is given by Table 5.18.

Table 5.19 Comparison matrix for angle ψ

	A_1 (0.5)	A_2 (0.2)	A_3 (0.3)
A_1 (0.5)	1	2.5	1.7
A_2 (0.2)	0.4	1	0.7
A_3 (0.3)	0.6	1.5	1

Using Table 5.17,the eigen vector for angle θ is given by

$$[A - \lambda I] = 0$$

The eigen values for angle θ can be obtained as

$$\begin{vmatrix} 1-\lambda & 0.6 & 1.5 \\ 1.7 & 1-\lambda & 2.5 \\ 0.7 & 0.4 & 1-\lambda \end{vmatrix} = 0$$

$$\Rightarrow \quad [1-\lambda]\left[1-\lambda^2-2\lambda-1\right]-0.6[1.7-1.7\lambda-1.75]+1.5[0.68-0.7+0.7\lambda]=0 \qquad \text{----(5.19)}$$

$$\Rightarrow \quad \left[\lambda^3-3\lambda^2-0.02\lambda\right]=0 \qquad \text{----(5.20)}$$

Solving the above Eq. (5.20), we get (λs) for angle θ

$$\lambda = 2.99, 0.01, 0 \qquad \text{----(5.21)}$$

Using Table 5.18, eigen vector for angle ϕ is given by

$$[A-\lambda I]=0$$

The eigen values for angle ϕ can be obtained as

$$\begin{vmatrix} 1-\lambda & 1 & 0.75 \\ 1 & 1-\lambda & 0.75 \\ 1.3 & 1.3 & 1-\lambda \end{vmatrix} = 0$$

$$\Rightarrow \quad [1-\lambda]\left[1-\lambda^2-2\lambda-1.3\times0.75\right]-\left[1-\lambda-1.3\times0.75\right]+0.75[1.3-1.3(1-\lambda)]=0 \qquad \text{----(5.22)}$$

$$\Rightarrow \quad \left[\lambda^3-3\lambda^2+0.04\lambda\right]=0 \qquad \text{----(5.23)}$$

Solving the above Eq. (5.23), we get (λs) for angle ϕ

$$\lambda = 2.98, 0.02, 0 \qquad \text{----(5.24)}$$

Using Table 5.19, eigen vector for angle ψ is given by

$$[A-\lambda I]=0$$

The eigen values for angle ψ can be calculated as

$$\begin{vmatrix} 1-\lambda & 2.5 & 1.7 \\ 0.4 & 1-\lambda & 0.7 \\ 0.6 & 1.5 & 1-\lambda \end{vmatrix}$$

$$\Rightarrow \quad [1-\lambda]\left[1-\lambda^2-2\lambda-1.05\right]-2.5[0.4-0.4\lambda-0.42]+1.7[0.6-0.6(1-\lambda)]=0 \qquad \text{----(5.25)}$$

$$\Rightarrow \quad \left[\lambda^3-3\lambda^2+0.03\lambda\right]=0 \qquad \text{----(5.26)}$$

Solving the above Eq. (5.26), we get (λs) for angle ψ

$$\lambda = 2.99, 0.01, 0 \qquad \text{----(5.27)}$$

Obtaining eigen values from these comparison matrices and then taking the maximum values of the eigen values, we have λ for θ (say λ_1) = 2.99, λ for ϕ (say λ_2) = 2.98 and λ for ψ

(say λ_3) = 2.99 for purpose of finding fitness used for ranking chromosomes for execution of genetic algorithms, the corresponding fitness function (f_c) is given by

$$f_c = (\theta \times \lambda_1 + \phi \times \lambda_2 + \psi \times \lambda_3) \quad \text{----(5.28)}$$

5.6 Implementation of Genetic Algorithms

To implement genetic algorithms, firstly solutions obtained from inverse kinematics are considered as search space and then chromosomes are generated to create initial population. Then, these results are fed to the genetic algorithms for generating the population of chromosomes. The values of all angles are considered as positive because the negative sign is only due to the right hand side or left hand side movement of the links.

The following steps are followed to optimise the value of joint movement of robotic arm.

1. [Start]: Generate random population of *n* chromosomes (suitable solutions for the problem).
2. [Fitness]: Evaluate the fitness $f_c(x)$ of each chromosome *x* in the population.
3. [New population]: Create a new population by repeating following steps until the new population is complete.
    a. Encoding: The encoding mainly depends on the solved problem. Various encodings are used depending upon the problem like value, binary, integer or real number, *etc*.
    b. Selection: Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chances to be selected).
    c. Crossover: With a crossover probability, crossover the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.
    d. Mutation: With a mutation probability, mutate new offspring at each locus (position in chromosome).
    e. Accepting: Place new offspring in the new population.
4. [Replace]: Use new generated population for a further run of the algorithm.
5. [Test]: If the end condition is satisfied, stop and return the best solution in current population.
6. [Loop]: Go to step 2 [Fitness].

The flow chart using genetic algorithms (Fig. 5.1) is given as

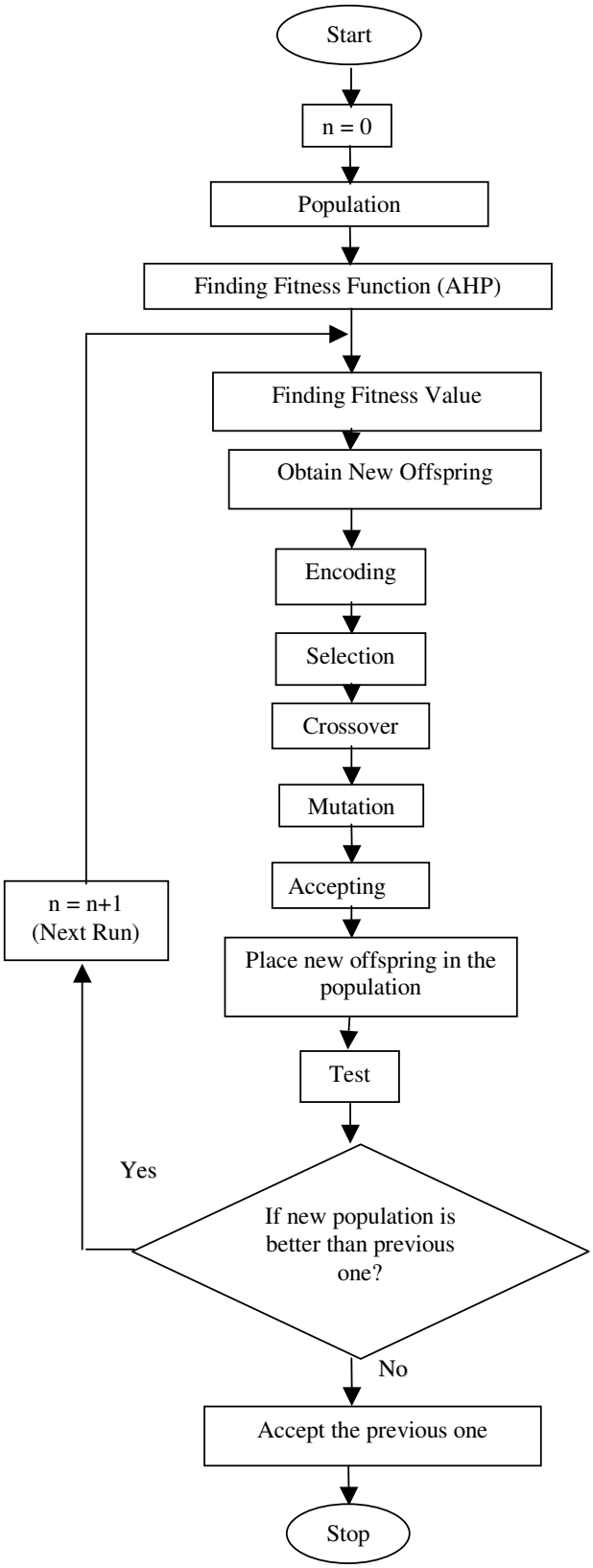


Fig. 5.1: Flowchart using genetic algorithms

5.6.1 Case I: 2 DOF Manipulator

We find the fitness of each chromosome with the help of fitness function as defined by Eq. (5.8). In case of 2 DOF, solutions obtained using inverse kinematics are mentioned below (refer to 4.2.3 Case study: 2 DOF manipulator).

The following values for the angles of the links from Eqs. (4.7), (4.8) and (4.9) are obtained.

$\theta_1 = 30.76^\circ,\ \theta_2 = 47.81^\circ$

$\theta_1 = 73.94^\circ,\ \theta_2 = -47.81^\circ$

First run

Step-1:

[Start]: The angles obtained above with inverse kinematics can be combined so as to form different solutions as population of chromosomes shown in Table 5.20.

Step-2:

[Fitness]: The fitness value is calculated by fitness function defined by Eq. (5.8) *i.e.,* $f_c = (\theta_1 \times \lambda_1 + \theta_2 \times \lambda_2)$, where λ_1=2.99 and λ_2=2.98.

Table 5.20: Fitness value of chromosomes obtained after applying inverse kinematics

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
1.	{30.76, 47.81}	91.97 + 142.47	234.44
2.	{30.76, 47.81}	91.97 + 142.47	234.44
3.	{73.94, 47.81}	221.08 + 142.47	363.55
4.	{73.94, 47.8}	221.08 + 142.47	363.55

It is concluded from the above table that more is the angle, more is the movement of the links and therefore more will be the fitness value of the chromosome. Hence, the ranking of the chromosome will be based on the criteria that less is the fitness value, higher will be the rank.

In case of 2 DOF manipulator, ranking of chromosomes are same as in initial Table 5.20, so there are no change in ranking as per fitness value. There is no other alternative solution in this case. The optimal solution is given as

$$\theta_1 = 30.76^\circ,\ \theta_2 = 47.81^\circ$$

$$\text{and } \theta_1 = 73.94^\circ,\ \theta_2 = -47.81^\circ \qquad \text{----(5.29)}$$

5.6.2 Case II: 3 DOF Manipulator

In case of 3 DOF, solutions obtained using inverse kinematics are mentioned below (refer to 4.3.3 Case study: 3 DOF manipulator).

The following values for the joint angles of the links for 3 DOF manipulator are obtained from Eqs. (4.41), (4.44), (4.47), (4.48) and (4.49).

$\theta_1 = 137.91^\circ$, $\theta_2 = -217.49^\circ$, $\theta_3 = 106.14^\circ$

and $\theta_1 = -84.78^\circ$, $\theta_2 = 37.49^\circ$, $\theta_3 = 73.85^\circ$

First run

Step-1:

[Start]: The angles obtained above with inverse kinematics can be combined so as to form different solutions as shown in Table 5.21.

Step-2:

[Fitness]: The fitness value is obtained by fitness function defined by Eq. (5.18) *i.e.,* $f_c = (\theta_1 \times \lambda_1 + \theta_2 \times \lambda_2 + \theta_3 \times \lambda_3)$, here, $\lambda_1 = 2.99$, $\lambda_2 = 2.98$ and $\lambda_3 = 2.99$. Fitness of each chromosome is found with the help of above fitness function.

Table 5.21: Fitness value of chromosomes obtained after applying inverse kinematics

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
1.	{137.91, 217.49, 106.14}	412.35 + 648.12 + 317.37	1377.84
2.	{137.91, 217.49, 73.85}	412.35 + 648.12 + 220.81	1281.28
3.	{137.91, 37.49, 106.14}	412.35 + 111.72 + 317.37	841.44
4.	{137.91, 37.49, 73.85}	412.35 + 111.72 + 220.81	744.88
5.	{84.78, 37.49, 106.14}	253.50 + 111.72 + 317.37	682.59
6.	{84.78, 37.49, 73.85}	253.50 + 111.72 + 220.81	586.03
7.	{84.78, 217.49, 106.14}	253.50 + 648.12 + 317.37	1218.99
8.	{84.78, 217.49, 73.85}	253.50 + 648.12 + 220.81	1122.43

It is concluded from the above table that more is the angle, more is the movement of the links and therefore more will be the fitness value of the chromosome. Hence, the ranking of the chromosome will be based on the criterion that less is the fitness value, higher will be the rank.

Hence, the chromosomes according to their ranking (in descending order) are given as.

Table 5.22: Ranking of chromosomes obtained after applying inverse kinematics

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 6 {84.78, 37.49, 73.85}	586.03
2	Chromosome 5 {84.78, 37.49, 106.14}	682.59
3	Chromosome 4 {137.91, 37.49, 73.85}	744.88
4	Chromosome 3 {137.91, 37.49, 106.14}	841.44
5	Chromosome 8 {84.78, 217.49, 73.85}	1122.43
6	Chromosome 7 {84.78, 217.49, 106.14}	1218.99
7	Chromosome 2 {137.91, 217.49, 73.85}	1281.28
8	Chromosome 1 {137.91, 217.49, 106.14}	1377.84

Step-3:

[New population]:

a. Encoding: As we have chromosomes having values too large, hence we will take value encoding. However, for this encoding, it is often necessary to develop some new crossover and mutation specific to the problem.

b. Selection: As the higher rank chromosomes have lower fitness values, so rank selection is done. According to rank selection, two chromosomes having highest ranks are chosen. So in this case, we have chosen first two chromosomes with the following specifications:

Ranking	Chromosome	Fitness value
1	Chromosome 6 {84.78, 37.49, 73.85}	586.03
2	Chromosome 5 {84.78, 37.49, 106.14}	682.59

c. Crossover: While operating crossover on the first two chromosomes, crossover points in the angle of third link (*i.e.*, in θ_3) are taken as shown below.

{84.78, 37.49, 7|3. 8|5}

{84.78, 37.49, 10|6. 1|4}

(bar shows crossover points)

After crossover, we get the following new offsprings:

Chromosome 9 {84.78, 37.49, 103.84}

Chromosome10 {84.78, 37.49, 76.15}

d. Mutation: We have value encoding for our purpose and hence mutation is ignored.

e. Accepting: New offsprings are placed in the new population.

Step-4:

[Replace]: As a result of crossover, the angle of link '3' has been increased or decreased by some value and hence an error appears in obtaining final destination point. So, new angles are calculated for link '1' and link '2'. Now, using these newly obtained chromosomes other angles are calculated.

Here, $\theta_3 = 103.84^\circ$ and 76.15°. The values for angle θ_1 and θ_2 are being obtained.

Values of other angles based on inverse kinematics using computer program are obtained as below.

$\theta_1 = -20.84^\circ,\ \theta_2 = -36.95^\circ,\ \ \theta_3 = 103.84^\circ$

and $\theta_1 = -40.33^\circ, \theta_2 = -28.75^\circ, \theta_3 = 76.15^\circ$

New population and fitness value of chromosomes obtained after first run are shown in Table 5.23.

Table 5.23: Fitness value of chromosomes obtained from first run

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
11.	{20.84, 36.95, 103.84}	62.31 + 110.11 + 310.48	482.90
12.	{20.84, 36.95, 76.15}	62.31 + 110.11 + 227.69	400.11
13.	{20.84, 28.75, 103.84}	62.31 + 85.68 + 310.48	458.47
14.	{20.84, 28.75, 76.15}	62.31 + 85.68 + 227.69	375.68
15.	{40.33, 36.95, 103.84}	120.59 + 110.11 + 310.48	541.18
16.	{40.33, 36.95, 76.15}	120.59 + 110.11 + 227.69	458.39
17.	{40.33, 28.75, 103.84}	120.59 + 85.68 + 310.48	516.75
18.	{40.33, 28.75, 76.15}	120.59 + 85.68 + 227.69	433.96

Step-5:

[Test]:

As it is clear from the above obtained new population, that the chromosomes in this population have much better fitness value than the previous one. So, this whole population is selected for further run.

Step-6:

[Loop]: Go to step 2 [Fitness].

Second run

Step-2:

[Fitness]: We have already found the fitness values in the previous run. The new population can be arranged according to their ranking (in descending order) as given in Table 5.24.

Table 5.24: Ranking of chromosomes obtained after first run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome14 {20.84, 28.75, 76.15}	375.68
2	Chromosome12 {20.84, 36.95, 76.15}	400.11
3	Chromosome18 {40.33, 28.75, 76.15}	433.96
4	Chromosome16 {40.33, 36.95, 76.15}	458.39
5	Chromosome13 {20.84, 28.75, 103.84}	458.47
6	Chromosome11 {20.84, 36.95, 103.84}	482.90
7	Chromosome17 {40.33, 28.75, 103.84}	516.75
8	Chromosome15 {40.33, 36.95, 103.84}	541.18

Step-3:

[New population]:

a. Encoding: Chromosomes are already in value encoding form.

b. Selection: According to rank selection, we first choose two chromosomes having highest ranks. So, in our case, we can choose first two chromosomes with the following specifications:

Ranking	Chromosome	Fitness value
1	Chromosome14 {-20.84, -28.75, 76.15}	375.68
2	Chromosome12 {-20.84, -36.95, 76.15}	400.11

c. Crossover: While operating crossover on the first two chromosomes, we can only make crossover points in the angle of second link (*i.e.*, in θ_2) as shown below.

{-20.84, -2|8.7|5, 76.15}

{-20.84, -3|6.9|5, 76.15}

(bar shows crossover points)

After crossover, we get the following new offsprings:

Chromosome 19 {-20.84**,** -26.95, 76.15}

Chromosome 20 {-20.84, -38.75, 76.15}

d. Mutation: We have taken value encoding and hence mutation can be ignored.

e. Accepting: In the new population, new offsprings are placed.

Step-4:

[Replace]: As a result of crossover, the angle of link '2' has been increased or decreased by some value and hence an error appears in obtaining final destination point. So, new angles

are calculated for link '1' and link '3'. Now, we use these newly obtained chromosomes for obtaining other angles.

Here, $\theta_2 = -26.95^\circ$ and -38.75°. The values for θ_1 and θ_3 are being obtained.

Values of other angles based on inverse kinematics are obtained through computer program as given below.

$\theta_1 = 137.911^\circ$, $\theta_2 = -26.95^\circ$, $\theta_3 = -84.4^\circ$

and $\theta_1 = -84.781^\circ$, $\theta_2 = -38.75^\circ$, $\theta_3 = 150.10^\circ$

New population and fitness values of chromosomes obtained after second run as shown in Table 5.25.

Table 5.25: Fitness value of chromosomes obtained after second run

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
21.	{137.91, 26.95, 84.40}	409.63 + 80.31 + 252.36	742.30
22.	{137.91, 26.95, 150.10}	409.63 + 80.31 + 448.80	938.74
23.	{137.91, 38.75, 84.40}	409.63 + 115.48 + 252.36	777.47
24.	{137.91, 38.75, 150.10}	409.63 + 115.48 + 448.80	973.91
25.	{84.78, 26.95, 84.40}	253.49 + 80.31 + 252.36	586.16
26.	{84.78, 26.95, 150.10}	253.49 + 80.31 + 448.80	782.60
27.	{84.78, 38.75, 84.40}	253.49 + 115.48 + 252.36	621.31
28.	{84.78, 38.75, 150.10}	253.49 + 115.48 + 448.80	817.77

Step-5:

[Test]: As it is clear from the above newly obtained population, that the chromosomes in this population have not good fitness values than the previous ones. So, we select the previous whole population obtained in the first run as final population.

Step-6:

[Loop]: Go to step 2 [Fitness]

Third run

Step-2:

[Fitness]: We have already found the fitness values in the previously in first run. The ranking of chromosomes obtained after first run (in descending order) as shown in Table 5.24.

