

Inverse Kinematic Analysis of Robot Manipulators

**A THESIS SUBMITTED IN FULFILMENT OF
THE REQUIREMENT FOR THE AWARD OF THE DEGREE**

OF

DOCTOR OF PHILOSOPHY

IN

INDUSTRIAL DESIGN

BY

PANCHANAND JHA

(Roll. No. 511ID101)



**NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA, INDIA**

July-2015

dedicated
to
my
parents

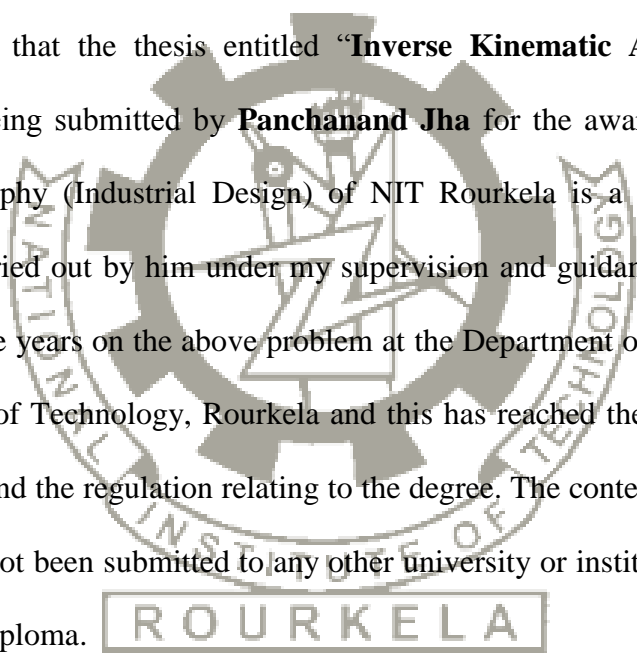


**NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA, INDIA**

Dr. Bibhuti Bhusan Biswal
Professor
Department of Industrial Design
NIT, Rourkela

CERTIFICATE

This is to certify that the thesis entitled “**Inverse Kinematic Analysis of Robot Manipulators**” being submitted by **Panchanand Jha** for the award of the degree of Doctor of Philosophy (Industrial Design) of NIT Rourkela is a record of bonafide research work carried out by him under my supervision and guidance. He has worked for more than three years on the above problem at the Department of Industrial Design, National Institute of Technology, Rourkela and this has reached the standard fulfilling the requirements and the regulation relating to the degree. The contents of this thesis, in full or part, have not been submitted to any other university or institution for the award of any degree or diploma.



(Dr. B. B. Biswal)

ACKNOWLEDGEMENT

This dissertation is a result of the research work that has been carried out at **National Institute of Technology, Rourkela**. During this period, the author came across with a great number of people whose contributions in various ways helped in the field of research and they deserve special thanks. It is a pleasure to convey the gratitude to all of them.

First of all the author expresses his heartiest gratitude to his supervisor and guide **Dr. B. B. Biswal**, Professor and Head, Department of Industrial Design, NIT, Rourkela for his valuable guidance, support and encouragement in the course of the present work. The successful and timely completion of the work is due to his constant inspiration and constructive criticisms. The author cannot adequately express his appreciation to him. The author records his gratefulness to Madam **Mrs. Meenati Biswal** for her constant support and inspiration during his work and stay at NIT, Rourkela.

The author take this opportunity to express his deepest gratitude to **Prof. M.R. Khan**, Head of the Department and **Prof. D. S. Bisht**, of the Department of Industrial Design, NIT Rourkela for constant advice, useful discussions, encouragement and support in pursuing the research work.

The author is grateful to **Prof. S.K. Sarangi**, Director, NIT, Rourkela, **Prof. R.K. Sahoo**, former Head of Mechanical Engineering Department, NIT, Rourkela, for their kind support and concern regarding his academic requirements.

The author also expresses his thankfulness to **Mr. H. Mishra, Mr. V. Mishra, Mr. A Jha, Mr. A. Agrawal, Mr. A. Ganguly and Mr. S. Chandrakar**, Department of Mechanical Engineering and **Mr. M. V. A. Raju, Mr. O. P. Sahu, Mr. R. N. Mahapatra, Mr. B. Balabantaray, and Mr. P. Parida**, researchers in NIT Rourkela for unhesitating cooperation extended during the tenure of the research programme.

The completion of this work came at the expense of author's long hours of absence from home. Words fail to express his indebtedness to his wife **Vibha**, loving son **Sambhav**, loving Niece **Mauli**, and nephew **Osho-Prince-Darsh** for their understanding, patience, active cooperation and after all giving their times throughout the course of the doctoral dissertation. The author thanks them for being supportive and caring. His **parents** and **relatives** deserve special mention for their inseparable support and prayers.

Last, but not the least, the author thank the one above all, the omnipresent **God**, for giving him the strength during the course of this research work.

Panchanand Jha

Abstract

An important part of industrial robot manipulators is to achieve desired position and orientation of end effector or tool so as to complete the pre-specified task. To achieve the above stated goal one should have the sound knowledge of inverse kinematic problem. The problem of getting inverse kinematic solution has been on the outline of various researchers and is deliberated as thorough researched and mature problem. There are many fields of applications of robot manipulators to execute the given tasks such as material handling, pick-n-place, planetary and undersea explorations, space manipulation, and hazardous field etc. Moreover, medical field robotics catches applications in rehabilitation and surgery that involve kinematic, dynamic and control operations. Therefore, industrial robot manipulators are required to have proper knowledge of its joint variables as well as understanding of kinematic parameters. The motion of the end effector or manipulator is controlled by their joint actuator and this produces the required motion in each joints. Therefore, the controller should always supply an accurate value of joint variables analogous to the end effector position. Even though industrial robots are in the advanced stage, some of the basic problems in kinematics are still unsolved and constitute an active focus for research. Among these unsolved problems, the direct kinematics problem for parallel mechanism and inverse kinematics for serial chains constitute a decent share of research domain. The forward kinematics of robot manipulator is simpler problem and it has unique or closed form solution. The forward kinematics can be given by the conversion of joint space to Cartesian space of the manipulator. On the other hand inverse kinematics can be determined by the conversion of Cartesian space to joint space. The inverse kinematic of the robot manipulator does not provide the closed form solution. Hence, industrial manipulator can achieve a desired task or end effector position in more than one configuration. Therefore, to achieve exact solution of the joint variables has been the main concern to the researchers.

A brief introduction of industrial robot manipulators, evolution and classification is presented. The basic configurations of robot manipulator are demonstrated and their benefits and drawbacks are deliberated along with the applications. The difficulties to solve forward and inverse kinematics of robot manipulator are discussed and solution of inverse kinematic is introduced through conventional methods. In order to accomplish the desired objective of the work and attain the solution of inverse kinematic problem an efficient study of the existing tools and techniques has been done.

A review of literature survey and various tools used to solve inverse kinematic problem on different aspects is discussed. The various approaches of inverse kinematic solution

is categorized in four sections namely structural analysis of mechanism, conventional approaches, intelligence or soft computing approaches and optimization based approaches. A portion of important and more significant literatures are thoroughly discussed and brief investigation is made on conclusions and gaps with respect to the inverse kinematic solution of industrial robot manipulators. Based on the survey of tools and techniques used for the kinematic analysis the broad objective of the present research work is presented as; to carry out the kinematic analyses of different configurations of industrial robot manipulators. The mathematical modelling of selected robot manipulator using existing tools and techniques has to be made for the comparative study of proposed method. On the other hand, development of new algorithm and their mathematical modelling for the solution of inverse kinematic problem has to be made for the analysis of quality and efficiency of the obtained solutions. Therefore, the study of appropriate tools and techniques used for the solution of inverse kinematic problems and comparison with proposed method is considered. Moreover, recommendation of the appropriate method for the solution of inverse kinematic problem is presented in the work.

Apart from the forward kinematic analysis, the inverse kinematic analysis is quite complex, due to its non-linear formulations and having multiple solutions. There is no unique solution for the inverse kinematics thus necessitating application of appropriate predictive models from the soft computing domain. Artificial neural network (ANN) can be gainfully used to yield the desired results. Therefore, in the present work several models of artificial neural network (ANN) are used for the solution of the inverse kinematic problem. This model of ANN does not rely on higher mathematical formulations and are adept to solve NP-hard, non-linear and higher degree of polynomial equations. Although intelligent approaches are not new in this field but some selected models of ANN and their hybridization has been presented for the comparative evaluation of inverse kinematic. The hybridization scheme of ANN and an investigation has been made on accuracies of adopted algorithms.

On the other hand, any Optimization algorithms which are capable of solving various multimodal functions can be implemented to solve the inverse kinematic problem. To overcome the problem of conventional tool and intelligent based method the optimization based approach can be implemented. In general, the optimization based approaches are more stable and often converge to the global solution. The major problem of ANN based approaches are its slow convergence and often stuck in local optimum point. Therefore, in present work different optimization based approaches are considered. The formulation of the objective function and associated constrained are discussed thoroughly. The comparison of all adopted algorithms on the basis of number

of solutions, mathematical operations and computational time has been presented. The thesis concludes the summary with contributions and scope of the future research work.

Table of Contents

Certificate	i
Acknowledgements	ii
Abstract.....	iii-v
Table of Contents	vi-x
List of Tables	xi-xii
List of Figures.....	xiii-xv
List of Symbols	xvi-xvii
Abbreviations	xviii
1 INTRODUCTION	1
1.1 Overview	1
1.2 Evolution of robot manipulators	3
1.3 Structural of industrial robots.....	5
1.3.1 Classification by Mechanism.....	5
1.3.2 Classification by dof and related componenets	8
a) General Manipulator	10
b) Redundant/Hyper-redundant Manipulator.....	10
c) Flexible Manipulator	10
d) Deficient Manipulator	11
1.3.3 Classification by Actuation.....	11
1.3.4 Classification by Workspace	11
a) Cartesian Robot	12
b) Cylindrical Robot	13
c) Spherical Robot	13
d) SCARA Robot	13
d) Articulated/Revolute Robot.....	14
1.3.5 Classification based on regional structure	15
1.3.6 Classification by Motion Characteristic	17
1.3.7 Classification by Application.....	17
1.4 Basic Kienmatics.....	18
1.5 Motivation	19
1.6 Broad Objective	20
1.7 Methodology	21

1.8	Organization of the Thesis	22
1.9	Summary	24
2	REVIEW OF LITERATURE	25
2.1	Overview	25
2.2	Survey of tools used for literature solution	26
2.2.1	Structural Analysis of Mechanism.....	40
2.2.2	Conventional Method for Kinematics.....	47
2.2.3	Intelligent or Soft-Computing Approach.....	59
2.2.4	Optimization Approach	68
2.3	Review Analysis and Outcome	75
2.4	Problem Statement	77
2.5	Scope of Work.....	77
2.6	Summary	78
3	MATERIALS AND METHODS.....	79
3.1	Overview	79
3.2	Materials.....	79
3.2.1	Description of planar 3-dof revolute manipulator	81
3.2.2	Description of 4-dof SCARA manipulator	82
3.2.3	Description of 5-dof revolute Pioneer2 manipulator	84
3.2.4	Description of 6-dof PUMA 560 manipulator.....	85
3.2.5	Description of 6-dof ABB IRb-1400 manipulator	86
3.2.6	Description of 5-dof ASEA IRb-6 manipulator.....	88
3.2.7	Description of 6-dof STAUBLI RX 160 L manipulator.....	89
3.3	Methods.....	90
3.3.1	Conventional approach	90
3.3.2	Intelligent based approaches	91
3.3.3	Optimization based approaches	93
3.4	Summary	95
4	MATHEMATICAL MODELLING AND KINEMATIC ANALYSIS	96
4.1	Overview	96
4.2	Representation methods and kinematics	98
4.2.1	Kinematic Variables and Parameters	98
4.2.2	DH-parameters	99
4.2.3	DH-Algorithm for frame assignment.....	102
4.2.4	Mathematical Modelling of 3-dof revolute manipulator	103
4.2.5	Mathematical Modelling of 4-dof SCARA manipulator	105
4.2.6	Mathematical Modelling of 5-dof revolute manipulator	107

4.2.7	Mathematical Modelling of 6-dof PUMA 560 manipulator.....	113
4.3	Quaternion Algebra Kinematics.....	115
4.3.1	Mathematical Background.....	116
a)	Conjugate of Quaternion.....	116
b)	Magnitude of Quaternion	117
c)	Norm.....	127
d)	Quaternion Inverse	117
4.3.2	Quaternion rotation and translation	117
4.3.3	Kinematic solution of SCARA Manipulator	118
4.3.4	Kinematic solution of 5-dof Revolute Manipulator.....	121
4.3.5	Kinematic solution of PUMA Manipulator	127
4.3.6	Kinematic solution of ABB IRB-1400 Manipulator.....	131
4.3.7	Kinematic solution of STAUBLI RX 160 L Manipulator.....	133
4.3.8	Kinematic solution of ASEA IRb-6 Manipulator	135
4.4	Summary	137
5	INTELLIGENT TECHNIQUES FOR INVERSE KINEMATIC SOLUTION	
	138	
5.1	Overview	138
5.2	Applications of ANN Models	139
5.2.1	Multi-layered Perceptron Network (MLPNN)	140
5.2.2	Polynomial Preprocessor Neural Network (PPNN).....	144
5.2.3	Pi-Sigma Neural Network.....	145
5.3	Application of Adaptive Neural Fuzzy Inference System (ANFIS)	147
5.3.1	Learning Algorithms.....	149
5.4	Hybridization of ANN with Metaheuristic Algorithms	150
5.4.1	Particle Swarm Optimization Algorithm(PSO)	152
5.4.2	Teaching learning based optimization(TLBO).....	153
5.4.3	Objective Function for training MLP	153
5.4.4	Objective Function.....	153
5.4.5	Weight and Bias Optimization Scheme	154
5.5	Summary	156
6	OPTIMIZATION SCHEME FOR INVERSE KINEMATIC SOLUTION ...	158
6.1	Overview	158
6.2	Metaheuristic Algorithms.....	161
6.2.1	Genetic Algorithm Reprerentation.....	164
a)	Initialization.....	165

b) Recombination.....	166
c) Mutation.....	167
d) Selection	167
6.2.2 Particle Swarm Optimization Overview	168
6.2.3 Grey Wolf Optimization Algorithm	169
6.3 Development of Novel Optimization Algorithm.....	171
6.3.1 Crab Intelligence Based Optimization (CIBO)Algorithm	172
6.3.2 Methodology of CIBO algorithm	174
6.3.3 Mathematical Modelling.....	175
a) Hypotheses.....	177
6.3.4 CIBO Algorithm	178
6.4 Implementation for Solving Inverse Kinematics	181
6.4.1 Mathematical Modelling of Objective Function.....	181
6.4.2 Position Based Error	183
6.4.3 Orientation Based Error	184
6.5 Solution Scheme of Inverse Kinematic Problem	186
6.6 Summary	186
7 RESULTS AND ANALYSIS.....	188
7.1 Overview	188
7.2 Inverse Kinematic Solution Using ANN.....	189
7.2.1 Results for 3-dof planar Revolute Manipulator	190
7.2.2 Results for 4-dof SCARA Manipulator	192
7.2.3 ANFIS Results for 4-dof SCARAManipulator	197
7.2.4 ANN Results for 5-dof Revolute Manipulator	199
7.2.5 ANFIS Results for 5-dof Revolute Manipulator.....	203
7.3 Hybrid ANN Approach for Inverse Kinematic Problem	206
7.3.1 Results for 4-dof SCARA Manipulator	206
7.3.2 Results for 5-dof revolute Manipulator	209
7.3.3 Results for 6-dof PUMA Manipulator	216
7.3.4 Results for 6-dof ABB IRb-1400 Manipulator	227
7.3.5 Results for 5-dof ASEA IRb-6Manipulator.....	232
7.3.6 Results for 6-dof STAUBLI RX 160 L Manipulator.....	235
7.4 Metahuristic Approach for Inverse Kinematic Problem	240
7.4.1 Results for 4-dof SCARA Manipulator	241
7.4.2 Results for 5-dof revolute Manipulator	245
7.4.3 Results for 6-dof PUMA Manipulator	250
7.4.4 Results for 6-dof ABB IRb-1400 Manipulator	263

7.4.5	Results for 5-dof ASEA IRb-6Manipulator	268
7.4.6	Results for 6-dof STAUBLI RX 160 L Manipulator.....	272
7.5	Discussions.....	277
7.6	Summary	281
8	CONCLUSIONS AND FURTHER WORK	283
8.1	Overview	283
8.2	Conclusions	284
8.3	Contributions.....	287
8.4	Future Research Work.....	288
9	REFERENCES	289
	Publications	311
	Curriculum Viate.....	313

List of Tables

Table 1.1: Different types of joints.....	9
Table 1.2: Configurations and workspace.	12
Table 1.3: Configurations of 6-dof revolute manipulators.	15
Table 1.4: Classification based on regional structure.	16
Table 2.1: List of some important literatures.....	27
Table 2.2: Different mechanisms and mobility.	42
Table 3.1: Configurations of robot manipulators.....	80
Table 3.2: Manipulator joint limits and kinematic parameters.	81
Table 3.3: Manipulator joint limits and kinematic parameters.	83
Table 3.4: Parm2 manipulator joint limits and kinematic parameters.....	85
Table 3.5: Maximum limit of joint variables.....	85
Table 3.6: Joint variable and parameters of PUMA 560 robot.	86
Table 3.7: ABB IRB-1400 manipulator joint limits and kinematic parameters.	87
Table 3.8: ASEA IRb-6 manipulator joint limits and kinematic parameters.....	88
Table 3.9: STAUBLI RX160L manipulator joint limits and kinematic parameters..	89
Table 3.10: Adopted materials and methods.	94
Table 4.1: DH parameters.....	100
Table 4.2: DH-parameters for 3-dof revolute manipulator.....	103
Table 4.3: The DH Parameters for SCARA manipulator.	105
Table 4.4: The DH parameters 5-dof revolute manipulator.....	107
Table 4.5: The DH parameters for PUMA manipulator.	113
Table 6.1: Shell selection.....	174
Table 7.1: Position of end effector and joint variables.	190
Table 7.2: Configuration of MLPNN.	193
Table 7.3: Comparison between analytical solution and MLPNN solution.	193
Table 7.4: Regression analysis.....	194
Table 7.5: Configuration of ANFIS.....	198
Table 7.6: Comparison of results.	198
Table 7.7: Desired joint variables determined through analytical solution.	200
Table 7.8: Configuration of MLPNN.	200
Table 7.9: Configuration of ANFIS.....	203
Table 7.10: Comparison of results.	203
Table 7.11: Mean square error for all training samples of hybrid MLPNN.	207
Table 7.12: Mean square error for all training samples of hybrid MLPNN.	210
Table 7.13: Configuration of MLPNN.	217
Table 7.14: Desired joint variables determined through quaternion algebra	217

Table 7.15: Mean square error for all training samples of hybrid MLPNN.	218
Table 7.16: Configuration of MLPNN	227
Table 7.17: Desired joint variables determined through quaternion algebra	228
Table 7.18: Mean square error for all training samples of hybrid MLPNN.	229
Table 7.19: Configuration of MLPNN.	232
Table 7.20: Desired joint variables determined through quaternion algebra.....	233
Table 7.21: Mean square error for all adopted algorithms.	233
Table 7.22: Configuration of MLPNN.	236
Table 7.23: Desired joint variables determined through quaternion algebra	236
Table 7.24: Mean square error for all training samples of hybrid MLPNN.	237
Table 7.25: Five different positions and joint variables.	241
Table 7.26: Five different positions and joint variables through adopted algorithm.	242
Table 7.27: Five different positions and joint variables.	245
Table 7.28: Five different positions and joint variables through adopted algorithm...	246
Table 7.29: Computational time for inverse kinematic evaluations.	246
Table 7.30: Five different positions and joint variables through quaternion.....	250
Table 7.31: Comparative results for joint variable and function value.	251
Table 7.32: Computational time for inverse kinematic evaluations.	260
Table 7.33: Five different positions and joint variables through quaternion.....	264
Table 7.34: Comparative results for joint variable and function value.	265
Table 7.35: Five different positions and joint variables.	268
Table 7.36: Comparative results for joint variable and function value.	269
Table 7.37: Five different positions and joint variables through quaternion.....	273
Table 7.38: Comparative results for joint variable and function value.	274
Table 7.39: Comparative analysis of conventional tools.....	278
Table 7.40: Comparative analysis of intelligent approaches.	279
Table 7.41: Comparative analysis of optimization algorithms.	280

List of Figures

Figure 1.1: (a) Serial [21], (b) Parallel and [21] (c) Hybrid mechanisms	6
Figure 1.2: Human arm structures.	7
Figure 1.3: Joint rotations of 7-dof robotic arm.....	7
Figure 1.4: SCARA robot.	13
Figure 1.5: Revolute robot.	14
Figure 1.6: Regional mechanism of robot manipulators	16
Figure 3.1: Model of 3-dof revolute manipulator.	81
Figure 3.2: Structure of Adept One SCARA manipulator.....	83
Figure 3.3: Structure of the Pinoneer arm2.....	84
Figure 3.4: Structure of PUMA 560 robot manipulator	86
Figure 3.5: Configurations of ABB IRB 1400.....	87
Figure 3.6: Configurations of ASEA IRb-6.....	88
Figure 3.7: Configurations of STAUBLI RX 160 L.....	89
Figure 4.1: Position and direction of a cylindrical joint in a Cartesian coordinate frame.....	99
Figure 4.2: kinematic pair and DH parameters.....	100
Figure 4.3: Denavit-Hartenberg parameters for successive translation and rotation of links	101
Figure 4.4: Planar 3-dof revolute manipulator.....	104
Figure 4.5: Coordinate frames of 3-dof revolute manipulator.....	104
Figure 4.6: DH frames of the SCARA robot.	106
Figure 4.7: Structure of SCARA manipulator through MATLAB.....	106
Figure 4.8: Model and coordinate frames of the manipulator.	108
Figure 4.9: Configuration of 5-dof revolute manipulator	108
Figure 4.10: Model and coordinate frames of manipulator.	113
Figure 4.11: Representation of rotation.	117
Figure 4.12: SCARA manipulator.	119
Figure 4.13: PUMA 560 manipulator model.	122
Figure 4.14: Base frame and model of 5-dof revolute manipulator.....	127
Figure 4.15: Configuration and model of ABB IRB-1440 robot manipulator.	131
Figure 4.16: Coordinate frame and model of STAUBLI RX160L robot manipulator.	133
Figure 4.17: Coordinate frame and model of ASEA IRb-6 robot manipulator.	135
Figure 5.1: Neural network models (a) Feed forward and (b) back propagation strategy.....	140
Figure 5.2: Multi-layered perceptron neural network structure.....	141

Figure 5.3: Flow chart for MLPBP.....	142
Figure 5.4: Polynomial perceptron network.....	145
Figure 5.5: Pi-Sigma neural network.....	146
Figure 5.6: Training of ANFIS structure.....	148
Figure 5.7: Architecture of ANFIS.....	149
Figure 5.8: Flow chart for PSO.....	152
Figure 5.9: Flow chart for MLPPSO.....	155
Figure 5.10: MLP network with structure 3-3-1.....	156
Figure 6.1: Binary representations of genes.....	164
Figure 6.2: Examples for simple crossover with two different cases.....	166
Figure 6.3: Mutation in genetic algorithm.....	167
Figure 6.4: Roulette wheel selections.....	168
Figure 6.5: Flow chart for grey wolf optimizer.....	170
Figure 6.6: Representation of swarm and searching behaviour.....	178
Figure 6.7: Flow chart for CIBO algorithm.....	180
Figure 6.8: Position based error.....	184
Figure 6.9: Orientation angle between two frames.....	185
Figure 7.1: Comparison of desired and predicted value of joint angles for 2-3-2-3 configuration using MLP model.....	191
Figure 7.2: Comparison of desired and predicted value of joint angles for 2-4-4-3 configuration using MLP model.....	191
Figure 7.3: Comparison of desired and predicted value of joint angles for 2-5-5-3 configuration using MLP model.....	191
Figure 7.4: Mean square error for joint angles using PPN model.....	192
Figure 7.5: Mean square error for joint angles using Pi-sigma network model.....	192
Figure 7.6: Mean square error for theta1.....	195
Figure 7.7: Mean square error for theta2.....	195
Figure 7.8: Mean square error for d3.....	196
Figure 7.9: Mean square error for theta4.....	196
Figure 7.10: Graphical view of regression.....	197
Figure 7.11: Mean square error for theta1.....	198
Figure 7.12: Mean square error for theta2.....	198
Figure 7.13: Mean square error for d3.....	199
Figure 7.14: Mean square error for theta4.....	199
Figure 7.15: Mean square error for theta1.....	201
Figure 7.16: Mean square error for theta2.....	201
Figure 7.17: Mean square error for theta3.....	201
Figure 7.18: Mean square error for theta4.....	202