Step-3:

[New population]:

a. Encoding: Now as the magnitudes of the angles are less in the second generation, binary encoding is performed (refer to Table 5.24). The population in decimal form is given as:

Chromosome 14 {20.84, 28.75, 76.15}
Chromosome 12 {20.84, 36.95, 76.15}
Chromosome 18 {40.33, 28.75, 76.15}
Chromosome 16 {40.33, 36.95, 76.15}
Chromosome 13 {20.84, 28.75, 103.84}
Chromosome 11 {20.84, 36.95, 103.84}
Chromosome 17 {40.33, 28.75, 103.84}
Chromosome 15 {40.33, 36.95, 103.84}

After binary encoding, we have

Chromosome 14 {10100.1010100, 11100.1001011, 1001100.1111}
Chromosome 12 {10100.1010100, 100100.1011111, 1001100.1111}
Chromosome 18 {101000.100001, 11100.1001011, 1001100.1111}
Chromosome 16 {101000.100001, 100100.1011111, 1001100.1111}
Chromosome 13 {10100.1010100, 11100.1001011, 1100111.1010100}
Chromosome 11 {10100.1010100, 100100.1011111, 1100111.1010100}
Chromosome 17 {101000.100001, 11100.1001011, 1100111.1010100}
Chromosome 15 {101000.100001, 100100.1011111, 1100111.1010100}

b. Selection: According to rank selection we had already choose two chromosomes having highest ranks in second run. Now we will choose next two chromosomes (chromosomes 12 and 18) in the increasing order of fitness.

Next, two chromosomes (chromosomes 12 and 18) are chosen in the increasing order of fitness.

Ranking	Chromosome	Fitness value
2	{10100.1010100, 100100.1011111, 1001100.1111}	400.11
3	{101000.100001, 11100.1001011, 1001100.1111}	433.96

c. Crossover: While crossover on the next two chromosomes, we can make crossover points in the angle of first and second link (*i.e.*, in θ_1 and θ_2) as shown below.

Chromosome 12 (101 00 | .1010100, 100 10 | 0.1011111, 1001100.1111}
Chromosome 18 (101 00 | 0.100001, 111 00 | .1001011, 1001100.1111}
(bar shows crossover points)

After crossover, the following new offsprings are obtained:

Chromosome 29 {101000.100001, 10000.1001011, 1001100.1111}

Chromosome 30 {10100.1010100, 111100. 1011111, 1001100.1111}

The above chromosomes are written as in value encoded form.

Chromosome 29 {-40.33, -16.75, 76.15}

Chromosome 30 {-20.84, -60. 95, 76.15}

From the above obtained chromosomes, it is judged that there is no such good change in the angles of link '1', but the angles of link '2' have changed largely. So, we will consider angles of link '2' for further run.

d. Mutation: With a mutation probability, new offspring are mutated at each locus. When mutation with binary encoding is done, selected bits are inverted. Arrow indicates mutated bits.

Before Mutation:

Chromosome 29 {101000.100001, 010000.1001011, 1001100.1111}

Chromosome 30 {10100.1010100, 111100.1011111, 1001100.1111}

After Mutation:

Chromosome 31 {101000.100001, 011000.1001011, 1001100.1111}

Chromosome 32 {10100.1010100, 110100.1011111, 1001100.1111}

e. Accepting: New offsrings are placed in the new population

Step-4:

[Replace]:

After the crossover, the angle of link '2' has been modified by some value and hence an error appears in obtaining final destination point. New angles are calculated for link '1' and link '3'. These newly obtained chromosomes are used for finding other angles.

Here, $\theta_2 = -16.75^\circ$ and -60.95°. The value for angles θ_1 and θ_3 are being calculated.

Values of other angles based on inverse kinematics are obtained using computer programe as below.

$\theta_1 = 137.91^\circ$, $\theta_2 = -16.75^\circ$ $\theta_3 = -84.40^\circ$

and $\theta_1 = -84.78^\circ$, $\theta_2 = -60.95^\circ$, $\theta_3 = 172.30^\circ$

New population and fitness values of chromosomes obtained after third run as given in Table 5.26.

Table 5.26: Fitness value of chromosomes obtained after third run

Chromosome No.	Chromosome (angles in degree)	Fitness value
33.	{137.91, 16.75, 84.40}	713.43
34.	{137.91, 16.75, 172.30}	977.44
35.	{137.91, 60.95, 84.40}	845.14
36.	{137.91, 60.95, 172.30}	1109.16
37.	{84.78, 16.75, 84.40}	555.57
38.	{84.78, 16.75, 172.30}	818.58
39.	{84.78, 60.95, 84.40}	686.28
40.	{84.78, 60.95, 172.30}	950.30

Step-5:

[Test]:

As it is clear from the above newly population, that the chromosomes in this population have not good fitness values than created in the previous run. So, we select the population obtained in the second run as final population.

In case of 3 DOF manipulator, lowest fitness having higher ranking is in the first run of the chromosomes. There is no alternative in this case. The optimal solutions are obtained after first run as:

$\theta_1 = -20.84^\circ,\ \theta_2 = -36.95^\circ,\ \ \theta_3 = 103.84^\circ$

and $\theta_1 = -40.33^\circ, \theta_2 = -28.75^\circ, \theta_3 = 76.15^\circ$ ----(5.30)

5.6.3 Case III: 4 DOF Manipulator

The following values for the angles of the links using inverse kinematics are obtained (refer to 4.4.3 Case study: 4 DOF manipulator). The following values for the joint angles of the links for 4 DOF manipulator are obtained from Eqs. (4.83), (4.90), (4.92), (4.99) and (4.101).

$\theta = 17.36^\circ,\ \phi = 30.09^\circ,\ \psi = 84.69^\circ$

and $\theta = 17.36^\circ,\ \phi = 26.26^\circ,\ \psi = 5.26^\circ$

First run

Step-1:

[Start]: The angles obtained above with inverse kinematics can be combined so as to form different solutions as shown in Table 5.27.

Step-2: The fitness value is obtained by fitness function defined by Eq. (5.28) *i.e.,* $f_c = (\theta \times \lambda_1 + \phi \times \lambda_2 + \psi \times \lambda_3)$, where, λ_1 = 2.99, λ_2 = 2.98 and λ_3 = 2.99. Fitness of each chromosome is found with the help of above fitness function.

Table 5.27: Fitness value of chromosomes obtained after applying inverse kinematics

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
1.	{17.36, 30.09, 84.69}	51.6363 + 89.9691+253.2231	394.83
2.	{17.36, 30.09, 5.26}	51.6363 + 89.9691 + 15.7274	157.35
3.	{17.36, 26.26, 84.69}	51.6363 + 78.5174 +253.2231	383.41
4.	{17.36, 26.26, 5.26}	51.6363 + 78.5174 + 15.7274	145.93

More is the movement of the links, more will be the fitness value of the chromosome. Hence, the ranking of the chromosome will be based on the criterion that less is the fitness value, higher will be the rank.

Hence, the chromosomes according to their ranking (in descending order) are shown in Fig. 5.28.

Table 5.28: Ranking of chromosomes obtained after applying inverse kinematics

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 4{17.36, 26.26, 5.26}	145.93
2	Chromosome 2{17.36, 30.09, 5.26}	157.35
3	Chromosome 3{17.36, 26.26, 84.69}	383.41
4	Chromosome 1{17.36, 30.09, 84.69}	394.83

Step-3:

[New population]:

a. Encoding: We have taken value encoding.

b. Selection: According to rank selection, two chromosomes having highest ranks are selected. So, in this case, we have selected first two chromosomes with the following specifications:

Ranking	Chromosome	Fitness value
1	Chromosome 4{17.36, 26.26, 5.26}	145.93
2	Chromosome 2{17.36, 30.09, 5.26}	157.35

c. Crossover: While crossover on the first two chromosomes, we can only make crossover points in the angle of second link (*i.e.,* in ϕ) as shown below.

{17.36, 2|6.2|6, 5.26}

{17.36, 3|0.0|9, 5.26}

(bar shows crossover points)

After crossover, we get the following new offsprings:

Chromosome 5 {17.36, 20.06, 5.26}

Chromosome 6 {17.36, 36.29, 5.26}

d. Mutation: We have taken value encoding. Hence, mutation is ignored.

e. Accepting: New offsprings are placed in the new population.

Step-4:

[Replace]: As a result of crossover, the angle for joint J_2 has been modified and hence an error appears in obtaining final destination point. So, new angles are calculated for joint J_1 and for joint J_2 using newly obtained chromosomes.

Hence, $\phi = 20.06^{\circ}$ and 36.29°. The values for θ and ψ are being obtained.

Values of other angle based on inverse kinematics are obtained using computer program as given below.

$\theta = 17.36^{\circ}$, $\phi = 20.06^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 36.29^{\circ}$, $\psi = 5.26^{\circ}$

New population and fitness value of chromosomes obtained after first run as shown in Table 5.29.

Table 5.29: Fitness value of chromosomes obtained after first run

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
7.	{17.36, 20.06, 84.69}	51.6363 + 59.7788 + 253.2231	364.92
8.	{17.36, 20.06, 5.26}	51.6363 + 59.7788 + 15.7274	127.45
9.	{17.36, 36.29, 84.69}	51.6363 + 108.1442 + 253.2231	413.29
10.	{17.36, 36.29, 5.26}	51.6363 + 108.1442 + 15.7274	175.81

Step-5:

[Test]: As it is clear from the above newly obtained population that the chromosomes in this population have much better fitness than the previous one. So, the previous whole population is selected for further run.

Step-6:

[Loop]: Go to step 2 [Fitness].

Second run

Step-2:

[Fitness]: We have already determined the fitness values in the previous run. The new population are arranged according to their ranking (in descending order) in Table 5.30.

Table 5.30: Ranking of chromosomes obtained after first run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 8{17.36, 20.06, 5.26}	127.45
2	Chromosome 10{17.36, 36.29, 5.26}	175.81
3	Chromosome 7{17.36, 20.06, 84.69}	364.92
4	Chromosome 9{17.36, 36.29, 5.26}	413.29

Step-3:

[New population]:

a. Encoding: Chromosomes are already in value encoding form.

b. Selection: According to rank selection, two chromosomes having highest ranks are selected. Two selected chromosomes are as mentioned below.

Ranking	Chromosome	Fitness value
1	Chromosome 8{17.36, 20.06, 5.26}	127.45
2	Chromosome 10{17.36, 36.29, 5.26}	175.81

c. Crossover: While operating crossover on the selected chromosomes, crossover points in the angle of second link (i.e. in ϕ) are considered as shown below.

{17.36, 2|0.0|6, 5.26}

{17.36, 3|6.2|9, 5.26}

(bar shows crossover points)

After crossover, we get the following new offsprings:

Chromosome 11 {17.36, 26.26, 5.26}

Chromosome 12 {17.36, 30.09, 5.26}

d. Mutation: We have taken value encoding for our problem. Hence mutation is ignored here.

e. Accepting: New offspring are placed in the new population.

Step-4:

[Replace]: As a result of crossover, the angle moved by joint J_2 has been increased or decreased by some value and hence an error appears in obtaining final destination point. New angles of joint J_1 and joint J_3 are calculated. Now, using newly obtained angles, values of other angles are calculated.

The following values for the angles ϕ of the links are obtained.

$\phi = 26.26^{\circ}$ and 30.09°

Values of other angles based on inverse kinematics have been obtained using computer program as given below.

$\theta = 17.36^{\circ}$, $\phi = 26.26^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 30.09^{\circ}$, $\psi = 5.26^{\circ}$

New population and fitness values of chromosomes obtained after second run as given in Table 5.31.

Table 5.31: Chromosomes obtained after second run

Chromosome No.	Chromosome (angles in degree)	Fitness value
13	{17.36, 26.26, 84.69}	383.40
14	{17.36, 26.26, 5.26}	145.93
15	{17.36, 30.09, 84.69}	394.82
16	{17.36, 30.09, 5.26}	157.34

Hence, the chromosomes according to their ranking (in descending order) as shown in Table 5.32.

Table 5.32: Ranking of chromosomes obtained after second run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 14 {17.36, 26.26, 5.26}	145.93
2	Chromosome 16{17.36, 30.09, 5.26}	157.34
3	Chromosome 13 {17.36, 26.26, 84.69}	383.40
4	Chromosome 15 {17.36, 30.09, 84.69}	394.82

Step-5:

[Test]: As it is clear from the above newly obtained population, that the chromosomes in this population have not good fitness values than the previous one. So the previous whole population obtained in the first run is selected as final population.

Step-6:

[Loop]: Go to step 2 [Fitness]

Third run

Step-2:

[Fitness]: We have already determined fitness in the previously in second run. The ranking of chromosomes obtained after first run (in descending order) as shown in Table 5.30.

Step-3:

[New population]:

a. Encoding: Now as the magnitudes of the angles are less in the second generation, binary encoding is performed (refer to Table 5.30). The population in decimal form is given as:

Chromosome 8 {17.36, 20.06, 5.26}

Chromosome 10 {17.36, 36.29, 5.26}

Chromosome 7 {17.36, 20.06, 84.69}

Chromosome 9 {17.36, 36.29, 5.26}

After binary encoding, we have

Chromosome 8 {10001.100100,10100.110, 101.11010}

Chromosome 10 {10001.100100, 100100.11101, 101.11010}

Chromosome 7 {10001.100100, 10100.110, 1010100.1000101}

Chromosome 9 {10001.100100, 100100.11101, 101.11010}

b. Selection: According to rank selection we had already choose two chromosomes having highest ranks in second run. Now we will choose next two chromosomes (chromosomes 10 and 7) in the increasing order of fitness.

Next, two chromosomes (chromosomes 10 and 7) are chosen in the increasing order of fitness.

Ranking	Chromosome	Fitness value
2	{10001.100100, 100100.11101, 101.11010}	175.81
3	{10001.100100, 10100.110, 1010100.1000101}	364.92

c. Crossover: While crossover on the next two chromosomes, we can make crossover points in the angle of first and second link (*i.e.*, in ϕ and ψ) as shown below.

Chromosome 10 {10001.100100, 10010|0.11101, 101|.11010}
Chromosome 7 {10001.100100, 10100|.110, 101|0100.1000101}
(bar shows crossover points)

After crossover, the following new offsprings are obtained:
Chromosome 17 {10001.100100, 10010 .110, 1010100.1000101}
Chromosome 18 {10001.100100, 10100 0.11101, 101.11010}
The above chromosomes are written as in value encoded form.
Chromosome 17 {17.36, 18.30, 84.81}
Chromosome 18 {17.36, 40.23, 06.11}

From the above obtained chromosomes, it is judged that there is no such good change in the angles in angle ψ, but the angles ϕ have changed largely. So, we will consider angles of link ϕ for further run.

d. Mutation: With a mutation probability, new offspring are mutated at each locus. When mutation with binary encoding is done, selected bits are inverted. Arrow indicates mutated bits.

Before Mutation:
Chromosome 17 {10001.100100, 10010 .110, 1010100.1000101}
Chromosome 18 {10001.100100, 10100 0.11101, 101.11010}

After Mutation:
Chromosome 19 {10001.100100, 10110 .110, 1010100.1000101}
Chromosome 20 {10001.100100, 10000 0.11101, 101.11010}

e. Accepting: New offsrings are placed in the new population

Step-4:

[Replace]:

After the crossover, the angle ϕ has been modified by some value and hence an error appears in obtaining final destination point. These newly obtained chromosomes are used for finding other angles.

Here, $\phi = 22.30^{\circ}$ and 32.23°. The value for angles θ and ψ are being calculated.

Values of other angles based on inverse kinematics are obtained using computer program as below.

$\theta = 17.36^{\circ}$, $\phi = 22.30^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 32.23^{\circ}$, $\psi = 5.26^{\circ}$

New population and fitness values of chromosomes obtained after third run as given in Table 5.33.

Table 5.33: Fitness value of chromosomes obtained after third run

Chromosome No.	Chromosome (angles in degree)	Fitness value
21	{17.36, 22.30, 84.69}	371.60
22	{17.36, 22.30, 5.26}	134.13
23	{17.36, 32.23, 84.69}	401.19
24	{17.36, 32.23, 5.26}	163.72

Step-5:

[Test]:

As it is clear from the above newly population, that the chromosomes in this population have not good fitness values than created in the previous first run. So, we select the population obtained in the first run as final population.

In case of 4 DOF manipulator, lowest fitness having higher ranking is in first run chromosomes. There is no alternative in this case. The optimal solutions are obtained after first run as:

$\theta = 17.36^{\circ}$, $\phi = 20.06^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 36.29^{\circ}$, $\psi = 5.26^{\circ}$ ----(5.31)

5.7 Implementation of Fuzzy Logic

Importance and values of following three attributes have been evaluated for the each angle separately using fuzzy logic. Joint movement (A_1), friction (A_2) and settling time (A_3) are three attributes, which are taken as inputs and weights λ_1, λ_2 and λ_3 are outputs of fuzzy models. The ranges of fuzzy input membership functions and output membership functions are from 0 to 1 (per unit basis). Table 5.34 shows input fuzzy expressions.

Table 5.34: Input fuzzy expressions

1st input (joint movement)		2nd input (friction)		3rd input (settling time)		Index representation for all three inputs
Abbreviation	Expression	Abbreviation	Expression	Abbreviation	Expression	
VS	Very small	VS	Very small	VS	Very small	0.00
S	Small	S	Small	S	Small	0.25
M	Medium	M	Medium	M	Medium	0.50
L	Large	L	Large	L	Large	0.75
VL	Very Large	VL	Very Large	VL	Very Large	0.10

5.7.1 Case I: For 2 DOF Manipulator

Table 5.34 and 5.35 show the inputs fuzzy expressions and output fuzzy expressions for angle θ_1 and angle θ_2. The ranges of fuzzy input membership functions and output membership functions are from 0 to 1 (per unit basis). Table 5.36 shows fuzzy rules considered for 2 DOF manipulator.