Figure 7.19: Mean square error for theta5	202
Figure 7.20: Figure 7.17: Mean square error for theta1	202
Figure 7.21: Mean square error for theta1	204
Figure 7.22: Mean square error for theta2	204
Figure 7.23: Mean square error for theta3	205
Figure 7.24: Mean square error for theta4	205
Figure 7.25: Mean square error for theta5	206
Figure 7.26: (a), (b), (c), (d) and (e) are mean square error curve for all joint angles using MLPGA.....	209
Figure 7.27: (a), (b), (c), (d) and (e) are mean square error curve for all joint angles using MLPGA.....	214
Figure 7.28: (a), (b), (c), (d) and (e) are mean square error curve for all joint angles using MLPTLBO	214
Figure 7.29: (a), (b), (c), (d) and (e) are mean square error curve for all joint angles using MLPTCIBO.....	216
Figure 7.30: (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPBP for all joint angles.....	220
Figure 7.31: (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPGA for all joint angles.....	222
Figure 7.32: (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPTLBO for all joint angles	224
Figure 7.33: (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPCIBO for all joint angles	226
Figure 7.34: (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPGA for all joint angles.....	231
Figure 7.35: (a), (b), (c), (d) and (e) are mean square error curve of MLPGA for all joint angles.....	235
Figure 7.36: (a), (b), (c), (d) and (e) are mean square error curve of MLPGA	240
Figure 7.37: Comparison of joint variables for position 1	243
Figure 7.38: Comparison of joint variables for position 2	243
Figure 7.39: Comparison of joint variables for position 3	244
Figure 7.40: Comparison of joint variables for position 4	244
Figure 7.41: Comparison of joint variables for position 5	245
Figure 7.42: Function value and joint variables for P1	247
Figure 7.43: Function value and joint variables for P2	247
Figure 7.44: Function value and joint variables for P3	248
Figure 7.45: Function value and joint variables for P4.....	248
Figure 7.46: Function value and joint variables for P5	248

Figure 7.47: Function value and joint variables for P1	248
Figure 7.48: Function value and joint variables for P2	249
Figure 7.49: Function value and joint variables for P3	249
Figure 7.50: Function value and joint variables for P4	249
Figure 7.51: Function value and joint variables for P5	249
Figure 7.52: Function value and joint variables for P1.....	252
Figure 7.53: Function value and joint variables for P2	253
Figure 7.54: Function value and joint variables for P3	253
Figure 7.55: Function value and joint variables for P4	253
Figure 7.56: Function value and joint variables for P5	254
Figure 7.57: Function value and joint variables for P1	254
Figure 7.58: Function value and joint variables for P2	254
Figure 7.59: Function value and joint variables for P3	255
Figure 7.60: Function value and joint variables for P4	255
Figure 7.61: Function value and joint variables for P5	255
Figure 7.62: Function value and joint variables for P1	256
Figure 7.63: Function value and joint variables for P2	256
Figure 7.64: Function value and joint variables for P3	256
Figure 7.65: Function value and joint variables for P4	257
Figure 7.66: Function value and joint variables for P5	257
Figure 7.67: Function value and joint variables for P1	257
Figure 7.68: Function value and joint variables for P2	257
Figure 7.69: Function value and joint variables for P3.....	258
Figure 7.70: Function value and joint variables for P4	258
Figure 7.71: Function value and joint variables for P5	258
Figure 7.72: Function value and joint variables for P1	258
Figure 7.73: Function value and joint variables for P2	259
Figure 7.74: Function value and joint variables for P3	259
Figure 7.75: Function value and joint variables for P4	259
Figure 7.76: Function value and joint variables for P5	260
Figure 7.77: Function value and joint variables for P1.....	261
Figure 7.78: Function value and joint variables for P2	261
Figure 7.79: Function value and joint variables for P3.....	262
Figure 7.80: Function value and joint variables for P4.....	262
Figure 7.81: Function value and joint variables for P5	263
Figure 7.82: Function value and joint variables for P1	266
Figure 7.83: Function value and joint variables for P2	266
Figure 7.84: Function value and joint variables for P3	267

Figure 7.85: Function value and joint variables for P4	267
Figure 7.86: Function value and joint variables for P5.....	268
Figure 7.87: Function value and joint variables for P2	270
Figure 7.88: Function value and joint variables for P3	271
Figure 7.89: Function value and joint variables for P4	271
Figure 7.90: Function value and joint variables for P5	271
Figure 7.91: Function value and joint variables for P1	272
Figure 7.92: Function value and joint variables for P2	275
Figure 7.93: Function value and joint variables for P3	275
Figure 7.94: Function value and joint variables for P4	276
Figure 7.95: Function value and joint variables for P5	276
Figure 7.96: Function value and joint variables for P1	277

List of Symbols

a_i	Link Length
α_i	Twist angle
d_i	Joint Distance
θ_i	Joint angle
c_i	$\cos \theta_i$, ($i = 1,2,3\dots n$)
s_i	$\sin \theta_i$, ($i = 1,2,3\dots n$)
δ, λ	Angular random number
Δw_{ij}^h	Hidden weight
\otimes	Quaternion multiplication
δ_k	Local gradient of the kth layer
η	Momentum parameter
$[R_{be}, T_{be}]$	Transformation quaternion of end effector
T	Transfer matrix
O	Origin Of Global coordinate system
$O_{1\dots 5}$	Origins of Local coordinate systems of thumb and fingers
$q_{1\dots 25}$	Joint angles of the hand model
${}^0A_{1\dots n}$	Transfer matrices from local coordinate system to global coordinate system.
n_e	Hidden layer neurons
δ	nonlinear activation functions
λ, μ	Lagrange multiplier

Abbreviations

dof	Degrees of freedom
DH	Denavit- Hartenberg
MLP	Multi layered perceptron
PSO	Particle swarm optimization
GWO	Grey wolf optimizer
TLBO	Teaching learning based optimization
PPN	Polynomial pre-processor network
CIBO	Crab intelligence based optimization
GA	Genetic algorithm
QA	Quaternion algebra
ANFIS	Adoptive Neuro-Fuzzy Inference System
LS	Least squares
BP	Back propagation
FIS	Fuzzy Interface System
MF	Membership function
GWS	Grasp wrench space
FC	Force closure
HT	Homogeneous transformation
R	Revolute
P	Prismatic
H	Quaternion vector

Chapter 1

INTRODUCTION

1.1 Overview

Over the last few decades, use of industrial robots can be seen worldwide and has significantly increased with a faster increasing trend. Mostly these are being used for material handling, welding, painting, assembling of parts, packaging, handling hazardous materials, undersea operations, etc. Robot manipulator implicates an electromechanical device that requires human dexterity to perform a variety of tasks. Although few manipulators are anthropomorphic and humanoid, most of these robots can be treated as electromechanical devices from their structure point of view. On the other hand, there are autonomous and semiautonomous robots that have a broad range of applications such as planetary space exploration, surgical robotics, rehabilitation, and household applications.

A common characteristic of such applications is that the robot needs to operate in unstructured environments rather than structured industrial work cells. Motion control and trajectory planning for robots in unstructured environments pose important challenges due to uncertainties in environment modelling, sensing, and robot actuation. At the present status, the broad area of robot applications deal with industrial robot arms operating in both structured and unstructured environments. A first introduction to the subject of robotics ought to include a rigorous treatment of the topics in this text.

Robots are also a concerned with and are slowly becoming a part of human life by assisting them in professional and personal life as well as insulating humans from a situation involving hazards, discomfort, repetitions, etc. With the advancement of various technology, the scope of the tasks performed by robots is widened so that it is desirable for machines to extend the capabilities of men and to replace them by robots in carrying out at tiresome as well as hazardous jobs. In order to accomplish the tasks in

human-like ways and to realize a proper and safe co-operation between humans and robots, the robots of the future must be thought of having human excellence in terms of its structure, intelligence, smartness and reactions. Therefore, a robot operating under some degree of autonomy can be extremely complex electromechanical systems whose analytic description requires advanced methods. Design and development of such devices present many challenging and interesting research problems. The most important thing is reprogramming ability of robot. It is computer controlled that gives the robot its utility and adaptability. The so-called robotics revolution is, in fact, part of the larger computer revolution. There are many fields for robot manipulators to perform a variety of tasks. Some of these are automobiles, household's products, pick-n-place, undersea and planetary explorations, satellite retrieval and repair, defusing of explosives and radioactive field. In the medical field robotics find applications in rehabilitation and surgery that involve kinematic, dynamic and control operations.

Robot manipulators move along pre-specified trajectories which are sequence of points where end effector position, and orientations are known. Trajectories may be joint space or Cartesian spaces that are a function of time. The industrial robots can be explicitly considered as open chain mechanisms that are systems of rigid bodies connected by various joints. Joints allow particular types of relative motions between the connected bodies. For example, a rotational joint acts as a hinge and allows only a relative rotation between the connected bodies about the axis of the joint. A system of rigid bodies interconnected by joints is called a kinematic chain. Individual rigid bodies within the kinematic chain are called links. A kinematic chain can be serial, parallel, or serial, and parallel combined, i.e. the kinematic chain can be open, closed, or branched. It is required to compute all the necessary points in Cartesian coordinate to perform the smooth operation. The conversion of trajectory locations from Cartesian coordinates to joint coordinates is referred to as the inverse kinematics problem.

Even though industrial robots are in the advanced stage, some of the basic problems in kinematics are still unsolved and constitute an active focus for research. Among these unsolved problems, the direct kinematics problem for parallel mechanism and inverse kinematics for serial chains constitute a decent share of research domain. The present research work primarily focuses on Kinematics of various industrial manipulators different configurations. The inverse kinematics problem is fundamental, not only in the design of manipulator but also in other applications including computer animations and molecular modelling. This problem is difficult due to its inherent computational complexity (i.e. NP-hard Problem) and due to mathematical complexity that does not guarantee closed form solution.

1.2 Evolution of robot manipulators

The concept of the robot was evidently recognized by the Czech playwright Karel Capek during the twentieth century in his play “Rossum’s Universal Robots (R.U.R.)”. The term “robot” is derived from “robota” which means subordinate labour in Slave languages. In 1940, the ethics of the interaction between robots and humans was envisioned to be governed by the well-known three fundamental laws of Isaac Asimov, the Russian science-fiction writer in his novel “Run-around”.

The middle of the twentieth century brought the first explorations of the connection between human intelligence and machines, marking the beginning of an era of fertile research in the field of artificial intelligence (AI). Around that time, the first robots were realized. They benefited from advances in the different technologies of mechanics, controls, computers and electronics. As always, new designs motivate new research and discoveries, which, in turn, lead to enhanced solutions and thus to novel concepts. This virtuous circle over time produced that knowledge and understanding that gave birth to the field of robotics, properly referred to as the science and technology of robots.

The early robots built in the 1960s stemmed from the confluence of two technologies: numerical control machines for precise manufacturing, and tele-operators for remote radioactive material handling. These master slave arms were designed to duplicate one-to-one the mechanics of the human arm and had rudimentary control and little perception about the environment. Then, during the mid-to-late twentieth century, the development of integrated circuits, digital computers and miniaturized components enabled computer-controlled robots to be designed and programmed. These robots, termed industrial robots, became essential components in the automation of flexible manufacturing systems in the late 1970s. Further to their wide application in the automotive industry, industrial robots were successfully employed in general industry, such as the metal products, the chemical, the electronics and the food industries. More recently, robots have found new applications outside the factories, in areas such as cleaning, search and rescue, underwater, space, and medical applications.