Table 5.35: Output fuzzy expressions

1st Output (λ_1)		2nd Output (λ_2)		Index representation for all three outputs
Abbreviation	Expression	Abbreviation	Expression	
EVS	Extremely very small	EVS	Extremely very small	0.00
ES	Extremely small	ES	Extremely small	0.10
VVS	Very, very small	VVS	Very, very small	0.20
VS	Very small	VS	Very small	0.30
S	Small	S	Small	0.40
M	Medium	M	Medium	0.50
L	Large	L	Large	0.60
VL	Very large	VL	Very large	0.70
VVL	Very, very large	VVL	Very, very large	0.80
EL	Extremely large	EL	Extremely large	0.90
EVL	Extremely very large	EVL	Extremely very large	1.00

Table 5.36: Fuzzy rules

If A_1 is VS and A_2 is VS and A_3 is VS then λ_1 is EVS and λ_2 is EVS
If A_1 is S and A_2 is VS and A_3 is VS then λ_1 is ES and λ_2 is EVS
If A_1 is M and A_2 is VS and $A_3$is VS then λ_1 is VVS and λ_2 is EVS
If A_1 is L and A_2 is VS and A_3 is VS then λ_1 is VS and λ_2 is ES

If A_1 is VL and A_2 is VS and A_3 is VS then λ_1 is S and λ_2 is ES
If A_1 is VS and A_2 is S and A_3 is VS then λ_1 is EVS and λ_2 is ES
If A_1 is S and A_2 is S and $A_3$is VS then λ_1 is ES and λ_2 is ES
If A_1 is M and A_2 is S and A_3 is VS then λ_1 is VVS and λ_2 is ES
If A_1 is L and A_2 is S and A_3 is VS then λ_1 is VS and λ_2 is ES
If A_1 is VL and A_2 is S and A_3 is VS then λ_1 is S and λ_2 is ES.
If A_1 is VS and A_2 is M and A_3 is VL then λ_1 is EVS and λ_2 is VVS
If $A_1$is S and A_2 is M and A_3 is VL then λ_1 is ES and λ_2 is VVS
If A_1 is M and A_2 is M and A_3 is VL then λ_1 is VVS and λ_2 is VVS
If A_1 is L and A_2 is M and $A_3$3 is VL then λ_1 is VS and λ_2 is VVS
If A_1 is VL and A_2 is M and A_3 is VL then λ_1 is S and λ_2 is VVS
.....
.....
.....
.....
If A_1 is VL and A_2 is L and $A_3$is VL then λ_1 is EVL and λ_2 is VL
If A_1 is VS and $A_2$is VL and A_3 is VL then λ_1 is L and λ_2 is EVL
If A_1 is S and A_2 is VL and A_3 is VL then λ_1 is VL and λ_2 is EVL
If A_1 is M and A_2 is V L and A_3 is VL then λ_1 is VVL and λ_2 is EVL
If A1 is L and A_2 is VL and A_3 is VL then λ_1 is EL and λ_2 is EVL
If A_1 is VL and A_2 is VL and A_3 is VL then λ_1 is EVL and λ_2 is EVL

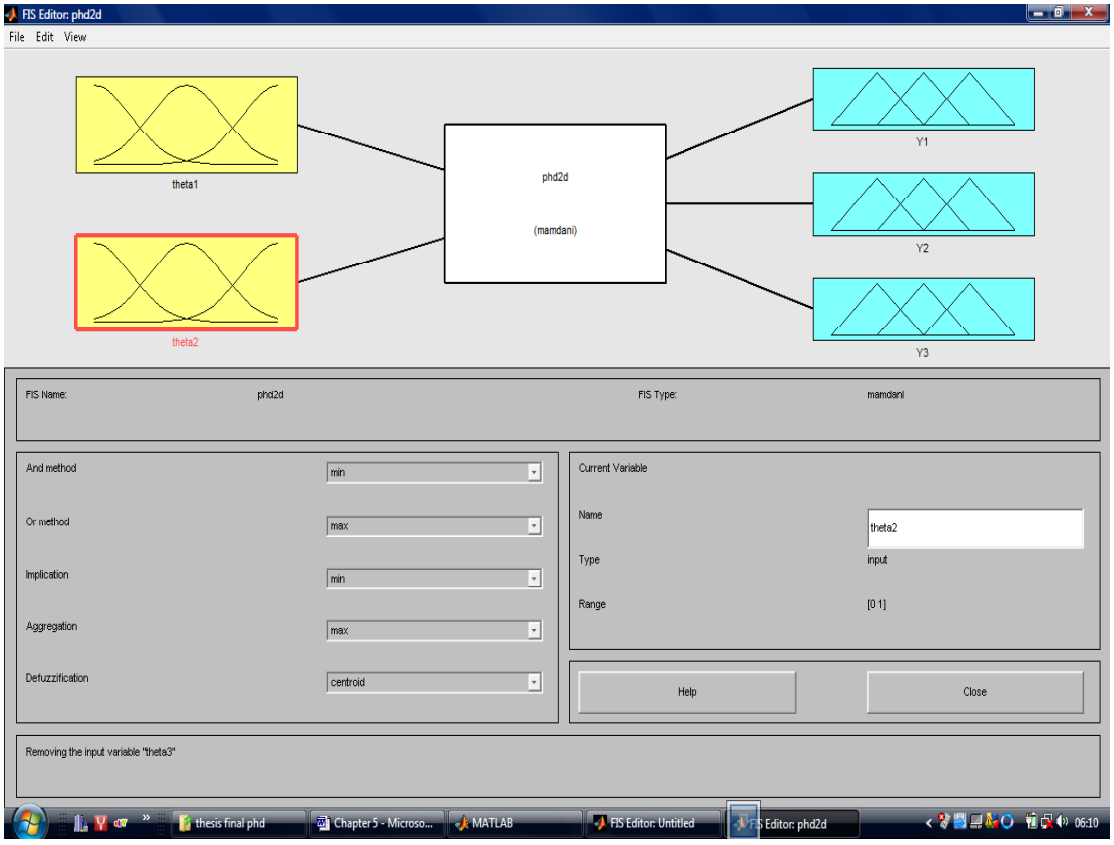


Fig. 5.2: Fuzzy editor for 2 DOF

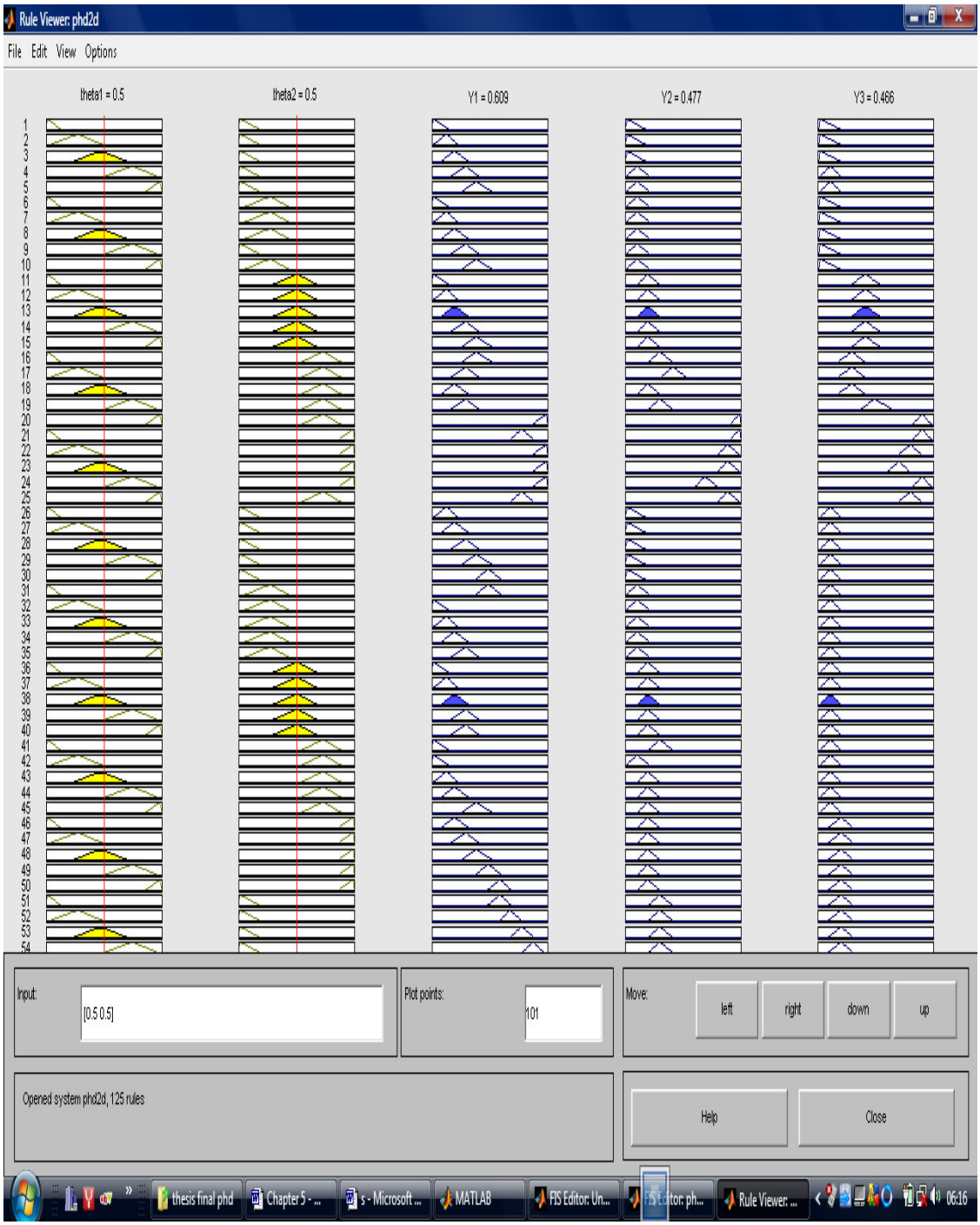


Fig. 5.3: Fuzzy rule viewer for 2 DOF

In Fig. 5.2 and Fig. 5.3 are snap shots of fuzzy editor and fuzzy rule viewer, respectively. Using fuzzy logic, the values of $\lambda_1 = 0.609$ and $\lambda_2 = 0.477$ as the output of fuzzy models, are calculated corresponding to three attributes, joint movement (A_1), friction (A_2) and settling time (A_3).

The corresponding fitness function (f_c) is given by Eq. (5.8), *i.e.*, $f_c = (\theta_1 \times \lambda_1 + \theta_2 \times \lambda_2)$.

5.7.2 Case II: For 3 DOF Manipulator

Importance and value of three attributes are evaluated for the each angle separately using fuzzy logic. Joint movement (A_1), friction (A_2) and settling time (A_3) are three attributes as inputs and weights λ_1, λ_2 and λ_3 are three outputs of fuzzy models. The ranges of fuzzy input membership functions and output membership functions are from 0 to 1 (per unit basis). Table 5.34 and 5.37 show the inputs fuzzy expressions and output fuzzy expressions, respectively. Table 5.38 shows fuzzy rules considered in this case.

Table 5.37: Output fuzzy expressions

1st Output (λ_1)		2nd Output (λ_2)		3rd Output (λ_3)		Index representation for all three outputs
Abbreviation	Expression	Abbreviation	Expression	Abbreviation	Expression	
EVS	Extremely very small	EVS	Extremely very small	EVS	Extremely very small	0.00
ES	Extremely small	ES	Extremely small	ES	Extremely small	0.10
VVS	Very, very small	VVS	Very, very small	VVS	Very, very small	0.20
VS	Very small	VS	Very small	VS	Very small	0.30
S	Small	S	Small	S	Small	0.40
M	Medium	M	Medium	M	Medium	0.50
L	Large	L	Large	L	Large	0.60
VL	Very large	VL	Very large	VL	Very large	0.70
VVL	Very, very large	VVL	Very, very large	VVL	Very, very large	0.80
EL	Extremely large	EL	Extremely large	EL	Extremely large	0.90
EVL	Extremely very large	EVL	Extremely very large	EVL	Extremely very large	1.00

Table 5.38: Fuzzy rules

If A_1 is VS and A_2 is VS and A_3 is VS then λ_1 is EVS and λ_2 is EVS and λ_3 is EVS
If A_1 is S and A_2 is VS and A_3 is VS then λ_1 is ES and λ_2 is EVS and λ_3 is EVS
If A_1 is M and A_2 is VS and $A_3$is VS then λ_1 is VVS and λ_2 is EVS and λ_3 is EVS
If A_1 is L and A_2 is VS and A_3 is VS then λ_1 is VS and λ_2 is ES and λ_3 is ES
If A_1 is VL and A_2 is VS and A_3 is VS then λ_1 is S and λ_2 is ES and λ_3 is ES
If A_1 is VS and A_2 is S and A_3 is VS then λ_1 is EVS and λ_2 is ES and λ_3 is EVS
If A_1 is S and A_2 is S and A_3 is VS then λ_1 is ES and λ_2 is ES and λ_3 is EVS
If A_1 is M and A_2 is S and $A_3$is VS then λ_1 is VVS and λ_2 is ES and λ_3 is EVS
If A_1 is L and A_2 is S and A_3 is VS then λ_1 is VS and λ_2 is ES and λ_3 is EVS
If A_1 is VL and A_2 is S and A_3 is VS then λ_1 is S and λ_2 is ES and λ_3 is EVS
If A_1 is VS and A_2 is M and A_3 is VL then λ_1 is EVS and λ_2 is VVS and λ_3 is S
If A_1 is S and A_2 is M and $A_3$is VL then λ_1 is ES and λ_2 is VVS and λ_3 is S
If A_1 is M and A_2 is M and A_3 is VL then λ_1 is VVS and λ_2 is VVS and λ_3 is S
If A_1 is L and A_2 is M and A_3 is VL then λ_1 is VS and λ_2 is VVS and λ_3 is S
If A_1 is VL and A_2 is M and A_3 is VL then λ_1 is S and λ_2 is VVS and λ_3 is S
……
……
……
……

If A_1 is VL and A_2 is L and A_3 is VL then λ_1 is EVL and λ_2 is VL and λ_3 is EVL
If A_1 is VS and A_2 is VL and A_3 is VL then λ_1 is L and λ_2 is EVL and λ_3 is EVL
If A_1 is S and A_2 is VL and A_3 is VL then λ_1 is VL and λ_2 is EVL and $\lambda_3$is EVL
If A_1 is M and A_2 is V L and A_3 is VL then λ_1 is VVL and λ_2 is EVL and λ_3 is EVL
If A_1 is L and A_2 is VL and A_3 is VL then $\lambda_1$is EL and λ_2 is EVL and λ_3 is EVL
If A_1 is VL and A_2 is VL and A_3 is VL then λ_1 is EVL and λ_2 is EVL and λ_3 is EVL

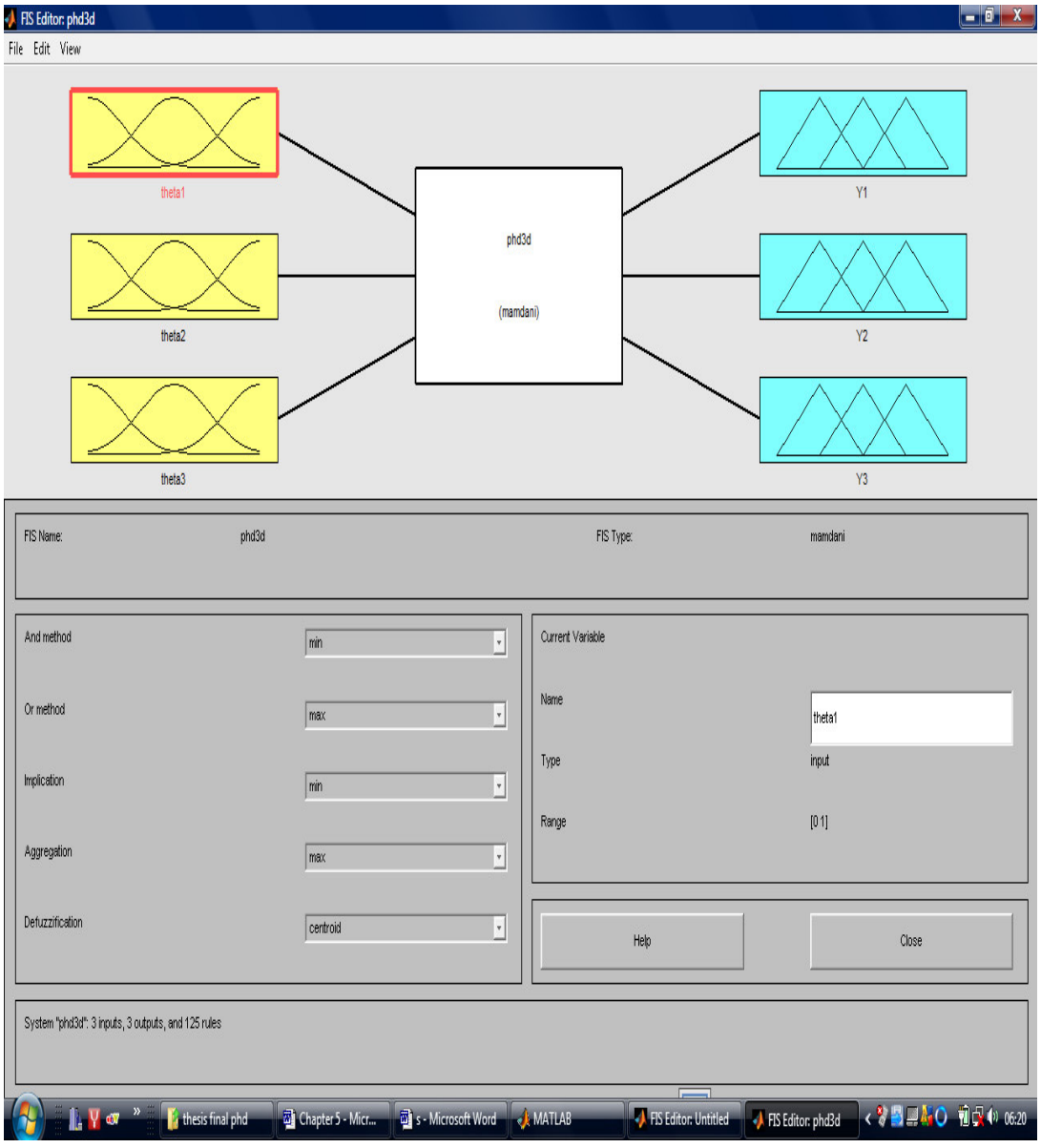


Fig. 5.4: Fuzzy editor for 3 DOF

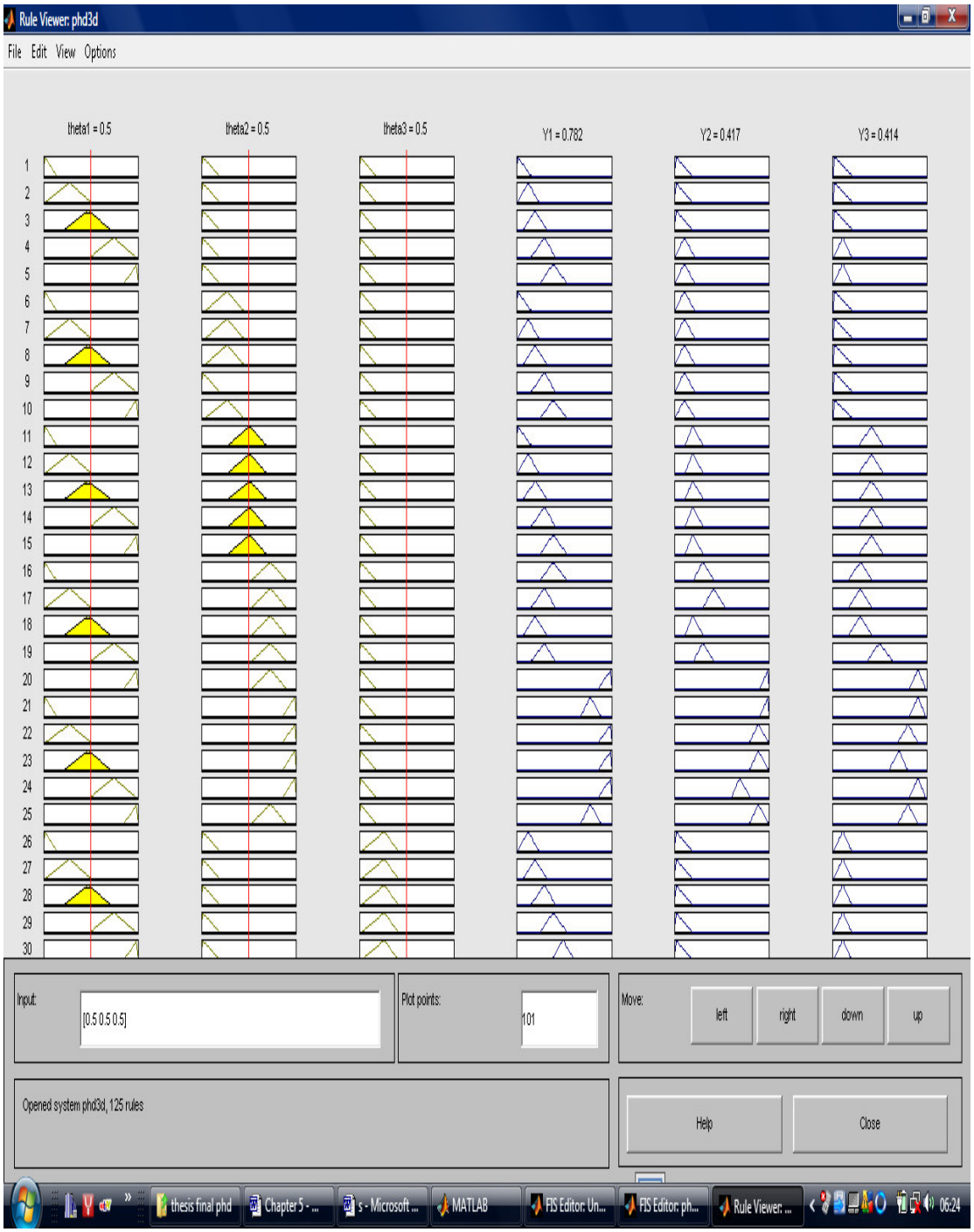


Fig. 5.5: Fuzzy rule viewer for 3 DOF

In Fig. 5.4 and 5.5 are snap shots of fuzzy editor and fuzzy rule viewer, respectively for 3 DOF manipulator. Using fuzzy logic, the values of $\lambda_1 = 0.782$, $\lambda_2 = 0.417$ and $\lambda_3 = 0.414$ as the output of fuzzy models, are calculated corresponding to three attributes, joint movement (A_1), friction (A_2) and settling time (A_3).

The corresponding fitness function (f_c) is given by Eq. (5.18), *i.e,* $f_c = (\theta_1 \times \lambda_1 + \theta_2 \times \lambda_2 + \theta_3 \times \lambda_3)$.

5.7.3 Case III: For 4 DOF Manipulator

For 4 DOF robotic arm manipulator, the importance and value of the attributes for the each angle separately using fuzzy logic has been evaluated. Here, again Attributes joint movement (A_1), friction (A_2) and settling time (A_3) (inputs) and weights λ_1, λ_2 and λ_3 (outputs) of fuzzy models. The ranges of fuzzy input membership functions and output membership functions are from 0 to 1 (per unit basis).

The importance and value of these three attributes for the each angle have been decided separately. Table 5.13 gives angle 'θ', moved by Joint '1'. Table 5.14 gives angle 'ϕ', moved by joint '2'. Table 5.15 gives angle 'ψ', moved by Joint '4'.

Table 5.34 and 5.39 show the inputs fuzzy expressions and output fuzzy expressions, respectively. Table 5.40 shows fuzzy rules taken in this case.

Table 5.39: Output fuzzy expressions

1st Output (λ_1) for θ		2nd Output (λ_2) for ϕ		3rd Output (λ_3) for ψ		Index representation for all three outputs
Abbreviation	Expression	Abbreviation	Expression	Abbreviation	Expression	
EVS	Extremely very small	EVS	Extremely very small	EVS	Extremely very small	0.00
ES	Extremely small	ES	Extremely small	ES	Extremely small	0.10
VVS	Very, very small	VVS	Very, very small	VVS	Very, very small	0.20
VS	Very small	VS	Very small	VS	Very small	0.30
S	Small	S	Small	S	Small	0.40
M	Medium	M	Medium	M	Medium	0.50
L	Large	L	Large	L	Large	0.60
VL	Very large	VL	Very large	VL	Very large	0.70
VVL	Very, very large	VVL	Very, very large	VVL	Very, very large	0.80
EL	Extremely large	EL	Extremely large	EL	Extremely large	0.90
EVL	Extremely very large	EVL	Extremely very large	EVL	Extremely very large	1.00

Table 5.40: Fuzzy rules

If A_1 is VS and A_2 is VS and A_3 is VS then λ_1 is EVS and λ_2 is EVS and λ_3 is EVS
If A_1 is S and A_2 is VS and A_3 is VS then λ_1 is ES and λ_2 is EVS and λ_3 is EVS
If A_1 is M and A_2 is VS and $A_3$is VS then λ_1 is VVS and λ_2 is EVS and λ_3 is EVS
If A_1 is L and A_2 is VS and A_3 is VS then λ_1 is VS and λ_2 is ES and λ_3 is ES
If A_1 is VL and A_2 is VS and A_3 is VS then λ_1 is S and λ_2 is ES and λ_3 is ES
If A_1 is VS and A_2 is S and A_3 is VS then λ_1 is EVS and λ_2 is ES and λ_3 is EVS
If A_1 is S and A_2 is S and A_3 is VS then λ_1 is ES and λ_2 is ES and λ_3 is EVS
If A_1 is M and A_2 is S and $A_3$is VS then λ_1 is VVS and λ_2 is ES and λ_3 is EVS
If A_1 is L and A_2 is S and A_3 is VS then λ_1 is VS and λ_2 is ES and λ_3 is EVS
If A_1 is VL and A_2 is S and A_3 is VS then λ_1 is S and λ_2 is ES and λ_3 is EVS
If A_1 is VS and A_2 is M and A_3 is VL then λ_1 is EVS and λ_2 is VVS and λ_3 is S
If A_1 is S and A_2 is M and $A_3$is VL then λ_1 is ES and λ_2 is VVS and λ_3 is S
If A_1 is M and A_2 is M and A_3 is VL then λ_1 is VVS and λ_2 is VVS and λ_3 is S

If A_1 *is L and* A_2 *is M and* A_3 *is VL then* λ_1 *is VS and* λ_2 *is VVS and* λ_3 *is S*
If A_1 *is VL and* A_2 *is M and* A_3 *is VL then* λ_1 *is S and* λ_2 *is VVS and* λ_3 *is S*
……
……
……
……
If A_1 *is VL and* A_2 *is L and* A_3 *is VL then* λ_1 *is EVL and* λ_2 *is VL and* λ_3 *is EVL*
If A_1 *is VS and* A_2 *is VL and* A_3 *is VL then* λ_1 *is L and* λ_2 *is EVL and* λ_3 *is EVL*
If A_1 *is S and* A_2 *is VL and* A_3 *is VL then* λ_1 *is VL and* λ_2 *is EVL and* λ_3*is EVL*
If A_1 *is M and* A_2 *is V L and* A_3 *is VL then* λ_1 *is VVL and* λ_2 *is EVL and* λ_3 *is EVL*
If A_1 *is L and* A_2 *is VL and* A_3 *is VL then* λ_1*is EL and* λ_2 *is EVL and* λ_3 *is EVL*
If A_1 *is VL and* A_2 *is VL and* A_3 *is VL then* λ_1 *is EVL and* λ_2 *is EVL and* λ_3 *is EVL*

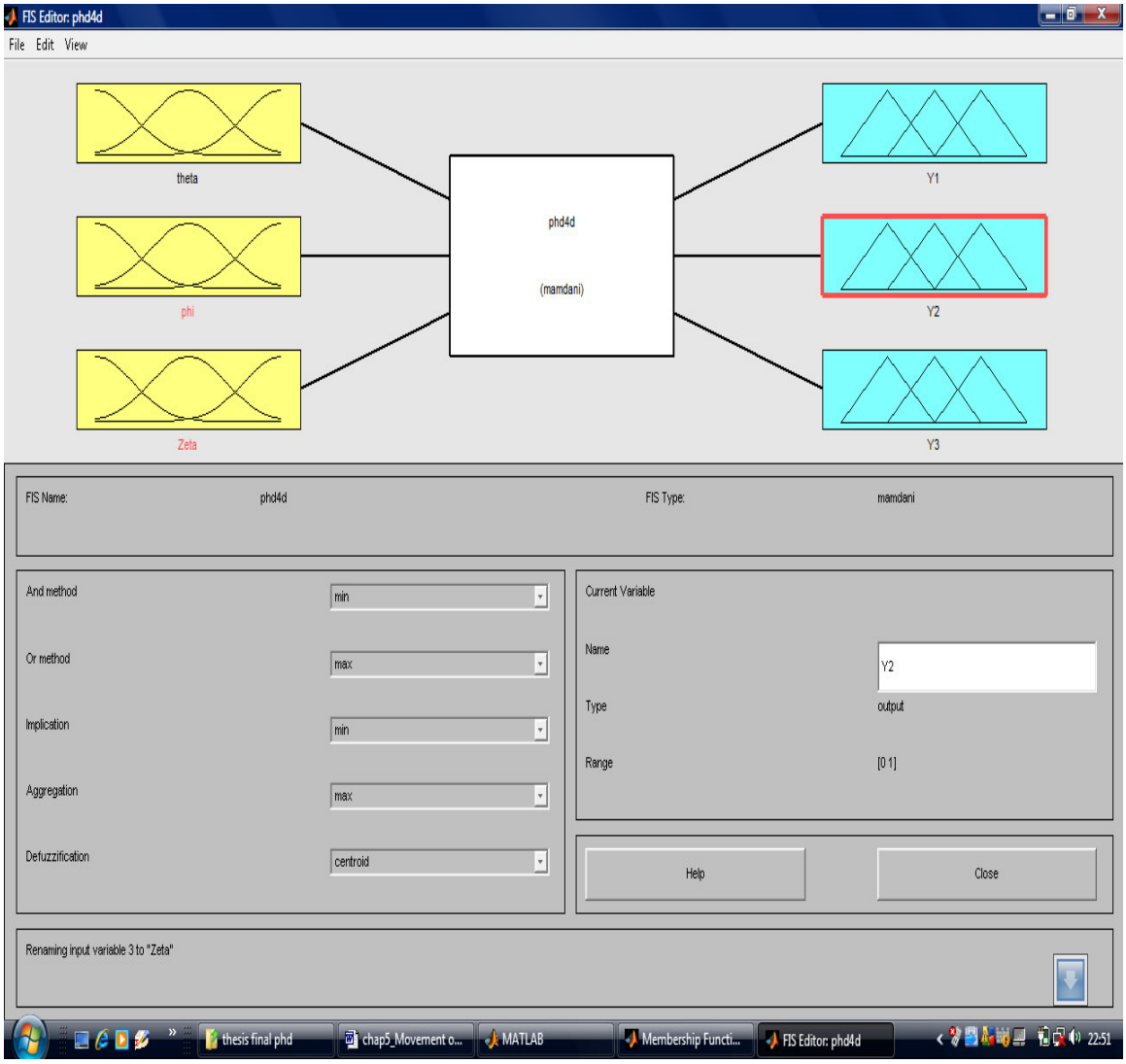


Fig. 5.6: Fuzzy editor for 4 DOF

Fig. 5.7: Fuzzy rule viewer for 4 DOF

In Fig. 5.6 and 5.7 snap shots of fuzzy editor and fuzzy rule viewer, respectively for 4 DOF manipulator has been shown. Here, the inputs are the attributes joint movement (A_1), friction (A_2) and settling time (A_3) and outputs are the weights $\lambda_1 = 0.741$, $\lambda_2 = 0.417$ and $\lambda_3 = 0.383$. The corresponding fitness function (f_c) is given by Eq. (5.28), *i.e.*, $f_c = (\theta \times \lambda_1 + \phi \times \lambda_2 + \psi \times \lambda_3)$.

5.8 Implementation of Fuzzy Logic and Genetic Algorithms

Conventional methods of optimisation require an accurate mathematical model. In robot manipulator any mathematical modeling inaccuracy will hamper the mathematical optimisation process. Also, as the configuration is changed, the optimisation needs to be redefined. Genetic Algorithms is an intelligent optimisation method. Here in this work, genetic algorithms are proposed to search the optimal angular displacement of robot arms as shown in flowchart given in Fig. 5.8.

The genetic algorithms is applied for generating the population of chromosomes having optimised values. The values of all angles is positive because the negative sign is only due to the right hand side or left hand side movement of the links.

1. [Start]: Generate random population of n chromosomes (suitable solutions for the problem).
2. [Fitness]: Evaluate the fitness $f_c(x)$ using fuzzy logic of each chromosome x in the population.
3. [New population]**:** Create a new population by repeating following steps until the new population is complete.
    a. Encoding: The encoding mainly depends on the solved problem. Various encodings are used depending upon the problem like value, binary, integer or real number, *etc*.
    b. Selection: Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected).
    c. Crossover: With a crossover probability, cross over the parents so as to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.
    d. Mutation: With a mutation probability, mutate new offspring at each locus (position in chromosome).
    e. Accepting: Place new offspring in the new population.
4. [Replace]: Use new generated population for a further run of the algorithm.
5. [Test]: If the end condition is satisfied, stop, and return the best solution in current population.
6. [Loop]: Go to step 2 [Fitness].

Solutions obtained from inverse kinematics are fed to the genetic algorithms for generating the population of chromosomes to be optimised.

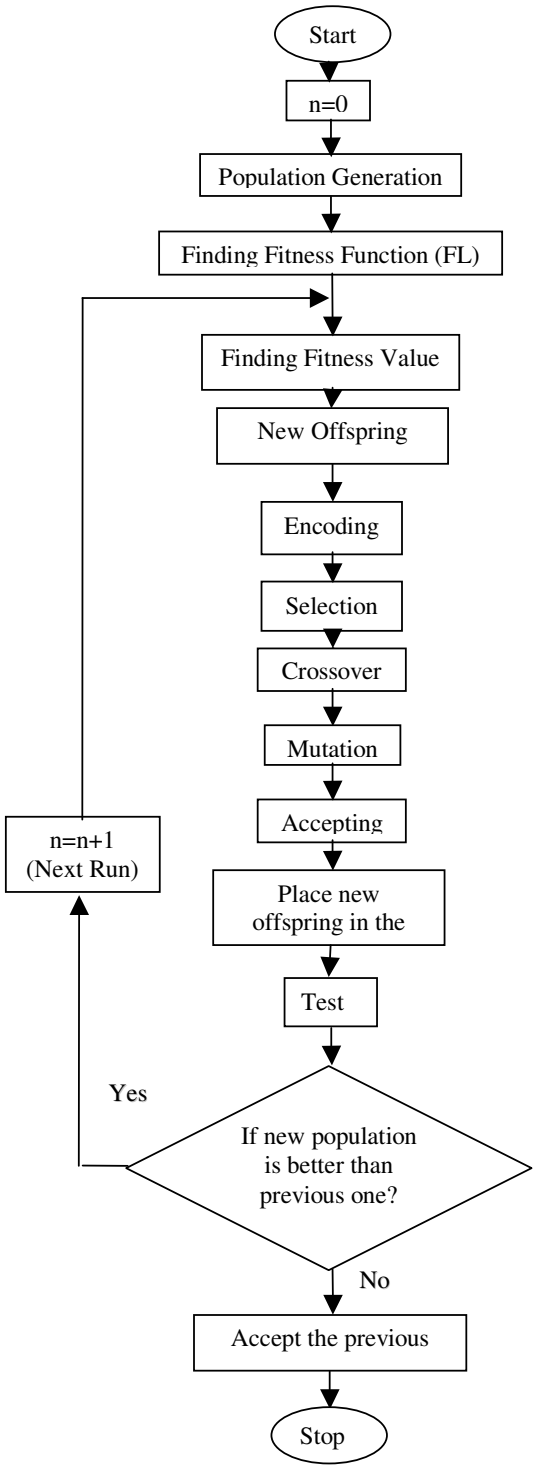


Fig. 5.8: Flowchart using fuzzy logic and genetic algorithms

Fitness of each chromosome depends upon many factors. Three mains factors on which the fitness function depends have been calculated by applying fuzzy logic. These three main factors are:

1. Joint Movement (A_1)
2. Friction (A_2)
3. Settling time (Minimum Vibration) (A_3)

The importance and values of these three attributes for the each angle have been decided.

If the new population is matched with the desired results, the population is stored in the search space, otherwise the inverse kinematics is again applied to the newly obtained population and the whole procedure is repeated till the required or desired results are obtained.

Output weights λ_1, λ_2 and λ_3 are obtained considering three attributes-joint movement (A_1), friction (A_2) and settling time (A_3) as inputs. Table 5.34 and 5.35 shows the inputs fuzzy expressions and output fuzzy expressions respectively.

5.8.1 Case I: 2 DOF Manipulator

Fitness of each chromosome is determined using the fitness function defined by Eq. (5.8). In case of 2 DOF manipulator, inverse kinematics solutions obtained from Eqs. (4.7), (4.8), and (4.9) (refer to 4.2.3 Case study: 2 DOF) are given as.

$\theta_1 = 30.76^\circ$, $\theta_2 = 47.81^\circ$

and $\theta_1 = 73.94^\circ$, $\theta_2 = -47.81^\circ$

First run

Step-1:

[Start]: The angles obtained above with inverse kinematics can be combined so as to form different solutions as shown in Table 5.41.

Step-2:

[Fitness]: Fitness value is calculated by fitness function defined by Eq. (5.8), i.e., $f_c = (\theta_1 \times \lambda_1 + \theta_2 \times \lambda_2)$, where, λ_1=0.609, and λ_2=0.477.

Fitness of each chromosome is evaluated with the help of above fitness function.

Table 5.41: Fitness value of chromosomes obtained after applying inverse kinematics

Chromosome No.	Chromosome	Fitness function	Fitness value
1.	{30.76, 47.81}	18.37 + 22.80	42.17
2.	{30.76, 47.81}	18.37 + 22.80	42.17
3.	{73.94, 47.81}	45.03+ 22.80	67.83
4.	{73.94, 47.81}	57.82 + 22.80	67.83

It is concluded from the above table that more is the angle, more is the movement of the links and therefore more will be the fitness value of the chromosome. Hence, the ranking of the chromosome will be based on the criteria that less is the fitness value, higher will be the rank.

In case of 2 DOF manipulator, ranking of chromosomes are same as in initial Table 5.41, so there are no change in ranking as per fitness value. There is no other alternative solution in this case. The optimal solution is given as

$\theta_1 = 30.76^\circ$, $\theta_2 = 47.81^\circ$

and $\theta_1 = 73.94^\circ$, $\theta_2 = -47.81^\circ$ ----(5.32)

5.8.2 Case II: For 3 DOF manipulator

In case of 3 DOF manipulator, solution obtained using inverse kinematics are mentioned below (refer to 4.3.3 Case study: 3 DOF manipulator).

The following values for the joint angles of the links for 3 DOF manipulator are obtained from Eqs. (4.41), (4.44), (4.47), (4.48) and (4.49).

$\theta_1 = 137.91^\circ$, $\theta_2 = -217.49^\circ$, $\theta_3 = 106.14^\circ$

and $\theta_1 = -84.78^\circ$, $\theta_2 = 37.49^\circ$, $\theta_3 = 73.85^\circ$

First run

Step-1:

[Start]: The angles obtained above with inverse kinematics can be arranged so as to form different solutions as shown in Table 5.42.

Step-2:

[Fitness]: The fitness function as defined by Eq. (5.18), *i.e.,* $f_c = (\theta_1 \times \lambda_1 + \theta_2 \times \lambda_2 + \theta_3 \times \lambda_3)$, where, λ_1=0.782, and λ_2=0.417 and λ_3 = 0.414. Fitness of each chromosome is determined with the help of above fitness function.

Table 5.42: Fitness value of chromosomes obtained after applying inverse kinematics

Chromosome No.	Chromosome (angles in degree)	Fitness Function	Fitness value
1.	{137.91, 217.49, 106.14}	107.84 + 90.69 + 43.94	242.47
2.	{137.91, 217.49, 73.85}	107.84 + 90.69 + 30.57	229.10
3.	{137.91, 37.49, 106.14}	107.84 + 15.63 + 43.94	167.41
4.	{137.91, 37.49, 73.85}	107.84 + 15.63 + 30.57	154.04
5.	{84.78, 37.49, 106.14}	66.29 + 15.63 + 43.94	125.86
6.	{84.78, 37.49, 73.85}	66.29 + 15.63 + 30.57	112.49
7.	{84.78, 217.49, 106.14}	66.29 + 90.69 + 43.94	200.92
8.	{84.78, 217.49, 73.85}	66.29 + 90.69 + 30.57	187.55

It can be concluded from the fitness values as shown in Table 5.42, more is the angle, more is the movement of the links and therefore, more will be the fitness value of the chromosome. Hence, the ranking of the chromosome is based on the criterion that less is the fitness value, higher will be the rank.

Hence, the chromosomes according to their ranking (in descending order) are shown in Table 5.43.

Table 5.43: Ranking of chromosomes obtained after applying inverse kinematics

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 6 {84.78, 37.49, 73.85}	112.49
2	Chromosome 5 {84.78, 37.49, 106.14}	125.86
3	Chromosome 4 {137.91, 37.49, 73.85}	154.04
4	Chromosome 3 {137.91, 37.49, 106.14}	167.41
5	Chromosome 8 {84.78, 217.49, 73.85}	187.55
6	Chromosome 7 {84.78, 217.49, 106.14}	200.92
7	Chromosome 2 {137.91, 217.49, 73.85}	229.10
8	Chromosome 1 {137.91, 217.49, 106.14}	242.47

Step-3:

[New population]:

a. Encoding: As we have chromosomes having values too large, hence value encoding is considered. However, for this encoding, it is often necessary to develop some new crossover and mutation specific for the problem.

b. Selection: According to rank selection, two chromosomes having highest ranks are selected. So in this case, we have selected first two chromosomes with the following specifications:

Ranking	Chromosome	Fitness value
1	Chromosome 6 {84.78, 37.49, 73.85}	112.49
2	Chromosome 5 {84.78, 37.49, 106.14}	125.86

c. Crossover: While crossover on the first two chromosomes, crossover points in the angle of third link (*i.e.,* in θ_3) are shown below:

{84.78, 37.49, 7|3. 8|5}

{84.78, 37.49, 10|6. 1|4}

(bar shows crossover points)

After crossover, the following new offsprings are obtained.