In the 1980s, robotics was defined as the science that studies the intelligent connection between perception and action. With reference to this definition, the action of a robotic system is entrusted to a locomotion apparatus to move in the environment (wheels, crawlers, legs, propellers) and/or to a manipulation apparatus to operate on objects present in the environment (arms, end effectors, artificial hands), where suitable actuators animate the mechanical components of the robot. The perception is extracted from the sensors providing information on state of the robot (position and speed) and its surrounding environment (force and tactile, range and vision). The intelligent connection is entrusted to a programming; planning and control architecture that relies

on the perception and available models of the robot and environment and exploits learning and skill acquisition.

In the 1990s research was boosted by the need to resort to robots to address human safety in hazardous environments (field robotics), or to enhance the human operator ability and reduce his/her fatigue (human augmentation), or else by the desire to develop products with wide potential markets aimed at improving the quality of life (service robotics). A common denominator of such application scenarios was the need to operate in a scarcely structured environment that ultimately requires increased abilities and a higher degree of autonomy.

By the dawn of the new millennium, robotics has undergone a major transformation in scope and dimensions. This expansion has been brought about by the maturity of the field and the advances in its related technologies. From a largely dominant industrial focus, robotics has been rapidly expanding into the challenges of the human world (human-centered and life-like robotics). The new generation of robots is expected to safely and dependably co-habitat with humans in homes, workplaces, and communities, providing support in services, entertainment, education, healthcare, manufacturing, and assistance.

Beyond its impact on physical robots, the body of knowledge robotics has produced is revealing a much wider range of applications reaching across diverse research areas and scientific disciplines, such as: biomechanics, haptics, neurosciences, and virtual simulation, animation, surgery, and sensor networks among others. In return, the challenges of the new emerging areas are proving an abundant source of stimulation and insights for the field of robotics. It is indeed at the intersection of disciplines that the most striking advances happen. Practical implementation of industrial robots was first started during the 1960s, along with numerical controlled and CAD/CAM systems. Now a day these manipulators reached the maturity stages. Some of the landmark developments in industrial robots are mentioned with this [1]:

1947 –The first servo controlled electric tele-operator launched

1948 –Introduction of force feedback in tele-operator

1954 –First programmable design from George Devol

1956 –Foundation of Unimation company by Josh Engelberger the Unimation Company

1961 –General Motors implementation of Unimate robot in New Jersey

1963 –First vision system developed for robots

1973 - Stanford University developed robot arm

1974 –First computer controlled manipulator introduced the MilacronT3
1978 –Development of PUMA 6 axis robot
1979 –First assembly line SCARA robot designed by Japanese
1981 - Mellon University developed first direct drive manipulator
1989- Hi-tech chess playing robot
1996 - Concept of Honda's P2 humanoid robot
1997 - Mars space exploration robot sojourner rover
2001 - Canadarm2 was implemented into ISS
2002- Introduction of humanoid robot ASIMO
2004 - Cornell University exposed a robot skilled of self-replication
2005- Development of wireless operated and computer controlled HUBO robot by KIST
2006- Starfish 4-legged robot developed by Cornell University
2007- Japanese company introduced entertainment robot TOMY
2013-to present- Kuka Robotics LBR iiwa, a lightweight robot Rob coaster for entertainment

Today, new communities of users and developers are forming, with growing connections to the core of robotics research. A strategic goal for the robotics community is one of outreach and scientific cooperation with these communities. Future developments and expected growth of the field will largely depend on the research community's abilities to achieve this objective.

1.3 Structure of industrial robots

This section is devoted to the classification of industrial robots, with attention to serial structures. Basic criteria for classification have been addressed stepwise, and concern mathematics behind the mechanism has also been proposed. The major aim is restricted to robots that are mainly anticipated for manipulation tasks and serial kinematic chains. Robots can usually be classified as per their number of degree of freedom (dof) or axes and their kinematic characteristic. Working proficiencies of robot manipulator can be evaluated from its degree of freedom. Common 6-dof robot manipulator can only achieve a general task in 3-dimension space containing arbitrarily position and orientation for any object. On the other hand, for specific application one needs to design robot manipulator as per dof as well as kinematics characteristic. However, there

are numerous criteria for the classification of robot manipulator but typically one can select dof or number of axes. On the other hand, Robotics Institute of America (RIA), Association Francaise de Robotique (AFR) and Japanese Industrial Robot Association broadly classified in 6 diverse modules that are as follows:

1. Manual handling devices
2. Fixed sequence robot
3. Variable sequence robot
4. Playback robot
5. Numerical control robot
6. Intelligent robot

Other than these above mentioned modules of industrial robot manipulator it can also be classified as per their mechanism, dof, actuation, workspace, control, motion and application.

1.3.1 Classification by mechanism

Typically a robot manipulator may be either a serial one having open loop or a parallel one having closed loop structure. In industrial robot manipulators the joint type may be either prismatic (P) or revolute (R) whereas the link type may be either rigid or flexible. Moreover, there can be hybrid structure that consists of both open and closed loop mechanical chains. The serial manipulator can be categorized based on the first joint will always starting from the fixed base and end of the link will free to move in space, see Figure 1.1 (a). There are many combinations of these joints and links that creates different configurations of robot manipulator simply due to the joints R and P, axes of two adjacent may be either parallel or orthogonal. Orthogonal joints intersect by 90 degrees with respect to their common normal and it can be parallel when one axis rotates 90 degrees, see Figure 1.1(b).

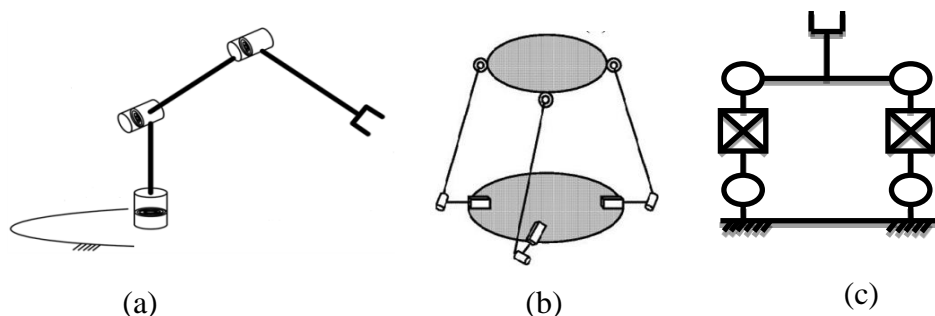


Figure 1.1 (a) Serial [1], (b) Parallel [1] and (c) Hybrid mechanisms

Examples of serial manipulators are PUMA, SCARA, KUKA, DENSO etc., Gough platform, Delta robot, 3-RPR planar parallel robot etc., are parallel manipulators and

Fanuc S-9000W is an example of hybrid manipulator as shown in Figure 1.1(c). Figure 1.1 shows further examples of mechanisms that result from open, closed and hybrid open/closed kinematic chains. Robotics and living organisms resemble the serial/parallel or hybrid mechanism. The most common and well known example is the human hand that resembles as a serial, parallel and hybrid manipulator as shown in Figure 1.2.

The human arm frame consists of different number of bones, as shown in Figure 1.2 that creates serial/hybrid manipulator or the kinematic chain. The human shoulder is attached to the stem having spherical joint. In the later chapter, human arm has been considered only 7-dof serial manipulator as shown in Figure 1.2 (a). The clavicle joint is connected to stem depicted as S in Figure and also with scapula via acromioclavicular joint (A). The scapula joint later connected with glenohumeral joint (G) to the upper arm. A summarized exemplary of arm mechanism identical to shoulder is shown in Figure 1.2 (b).

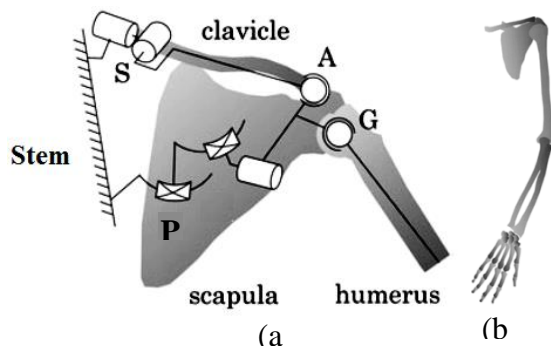


Figure 1.2 Human arm structures [1]

As per depicted figure the arm manipulator structure have 11-dof. The upper arm with humeral bone can be assumed as a serial mechanism and elbow joint with a humeral bone that connects the ulna can be considered as a parallel manipulator.

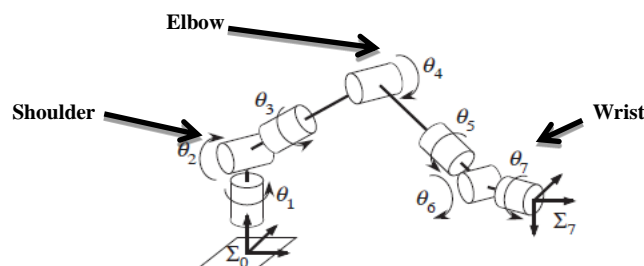


Figure 1.3 Joint rotations of 7-dof robotic arm

Most of the manipulator or common industrial manipulators are based on the above discussed human arm mechanism. As per Figure 1.3 is the elaborated view of the human arm with 7-dof manipulator that includes the shoulder, elbow and wrist.

1.3.2 Classification by degree of freedom and related components

The specific motion of links related to any mechanism or machine can be defined as the degree of freedom. To execute specific task degree of freedom will always play the main role. The total number of dof will always equal the number of independent displacement of links. As we know that 6-dof robot manipulator is the basis to execute the specific task in 3-dimensional space. On the other hand, mathematical definition of degrees of freedom will be a minimum number of independent joint parameters of any mechanism that exclusively describe the spatial position and orientation of system/body. On the other hand, no. of dof in any mechanism can be obtained by summation of the available dof of moving links that would be then λN . This is no. of dof if there are no joints and from this we can subtract the constraint C_i . Now this can be expressed as follows

$$\text{dof} = \lambda N - \sum_{i=1}^n C_i \quad (1.1)$$

Where, a constraint $C_i = \lambda - f_i$ that is the difference between the potential dof (λ) and no. of dof permitted by joint (f). Suppose there are f independent joint variables associated with a joint. We would propose that the joint permits f degrees of freedom.

$$\text{dof} = \lambda(N - J - 1) + \sum_{i=1}^J F_i \quad (1.2)$$

This is known as Grübler's formula for the degree of freedom. Where N is a number of links including fixed or base link, J is no. of joints F_i is dof at the i^{th} Joint.

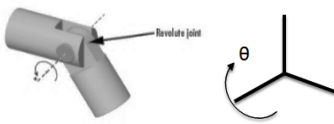
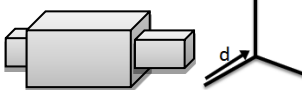
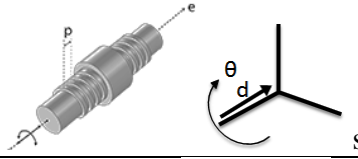
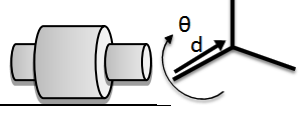
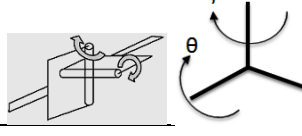
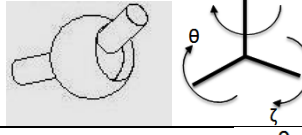
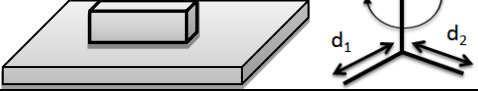
$$\lambda = \begin{cases} 6 & \text{for spatial manipulator and mechanisms} \\ 3 & \text{for planar manipulator and mechanisms} \end{cases} \quad (1.3)$$

A rigid body moving freely in 3-Dimensional space contains 6-dof and its position in the space can be separated with three positional and three orientational coordinates, i.e. $\lambda = 6$ parameters as given in equation (1.3). But in case if $\lambda = 3$ dof then there will be two positional and one orientational coordinate will be there to explain. In this context we can explain no. of dof in case of rotational joint, for example in this joint we know $f=1$ and $\lambda = 6$ therefor $c=6-1=5$, means rotational joint reduces 5 dof of relative movement between two links.