Chromosome 9 {84.78, 37.49, 103.84}

Chromosome10 {84.78, 37.49, 76.15}

d. Mutation: Value encoding is considered here and hence, mutation is ignored.

e. Accepting: New offsprings are placed in the new population.

Step-4:

[Replace]: As a result of crossover, the angle of link '3' has been modified and hence an error appears in obtaining final destination point. So, new angles are calculated for link '1' and link '2'.

Here, $\theta_3 = 103.84^\circ$ and 76.15°. The values for angle θ_1 and angle θ_2 are being obtained.

Values of other angles based on inverse kinematics using computer program are obtained as below.

$\theta_1 = -20.84^\circ,\ \theta_2 = -36.95^\circ, \theta_3 = 103.84^\circ$

and $\theta_1 = -40.33^\circ, \theta_2 = -28.75^\circ,\ \theta_3 = 76.15^\circ$

New population and fitness values of chromosomes obtained after first run are shown in Table 5. 44.

Table 5.44: Fitness value of chromosomes obtained from first run

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
11.	{20.84, 36.95, 103.84}	16.29 + 15.40 + 42.98	74.67
12.	{20.84, 36.95, 76.15}	16.29 + 15.40 + 31.52	63.21
13.	{20.84, 28.75, 103.84}	16.29 + 11.98 + 42.98	71.25
14.	{20.84, 28.75, 76.15}	16.29 + 11.98 + 31.52	59.79
15.	{40.33, 36.95, 103.84}	31.53 + 15.40 + 42.98	89.91
16.	{40.33, 36.95, 76.15}	31.53 + 15.40 + 31.52	78.45
17.	{40.33, 28.75, 103.84}	31.53 + 11.98 + 42.98	86.49
18.	{40.33, 28.75, 76.15}	31.53 + 11.98 + 31.52	75.03

Step-5

[Test]: As it is clear from the above obtained new population, that, the chromosomes in this population have much better fitness values than the previous one. So this whole population is selected for further run.

Step-6

[Loop]: Go to step 2 [Fitness].

Second run

Step-2:

[Fitness]: We have already determined fitness values in the previous run. The new population is arranged according to there ranking (in descending order) as given in Table 5.45.

Table 5.45: Ranking of chromosomes obtained after first run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome14 {20.84, 28.75, 76.15}	59.79
2	Chromosome12 {20.84, 36.95, 76.15}	63.21
3	Chromosome13 {20.84, 28.75, 103.84}	71.15
4	Chromosome11 {20.84, 36.95, 103.84}	74.67
5	Chromosome18 {40.33, 28.75, 76.15}	75.03
6	Chromosome16 {40.33, 36.95, 76.15}	78.45
7	Chromosome17 {40.33, 28.75, 103.84}	86.49
8	Chromosome15 {40.33, 36.95, 103.84}	89.91

Step-3:

[New population]:

a. Encoding: Chromosomes are already in value encoding form.

b. Selection: According to rank selection, we first choose two chromosomes having highest ranks. So, in this case, we have chosen first two chromosomes with the following specifications:

Ranking	Chromosome	Fitness value
1	Chromosome14 {-20.84, -28.75, 76.15}	59.79
2	Chromosome12 {-20.84, -36.95, 76.15}	63.21

c. Crossover: While crossover on the first two chromosomes, crossover points in the angle of second link (*i.e.*, in θ_2) are shown below.

{-20.84, -2|8.7|5, 76.15}
{-20.84, -3|6.9|5, 76.15}
(bar shows crossover points)

After crossover, we get the following new offsprings.

Chromosome 19 {-20.84, -26.95, 76.15}

Chromosome 20 {-20.84, -38.75, 76.15}

d. Mutation: We have taken value encoding and hence mutation is ignored here.

e. Accepting: In the new population, new offsprings are placed.

Step-4:

[Replace]: As a result of crossover, the angle of link '2' has been modified and hence an error appears in reaching final destination point. So, new angles are calculated for link '1' and link '3'.

Here, $\theta_2 = -26.95^\circ$ and -38.75°. The values for θ_1 and θ_3 are being obtained.

Values for θ_1 and θ_3 are obtained using inverse kinematics through computer program as given below.

$\theta_1 = 137.91^\circ$, $\theta_2 = -26.95^\circ$, $\theta_3 = -84.4^\circ$

and $\theta_1 = -84.78^\circ$, $\theta_2 = -38.75^\circ$, $\theta_3 = 150.10^\circ$

New population of chromosomes and fitness values after second run as shown in Table 5.46.

Table 5.46: Fitness value of chromosomes obtained after second run

Chromosome No.	Chromosome (angles in degree)	Fitness Function	Fitness value
21.	{137.91, 26.95, 84.40}	107.84 + 11.23 + 34.94	64.01
22.	{137.91, 26.95, 150.10}	107.84 + 11.23 + 62.14	181.21
23.	{137.91, 38.75, 84.40}	107.84 + 16.15 +34.94	158.93
24.	{137.91, 38.75, 150.10}	107.84 + 16.15 +62.14	186.13
25.	{84.78, 26.95, 84.40}	66.29 + 11.23 + 34.94	112.46
26.	{84.78, 26.95, 150.10}	66.29 + 11.23 + 62.14	139.66
27.	{84.78, 38.75, 84.40}	66.29 + 16.15 +34.94	117.38
28.	{84.78, 38.75, 150.10}	66.29 + 16.15 +62.14	144.58

Step-5:

[Test]: As it is clear from the above newly obtained population, that the chromosomes in this population have not good fitness values than the previous one. So, we select the previous whole population obtained in the first run as final population.

Step-6:

[Loop]: Go to step 2 [Fitness]

Third run

Step-2:

[Fitness]: We have already determined the fitness values in the previous run. The ranking of chromosomes obtained after first run (in descending order) as shown in Table 5.45.

Step-3:

[New population]:

a. Encoding: Now, as the magnitude of the angles are less in the second generations, binary encoding is performed. The population in value encoded form is given as.

Chromosome 14 {- 20.84, - 28.75, 76.15}

Chromosome 12{- 20.84, - 36.95, 76.15}

Chromosome 13 {- 20.84, - 28.75, 103.84}

Chromosome 11 {- 20.84, - 36.95, 103.84}

Chromosome 18{- 40.33, - 28.75, 76.15}

Chromosome 16 {- 40.33, - 36.95, 76.15}

Chromosome 17{- 40.33, - 28.75, 103.84}

Chromosome 15 {- 40.33, - 36.95, 103.84}

After binary encoding, we have

Chromosome 14 {10100.1010100, 11100.1001011, 1001100.1111}

Chromosome 12 {10100.1010100, 100100.1011111, 1001100.1111}

Chromosome 13 {10100.1010100, 11100.1001011, 1100111.1010100}

Chromosome 11 {10100.1010100, 100100.1011111, 1100111.1010100}

Chromosome 18 {101000.100001, 11100.1001011, 1001100.1111}

Chromosome 16{101000.100001, 100100.1011111, 1001100.1111}

Chromosome 17 {101000.100001, 11100.1001011, 1100111.1010100}

Chromosome 15 {101000.100001, 100100.1011111, 1100111.1010100}

b. Selection: According to rank selection we had already choosen two chromosomes having highest ranks in second run. Now we will choose next two chromosomes (chromosomes 12 and 13) in the increasing order of fitness for further run.

Next, two chromosomes (chromosomes 12 and 13) are chosen in the increasing order of fitness.

Ranking	Chromosome	Fitness value
2	12 {10100.1010100, 100100.1011111, 1001100.1111}	63.21
3	13 {10100.1010100, 11100.1001011, 1100111.1010100}	71.15

c. Crossover: While crossover on the next two chromosomes, we can make crossover points in the angle of first and second link (*i.e.,* in θ_2 and θ_3) as shown below.

Chromosome 12 {10100.1010100, 100|100.1011111, 100|1100.1111}

Chromosome 13 {10100.1010100, 111|00.1001011, 110|0111.1010100}

(Bar shows crossover points)

After crossover, the following new offsprings are obtained.

Chromosome 29 {10100.1010100, 10000.1001011, 1000111.1010100}

Chromosome 30 {10100.1010100, 111100.1011111, 110 1100.1111}

The above chromosomes are written as in value encoded form.

Chromosome 29 {-20.84, -16.10, 71.11}

Chromosome 30 {-20.84, -60.12, 108.15}

From the above obtained chromosome, it is concluded that there is no such good change in the angles of link '3', but the angles of link '2' have changed largely. So, we consider angles of link '2' for further run.

d. Mutation: Random selected bits are inverted, which are indicated by arrow.

Before Mutation

Chromosome 29 {10100.1010100, 10000.1001011, 1000111.1010100}

Chromosome 30 {10100.1010100, 111100.1011111, 110 1100.1111}

After Mutation

Chromosome 31 {10100.1010100, 10100.1001011, 1000111.1010100}

Chromosome 32 {10100.1010100, 110100.1011111, 110 1100.1111}

e. Accepting: New offspring are placed in the new population.

Step-4:

[Replace]: As a result of crossover, the angle of link '2' has been modified and hence an error appears in obtaining final destination point. New angles are calculated for link '1' and link '3'.

Here, $\theta_2 = -20.10^\circ$ and -52.12°. The value for angle θ_1 and θ_3 are being calculated.

Using inverse kinematics, values of θ_1 and θ_3 are obtained using computer program are as below.

$\theta_1 = -137.97^\circ, \theta_2 = -20.10^\circ, \theta_3 = -91.29^\circ$

$\theta_1 = -84.81^\circ, \ \theta_2 = -52.12^\circ, \ \theta_3 = 163.51^\circ$

New population and fitness values of chromosomes obtained after third run as given in Table 5.47.

Table 5.47: Fitness value of chromosomes obtained after third run

Chromosome No.	Chromosome (angles in degree)	Fitness value
33.	{137.97, 20.10, 91.29}	154.07
34.	{137.97, 20.10, 163.51}	183.97
35.	{137.97, 52.12, 91.29}	167.42
36.	{137.97, 52.12, 163.51}	197.32
37.	{84.81, 20.10, 91.29}	112.50
38.	{84.78, 20.10, 163.51}	142.40
39.	{84.78, 52.12, 91.29}	125.85
40.	{84.78, 52.12, 163.51}	155.75

Step-5:

[Test]: It is clear from the above, newly obtained population, that the chromosomes in this population have not good fitness values than the previous one. The population obtained in first run is selected as final population.

The optimal solution for 3 DOF robotic arm manipulator is as:

$\theta_1 = -20.84^\circ,\ \theta_2 = -36.95^\circ, \theta_3 = 103.84^\circ$

and $\theta_1 = -40.33^\circ, \theta_2 = -28.75^\circ,\ \theta_3 = 76.15^\circ$ ----(5.33)

5.8.3 Case III: 4 DOF Manipulator

The values of θ, ϕ and ψ have been obtained for 4 degree-of-freedom (refer to 4.4.3 Case study: 4 DOF manipulator) manipulator from Eqs. (4.83), (4.90), (4.92), (4.99) and (4.101).

$\theta = 17.36^\circ,\ \phi = 30.09^\circ,\ \psi = 84.69^\circ$

and $\theta = 17.36^\circ,\ \phi = 26.26^\circ,\ \psi = 5.26^\circ$

First run

Step-1:

[Start]: The angles obtained above with inverse kinematics can be combined so as to form different solutions as shown in Table 5.48.

Step-2: The fitness value is obtained by fitness function defined by Eq.(5.28), *i.e.,* $f_c = (\theta \times \lambda_1 + \phi \times \lambda_2 + \psi \times \lambda_3)$, where, λ_1=0.741, λ_2=0.417 and λ_3=0.383. Fitness of each chromosome is determined with the help of above fitness function.

Table 5.48: Fitness value of chromosomes after applying inverse kinematics

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
1.	{17.36, 30.09, 84.69}	12.86+12.55+32.44	57.85
2.	{17.36, 30.09, 5.26}	12.86+12.55+2.02	27.43
3.	{17.36, 26.26, 84.69}	12.86+10.95+32.44	56.25
4.	{17.36, 26.26, 5.26}	12.86+10.95+2.02	25.83

It can be concluded from the above Table 5.48, that more is the movement of the links and more will be the fitness value of the chromosome. Hence, the ranking of the chromosome is based on the criterion that less is the fitness value, higher is the rank

Hence, the chromosomes according to their ranking (in descending order) are shown in Table 5.49.

Table 5.49: Ranking of chromosomes obtained after applying inverse kinematics

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 4{17.36, 26.26, 5.26}	25.83
2	Chromosome 2{17.36, 30.09, 5.26}	27.43
3	Chromosome 3{17.36, 26.26, 84.69}	56.25
4	Chromosome 1{17.36, 30.09, 84.69}	57.85

Step-3:

[New population]:

a. Encoding: We have taken value encoding.

b. Selection: According to rank selection two chromosomes having highest ranks are chosen. So, in this case, we have selected first two chromosomes with the following specifications:

Ranking	Chromosome	Fitness value
1	Chromosome 4{17.36, 26.26, 5.26}	25.83
2	Chromosome 2{17.36, 30.09, 5.26}	27.43

c. Crossover: While crossover on the first two chromosomes, crossover points in the angle moved by joint J_2 (*i.e.,* in ϕ) are shown below.

{17.36, 2|6.2|6, 5.26}

{17.36, 3|0.0|9, 5.26}

(bar shows crossover points)

After crossover, we get the following new offsprings:

Chromosome 5 {17.36, 20.06, 5.26}

Chromosome 6 {17.36, 36.29, 5.26}

d. Mutation: Here, we have considered value encoding. Hence, mutation is ignored.

e. Accepting: New offsprings are placed in the new population.

Step-4:

[Replace]: As a result of crossover, the angle moved by joint J_2 has been modified and hence an error appears in obtaining final destination point. New movement of angles for joint J_1 and joint J_2 are obtained using newly obtained chromosomes.

Hence, $\phi = 20.06^{\circ}$ and 36.29°. The values for θ and ψ are being obtained.

Values of other angle based on inverse kinematics are obtained using computer program as given below.

$\theta = 17.36^{\circ}$, $\phi = 20.06^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 36.29^{\circ}$, $\psi = 5.26^{\circ}$

New population and fitness value of chromosomes obtained after first run as shown in Table 5.50.

Table 5.50: Fitness of chromosomes obtained after first run

Chromosome No.	Chromosome (angles in degree)	Fitness value
7.	{17.36, 20.06, 84.69}	53.67
8.	{17.36, 20.06, 5.26}	23.25
9.	{17.36, 36.29, 84.69}	60.45
10.	{17.36, 36.29, 5.26}	30.02

Step-5:

[Test]: It is clear from the above, newly obtained population, that the chromosomes in this population have much better fitness values than the previous one. So, this population is selected for the second run.

Second run

Step-2:

[Start]: We have already determined the fitness values in the previous run. The new population are arranged according to their ranking (in descending order) in Table 5.51.

Table 5.51: Ranking of chromosomes obtained after first run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 8 {17.36, 20.06, 5.26}	23.25
2	Chromosome 10{17.36, 36.29, 5.26}	30.02
3	Chromosome 7{17.36, 20.06, 84.69}	53.67
4	Chromosome 9{17.36, 36.29, 84.69}	60.45

Step-3:

[New population]**:**

a. Encoding: Chromosomes are already in value encoding form.

b. Selection: According to rank selection, two chromosomes having highest ranks are chosen. Two selected chromosomes are as mentioned below.

Ranking	Chromosome	Fitness value
1	Chromosome 8 {17.36, 20.06, 5.26}	23.25
2	Chromosome10 {17.36, 36.29, 5.26}	30.02

c. Crossover: While crossover on selected chromosomes, crossover points in the angle for joint J_2 (*i.e.,* in ϕ) are considered as shown below.

{17.36, 2|0.0|6, 5.26}

{17.36, 3|6.2|9, 5.26}

(bar shows crossover points)

After crossover, we get the following new offsprings:

Chromosome 11 {17.36, 26.26, 5.26}

Chromosome 12 {17.36, 30.09, 5.26}

d. Mutation: We have taken value encoding in our problem. Hence mutation is ignored.

e. Accepting: New offspring are placed in the new population.

Step-4:

[Replace]: As a result of crossover, the angle of joint J_2 has been modified and hence an error appears in obtaining final destination point. New angles are calculated for joint J_1 and joint J_2.

Here, $\phi = 26.26^{\circ}$ and 30.09°. The values for angle θ and ψ are being obtained.

Values of other angles based on inverse kinematics have been obtained using a computer program as given below.

$\theta = 17.36^{\circ}$, $\phi = 26.26^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 30.09^{\circ}$, $\psi = 5.26^{\circ}$

Step-2:

New population and fitness values of chromosomes obtained after second run as given in Table 5.52.

Table 5.52: Fitness value of chromosomes obtained after second run

Chromosome No.	Chromosome (angles in degree)	Fitness value
13.	{17.36, 26.26, 84.69}	56.25
14.	{17.36, 26.26, 5.26}	25.84
15.	{17.36, 30.09, 84.69}	57.85
16.	{17.36, 30.09, 5.26}	27.43

Hence, the chromosomes according to their ranking (in descending order) as shown in Table 5.53.

Table 5.53: Ranking of chromosomes obtained after second run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 14 {17.36, 26.26, 5.26}	25.84
2	Chromosome 16{17.36, 30.09, 5.26}	27.43
3	Chromosome 13 {17.36, 26.26, 84.69}	56.25
4	Chromosome 15 {17.36, 30.09, 84.69}	57.85

Step-5:

[Test]: As it is clear from the above newly obtained population that, the chromosomes in this population have not good fitness values than the previous one. So the previous whole population obtained in the first run is selected as final population.

Step-6:

[Loop]: Go to step 2 [Fitness]

Third run

Step-2:

[Fitness]: We have already determined fitness in the previously in second run. The ranking of chromosomes obtained after first run (in descending order) as shown in Table 5.51.

Step-3:

[New population]:

a. Encoding: Now as the magnitudes of the angles are less in the second generation, binary encoding is performed (refer to Table 5.51). The population in decimal form is given as:

Chromosome 8 {17.36, 20.06, 5.26}

Chromosome 10 {17.36, 36.29, 5.26}

Chromosome 7 {17.36, 20.06, 84.69}

Chromosome 9 {17.36, 36.29, 5.26}

After binary encoding, we have

Chromosome 8 {10001.100100,10100.110, 101.11010}

Chromosome 10 {10001.100100, 100100.11101, 101.11010}

Chromosome 7 {10001.100100, 10100.110, 1010100.1000101}

Chromosome 9 {10001.100100, 100100.11101, 101.11010}

b. Selection: According to rank selection we had already choose two chromosomes having highest ranks in second run. Now we will choose next two chromosomes (chromosomes 10 and 7) in the increasing order of fitness.

Next, two chromosomes (chromosomes 10 and 7) are chosen in the increasing order of fitness.

Ranking	Chromosome	Fitness value
2	{10001.100100, 100100.11101, 101.11010}	30.02
3	{10001.100100, 10100.110, 1010100.1000101}	53.67

c. Crossover: While crossover on the next two chromosomes, we can make crossover points in the angle moved by joint J_2 and joint J_4 (*i.e.*, in ϕ and ψ) as shown below.