The no. of dof's permitted by a joint and their characteristic can be determined by the design constraints imposed on body or link. There are many different types of joints as

shown in Table 1.2. Among these different types of joint the two common joints that permit $f=1$ dof and $c=5$ constraints in spatial motion or another $c=2$ constraints for planar motion. From Table 1.1 basic notations for joints are given for example revolute and prismatic joints can be denoted as R and P. These joints can be described by a unit vector, which defines their axis of either rotation or translation. For example, revolute and prismatic joints having one dof while cylindrical and Hooke joints contain two degrees of freedoms. The diversity of joints in many mechanisms is larger, but these joints are commonly used in the field of robotics.

Table 1.1 Different types of joints

Joints/Pair	Symbol	dof	Representation
Revolute	R	1	
Prismatic	P	1	
Screw	H	1	
Cylindrical	C	2	
Hooke joint	T	2	
Spherical	S	3	
Planar	E	3	

In cylindrical and screw, joint translation takes place in d direction and rotation is about the coincident axis with an angle θ . Wherein, joint translation and rotations θ and d are independent parameters. Hence c will be 4 and $f=2$. Therefore, independence of screw joint can be explained with the relation between $\Delta d = \gamma \Delta \theta$, where γ and $\Delta \theta$ are the variations of joint and is pitch of the screw. Although screw joint is having two

independent joint parameters and one joint either θ or d , therefore in this case f will be 1 and $c=5$. Similarly, other joints can be elaborated as per the degree of freedom and imposed constraints. These notations and representations have been adopted throughout the text. Therefore on the basis of dof the robots can be classified as follows:

a) General manipulator

General robots can normally have 6-dof due to the vast application in various fields. There are many robots which possess 6-dof for example Fanuc S-900W, where last three joint axes intersect at the wrist centre. The kinematics solution for this class of manipulator can be separately solved considering first three links and then last three links can be solved independently. A therefor solution of inverse kinematics will be much easier than the other class of manipulators.

b) Redundant/hyper redundant manipulator

Kinematic redundancy of any mechanisms arises when it has more dof than those rigorously necessary to perform a desired task. Most of the industrial application can be executed by 6-dof but if it is 7-dof robot manipulator, it can be considered as the distinctive example of inherent redundancy. It is not always necessary that the robot with more dof will be redundant, but sometimes it occurs for less dof for specific tasks, such as simple manipulator tool positioning without having constraints for the orientation. Hyperredundant manipulators for any mechanism occur when it has a larger number of joints. Its joint configurations dof are exceeded to its task space dof. Therefore, 7-dof or 8-dof spatial manipulator usually not considered as a hyperredundant manipulator. A typical example of hyperredundant is snake robot. In fact, redundant manipulators are mainly used due to its increased dexterity; it may tolerate singularities, joint variable limits, and obstacle avoidance, but also for minimizing torque/energy for a given task.

c) Flexible manipulator

The standard hypothesis relating robot kinematics, design of manipulator and dynamics is that robot manipulator generally comprises of rigid links and transmission components. However it can be assumed as standard condition for general application which may be effective for less payloads or less interacting forces and slow motions. Practically speaking, flexible robot manipulator can be useful due to the reduced weight of moving links and slender design of links as well as use of compliant transmission elements. This concept of flexibility usually having major application in the area of space robot because of very long links of manipulator further requires resolution of time with respect to elastic deformations and also inferior link weight to payload ratio along with the

enhanced energy efficiency. On the other hand, in case of medical surgery or nuclear hazard applications tele-operated manipulators depicts similar concept like space manipulator.

Therefore it can be understood that in case of flexible robot which is having less control inputs as compared to number of dof which explains the design control parameters for flexible manipulator is more difficult than rigid link manipulator. Moreover, the execution of a whole system will definitely require more number of sensors. Among these limitations the flexible robot manipulator is unlikely used in various industrial applications due to the benefits of inertial decoupling of the joint actuator and the link, reduced kinetic energy consumption and undesired collisions offered by obstacles as well as humans.

d) Deficient manipulator

A robot is called a deficient robot if it possesses less than six degrees of freedom and it cannot be positioned or oriented freely in space. Adept-one SCARA manipulator is an example of a deficient robot.

1.3.3 Classification by actuation

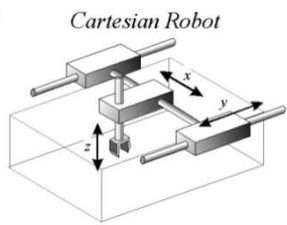
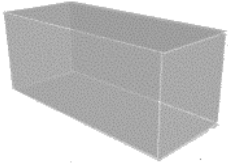


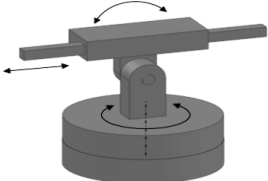
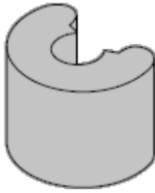
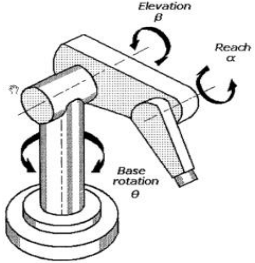

Actuators are basically transmitting power as a motion to drive rigid or flexible links attached to any mechanism or manipulator. Actuators can be categorized mainly as electrical, pneumatic and hydraulic. There are other types of actuation that can be considered as shape memory alloys (SMA), piezoelectric, magnetostriction and polymeric. Among all considered actuators the basic and most preferred actuators are electric which are powered by AC or DC motors because of their cleaner, precise and quieter operations as compared to other actuators. Electric drives are more efficient and precise at high speed because of gear box used and also in case of stepper motor precise motion and high torque are possible. However, for high speed and heavy load carrying capacity electric motors do not support as compared to hydraulic or pneumatic actuators. Hydraulic drives are reasonable because of their high speed and efficient torque or power ratios. Therefore, hydraulic actuators focused manipulators are mainly used for lifting heavy loads. Major drawbacks of hydraulic actuators are noisiness, leakiness of fluid used and heavy pumps. Besides hydraulic actuated manipulators pneumatic actuators are similar but they do not have precise motion and difficulty in control of end effector.

1.3.4 Classification by workspace

In general, workspace of any manipulator can be defined as the total volume covered by the end effector as the manipulator finishes maximum possible movements. Workspace

can be determined by the limits of joint variables and geometry of the manipulator. There are basically two types of work spaces which are reachable and dextrous; reachable workspace can be understood by the total locus point traced by end effector and subset of these traced point of end effector while giving arbitrary orientation is known as dextrous workspace. But practically dextrous workspace is suitable only for idealized geometries and generally it does not possess for industrial manipulators. The above mentioned configurations and their corresponding workspaces are given in Table 1.2.

Table 1.2 Configurations and workspace

Types of robot	Structure	Joint type	Shape of the workspace
Cartesian		P-P-P	
Cylindrical		R-P-P	
Spherical		R-R-P	
Revolute		R-R-R	

a) Cartesian robot

Cartesian robots are also known as gantry robots, having three orthogonal arrangements of prismatic joints as shown in Table 1.2. Position of wrist centre point of Cartesian robot can be appropriately determined by associated coordinate with the three prismatic joints. Workspace of Cartesian manipulator

will be rectangular or cube in shape, so that performed work will always be within the space of joint motion. The robot configuration will be PPP linearly arranged three mutual axes, and the motion will be in X, Y and Z direction.

b) Cylindrical robot

Cylindrical robot will possess at least one revolute joint along with two prismatic joints (RPP) that completely creates cylindrical coordinates of end effector. The workspace of this configuration is limited by two concentric structure of cylinder of finite length as shown in Table 1.2. This robot comprises of one revolute joint in base and other two joints having linear motion along Z and Y directions. First joint of rotation along Z direction gives advantage to move rapidly and efficient pick and place operation in assembly.

c) Spherical robot

Spherical robots having first two joints revolute with intersecting axes and last joint will be linear or prismatic joint (RRP) that resembles spherical coordinates of all three joints. The workspace of this robot is limited by two concentric spheres as shown in Table 1.2. First link rotates along the Z-direction with the base and second joint rotates in Y-direction while last joint moves left and right linearly overall crates sphere envelope.

d) SCARA robot

SCARA (Selective Compliance Assembly Robot Arm) robot manipulator is basically designed for assembly tasks as it provides vertical axis rigidity and compliance in the horizontal axis. It mainly contains three revolute (3-dof revolute) and one prismatic (P) joints altogether known as RRRP manipulator as sown in Figure 1.4. In this type robot first three joints are parallel to each other and having downward direction gravity. This manipulator is used mainly in aeroplane parts and electronic parts assemblies. Adept one and SCARA AR-i350 is the example of this robot configuration.

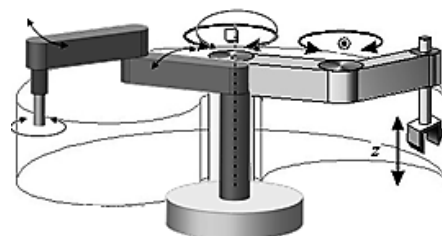


Figure 1.4 SCARA robot

The major advantage of this manipulator is small installation area and works as higher dof manipulator. The minimal acquired area of this manipulator design leads to minimizing cost and maintenance. However, having less dof or joints will be limitations for real world application in addition with singularity, obstacle avoidance and limited workspace.

e) Articulated/revolute manipulator

A robot manipulator having all three joints revolute (RRR) is said to be articulated or anthropomorphic manipulator. The anthropomorphic resembles the design of human hand that includes shoulder, waist and elbow joints. The workspace of this type of robot is quite complex, mainly crescent-shaped cross-section. This can swept the volume in space bounded by spherical outer surface and consisting scallops of inner surface to the constraint joints. Very well-known examples of this category are PUMA, KUKA, DENSO, IRB 6 etc as shown in Figure 1.5. This type of manipulator generally having 6-dof consisting first three revolute joint in X, Y and Z axes and last three joint will be pitch, yaw and roll.

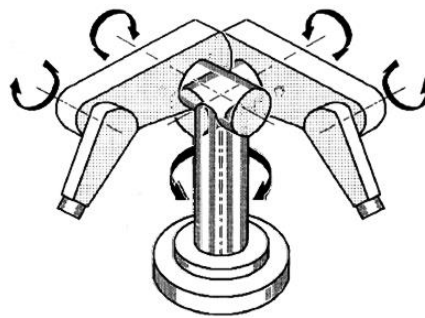
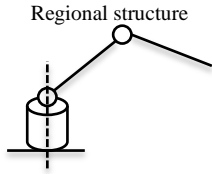
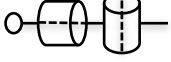
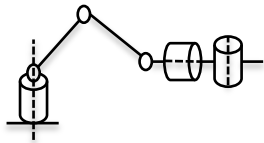

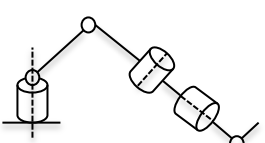


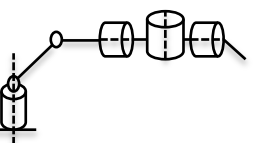

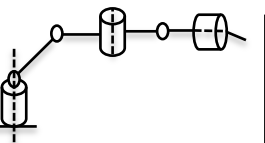

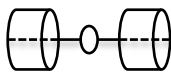
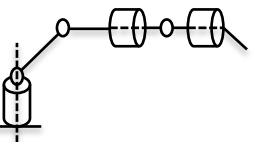

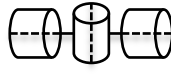
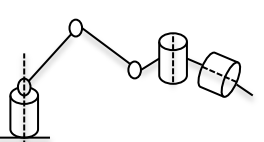



Figure 1.5 Revolute robot

Many serial manipulators are designed in regional and orientational structure so that it can overcome the complexity of kinematic analysis. The joint variables in regional structure will help for major displacement or positioning of end effector but in case of prismatic joint its doesn't support for orientation of end effector. Now the revolute manipulators can also be classified as type A1, A2, B1, B2, C and D. The structures of the types of robots are shown in the Figure 1.5. Examples of different types of robot are given in Table 1.3.