Chromosome 10 {10001.100100, 10010|0.11101, 101|.11010}

Chromosome 7 {10001.100100, 10100|.110, 101|0100.1000101}

(bar shows crossover points)

After crossover, the following new offsprings are obtained:

Chromosome 17 {10001.100100, 10010 .110, 1010100.1000101}

Chromosome 18 {10001.100100, 10100 0.11101, 101.11010}

The above chromosomes are written as in value encoded form.

Chromosome 17 {17.36, 18.30, 84.81}

Chromosome 18 {17.36, 40.23, 06.11}

From the above obtained chromosomes, it is judged that there is no such good change in the angles in angle ψ, but the angles ϕ have changed largely. So, we will consider angles of link ϕ for further run.

d. Mutation: With a mutation probability, new offspring are mutated at each locus. When mutation with binary encoding is done, selected bits are inverted. Arrow indicates mutated bits.

Before Mutation:

Chromosome 17 {10001.100100, 10010 .110, 1010100.1000101}

Chromosome 18 {10001.100100, 10100 0.11101, 101.11010}

After Mutation:

Chromosome 19 {10001.100100, 10110 .110, 1010100.1000101}

Chromosome 20 {10001.100100, 10000 0.11101, 101.11010}

e. Accepting: New offsrings are placed in the new population

Step-4:

[Replace]:

After the crossover, the angle ϕ has been modified by some value and hence an error appears in obtaining final destination point. These newly obtained chromosomes are used for finding other angles.

Here, $\phi = 22.30^{\circ}$ and 32.23°. The value for angles θ and ψ are being calculated.

Values of other angles based on inverse kinematics are obtained using computer programe as below.

$\theta = 17.36^{\circ}$, $\phi = 22.30^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 32.23^{\circ}$, $\psi = 5.26^{\circ}$

New population and fitness values of chromosomes obtained after third run as given in Table 5.54.

Table 5.54: Fitness value of chromosomes obtained after third run

Chromosome No.	Chromosome (angles in degree)	Fitness value
21	{17.36, 22.30, 84.69}	54.58
22	{17.36, 22.30, 5.26}	24.16
23	{17.36, 32.23, 84.69}	58.72
24	{17.36, 32.23, 5.26}	28.30

Step-5:

[Test]:

As it is clear from the above newly population, that the chromosomes in this population have not good fitness values than created in the previous first run. So, we select the population obtained in the first run as final population.

In case of 4 DOF manipulator, lowest fitness having higher ranking is in first run chromosomes. There is no alternative in this case. The optimal solutions are obtained after first run as:

$\theta = 17.36^{\circ}$, $\phi = 20.06^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 36.29^{\circ}$, $\psi = 5.26^{\circ}$ ----(5.34)

5.8 Summary

In this chapter, initially we have discussed the role fuzzy logic to evaluate the weightages of the variables. Then the role of genetic algorithm has been discussed for optimisation of angle movement. The role of AHP to find weightage of input variables has also been discussed. We had implemented the AHP for 2 DOF, 3 DOF and 4 DOF robotic arm manipulators to find weightage of variables. Using weightages from AHP, we have implemented genetic algorithms to optimise the movement of robotic arm manipulator for 2 DOF, 3 DOF and 4 DOF. Then, we have implemented fuzzy logic to evaluate the weightage of variables. Using variables weightages from fuzzy logic, we have implemented genetic algorithms to optimise the movement of robotic arm manipulators for 2 DOF, 3 DOF and 4 DOF. In this chapter, we have implemented fuzzy logic and genetic algorithms for finding optimal solution from possible solutions of given path problems, while achieving the target of minimum energy consumption criterion.

Chapter 6

Results and Discussions†

6.1 Preamble

Genetic algorithms and fuzzy-genetic algorithms have been extensively used to explore their capabilities to optimise robotic arm movement, while considering the optimisation criterion. GAs search global optimal solution for three constraints taken into account. Software codes have also been developed which offer off-line testing and simulation for various case studies, *i.e.,* 2 DOF, 3 DOF and 4 DOF robotic arm manipulators.

For this purpose, the inverse kinematics solutions have been considered because it provides multiple solutions for each link's angle, which are further used to generate chromosomes population to be fed into the GAs and fuzzy-GAs. The inverse kinematics, fitness value evaluation and binary encoding like tasks have been simulated. Three factors, *viz.*, movement, friction and settling time (or minimum vibration) are considered for finding the fitness function / fitness values.

---

† **The major findings of this chapter have been published in International Journal of Advances in Modeling, AMSE, France, vol. 77, issue 1, 70-77, 2008 and in International Journal of soft Computing Apllications (IJSCA), vol. 3, 69-76, 2008.**

Firstly, results using only GAs for various case studies are presented. The results using fuzzy-GAs are presented later.

6.2 Results using GAs

The results of 2 DOF, 3 DOF and 4 DOF case studies are compared and discussed using genetic algorithms. The optimal solutions have been obtained.

6.2.1 For 2 DOF Manipulators

In case of 2 DOF, solutions are obtained using inverse kinematics.

The following values for the angles of the links from Eqs. (4.7), (4.8) and (4.9) are given as:

$\theta_1 = 30.76^\circ,\ \theta_2 = 47.81^\circ$

$\theta_1 = 73.94^\circ,\ \theta_2 = -47.81^\circ$

After applying GAs the chromosomes obtained on application of inverse kinematics are shown in Table 5.20.

Table 5.20: Fitness value of chromosomes obtained after applying inverse kinematics

Chromosome No.	Chromosome (angles in degree)	Fitness function	Fitness value
1.	{30.76, 47.81}	91.97 + 142.47	234.44
2.	{30.76, 47.81}	91.97 + 142.47	234.44
3.	{73.94, 47.81}	221.08 + 142.47	363.55
4.	{73.94, 47.8}	221.08 + 142.47	363.55

In case of 2 DOF manipulator, ranking of chromosomes are same as in initial Table 5.20, so there are no change in ranking as per fitness value. There is no other alternative solution in this case. The optimal solution is given by Eq. (5.29).

$\theta_1 = 30.76^\circ,\ \theta_2 = 47.81^\circ$

and $\theta_1 = 73.94^\circ,\ \theta_2 = -47.81^\circ$

In this case, there are lesser number of inverse kinematics solutions and is quite easy to find optimal solution.

6.2.2 For 3 DOF Manipulator

In the case of 3 DOF, solutions are found using inverse kinematics.

The following values for the joint angles of the links for 3 DOF manipulator from Eqs. (4.41), (4.44), (4.47), (4.48) and (4.49) are given as:

$\theta_1 = 137.911^\circ,\ \theta_2 = -217.49^\circ,\ \theta_3 = 106.144^\circ$

and $\theta_1 = -84.781^\circ,\ \theta_2 = 37.491^\circ,\ \ \theta_3 = 73.855^\circ$

After applying GAs, the ranking of chromosomes obtained after from first run are have been shown in Table 5.24. The chromosomes obtained after third run are shown in Table 5.26.

Table 5.24: Ranking of chromosomes obtained after first run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome14 {20.84, 28.75, 76.15}	375.68
2	Chromosome12 {20.84, 36.95, 76.15}	400.11
3	Chromosome18 {40.33, 28.75, 76.15}	433.96
4	Chromosome16 {40.33, 36.95, 76.15}	458.39
5	Chromosome13 {20.84, 28.75, 103.84}	458.47
6	Chromosome11 {20.84, 36.95, 103.84}	482.90
7	Chromosome17 {40.33, 28.75, 103.84}	516.75
8	Chromosome15 {40.33, 36.95, 103.84}	541.18

Table 5.26: Fitness value of chromosomes obtained after third run

Chromosome No.	Chromosome (angles in degree)	Fitness value
33.	{137.91, 16.75, 84.40}	713.43
34.	{137.91, 16.75, 172.30}	977.44
35.	{137.91, 60.95, 84.40}	845.14
36.	{137.91, 60.95, 172.30}	1109.16
37.	{84.78, 16.75, 84.40}	555.57
38.	{84.78, 16.75, 172.30}	818.58
39.	{84.78, 60.95, 84.40}	686.28
40.	{84.78, 60.95, 172.30}	950.30

By comparing the fitness value from Table 5.24 and Table 5.26, it is concluded that there is no improvement in fitness value. Table 5.24 is considered as final solution. Selecting the values of angles having lowest fitness value, the optimal solution for 3 DOF manipulator is given by Eq. (5.30).

$\theta_1 = -20.84^\circ,\ \theta_2 = -36.95^\circ,\ \ \theta_3 = 103.84^\circ$

and $\theta_1 = -40.33^\circ, \theta_2 = -28.75^\circ, \theta_3 = 76.15^\circ$

6.2.3 For 4 DOF Manipulator

The angles of the links are obtained using inverse kinematics. From Eqs. (4.83), (4.90), (4.92), (4.99) and (4.101), the solutions are given as:

$\theta = 17.36°$, $\phi = 30.09^{\circ}$, $\psi = 84.694^{\circ}$

and $\theta = 17.36°$, $\phi = 26.26^{\circ}$, $\psi = 5.2696^{\circ}$

On applying GAs, the ranking chromosomes obtained after first run are shown in Table 5.30. Also, the chromosomes obtained after third run are shown in Table 5.33.

Table 5.30: Ranking of chromosomes obtained after first run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 8{17.36, 20.06, 5.26}	127.45
2	Chromosome 10{17.36, 36.29, 5.26}	175.81
3	Chromosome 7{17.36, 20.06, 84.69}	364.92
4	Chromosome 9{17.36, 36.29, 5.26}	413.29

Table 5.33: Fitness value of chromosomes obtained after third run

Chromosome No.	Chromosome (angles in degree)	Fitness value
21	{17.36, 22.30, 84.69}	371.60
22	{17.36, 22.30, 5.26}	134.13
23	{17.36, 32.23, 84.69}	401.19
24	{17.36, 32.23, 5.26}	163.72

By comparing the fitness value from Table 5.30 and Table 5.33, it is found that there is no improvement in fitness value. So we have considered Table 5.30 as final solution. We selects the values of angles having lowest fitness value. The optimal solution is given by Eq. (5.31).

$\theta = 17.36^{\circ}$, $\phi = 20.06^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 36.29^{\circ}$, $\psi = 5.26^{\circ}$

6.3 Results with Fuzzy-GAs

Here, results using fuzzy-GAs are discussed for 2 DOF, 3 DOF and 4 DOF robotic arm manipulators. The optimal solutions have also been obtained.

6.3.1 For 2 DOF Manipulator

The following values for the angles of the links are obtained from Eqs. (4.7), (4.8), and (4.9).

$\theta_1 = 30.76^\circ$, $\theta_2 = 47.81^\circ$

and $\theta_1 = 73.94^\circ$ $\theta_2 = -47.81^\circ$

After applying fuzzy-GAs, the chromosomes obtained after using inverse kinematics are shown in Table 5.19.

Table 5.41: Fitness value of chromosomes obtained after applying inverse kinematics

Chromosome No.	Chromosome	Fitness function	Fitness value
1.	{30.76, 47.81}	18.37 + 22.80	42.17
2.	{30.76, 47.81}	18.37 + 22.80	42.17
3.	{73.94, 47.81}	45.03+ 22.80	67.83
4.	{73.94, 47.81}	57.82 + 22.80	67.83

In case of 2 DOF manipulator, ranking of chromosomes are same as in initial Table 5.41, so there are no change in ranking as per fitness value. The chromosome having lowest fitness value will have higher ranking. The optimal solution is given by Eq. (5.32)

$\theta_1 = 30.76^\circ$, $\theta_2 = 47.81^\circ$

and $\theta_1 = 73.94^\circ$, $\theta_2 = -47.81^\circ$

6.3.2 For 3 DOF Manipulator

The following values, for the joint angles of the links for 3 DOF manipulator are calculated from Eqs. (4.41), (4.44), (4.47), (4.48) and (4.49).

$\theta_1 = 137.911^\circ$, $\theta_2 = -217.49^\circ$, $\theta_3 = 106.144^\circ$

and $\theta_1 = -84.781^\circ$, $\theta_2 = 37.491^\circ$, $\theta_3 = 73.855^\circ$

On applying fuzzy-GAs, the ranking of chromosomes obtained after first run are shown in Table 5.45. The chromosomes obtained after third run are shown in Table 5.47.

Table 5.45: Ranking of chromosomes obtained after first run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome14 {20.84, 28.75, 76.15}	59.79
2	Chromosome12 {20.84, 36.95, 76.15}	63.21
3	Chromosome13 {20.84, 28.75, 103.84}	71.15
4	Chromosome11 {20.84, 36.95, 103.84}	74.67
5	Chromosome18 {40.33, 28.75, 76.15}	75.03
6	Chromosome16 {40.33, 36.95, 76.15}	78.45
7	Chromosome17 {40.33, 28.75, 103.84}	86.49
8	Chromosome15 {40.33, 36.95, 103.84}	89.91

Table 5.47: Fitness value of chromosomes obtained after third run

Chromosome No.	Chromosome (angles in degree)	Fitness value
33.	{137.97, 20.10, 91.29}	154.07
34.	{137.97, 20.10, 163.51}	183.97
35.	{137.97, 52.12, 91.29}	167.42
36.	{137.97, 52.12, 163.51}	197.32
37.	{84.81, 20.10, 91.29}	112.50
38.	{84.78, 20.10, 163.51}	142.40
39.	{84.78, 52.12, 91.29}	125.85
40.	{84.78, 52.12, 163.51}	155.75

From Tables 5.46 and 5.48, it is clear that the chromosomes obtained after third run in this population have not good fitness values than the previous one. So, the population obtained after first run is the final population having lowest fitness values. The optimal solution is given by Eq. (5.33)

$\theta_1 = -20.84^\circ,\ \theta_2 = -36.95^\circ, \theta_3 = 103.84^\circ$

and $\theta_1 = -40.33^\circ, \theta_2 = -28.75^\circ,\ \theta_3 = 76.15^\circ$

6.3.3 For 4 DOF manipulator

The values of θ, ϕ and ψ have been obtained for 4 degree-of-freedom manipulator from Eqs. (4.83), (4.90), (4.92), (4.99) and (4.101).

$\theta = 17.36^\circ,\ \phi = 30.09^\circ,\ \psi = 84.69^\circ$

and $\theta = 17.36^\circ,\ \phi = 26.26^\circ,\ \psi = 5.26^\circ$

On applying fzzy-GAs, ranking of the chromosomes obtained after first are shown in Table 5.51. The chromosomes obtained after third run are shown in Table 5.54.

Table 5.51: Ranking of chromosomes obtained after first run

Ranking	Chromosome (angles in degree)	Fitness value
1	Chromosome 8 {17.36, 20.06, 5.26}	23.25
2	Chromosome 10{17.36, 36.29, 5.26}	30.02
3	Chromosome 7{17.36, 20.06, 84.69}	53.67
4	Chromosome 9{17.36, 36.29, 84.69}	60.45

Table 5.54: Fitness value of chromosomes obtained after third run

Chromosome No.	Chromosome (angles in degree)	Fitness value
21	{17.36, 22.30, 84.69}	54.58
22	{17.36, 22.30, 5.26}	24.16
23	{17.36, 32.23, 84.69}	58.72
24	{17.36, 32.23, 5.26}	28.30

As it is clear from the above obtained population that, the chromosomes in this population have not good fitness values than the population obtained after first run. So, the whole population of first run is taken as final population. The optimal solution is given by Eq. (5.34).

$\theta = 17.36^{\circ}$, $\phi = 20.06^{\circ}$, $\psi = 84.69^{\circ}$

and $\theta = 17.36^{\circ}$, $\phi = 36.29^{\circ}$, $\psi = 5.26^{\circ}$

6.4 Discussions

From the above results, it is observed that GAs and fuzzy-GAs have been implemented effectively on 2, 3 and 4 DOF robotic arm manipulators. By using fuzzy-GAs, fitness value has been reduced as compared with that for only GAs. The success rate of GAs depends upon crossover and mutation. In this work, AHP has been selected in place of conventional techniques like roulette wheel, rank selection, steady state selection, *etc.,* AHP is a pragmatic method of mathematically ranking the various available alternatives, which evolve during the execution of GAs in various test runs / iterations along with approximate reasoning.

As apparent from the simulated test results, given in the various tables, the GAs converge quickly in two runs. However, the convergence may even take more test runs and thus,

imperiling the controllability aspect of three degree-of-freedom and four degree-of-freedom manipulator movements.

6.5 Comparison of attributes with fitness

Comparison of attributes in three degree-of-freedom and four degree-of-freedom robotic arm movements with respect to cost function are shown in Table 6.1, 6.2 and 6.3. Their comparison graphs are given in Fig. 6.1, 6.2 and 6.3. Here, λ_1, λ_2 and λ_3 represent the movement, friction and settling time respectively.

Table 6.1: Comparison of movements in 3 DOF and 4 DOF

λ_1	3 DOF	4 DOF
0.5	375	84
1	417	106
1.5	245	104
2	355	113
2.5	352	120
3	376	126

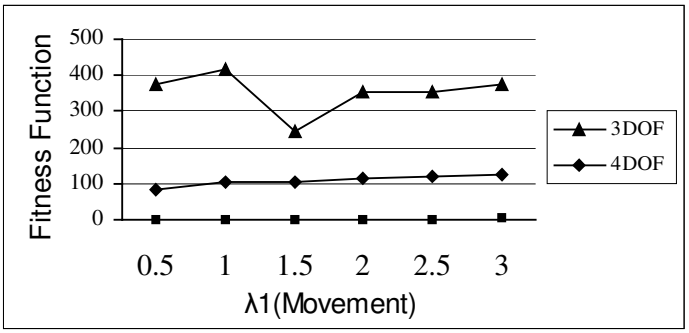


Fig. 6.1: Comparison of movements in 3 DOF and 4DOF

Table 6.2: Comparison of friction in 3 DOF and 4 DOF

λ_2	3 DOF	4 DOF
0.5	413	84
1	319	92
1.5	333	101
2	348	110
2.5	362	118
3	376	127

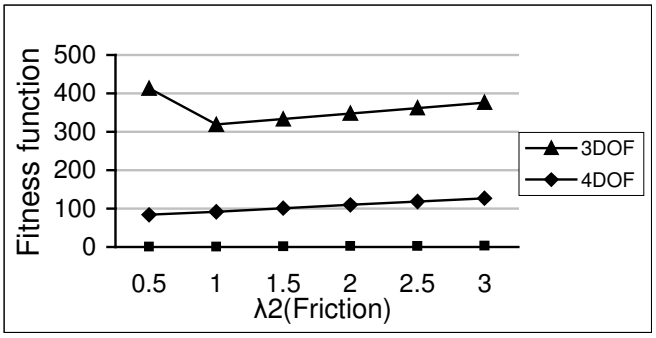


Fig. 6.2: Comparison of friction in 3DOF and 4DOF

Table 6.3: Comparison of settling time in 3DOF and 4 DOF

λ_3	3 DOF	4 DOF
0.5	175	114
1	222	116
1.5	260	119
2	297	122
2.5	336	129
3	376	132

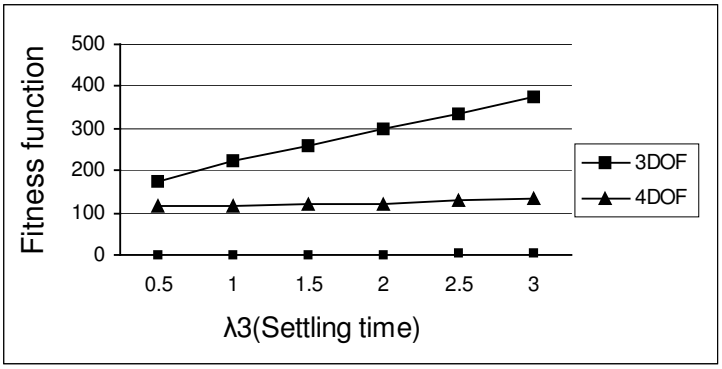


Fig. 6.3: Comparison of settling time in 3DOF and 4 DOF

6.5.1 Discussions

From Table 6.1 and Fig. 6.1, at 1.5 the value of movement (λ_1) and fitness are minimum for 3 DOF. It equals 245. At 0.5 the value of movement (λ_1) and fitness are minimum for 2 DOF. It equals 84.

Similarly, from Table 6.2, and Fig. 6.2, at 0.1 the value of friction (λ_2) and fitness are minimum for 3 DOF. It equals 319. At 0.5 the value of movement (λ_2) and fitness are minimum for 2 DOF. It equals 84.

Similarly, from Table 6.3 and Fig. 6.3 at 0.5 at the value of settling time (λ_3) and fitness are minimum for 3 DOF. It equals 175. At 0.5 the value of settling time (λ_3) and fitness are minimum for 2 DOF. It equals 114.

6.6 Summary

In this chapter, we have discussed the results obtained using only GAs for 2, 3 and 4 DOF manipulators. The optimal results are obtained in each case. These results are discussed using fuzzy-GAs for 2, 3 and 4 DOF manipulators. Results also indicate the improvements using fuzzy-GAs. Decreased fitness value means more optimal value with hybrid system (fuzzy-GAs) than only GAs. The dependence of optimised variables on fitness function is evident. Comparison of attributes movement (λ_1), friction (λ_2) and settling time (λ_3) effect on 3 DOF and 4 DOF are presented. The optimal values of variables are evaluated using graphs and with 4 DOF manipulator showing lower fitness values.

Chapter 7

Conclusions and Future Scope

7.1 Conclusions

It is concluded that in the presence of several optimisation attributes for a physical system of higher order manipulators as in this case, the GAs is a practical way of finding the globally optimal solutions.

Uncertainties like movement, friction and settling time in robotic arm movements have been compensated using genetic algorithms. Augmentation of GAs with AHP reduces any chance of GAs getting converted into a random search method as evident from the results obtained in this work. It is also concluded from the results, that GAs shows better optimal arm movement as degree-of-freedom is increased. As evident, higher order robotic arm (four degree-of-freedom) movements with respect to lower order (three degree-of-freedom) are more linear. Also, there is significant improvement in optimisation attributes of robotic arm as shown in the results. There is 69% improvement in movement, 72% improvement in friction and 56% in settling time.

Finally, it is concluded that application of genetic algorithms for the optimisation of robotic arm movement in real world brings out major changes and also helps the users to minimise the losses to a greater extent.

The optimal movement of robotic arms for two degree-of-freedom using inverse kinematics has been studied, analysed and implemented. An optimisation method based on the genetic algorithms is proposed. In the developed genetic algorithms, in order to obtain the optimal angular displacements for the robotic arms in the whole work space, elitism has been retained from the previous generation to the next. Simulations, testing and comparisons have been carried out. GAs do not require complete knowledge of system. It is concluded that GAs are quite practical and effective method for achieving optimisation of robotic arm angular displacements.

This research work on flexible link robotic arms focuses on how to solve optimal movement in slow and fast mode control, respectively. For slow mode control, the problem has been dealt with previously by soft computing tools in which some parameters are designed manually. As a result, system performances are often tiresome and intractable. This research work has introduced a scheme to improve the system performance by applying genetic algorithms.

A common approach to the control of flexible link manipulator is to design separate controller for the slow and fast modes. Fuzzy logic has shown to have great potential for realising a slow mode controller, but the need for human input in the form of engineering insight and a tendency towards inaccuracy in the controller are serious shortcomings. This work has described techniques, which avoid the need for engineering insight and increases accuracy by using genetic algorithms in order to optimise fuzzy logic controller for robot arms. The results as obtained in this research work show the effectiveness and practicality. Best results have been obtained from fuzzy-GA hybrid scheme.

The essential problem in controlling robots is to make the manipulator reach a destination coordinate in space. This work has implemented fuzzy-GAs for following the destination for two, three and four degree-of-freedom robotic arms. Numerical simulations using the dynamic model of two, three and four degree-of-freedom robotic arm show the effectiveness of the developed approach. The results presented in this research work emphasise that a satisfactory precision is achieved using fuzzy-GAs hybrid algorithms.

On the other hand, it also requires very less knowledge of the system description in order to apply this technique over other artificial intelligence techniques like artificial neural

networks (ANN) and fuzzy logic (FL). This algorithm can be easily extended to more higher degrees-of-freedom robotic arms with little modifications.

7.2 Future Scope

For the future research, improvement in robot path planning can be done so as to develop autonomous robots. With increase in degree-of-freedom and higher dimensional work spaces, complexity increases. Thus, it becomes very difficult to find the solutions of path planning problems with classical or mathematical techniques. It is also very difficult to develop accurate mathematical models. So by implementing GAs and fuzzy-GAs, path planning problems can be solved efficiently. This research work can be extended to more degrees-of-freedoms. While optimising, we have also compensated the non-deterministic parameters like friction, settling time and movements. These parameters have been successfully compensated and minimised for their effects on the performance of robotic arm movements in order to achieve the targets quite effectively. There may be more number of non-deterministic parameters, which may be compensated.

Due to the strong non-linear characteristic and parameter variation in real environments, optimal movement of a robot arm is difficult. Fuzzy-GAs perform it well but the only problem faced with these controllers is that genetic algorithms do not rely on the designer and also they involve substantial processing time. Tuning of fuzzy-GAs PID controller with fuzzy PID tuner in which sequential quadratic programming algorithms can be further investigated.

GAs and fuzzy-GAs carry out important role in path planning. It is evident that this research work contributes well to the ultimate goal of the autonomous robots and provides great motivation to the other researches in this area.

REFERENCES

1. Aarno D., Kragic D., Christensen H. I., 2004. Artificial potential biased probabilistic roadmap method. IEEE International Conference on Robotics and Automation, New Orleans, LA, USA, 1, 461–466.
2. Ahuactzin J. M., Gupta K. K., 1999. The kinematic roadmap: a motion planning based global approach for inverse kinematics of redundant robots. IEEE Transactions on Robotics and Automation, 15(4), 653-669.
3. Ahuactzin J. M., Portilla. A., 2000. A basic algorithm and data structures for sensor-based path planning in unknown environments. IEEE/RSJ International Conference on Intelligent Robots and Systems, Takamatsu, Japan, 2, 903–908.
4. Ahuactzin, J. M., Talbi E. G., Bessiere P., Mazer E., 1992. Using genetic algorithms for robot motion planning. Proceedings of the 10th European Conference on Artificial Intelligence, Vienna, Austria, 671-675.
5. Alam M. S. and Tokhi M. O., 2008. Hybrid fuzzy logic control with genetic optimisation for a single-link flexible manipulator. Elsevier Engineering Applications of Artificial Intelligence 21(6), 858-873.
6. Alexei Zakharov and Sindor Halhsz, 1999. Genetic algorithms based identification method for a robot arm. IEEE ISIE'99 - Bled, Slovenia, 1014-1019.
7. Antonelli G., Chiaverini S., Fusco G., 2003. A new on-line algorithm for inverse kinematics of robot manipulators ensuring path tracking capability under joint limits. IEEE Transactions on Robotics and Automation 19(1),162- 167.
8. Appuu Kuttan K. K., 2007. Robotics. I. K. International, India.
9. Arteaga M.A. and Kelly R., 2004. Robot control without velocity measurements: new theory and experimental results. IEEE Transactions on Robotics and Automation 20(2), 297- 308.
10. Ata A. A. and Myo T. R., 2006. Collision-free trajectory planning for manipulators using generalised pattern search. International Journal of simulation and modelling 5(4), 145-154.

11. Autere A. and Lehtinen J., 1997. Robot motion planning by a hierarchical search on a modified discretized configuration space. In IEEE/RSJ International Conference on Intelligent Robots and Systems Grenoble, France 2, 1208–1213.

12. Bagchi A. and Hatwal H., 1990. A solution strategy for collsion avoidance of multiple bodies moving on a plane using fuzzy logic. Proceedings of the 29th IEEE Conference on Decision and Control, 452 – 461.

13. Bansal R. C., Kothari D. P., Bhatti T. S., 2003. Artificial intelligence techniques for reactive power/voltage control in power systems: A review. International Journal of Power and Energy Systems 23(2), 81-89.

14. Barai Ranjit Kumar and Nonami Kenzo, 2007. Optimal two-degree-of-freedom fuzzy control for locomotion control of a hydraulically actuated hexapod robot. Elsevier International Journal of Information Sciences 177, 1892–1915.

15. Barraquand J. and Latombe J.C., 1991. Robot motion planning: A distributed representation algorithm. International Journal of Robotics Research 10(6), 628–49.

16. Belarbi K. and Titel F., 2000. Genetic algorithm for the design of a class of fuzzy controllers: an alternative approach. IEEE Transactions on Fuzzy Systems 8(4), 398-405.

17. Bicchi A., Casalino G., Santilli. C., 1995. Planning shortest bounded curvature paths for a class of nonholonomic vehicles among obstacles. IEEE International Conference on Robotics and Automation, Nagoya, Japan, 2, 1349–1354.

18. Bohlin R. and Kavraki. L. E. , 2000. Path planning using lazy PRM. In IEEE International Conference on Robotics and Automation, San Francisco, CA, USA, 1, 521–528.

19. Brooks R., 1983. Solving the find-path problem by good representation of free space. IEEE Transactions on Systems, Man and Cybernetics 13(3), 190-197.

20. Brunn P., 1996. Robot collision avoidance. Industrial Robot, 23(1), 27-33.

21. Cameron S., 1994. Obstacle avoidance and path planning. International Journal of Industrial Robot 21(5), 9-14.

22. Cameron S., 1998. Motion planning and collision avoidance with complex geometry. In Proceedings of the IEEE Industrial Electronics Society, Aachen, Germany 4, 2222–2226,

23. Canny J., 1987. A new algebraic method for robot motion planning and real geometry. Proceedings of the 28th IEEE Annual Symposium on Foundations of Computer Science 39-48.

24. Chang P. H., Park K. C., Lee S., 2000. An extension to operational space for kinematically redundant manipulators: kinematics and dynamics. IEEE Transactions on Robotics and Automation 16(5), 592-596.

25. Chen Jigien and Chao Lih-Ming, 1987. Positioning error analysis for robot manipulators with all rotary joints. IEEE Journal Of Robotics and Automation 3(6), 539-545.

26. Chiang C. K., Chung H.Y., Lin J.J., 1997. A self-learning fuzzy logic controller using genetic algorithms with reinforcements. IEEE Transactions on Fuzzy Systems 5(3), 460-467.

27. Christophe Collewet, Guylaine Rault, Stephane Quellee, Marchal P., 1998. Fuzzy adaptive controller design for the joint space control of an agriculture robot. Elsevier Fuzzy sets and systems 99(1), 1-25.

28. Chung W. K., Youm Y., Cheong J., 2004. Inverse kinematics of multilink flexible robots for high-speed applications. IEEE Transactions on Robotics and Automation 20(2), 269- 282.

29. Chung W. J., Chung W. K., Youm Y., 1991. Inverse kinematics of planar redundant manipulators using virtual links and displacement distribution schemes. IEEE International Conference on Robotics and Automation 926 - 932.

30. Clarke Roger, 1993. Asimov's Laws for Robotics: Implications for Information Technology. IEEE Computer, 53-61.

31. Craig Eldershaw and Stephen Cameron, 2000. Genetic algorithms to solve the motion planning problem. Journal of Universal Computer Science 6(4), 422-432.

32. Craig J. J., 2004. Introduction to Robotic Mechanics and Control. Pearson Education, Singapore.

33. Danner T. and Kavraki L. E., 2000. Randomized planning for short inspection paths. In IEEE International Conference on Robotics and Automation San Francisco, CA, USA, 2, 971–976.

34. Deb S. R., 2002. Robotics Technology and Flexible Automation. Tata Mc-Graw Hill.

35. Devendra P. Garga and Manish Kumar, 2002. Optimization techniques applied to multiple manipulators for path planning and torque minimization. Elsevier Engineering Applications of Artificial Intelligence 15(3-4), 241–252.

36. Dolinsky Jens Uwe, 2001. The development of a genetic programming method for kinematic robot calibration. Ph.D. thesis. Liverpool John Moores University.

37. Dongbing Gu, Huosheng Hu, Libor Spacek, 2003. Learning fuzzy logic controller for reactive robot behaviours. Proceedings of IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Kobe, Japan, 20-24.

38. Doyle A. B., 1995. Algorithms and computational techniques for robot path planning. Ph.D. thesis submitted in School of Electronic Engineering and Computer Systems, University of Wales, Bangor, United Kingdom.

39. Edward T. Lee, 1995. Applying fuzzy logic to robot navigation. Journal of Kybernetes 24(6), 38–43.

40. Euntai Kim, Ji M. P. S., Park M., 1997. A new approach to fuzzy modeling. IEEE Transactions on Fuzzy Systems 5(3), 328-337.

41. Fahimi F., Ashrafiuon H., Nataraj C., 2002. An improved inverse kinematic and velocity solution for spatial hyper-redundant robots. IEEE Transactions on Robotics and Automation 18(1), 103-107.

42. Feddema J. T., Lewis C., Schoenwald D. A., 2002. Decentralized control of cooperative robotic vehicles: theory and application. EEE Transactions on Robotics and Automation 18(5), 852- 864.

43. Feliu V., Somolinos J. A., Garcia A., 2003. Inverse dynamics based control system for a three-degree-of-freedom flexible arm. IEEE Transactions on Robotics and Automation 19(6), 1007-1014.

44. Ferrari C. and Canny J., 1992. Planning optimal grasps. IEEE International Conference on Robotics and Automation, Nice, France, 3, 2290–2295.

45. Fogel D. B., 1997. Evolutionary computation: A New Transactions."IEEE Transactions on Evolutionary Computation, 1(1), 1-2.

46. Fonseca C. M. and Fleming P. J., 1993. Genetic algorithms for multi-objective optimization: formulation, discussion and generalization, Fifth International Conference on Genetic Algorithms 416–423.

47. Fonseca C.M. and Fleming P. J., 1995. An overview of evolutionary algorithms in multi-objective optimization, Journal of Evolutonary . Computation. 3(1), 1–16.

48. Fred E. Sistler, 1987. Robotics and intelligent machines in agriculture. IEEE Journal of Robotics and Automation 3(1), 3-6.

49. Fu F .Yhl C., 2000. Intelligent robot deburring using adaptive fuzzy hybrid position/force control. IEEE Transactions on Robotics and Automation 16(4), 325-335.

50. Fu K.S., Gonzalez R. C., Lee C. S. G., 1987. Robotics: control, sensing, vision, and intelligence. McGraw-Hill International Editions, Singapore.

51. Fukao T., Nakagawa H., Adachi N., 2000. Adaptive tracking control of a nonholonomic mobile robot. IEEE Transactions on Robotics and Automation 16(5), 609-615.

52. Galantucci L. M., Percoco G., Spina R., 2004. Assembly and disassembly planning by using fuzzy logic & genetic algorithms. International Journal of Advanced Robotic Systems 1(2), 67–74.

53. Galicki M., 2000. Time-optimal controls of kinematically redundant manipulators with geometric constraints. IEEE Transactions on Robotics and Automation 16(1), 89-93.

54. Galicki M., 1992. Optimal planning of a collision-free trajectory of redundant manipulators. The International Journal of Robotics Research, 11(6), 549-559.

55. Ge S.S. and Cui Y. J., 2000. New potential functions for mobile robot path planning. IEEE Transactions on Robotics and Automation 16(5), 615-620.

56. Gemeinder M. and Gerke M., 2003. GA-based path planning for mobile robot systems employing an active search algorithm. Elsevier Applied Soft Computing 3(2), 149-158.
57. Gillespie R. B., Colgate J. E., Peshkin M. A., 2001. A general framework for robot control. IEEE Transactions on Robotics and Automation 17(4), 391-401.
58. Goldberg, D. E., 2007. Genetic algorithms in search, optimization and machine learning. Pearson Education, India.
59. Green A. and Sasiadek J. Z., 2001. Fuzzy and optimal control of a two-link flexible manipulator. IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 1169–1174.
60. Green A. and Sasiadek J. Z., 2004. Dynamics and trajectory tracking control of a two-link robot manipulator. Journal of Vibration Control **10** (10), 1415–1440.
61. Groom K. N., Maciejewski A. A., Balakrishnan V., 1999. Real-time failure-tolerant control of kinematically redundant manipulators. IEEE Transactions on Robotics and Automation 15(6), 1109-1115.
62. Groover M. P., Weiss M., Nagel R. N., Odrey N. G., 1986. Industrial robotics technology, programming, and applications. McGraw-Hill International Edition, Singapore.
63. Guo Tong-ying, Dao-kui Q. U., Dong Zai-li, 2004. Research of path planning for polishing robot based on improved genetic algorithm. IEEE International Conference on Robotics and Biomimetics Aug. 22 - 26, Shenyang, China, 334-338.
64. Han Y., 2004. Simultaneous translational and rotational tracking in dynamic environments: theoretical and practical viewpoints. IEEE Transactions on Robotics and Automation 20(2), 309-318.
65. Handley S. G., 1993. The genetic planner: The automatic generation of plans for a mobile robot via genetic programming. In Proceedings of IEEE International Symposium on Intelligent Control 190-195.
66. Hemami A. and Cheng R. M. H.**,** 1992. A preliminary step for path tracking of coordinated robot arms based on kinematics. International Journal of Robotics Research, Sage publications 11(3), 185-195.
67. Helguera C. and Zegloul S., 2000. A local-based method for manipulators path planning in heavy cluttered environments. IEEE International Confernce on Robotics and Automation, San Francisco, CA, USA, 3467–3472.

68. Hemami A., 1986. Kinematics of two-arm robots. IEEE Journal of Robotics and Automation 2(4), 225-228.
69. Hernando M. and Gambao E., 2002. Visibility analysis and genetic algorithms for fast robot motion planning. In IEEE/RSJ International Conference on Intelligent Robots and Systems, EPFL, Lausanne, Switzerland, 3, 2413–2418
70. Ho H. F., Wong Y. K., Rad A. B., 2007. Robust fuzzy tracking control for robotic manipulators. Simulation Modelling Practice and Theory 15(7), 801-816.
71. Homaifar A. and McCormick E., 1995. Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms. IEEE Transactions on Fuzzy Systems 3(2), 129-139.
72. Hocaoglu C. and Sanderson A. C., 1998. Multi-dimensional path planning using evolutionary computation. In Proceedings of IEEE World Congress on Computational Intelligence An- chorage, Alaska, USA, 165-170.
73. Hsu D., Latombe J. C., Sorkin S., 1999. Placing a robot manipulator amid obstacles for optimized execution. In IEEE International Symposium on Assembly and Task Planning, Porto, Portugal, 280–285.
74. Hwang Y. K. and Ahuja N., 1992. Gross motion planning-a survey. ACM Computing Surveys 24(3), 219–291.
75. Ioannidis S., Tsourveloudis N., Valavanis K., 2004. Fuzzy supervisory control of manufacturing systems. IEEE Transactions on Robotics and Automation 20(3), 379- 389.
76. Isto P., 1996. Path planning by multiheuristic search via subgoals. Proc. 27th Int. Symposium on Industrial Robots, 712–726.
77. Jakopec M., Rodriguez Y., Baena F., Harris S. J., Gomes P., Cobb J., Davies B. L., 2003. The hands-on orthopaedic robot "acrobot": Early clinical trials of total knee replacement surgery. IEEE Transactions on Robotics and Automation 19(5), 902-911.
78. Johansen T. A., 1994. Fuzzy model based control: stability, robustness, and performance issues. IEEE Transactions on Fuzzy Systems 2(3), 221-234.
79. Johnson J. A. and Smartt H. B., 1995. Advantages of an alternative form of fuzzy logic. IEEE Transactions on Fuzzy Systems 3(2), 149-157.