Table 1.3 Configurations of 6-dof revolute manipulators

<p>Orientation structure</p>	<p>Regional structure</p> 	
	<p>Type: A1 IRb6</p>  <p>Example ASEA</p> 	<p>Type: A2 I400</p>  <p>Example ABB IRb-</p> 
	<p>Type: B1 T3</p>  <p>Example Cincinnati Milacron</p> 	<p>Type: B2</p>  <p>Example STÄUBLI RX160</p> 
	<p>Type: C</p>  <p>Example Unimation PUMA 560</p> 	
	<p>Type: D</p>  <p>Example Fanuc-M-10iA-850p</p> 	

1.3.5 Classification based on regional structure

In case of regional structure of manipulator determines the major displacement of end effector having three degrees of freedom. Now let us observe the first regional part of manipulator. In this regional part of manipulator both rotational and translational can be

used to position the end effector of manipulator. The direction of axes can be arbitrarily in space. But good practice is to position of mechanism will always be parallel joint axes along with fixed joint coordinate frame x, y and z . Therefore the translation and rotation can be represented as R_x, R_y, R_z and P_x, P_y, P_z , (see Table 1.4). This representation can make 6^3 possible combinations of these six joints. whereas it is not always important that it could make spatial mechanism for example $P_x P_x P_x$ or $P_x P_y R_z$ combinations cannot move in at least in one direction of the x, y and z axis. So to have spatial mechanism it is required to have motion in all three directions. Therefore motion of a single joint should always be independent of other two joint motions.

Table 1.4 Classification based on regional structure

Different configurations					
$R_x R_x R_y$	$R_x R_x R_x$	$R_x P_x R_y$	$R_x P_y P_x$	$P_x R_x P_z$	
$R_x R_x R_z$	$R_x R_y P_x$	$R_x P_x R_z$	$R_x P_z P_x$	$P_x R_y P_y$	
$R_x R_y R_z$	$R_x R_y P_y$	$R_x P_y R_y$	$P_x R_x R_x$	$P_x P_y R_x$	
$R_x R_y R_y$	$R_x R_y P_z$	$R_x P_y R_z$	$P_x R_x R_y$	$P_x P_y R_y$	
$R_x R_y R_z$	$R_x R_z P_x$	$R_x P_z R_y$	$P_x R_x R_x$	$P_x P_y P_z$	
$R_x R_z R_x$	$R_x R_z P_y$	$R_x P_z R_z$	$P_x R_y R_x$		
$R_x R_z R_y$	$R_x R_z P_z$	$R_x P_x P_y$	$P_x R_y R_z$		
$R_x R_z R_z$	$R_x P_x R_x$	$R_x P_x P_z$	$P_x R_x P_y$		

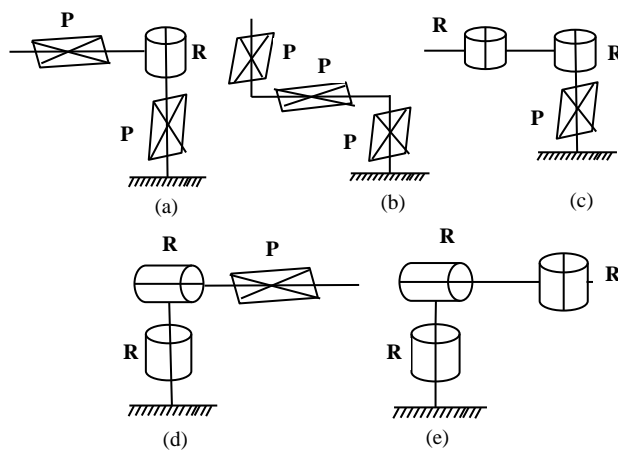


Figure 1.6 Regional mechanism of robot manipulators (a) PRP cylindrical manipulator, (b) PPP Cartesian manipulator, (c) PRR SCARA manipulator, (d) RRP spherical manipulator and (e) RRR revolute manipulator.

As per Table 1.4, only few of these combinations are used to form industrial manipulator. In general there are five types of positioning mechanism are found in industrial manipulator as shown in Figure 1.6. Another part of manipulator is orientation part that is required to have at least 3-dof joints to achieve desired task and combination of three rotations variables can yield 27 different configurations of wrist. However, configuration with consecutively perpendicular axes can be considered as

shown in Figure 1.6. In case the first rotation is about x axis then next alignment of axis should be in y or z axis direction. If the next rotation is about y axis, then last axis of rotation should be about z or x axis. Considering these criteria there can be 12 different configurations, only the difference will be there on the basis of attachment orientation with moving link of the manipulator.

1.3.6 Classification by motion characteristics

Robot manipulators can also be classified according to their nature of motion such as; Robot manipulator can possess three different characteristics of motion namely planar, spherical and spatial. A manipulator will be known as planar if the joint associated with the links arbitrary translates and rotates in the plane. In planar manipulator all moving links performs planar motion that is all joint axis are parallel to each other. Prismatic and revolute joints are only allowable lower pairs for planar mechanism. The motion of planar joints are limited to SE (2) group, considered as a 3D subgroup of SE (3). But in case of spherical manipulator all the links accomplish spherical motion with respect to common fixed point and all other motion of joints can be determined by the radial projection of unit sphere. Revolute joints are limited to the construction of spherical linkage that can be used as pointing device. On the other hand, manipulator moving in 3-dimensional space or belongs to SE (3) group and possesses three coordinates is said to be spatial manipulator. Based on the observation of growing trend of industrial manipulator application the commonly used manipulator are spatial.

1.3.7 Classification by application

Regardless of structure, dof and workspace, manipulator can also be categorized as per their application. It can be classified as follows;

- a) Assembly robot manipulator
- b) Underwater
- c) Space
- d) Agriculture
- e) Mining
- f) Surgical and rehabilitation
- g) Domestic
- h) Educational

In case of assembly robot manipulator the major application will always pick and place, loading/unloading, welding, painting, inspection, sampling, manufacturing etc. As per configuration and design, mostly industrial robots are anthropomorphic, which includes shoulder, an elbow and wrist. Therefore, finally most of the manipulator possesses 6-

dof to achieve desired position and orientation the basic difference is the application. The basic objective of manipulator is to have high resolution, energy efficient and can carry maximum load. Hence, all distinguished manipulator possesses different kinematic structures.

1.4 Basic kinematics

This section discussed some basics of kinematics of rigid body and further introduced different types of mechanism and parameters associated with it. Kinematic Chain may consist of rigid/ flexible links which are connected with joints or kinematics pair permitting relative motion of the connected bodies. For example, a rotational joint acts as a hinge and allows only a relative rotation between the connected bodies about the axis of the joint. The relative movements allowed by a joint are referred to as the joint variables or the internal coordinates. The rotational joint has only one joint variable and that is the relative rotation between the connected bodies.

As we know about the different types of kinematic chains for example serial, parallel or hybrid which may be open, closed or branched. For the positioning of end effector or base it is required to have understanding of kinematics of rigid body systems. The design of the links and joints of any mechanism decides the orientation or positional properties that affect the overall kinematic chain. There are basically two types of kinematics of any mechanism namely forward kinematics and inverse kinematics. The forward kinematics problem is concerned with the relationship between the individual joints of the robot manipulator and the position and orientation of the tool or end-effector. The forward kinematics of any manipulator or mechanism can be determined with given joint variables that yield the position and orientation of end effector. The joint variables may be revolute or prismatic depending of types of joints used. On the other hand the second problem of kinematic is resolution of inverse kinematics. Inverse kinematics can be defined as resolution of joint variables in terms of given end effector position and orientation.

Systematic and generalized ways of kinematic analysis are vectors and matrix algebras that represents and describe the location of end effector and joint variables with respect to defined reference frame. As we know that the joints can be rotate or translates so there is basic matrix algebra known as 3X3 rotation matrix is used to resolve the position and orientation of end effector or tool. This rotation matrix further modified with 4X4 homogeneous transformation matrix to evaluate the translation of the links in 3-dimensional space. This representation concept was first applied by Denavit-Hartenberg.

The second problem associated with robot manipulators is inverse kinematic solution. In order to calculate the exact position and orientation of the end effector of robot manipulator to reach its task the inverse kinematics solution is mandatory. The inverse kinematics of manipulator is essential not only for design synthesis but also to reach desired position. The major problem associated with inverse kinematic formulations is computational and mathematical complexity due to higher degree of polynomial half tangent equations which does not guarantee closed form solution. This problem is main area of research now a day in the field of animations and molecular mathematical modelling. Overall it can be summarized that there are two basic problems of kinematics which are forward and inverse kinematics.

Now is it significant to describe different parameters that create kinematic and mathematical modelling of various manipulator of any mechanism. This section pertain only brief introduction of degree of freedom (dof) and various types of joints that altogether conclude the mechanism of a system without considering any forces/torque. In the later chapter different methodologies to obtain kinematics solutions will be discussed in detail.

1.5 Motivation

As we know that in the track of kinematic analysis, rotational, translational, DH-algorithm and homogeneous matrices have shown their importance in the application of positional analysis of different manipulators. From many decades these method have been adopted by various researchers and implemented in different number of manipulators. However, ensuring the absence of proper mathematical formulations with less computational and mathematical cost which leads to decreasing in many applications, where quick calculations are required. These techniques fail to prove when the manipulator having higher number of dof's. In general when there are higher dof manipulator the inverse kinematic formulations are much more difficult due to non-linear, time varying and transcendental functions. There are many other tools and techniques are available to solve inverse kinematic problem for example algebraic, Jacobian, or geometric, analytical, pseudo inverse Jacobian etc. These methods are conventional and they do not provide exact solution. On the other hand, alternative of these techniques for representing and solving kinematics problem are quaternion algebra, Lie algebra, exponential algebra, epsilon algebra and screw theory which are being used from many years due to less mathematical operations. So the quaternion algebra is much more powerful method for resolving kinematic problem of any manipulator. Quaternion can be used for rotation as well translation of rigid body in Euclidian space.

Considering the complexities involved in the process of modelling and consequently solving the inverse kinematic problem for achieving precise, optimized and faster solution for real time application. The present research problem in design, the research issues will primarily focus on selecting/developing an appropriate tool for achieving the objectives after validating them on various configurations of industrial robots.

On the other side of these conventional techniques, intelligent or soft computing techniques are widely used to find out the inverse kinematic solutions. This intelligent technique includes artificial neural network, hybrid ANN, fuzzy logic, hybrid fuzzy, metaheuristic algorithms and biologically-inspired approaches. In past decades, many others have adopted these technique because of their less computational and mathematical cost. These techniques are useful when the manipulator having higher number of dof's where generally conventional method fails. So the ultimate goal of this dissertation is to find out inverse kinematic solution using these techniques and to develop novel method for resolving inverse kinematic problem for any configuration of robot manipulator.

1.6 Broad objective

The major objective of this dissertation is to resolve inverse kinematic problem. As per survey and analysis of various literatures in this field of manipulator kinematics recommends that there is obvious requirement of some novel technique for solving higher dof manipulator kinematics. It is also requires to produce inverse kinematic solution efficiently and should be capable of online control of manipulator. Therefore, this work is planned with following major objectives:

- 1) To carry out critical study of different tools and techniques suitable for solving inverse kinematic problems.
- 2) To develop the inverse kinematic model of various robot manipulators and to adopt some existing techniques for solution of inverse kinematics of selected robot manipulator configurations.
- 3) Development of new algorithm and mathematical model for resolving and simulating inverse kinematics.
- 4) To analyze the efficiency of newly developed method and comparison with the obtained solution through other existing techniques.
- 5) To recommend the appropriate techniques for solving inverse kinematics problem for various application.

1.7 Methodology

Kinematic analysis and synthesis of planar or spatial manipulators always need to follow through the nonlinear equations which can be complex and time consuming. The conventional methods are less efficient which can be the greater objective for any researcher to develop novel method to overcome the stated problem. Considering the above stated objective, kinematic relationship and mathematical modelling is required to develop. Therefore, to accomplish the aforesaid major objectives of this research work and to resolve perfect solution of inverse kinematic one should develop efficient method. The adopted methods and steps for achieving objective have been planned as follows:

- *Review of literature:* Considering various configurations of revolute and prismatic joints of manipulators and their classification on the basis of their structures, mechanism, actuations, workspaces, motion properties, applications etc. have been studied. Analysis of the literature survey with the prime importance of mathematical modelling and kinematics analysis of robot manipulator along with the problem associated with the developed techniques has been done. Literature review has been done related to different methods like algebraic, analytical, intelligent techniques and optimization algorithms etc.
- *Configurations of manipulator:* On the basis of literature survey it has been deliberated their outcome and associated problem so as to select appropriate model of manipulator. Starting from 3-dof manipulator up to 7-dof redundant manipulator has been taken for the resolution of inverse kinematic. Different configurations for 3-dof manipulator and 6-dof revolute manipulator have been proposed and among them few configuration has been selected for kinematic analysis.
- *Mathematical modelling and kinematic analysis:* Different configurations of robot manipulator have been considered and their mathematical modelling is presented. All considered manipulator belongs to the category of serial manipulator and different configurations for 6-dof and 3-dof manipulator has been analysed. Denavit-Hartenberg algorithms have been used for kinematics formulation and simulation of different joint configurations of robot manipulator and later the obtained solution of inverse kinematics has been compared with quaternion algebra. The forward and inverse kinematics of adopted configurations has been done along with their workspace analysis and detailed derivation of kinematics.
- *Intelligent approach:* Artificial neural network (ANN), fuzzy logic and hybrid techniques from the soft computing domain have been widely used in last

decades. These intelligent techniques do not required higher mathematical formulation and are capable of solving NP-hard, nonlinear and higher degree of polynomial equations. As per review of literature different models of ANN, adaptive neural fuzzy inference system (ANFIS) has been adopted for the resolution of inverse kinematics of robot manipulator. Although these intelligent techniques are not new in this field but few selected models of ANN along with ANFIS and hybrid ANN methods has been adopted for the comparison. There are different optimization techniques like Particle swarm optimization (PSO), genetic algorithm(GA), artificial bee colony (ABC), biogeography based optimization(BBO), teachers learners base optimization(TLBO) etc. have been applied for training of multi-layer perceptrons (MLP) neural network for the prediction of invers kinematic solution of robot manipulator.

- *Optimization algorithm:* Different optimization algorithms like GA, BBO, PSO, TLBO and ABC have been adopted for the solution of inverse kinematic of robot manipulator and novel Crab intelligence based optimization algorithm (CIBO) has been proposed. These algorithms are compared with new developed CIBO algorithm. These adopted optimization algorithms does not requires any computation of Jacobian matrix only it needs forward kinematic equations which can be easily developed.
- *Discussion and recommendations:* Conversation about the results obtained through the adopted methods and proposed method for robot manipulator kinematics. Simulation results for kinematics and workspace analysis have been addressed. Future recommendations for the adopted configuration of manipulator and scope of the future work considering improvements of the quality and efficiency are given.

1.8 Organization of the thesis

Current *chapter 1* is the Introduction part of the dissertation that provides brief description of history of evolution of robots, types of manipulator, classifications and application in various fields. Forthcoming chapters apart from introduction chapter are organized as follows:

Chapter 2 delivers review of literature on the basis of various aspects of the robot manipulator like mechanism, actuation, workspace analysis, motion types, different components considered, application, intelligent controls and optimization. Some of the significant literatures are summarized in table and brief explanations of the outcome and deficits with respect to manipulator and configurations are discussed. Finally the

objectives of the research work are determined and explained on the basis of literature analysis.

Chapter 3 provides the brief description of selected configurations of industrial manipulators for inverse kinematic analysis. In later section, different methodologies to solve inverse kinematic problem is discussed in brief.

Chapter 4 delivers the kinematic analysis and mathematical modelling of various configurations of robot manipulator. A brief discuss of various conventional techniques like algebra, analytical method, iterative method, numerical method, geometric method, homogeneous matrix, DH algorithm and quaternion algebra are presented. Classification of 3-dof and 6-dof serial manipulators along with DH parameters and their mathematical modelling has been discussed. The inverse and forward kinematics of adopted manipulator is derived using adopted method.

Chapter 5 proposes various intelligent techniques like ANN, ANFIS and hybrid ANN for the prediction of inverse kinematic solution of robot manipulator. Different types of ANN models like multi-layered perceptron (MLP), polynomial perceptron network (PPN) and Pi network are explained in brief and their application towards the solution of inverse kinematics has been presented. MLP model is hybridized with many optimization techniques like GA, GWO, PSO, TLBO and proposed CIBO algorithm to increase the performance of MLP network. The end effector position is considered as input for the training of ANN models and ANFIS training is also completed similar to ANN training. Application of these algorithms and strategies to use ANN and ANFIS is addressed.

Chapter 6 discusses about the adopted optimization algorithms for the solution of inverse kinematics of robot manipulators. In this chapter forward kinematics equations are used to find out the joint variables of robot manipulator using Euclidean distance norm. Crab Intelligence Based novel Optimization algorithm (CIBO) has been proposed in detail for the solution of inverse kinematics of robot manipulator. For the comparison of the developed optimization algorithm various metaheuristic algorithms are briefly explained.

In *Chapter 7* presents the kinematic results achieved through all adopted techniques and comparison has been made with other existing techniques. Forward and inverse kinematics along with the workspace analysis and joint angle behaviour has been addressed and compared. Programmed output in the form of tables and graphs are depicted in this chapter.

Chapter 8 presents the conclusions of the dissertation and future research guidance with summary of contribution.

1.9 Summary

In the current chapter, the general synopsis of the different types of robot manipulator, classifications, history of developments are presented. Configurations of 6-dof revolute manipulators and combination of 3-dof manipulator are presented. The chronological progresses of some selected manipulators are presented and also current status has been briefed. Basic applications of kinematics and objectives are also discussed in this chapter.

Chapter 2

REVIEW OF LITERATURE

2.1 Overview

With advancement of robot technology and ever increasing application of robots in various walks of life, robotic research has gaining appreciable momentum over the years. Newer configurations, smart behaviours, autonomous robotics, high level intelligence, uncertainty in environments have been the various areas for researcher in robotics. All these areas are naturally connected with the subject of robot kinematics; both forward and inverse. Inverse kinematics solution for serial manipulator is difficult task, because the solution is not unique due to nonlinear, uncertain and time varying nature of the governing equations. There are various software's and algorithms to simplify the inverse kinematics of robot manipulator. During 1980s wrist orientation was decoupled from the translation by the arm by using wrist axes that intersect with the arm axes. But the major problem singularities with the robot arm were no longer back driven, it limits with the structure of the manipulator. There are various techniques for solving inverse kinematic problem. Since many methods have been presented to solve the IK problem such as homogeneous transformation method, geometric method, dual number approach and continuation method. However, the problem involves the solving of highly non-linear equations. Many papers have presented algorithms giving analytical solutions for the Inverse Jacobean. The formulation of the Jacobean matrix and its inverse has to be done within a very short span of time for real-time implementations. Some mathematical methods (such as MACSYMA, REDUCE, SMP and SEGM) are well- known, efficient tools in terms of their speed and accuracy. In the area of robotics, researchers such as (Kircanski, 1985 and Vukobratovic, 1986), (Morris 1987), (Hussain and Nobie, 1985), and (Tsai and Chiou, 1989), have used these kinds of tools for deriving the direct kinematics, Jacobian, and reverse Jacobian closed- form equations. Analytic solutions, however, are only used for simple robot manipulators.

2.2 Survey of tools used for inverse kinematic solution

Besides above mentioned approaches, researchers are up to developing newer techniques which would make the process easier and faster. In the current research different methodologies used by researchers have been studied. They are as follows:

- Analytical solution
- Iterative solution
- Geometric solution
- Quaternion algebra
- Theory of screws
- Exponential rotational algebra
- Lie algebra
- Artificial neural network (ANN)
- Hybrid ANN
- Adaptive neuro-fuzzy inference system (ANFIS)
- Genetic algorithm
- Simulated annealing
- Particle swarm optimization
- Bee algorithm
- Fuzzy learning algorithm
- Neuro-fuzzy
- Fuzzy-neuro

Based on a comprehensive survey of literature, a list of some the major work done in the area is presented in Table 2.1.

Table 2.1 List of some important literatures

Sl.	Year	Author	Title	Type	Contribution
1	1990	Funda and Paul [2]	A computational analysis of screw transformations in robotics	PUMA 560	Proposed a representation method based on screw displacement and made their comparison on the basis of computational cost. In this work they have determined the rotational and translation representation of line for the application of general displacement of rigid body. They have compared four different mathematical formulizations which direct affects the rotation and translation of rigid body. The proposed methods are dual orthogonal matrix, dual unit quaternion, dual special unitary matrix and dual Pauli spin matrix.
2	1990	Funda et al. [3]	On homogeneous transform, quaternions, and computational efficiency	PUMA 560	Proposed work is based on the inverse kinematic solution of robot manipulator using quaternion vector pair based method. In this work the proposed method is applied to solve inverse kinematic of the PUMA robot manipulator and comparison has been made on the basis of computational cost.
3	1995	Mitsi et al. [4]	Optimization of robot link motion in inverse kinematic	5-dof revolutespatial and	Proposed inverse kinematic solution of 5-dof redundant robot manipulator based on conventional optimization

			solution considering collision avoidance and joint limit.	redundant	method. In this work penalty function optimization method adopted for the problem resolution. Moreover forward kinematic solution is done using standard analytical method which is later used to formulate the objective function for the proposed optimization algorithm.
4	1998	Nearchou [5]	Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm	Puma 566	Proposed inverse kinematic solution of redundant manipulator using modified genetic algorithm. They have implemented some assumptions; first they considered that the manipulator may be redundant and articulated. Then the second assumption is that the manipulator is in moving object of its workspace, and last assumption is that they are not considering dynamics of the manipulator. Thereafter, genetic algorithm is used in two different manners, first joint displacement ($\Delta\theta$) error minimization and the second approach is based on positional error of end effector.
5	1999	Ozgoren [6]	Kinematic analysis of a manipulator with its position and velocity related singular configurations	6-dof revolute	Proposed inverse and forward kinematics of general 6R manipulator considering both position and velocity using exponential rotation matrix method. They have also investigated the singular configuration related to the inverse position analysis termed as (POSCs) i.e. position

					related singular configurations and other with the velocity is termed as (VESOs).
6	2000	Karlik and Aydin [7]	An improved approach to the solution of inverse kinematics problems for robot manipulators	6-dof revolute	Proposed inverse kinematic solution of 6-dof robot manipulator using structured artificial neural network based method. In this work they have used DH-algorithm to formulate forward kinematic equation so as to complete the dataset for training ANN model and the adopted ANN model is MLP.
7	2002	Her et al. [8]	Approximating a robot inverse kinematics solution using fuzzy logic tuned by genetic algorithms	2-dof and 4-dof planner	Proposed inverse kinematic solution of 2 and 4-dof planar robot manipulator using fuzzy logic together with the genetic algorithm. They have used triangular membership function for fuzzy logic and center of gravity is used for the defuzzification. These parameters are later tuned by genetic algorithm for the surety of exact inverse kinematic solution.
8	2002	Rueda [9]	Manipulator kinematic error model in a calibration process through quaternion-vector pairs	PUMA 560	Proposed inverse kinematic solution of PUMA 560 robot manipulator using quaternion vector pair based method. In this work they have calculated the geometric error for each joint variables and link. They have used differential algorithm for the resolution of inverse kinematic to

					achieve they formulated the objective function as position error and orientational error.
9	2004	Koker et al. [10]	Study of neural network based inverse kinematics solution for a three-joint robot	3-dof revolute	Proposed inverse kinematic solution of 3R robot manipulator using artificial neural network technique. In this work forward kinematic is resolved using analytical solution which is later used to generate input dataset for the ANN training.
10	2005	Bingul [11]	Comparison of inverse kinematics solutions using neural network for 6r robot manipulator with offset	6-dof revolute	Proposed inverse kinematic solution of 6-dof robot manipulator using artificial neural network technique. In this work forward kinematic is resolved using analytical solution which is later used to generate input dataset for the ANN training. Back propagation algorithm is used to calculate the output error in this work.
11	2005	Xu et al. [12]	An analysis of the inverse kinematics for a 5-dof manipulator	PArm (PRRPP) 5-dof	Proposed inverse kinematic solution of 5-dof robot manipulator using analytical method. In this work DH-algorithm is used to resolve the forward and inverse kinematic of adopted manipulator later they have discussed on trajectory planning and singularity analysis of the manipulator.