80. Jones L. Joseph and Lozano-P´erez T., 1990. Planning two-fingered grasps for pick-and-place operations on polyhedra. IEEE International Conference on Robotics and Automation, Cincinnati, OH, 1, 683–688.

81. Jou C. C. and Wang N. C., 1993. Training a fuzzy controller to back up an autonomous vehicle. Proceedings of the IEEE International Conference on Robotics and Automation 1, 923 - 928.

82. Kelly-III W. E., 1994. Neuro-Fuzzy control of a robotic arm. Ph.D thesis. Texas A & M University–Kingsville.

83. Kelly-III W. E., Rajab C., Mclauchlan Robert, Omar S. Iqbal, 1996. Neuro-fuzzy control of robotic arm. Proceeding of the artificial Neural networks in Engineering conference, St. Louis, MO, Nov. 10-13, 837-842.

84. Kent S., 1999. Evolutionary approaches to robot path planning. Ph.D. Thesis. Brunel University, United Kingdom.

85. Khatib O., 1986. Real-time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics Research, 5, 90-98.

86. Khoury G. M., Saad M., Kanaan. Y., Asmar C., 2004. Fuzzy PID control of a five DOF robot arm. Journal of Intelligent and Robotic Systems 40, 299–320.

87. Kim E. and Tafazoli S., 2003. A discrete-time fuzzy disturbance observer and its application to control. IEEE Transactions on Fuzzy Systems 11(3), 399- 410.

88. Kubilay K. Aydin and Erol Kocaoglan, 1995. Genetic algorithm based redundancy resolution of robot manipulators. IEEE Proceedings of ISUMA-NAFIPS '95, 322-327.

89. Kuffner Jr. J. J. and LaValle S. M., 2000. RRT-Connect: An efficient approach to single-query path planning. In IEEE International Conference on Robotics and Automation, San Francisco, CA, USA, 2, 995–1001.

90. Kwok, D. P. and Sheng F., 1994.Genetic algorithm and simulated annealing for optimal robot arm PID control. Proceedings of the First IEEE Conference on Evolutionary Computation 27-29 Jun, Orlando, FL, USA 2, 707-713.

91. Lamiraux F., Ferr´e E., Vall´ee E., 2004. Kinodynamic motion planning: Connecting exploration trees using trajectory optimisation methods. In IEEE International Conference on Robotics and Automation, New Orleans, LA, USA, 4, 3987–3992.

92. Lea R., Jani Y., Berenji H., 1990. Fuzzy logic controller with reinforcement learning for proximity operations and docking. Fifth IEEE International Symposium on Intelligent Control.

93. Lee Sung-Hee, Kim Junggon, Park F. C., Kim Munsang, Bobrow E. James, 2005. Newton-type algorithms for dynamics-based robot movement optimization. IEEE Transactions on Robotics 21(4), 657-667.

94. Leu Y. G., Wang W. Y., Lee T. T., 1999. Robust adaptive fuzzy-neural controllers for uncertain nonlinear systems. IEEE Transactions on Robotics and Automation 15(5), 805-817.

95. Lewis M. Anthony, Fagg Andrew H., Bekey George A., 1994. Genetic algorithms for gait synthesis a hexapod robot. Recent Trends in Mobile Robots, World Scientific, New Jersey, 317-331.

96. Li Z. and Sastry S. S., 1988. Task-oriented optimal grasping by multifingered robot hands. IEEE Journal of Robotics and Automation, 4(1), 32–44.

97. Liao D. X., Sung C.K., Thompson B.S., 1987. The design of flexible robotic manipulators with optimal arm geometries fabricated from composite laminates with optimal material properties. International Journal of Robotics Research, SAGE Publications 6(3) 116-130.

98. Lingelbach. F., 2004. Path planning using probabilistic cell decomposition. In IEEE International Conference on Robotics and Automation, New Orleans, LA, USA, 1, 467–472.

99. Lozano-P´erez T., 1983. Spatial planning: A configuration space approach. IEEE Transaction on computers, 32(2), 108–120.

100. Lozano P. and Erez T., 1987. A simple motion-planning algorithm for general robot manipulators. IEEE Journal of Robotics and Automation 3(3), 224–238.

101. Lozano-P´erez T., Jones J. L., Mazer E., P. Tournassoud, Handey A. Lanusse., 1987. A robot system that recognizes, plans, and manipulates. In IEEE International Conference on Robotics and Automation 4, 843–849.

102. Lu L. and Akella S., 2000. Folding cartons with fixtures: A motion planning approach. IEEE Transactions on Robotics and Automation, 16(4), 346–356.

103. Lumelsky V. J., 1987. Effect of kinematics on motion planning for planar robot arms moving amidst unknown obstacles. Journal of Robotics and Automation 3, 207-222.

104. Ma X. J., Sun Z. Q., He Y. Y., 1998. Analysis and design of fuzzy controller and fuzzy observer. IEEE Transactions on Fuzzy Systems 6(1), 41-51.

105. Mclean A. W. and Cameron S. A., 1996. The virtual springs method: Path planning and obstacle avoidance for redundant manipulators. International Journal of Robotics Research, 15(4), 300-307.

106. Messom Chris, 2002.Genetic algorithms for auto-tuning mobile robot motion control. Research. Letter Information Mathematics Science, 3, 129-134.

107. Mita T, Hyon S. H., Nam T. K., 2001. Analytical time optimal control solution for a two-link planar aerobot with initial angular momentum. IEEE Transactions on Robotics and Automation 17(3), 361-366.

108. Mittal R. K. and Nagrath I. J., 2007. Robotics and control. Tata McGraw-Hill, India.

109. Momotaz Begum, George K. I. Mann, Raymond G. Gosai, 2008. Integrated fuzzy logic and genetic algorithmic approach for simultaneous localization and mapping of mobile robots. Elsevier Applied Soft Computing 8(1), 50–165.

110. Monteiro D. C., Madrid M. K., 1999. Planning of robot trajectories with genetic algorithms. IEEE Proceedings of the first workshop on Robot Motion and Control, RoMoCo '99, Kiekrz, Poland 223–228.

111. Moudgal V. G., Kwong W. A., Passino K. M., Yurkovich S., 1995. Fuzzy learning control for a flexible-link robot. IEEE Transactions on Fuzzy Systems 3(2), 199-210.

112. Mucientes M., Moreno D. L., Bugarín A., Barro S., 2007. Design of a fuzzy controller in mobile robotics using genetic algorithms. Elsevier Applied Soft Computing 7 (2), 540-546.

113. Muller-Karger C. M., Leonell Granados Mirena A., Scarpati Lopez J. T., 2000. Hyperbolic trajectories for pick-and-place operations to elude obstacles. IEEE Transactions on Robotics and Automation 16(3), 294-300.

114. Murray R. M., Li Z., S. S. Sastry, 1994. A mathematical introduction to robotic manipulation. CRC Press.

115. Nasser Sadati and Javid Taheri, 2002. Genetic algorithm in robot path planning problem in crisp and fuzzified environments. IEEE ICiT'02, Bangkok, Thailand, 175-180.

116. Neffenger C., 1993. Fuzzy Logic in Motor Control. Fuzzy Logic '93 Proceedings A111(1-10).

117. Nguyen V. B. and Morris A. S., 2007. Genetic algorithm tuned fuzzy logic controller for a robot arm with two-link flexibility and two-joint elasticity. Springer Journal Intelligent Robot Systems 49, 3–18.

118. Niku, Saeed B., 2003. Introduction to robotics analysis, systems, applications. Prentice-Hall, India.

119. Olsen M. M. and Petersen H.G., 2001. A new method for estimating parameters of a dynamic robot model. IEEE Transactions on Robotics and Automation 17(1), 95-100.

120. Olson C. F. and Pasadena C., 2000. Probabilistic self-localization for mobile robots. IEEE Transactions on Robotics and Automation 16(1), 55-66.

121. Oriolo G., Ottavi M., Vendittelli M., 2002. Probabilistic motion planning for redundant robots along given end-effector paths. IEEE/RSJ International Conference on Intelligent Robots and Systems, EPFL, Lausanne, Switzerland, 2, 1657–1662.

122. Ostrowski J. P., 1999. Computing reduced equations for robotic systems with constraints and symmetries. IEEE Transactions on Robotics and Automation 15(1), 111-123.

123. Oya M., Su. C. Y., Katoh R., 2003. Robust adaptive motion/force tracking control of uncertain nonholonomic mechanical systems. IEEE Transactions on Robotics and Automation 19(3), 175- 181.

124. Pack D., Toussaint G., Haupt R., 1996. Robot trajectory planning using a genetic algorithm. SPIE 2824, 171–182.

125. Paredis C. J. J. and Khosla P. K., 1991. On Kinematic Design of Serial Link Manipulators. Proceedings of the 30th IEEE Conference on Decision and Control 517 - 531.

126. Ploen S. R. and Park F. C., 1999. Coordinate-invariant algorithms for robot dynamics. IEEE Transactions on Robotics and Automation 15(6), 1130-1135.

127. Qin C., Cameron S., McLean A., 1995. Towards efficient motion planning for manipulators with complex geometry. IEEE International Symposium on Assembly and Task Planning, Pittsburg, PA, USA, 207–212.

128. Rajasekaran S. and Vijayalakshmi Pai G. A., 2007. Neural networks, fuzzy logic, and genetic algorthms synthesis and applications. Prentice-Hall, India.

129. Reggiani M., Mazzoli M., Caselli S., 2002. An experimental evaluation of collision detection packages for robot motion planning. In IEEE/RSJ International Conference on Intelligent Robots and Systems, EPFL, Lausanne, Switzerland, 3, 2329–2334.

130. Rodr´ıguez M., Iglesias R., Regueiro C.V., Correa J., Barro S., **2007**. Autonomous and fast robot learning through motivation. Elsevier Robotics and Autonomous Systems 55, 735–740.

131. Rosales E. M., Gan J. Q., 2003. Forward and inverse kinematics models for a 5-dof Pioneer 2 Robot Arm. Technical report CSM-412, University of Essex.

132. Rosales E. M., Gan J. Q., Huosheng Hu, Oyama E., 2003 A hybrid appproach to inverse kinematics modelling and control of pioneer 2 robotic arms. Technical report CSM-413, University of Essex.

133. Rostami S., Hamidzadeh B., Camporese D., 2001. An optimal periodic scheduler for dual-arm robots in cluster tools with residency constraints. IEEE Transactions on Robotics and Automation 17(5), 609-618.

134. Rouvinen A. and Handroos H., 1997. Robot positioning of a flexible hydraulic manipulator utilizing genetic algorithm and neural networks. 4th Annual Conference on Mechatronics and Machine Vision in Practice (M2VIP '97), 182-186.

135. Rovatti R. and Guerrieri R., 1996. Fuzzy sets of rules for system identification. IEEE Transactions on Fuzzy Systems 4(2), 89-102.

136. Roy S. S. and Pratihar D. K., 2003. A genetic-fuzzy approach for optimal path-planning of a robotic manipulator among static obstacles. IE (I) Journal. CP 84, 15-22.

137. Rich E. and Knight K., 2004. Artificial intelligence. Tata McGraw-Hill, India.

138. Saffiotti A., 1997. The uses of fuzzy logic in autonomous robot navigation: a catalogue raisonn'e. Soft Computing Research Journal

139. Saxena A. K. and Saini Ashish, 2008. Enhanced GA-Fuzzy OPF under both normal and contingent operation states. International Journal of Electrical, Computer, and Systems Engineering 2(3), 208-216.

140. Seraji H. and Bon B., 1999. Real-time collision avoidance for position-controlled manipulators. IEEE Transactions on Robotics and Automation 15(4), 670-677.

141. Seraji H. and Howard A., 2002. Behavior-based robot navigation on challenging terrain: A fuzzy logic approach. IEEE Transactions on Robotics and Automation 18(3), 308-321.

142. Shamma J. S. and D. E. Whitney. 1987. A method for inverse robot calibration. Transactions of the ASME Journal of Dynamical Systems, Measurement and Control 109, 36-43.

143. Shibata T., Abe T., Nose M., 1997. Motion planning by genetic algorithm for a redundant manipulator using a model of criteria of skilled operators. Journal of Information Science 102, 171–186.

144. Shintaku E., 1999. Minimum energy trajectory for an underwater manipulator and its simple planning method by using a genetic algorithm. Adv. Robotics 13, 115–138.

145. Sidhu T. S., Singh H., Sachdev M. S., 1997. An artificial neural network for directional comparison relaying oftransmission lines. IEEE International conference on Developments in Power System Protection 434, 282-285.

146. Singh Chanan, Luo-X, Patton A. D., 2000. Real power transfer capability calculation using multilayer feed forward Neural Networks. IEEE transaction on Power system 15(2), 903-908.

147. Singh Chanan, 2002**.** Including uncertainty in LOLE calculation using fuzzy set theory. IEEE transaction on Power system, 17(1), 19-25.

148. Singh S. N., Srivastava L., Sharma J., 2000. Fast voltage contingency screening and ranking using cascade neural network. Electric Power Systems Research 53(3),197-205.

149. Srivastava L., Singh S. N., Sharma J., 2000. A hybrid model for fast voltage contingency screening and ranking. International Journal of Electric Power and Energy Systems 22(1), 35-42.

150. Sirouspour M. R. and Salcudean S. E., 2001. Nonlinear control of hydraulic robots. IEEE Transactions on Robotics and Automation 17(2), 173-182.

151. Solteiro Pires E. J., Moura Oliveira P. B. de, Tenreiro Machado J. A., 2007. Manipulator trajectory planning using a MOEA. Elsevier Applied Soft Computing 7(3), 659-667.

152. Strandberg M., 2004. Robot path planning: An object-oriented approach. Ph.D. thesis. Royal Institute of Technology (KTH) Stockholm, Sweden.

153. Sugar T.G. and Kumar V., 2002. Control of cooperating mobile manipulators. IEEE Transactions on Robotics and Automation 18(1), 94-103.

154. Sugie T., Fujimoto K., Kito Y., 2003. Obstacle avoidance of manipulators with rate constraints. IEEE Transactions on Robotics and Automation 19(3), 168- 174.

155. Suh S.H. and Kang G. S., 1988. A variational dynamic programming approach to robot-path planning with a distance-safety criterion. IEEE Transactions on Robotics and Automation 4(3), 334-349.

156. Sun Y.L. and Er M.J., 2004. Hybrid fuzzy control of robotics systems. IEEE Transactions on Fuzzy Systems **12** (6), 755–765.

157. Sundaram S., Remmler I., Amato N. M., 2001. Disassembly sequencing using a motion planning approach. IEEE International Conference on Robotics and Automation, Seoul, Korea, 2, pages 1475–1480.

158. Szilveszter Pletl and Bela Lantos, 2001. Advanced robot control algorithms based on fuzzy, neural and genetic methods. Journal of Advanced Computational Intelligence and Intelligent Informatics 5(2), 81-89.

159. Tafazoli S., Lawrence P. D., Salcudean S. E., 1999. Identification of inertial and friction parameters for excavator arms. IEEE Transactions on Robotics and Automation 15(5), 966-971.

160. Tang Y. and Velez-Diaz D., 2003. Robust fuzzy control of mechanical systems. IEEE Transactions on Fuzzy Systems 11(3), 411- 418.

161. Tarek A. El-Mihoub, Adrian A. Hopgood, Lars Nolle, Alan Battersby, 2006. Hybrid genetic algorithms: A Review Engineering letter 13(2).

162. Tian L., Collins C., 2004. An effective robot trajectory planning method using a genetic algorithm. Elsevier Mechatronics 14 (5), 455-470.

163. Tokhi M.O. and Azad A. K. M., 1997. Design and development of an experimental flexible manipulator system. Robotica 15 (3), 283–292.

164. Trinkle J. C. and Milgram R. James, 2002. Complete path planning for closed kinematic chains with spherical joints. The International Journal of Robotics Research, Sage Publications 21(9) 773-789.

165. Tsuji T., Hachino T., Oguro R., Umeda N., Takata H., 1998. A control design of robotics using the genetic algorithm. Springer Artificial Life and Robotics 2(1), 24-27.

166. Wang Li-Xin and Mandel Jerry M., 1992. Generating fuzzy rules by learning from examples. IEEE Transactions on System Man and Cybernatics 22(6), 534-539.

167. Wang D. and Hamam Y., 1992. Optimal trajectory planning of manipulators with collision detection and avoidance. The International Journal of Robotics Research 11(5), 460 - 468.

168. Wang Q. and Zalzala A.M.S., 1996. Genetic control of near time-optimal motion for an industrial robot arm, IEEE International Conference on Robotics and Automation Minneapolis, Minnesota, 2592–2597.

169. Wilson Lucas A., Moore Michelle D., Picarazzi Jason P., Simon D. San Miquel, 2004. Parallel genetic algorithm for search and constrained multi-objective

optimization. 18th IEEE International Parallel and Distributed Processing Symposium, Santa Fe, USA (IPDPS'04) 165-173.

170. Wise E., 2003. Applied robotics-II. Thomson Delmar Learning, Canada.

171. Wong S. V. and Hamouda A. M. S., 2004. Development of genetic algorithm-based fuzzy rules design for metal cutting data selection. Elsevier Robotics and Computer Integrated Manufacturing 18, 1–12.

172. Wu L., Cui K., Chen S. B., 1999. Redundancy coordination of multiple robotic devices for welding through genetic algorithm. Robotica 18, 669–676.

173. Yano F and Tooda Y., 1999. Preferable movement of a multi-joint robot arm using genetic algorithm. SPIE Confernce on Intelligent Robots and Computer Vision, 3837, 80–86.

174. Yoshida E., Murata S., Kamimura A., Tomita k., Kurokawa H., Kokaji S., 2003. Evolutionary motion synthesis for a modular robot using genetic algorithm. Journal of Robotics and Mechatronics 15(2), 227–237.

175. Yuan. J. and Yu S. L., 1999. End-effector position-orientation measurement. IEEE Transactions on Robotics and Automation 15(3), 592-595.

176. Yuefa Fang and Lung-Wen Tsai, 2002. Structure synthesis of a class of 4-DoF and 5-DoF parallel manipulators with identical limb structures. The International Journal of Robotics Research, Sage Publications 21(9), 799-810.

177. Yuval Davidor, 1990. Robot programming with a genetic Algorithm. IEEE CH2867-, 186-191.

178. Zacharia P. Th. and Aspragathos N. A., 2005. Optimal robot task scheduling based on genetic algorithms. Elsevier Robotics and Computer-Integrated Manufacturing 21, 67–79.

179. Zadeh L. A., 1965. "Fuzzy sets", Information and control. IEEE Transactions on Fuzzy systems 8(3), 338-353.

180. Zadeh L. A., 1996. Fuzzy logic=computing with words. IEEE Transactions on Fuzzy Systems 4(2), 103-111.

181. Zakharov A. and Halasz S., 1999**.** Genetic algorithms based identification method for a robot arm. Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE apos; 99) 3, 1014–019.

182. Zelinsky A., 1992. A mobile robot exploration algorithm. IEEE Transactions on Robotics and Automation, 8(6), 707-717.

183. Zhang Nian and Wunsch-II Donald C., 2003. Fuzzy logic in collective robotic search. 12th IEEE International Conference on Fuzzy Systems 2, 1471-1475.

184. Zhao Min., Ansari Nirwan, Edwin S. H. Hou, 1994. Mobile manipulator path planning by a Genetic Algorithm. Journal of Robotic Systems, 11 (3), 143–153.

185. Zhu X., Ding H., Wang J., 2003. Grasp analysis and synthesis based on a new quantitative measure. IEEE Transactions on Robotics and Automation 19(6), 942-953.

186. http://www.obitko.com/tutorials/genetic-algorithms/ga-basic-description.php

187. http://www.genetic-programming.com

188. http://www.cs.toronto.edu/~sme/CSC340F/slides/tutorial-prioritization.pdf

189. http://www.seattlerobotics.org/Encoder/mar98/fuz/flindex.html