12	2005	Koker [13]	Reliability-based approach to the inverse kinematics solution of robots using elman's networks	6-dof revolute	Proposed inverse kinematic solution of 6-dof robot manipulator using reliability based artificial neural network technique. In this work forward kinematic is resolved using analytical solution which is later used to generate input dataset for the ANN training. Back propagation algorithm is used to calculate the output error in this work.
13	2005	Mayorga and Sanongboon [14]	Inverse kinematics and geometrically bounded singularities prevention of redundant manipulators: an artificial neural network approach	3-dof revolute planar redundant	Proposed artificial neural network technique to solve inverse kinematics of the 3-dof revolute planar manipulator and also calculated the effective geometrically bounded singularities prevention of redundant manipulators.
14	2006	Aydin and Kucuk [15]	Quaternion based inverse kinematics for industrial robot manipulators with euler wrist	6-dof revolute	Proposed inverse kinematic solution of 6-dof industrial manipulator with Euler wrist using quaternion vector pair method. They have given detail derivation of forward and inverse kinematic of RS, RN and NS type robot manipulators.
15	2006	Hasan et al. [16]	An adaptive-learning algorithm to solve the inverse kinematics problem of a 6 d.o.f serial	FANUC M710i	Proposed adaptive learning plan of ANN for the solution of inverse kinematic of 6-dof manipulator. Moreover they have tried to resolve singularity and uncertainty problem

			robot manipulator.		of the adopted configuration of the manipulator. In this work ANN model have been trained using analytical solution of the adopted manipulator. Generated datasets using kinematics equations are used to trained and test the adopted model of ANN. They have concluded that the proposed model of ANN does not need to have previous information of the kinematics of the system that learns through the ANN model application.
16	2006	Tabandeh et al. [17]	A genetic algorithm approach to solve for multiple solutions of inverse kinematics using adaptive niching and clustering.	3-dof revolute	Proposed inverse kinematic solutions of 3-dof PUMA manipulator for the major displacement propose. In this work they have adopted genetic algorithm with adaptive niching and clustering. Genetic algorithm's parameters are set by adaptive niching method which is later required the forward kinematic equations for the solution of inverse kinematic of adopted manipulator. Forward kinematic is simply calculated by standard analytical method. Thereafter for processing the output filtering and clustering is also added to the genetic algorithm.
17	2007	Xie et al. [18]	Inverse kinematics problem for 6-dof space manipulator based on the theory of screws	6-dof revolute	Presented inverse kinematic solution of 6-dof mechanical arm with the application of free flying space using screw algebra based method. In this work they have completed the simulation model for adopted manipulator along with

					kinematic analysis.
18	2007	Husty et al. [19]	A new and efficient algorithm for the inverse kinematics of a general serial 6r manipulator	6-dof revolute	Proposed inverse kinematic solution of 6-dof revolute robot manipulator using new efficient algorithm based on analytical method. In this work they have used elimination technique to reduce the complexity of inverse kinematic formulation. They have used general 6-dof revolute robot manipulator geometry for the elimination information.
19	1994	Park [20]	Computational aspects of the product-of-exponentials formula for robot kinematics	3-dof spatial revolute	Proposed forward kinematic analysis of 3-dof revolute spatial robot manipulator using product of exponential algebra based method. In this work they have also focused on the calculation of Jacobian matrix using POE method.
20	2008	Pham [21]	Learning the inverse kinematics of a robot manipulator using the bees algorithm	3-dof revolute	Proposed inverse kinematic solution of 3-dof robot manipulator using Bee algorithm. In this work they have compared three different methods like evolutionary algorithm, neural network back propagation method and bee algorithm. Neural network structure is optimized by using bee algorithm to predict joint variables of the robot

					manipulator.
21	2008	Alavandar and Nigam [22]	Neuro-fuzzy based approach for inverse kinematics solution of industrial robot manipulators	2-dof revolute and 3-dof revolute	Proposed inverse kinematic solution of 2-dof and 3-dof planar manipulator using adaptive neural fuzzy inference system (ANFIS). In this work, they have adopted Sugeno type fuzzy architecture and hybridized with simple neural network for the prediction of inverse kinematic of planar manipulator.
22	2008	Albert et al. [23]	Inverse kinematic solution in handling 3r manipulator via real-time genetic algorithm	3-dof revolute	Proposed inverse kinematic solution of 3-dof revolute robot manipulator using real time genetic algorithm. In this work end-effector displacement from its initial point to desired point has been optimized using genetic algorithm. Genetic algorithm crossover selection is based on new method which is known as dynamic multi-layered chromosome (DMCC) to produce two offspring's. The GUI simulation has been verified with GA and DMCC.
23	2008	Dutra [24]	New technique for inverse kinematics problem using simulated annealing	2-dof revolute	Proposed inverse kinematic solution of 2-link planar manipulator using simulated annealing method. In this work standard analytical solution of forward kinematic is presented using forward kinematic equation the

					displacement based error minimization objective function is used for the simulated annealing approach.
24	2009	Sariyildiz and Temeltas [25]	Solution of inverse kinematic problem for serial robot using quaternions	6-dof revolute	Proposed inverse kinematic solution of 6-dof revolute robot manipulator based on quaternion in the framework of screw theory. In this work they used quaternion with the screw theory to reduce the computational cost for inverse kinematics derivation.
25	2009	Ayiz and Kucuk [26]	The kinematics of industrial robot manipulators based on the exponential rotational matrices	6-dof revolute	Proposed inverse and forward kinematics of 6-dof robot manipulator based on exponential rotation matrix method. In this work they have used the exponential based method for derivation of inverse kinematics of NS and RS type robot manipulator.
26	2009	Martin [27]	A method to learn the inverse kinematics of multi-link robots by evolving neuro-controllers	SCARA	Proposed inverse kinematic learning of 3-dof planar and SCARA manipulator using neuro-controller. Furthermore, they have presented the some issues of neural network learning such as classical supervised learning scheme which generally converge in local optimum solution. Therefore they have applied neuro-evolution algorithm for the global optimum solution of the inverse kinematics of the selected manipulator. In this work DH-algorithm is used to generate the input data set

					for the neural network algorithm. They have reduced the drawback of the gradient descent learning of ANN model with the help of evolutionary algorithm.
27	2010	Wenjun et al. [28]	Numerical study on inverse kinematic analysis of 5R serial robot	5-dof revolute	Proposed a mathematical modelling of 5-dof robot manipulator using conventional method. In this work they have focused on the inverse kinematic and forward kinematic solution of robot manipulator. Later section deals with the application of genetic algorithm for the optimization of joint variables of the adopted manipulator.
28	2010	Chiddarwar and Babu [29]	Comparison of RBF and MLP neural networks to solve inverse kinematic problem for 6R serial robot by a fusion approach	6-dof revolute	Proposed MLP and RBF neural network model for the solution of inverse kinematic of the 6-dof serial manipulator. In this work, a fusion approach of these ANN models is used with the forward kinematics of the manipulator. Forward kinematics equations are used to generate the data for training adopted models of ANN. They have proposed the Cartesian path to be followed by the manipulator end effector using the generated ANN inverse kinematic solution. KUKA 6-dof manipulator is tested with the obtained results wherein DH-algorithm is used to generate the input for the ANN models.

29	2010	Hasan et al. [30]	Artificial neural network-based kinematics Jacobian solution for serial manipulator is passing through singular configurations	6-dof revolute	Proposed inverse kinematic solution of 6-dof revolute robot manipulator using artificial neural network based technique. In this work they have also focused on the singularity avoidance using Jacobian based method together with the ANN approach.
30	2010	Cui [31]	Kinematics simulation of an aided fruit-harvesting manipulator based on ADAMS	4-dof	Proposed virtual model of an agricultural robot for fruit harvesting and their kinematics analysis using DH algorithm. In this work, the inverse kinematic is obtained using algebraic method and simulations are carried out using ADAMS.
31	2011	Olaru et al. [32]	Assisted research and optimization of the proper neural network solving the inverse kinematics problem	3-dof revolute	Proposed inverse kinematic solution of 3-dof revolute robot manipulator using neural network technique. In this work DH-algorithm is used to calculate the forward kinematic of the adopted manipulator.
32	2011	Ramírez and Rubiano [33]	Optimization of inverse kinematics of a 3r robotic manipulator using genetic algorithms.	3-dof revolute	Proposed inverse kinematic solution of 3-dof revolute spatial manipulator using genetic algorithm. Forward kinematics formulation has been completed by using DH-algorithms and homogeneous matrix multiplication based method. Fitness function for the inverse kinematic solution is based on the end effectors initial and desired position error which is also known as Euclidean distance

					norm.
33	2011	Zhang [34]	A psgo-based method for inverse kinematics analysis of serial dangerous articles disposal manipulator	Mobile robot with 6-dof revolute manipulator	Proposed inverse kinematic solution of the serial dangerous articles disposal manipulator with multiple degrees of freedom using particle swarm gene optimization algorithm. In this work position and orientation error of end effector is used as an objective function for traditional PSO and modified PSGO method.
34	2012	Köker [35]	A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization	Stanford robot	Proposed hybrid approach which is combination of neural networks and evolutionary techniques (genetic algorithms) to obtain more precise solutions. Three Elman neural networks were trained using separate training sets.
35	2013	Morishita and Tojo [36]	Integer inverse kinematics method using fuzzy logic	3-dof revolute	Proposed integer inverse kinematic solution of multi-joint robot manipulator using fuzzy logic based method. They have evaluated the efficiency of the adopted technique and tested it for trajectory generation and control application
36	2004	Perez and McCarthy [37]	Dual quaternion synthesis of constrained robotic systems	2,3and 4-dof	Proposed dual quaternion algebra based kinematic synthesis of constrained robotic system. They have proposed this method for one or more serial chain manipulator considering both prismatic and revolute

					joints. In this research they have used DH algorithm and successive screw displacement for determining the joint variables for the resolution of end effector position. Then dual quaternions are used to define the transformation matrices obtained through DH algorithm to simplify the design formulations of different types of manipulators.
37	1988	Bendezu et al. [38]	Symbolic computation of robot manipulator kinematics	7-dof anthropomorphic	Symbolic robot arm tool software is introduced to solve inverse kinematic problem.
38	1990	Smith and Lipkin [39]	Analysis of fourth order manipulator kinematics using conic sections	6-dof revolute	Introduced new technique based on fourth order inverse kinematic solution. In this work solution of inverse kinematics problem is considered as pencil of conics.

Study confirms that the number of research publications which appears in various journals, conference proceedings and technical articles verify various aspects of inverse kinematic analysis of robot manipulator. Inverse kinematic solution of robot manipulator can be classified on the basis of different methodology. A lot of literature survey has been done regarding this area, some of which are discussed as follows.

- i) Structural analysis of mechanism
- ii) Conventional method for kinematics
- iii) Intelligent or soft computing approach
- iv) Optimization approach

2.2.1 Structural analysis of mechanism

The major aim of this literature survey is limited to the mechanism serial, parallel or hybrid mainly expected for mobility's of the kinematic chains. As we know that dof or mobility of any mechanism or manipulator is the basic approach for classification or kinematic analysis. Therefore it is require having understanding of different way of dof or mobility for various mechanism and their formulations. On the other hand, working abilities of manipulator or any mechanism can be evaluated from its dof/mobility. A general rigid body in a space having 6-dof that is the maximum know dof of the system. However, there are numerous criteria for the literature survey but typically one can go for the dof/mobility analysis of different mechanism or structure of the manipulator.

The history of the structural analysis related to mobility and about the no. of independent kinematic chains was done by L. Euler. Then afterward in 19th century, the first mechanism analysis and structural formula was generated by [40]-[44] as depicted in Table 2.2. The basic concept of dof is the total number of independent loops (l), dof of the mechanism (M), number of joints (j), moving links (n) dof of kinematic pairs (f), joint constraints (s), no. of passive dof (J_p), no. of over closing constraints (q), loop motion variables (λ) etc. Therefore brief literature survey has been done related to structural formula and the parameters are presented in Table 2.2. Later in the 20th century structural formula and simple groups have developed by [45]-[52] as given below in Table 2.2. Furthermore several novel concept had been generated for the problem of configuration analysis and design synthesis of mechnaism and manipulators such as screw pairs (Sc), some configurations with zero dof ($M=0$), no of variable length links (n_v), general variable constraints (λ_k) and the closed loop constraints ($d_k = 6 - \lambda_k$).

Afterwards in 20th century general mathematical modelling of the determination of dof for any mechanism had been achieved by [53]-[72] as shown in Table 2.2. Therefore following up these research several new parameter for calculation of dof of mechanism

had been introduced these new parameters are screw system for closed loop (r), no. of independent dof's (λ_k), relative displacement of joints (m), coefficient matrix rank ($r(j)$), new formula for independent loops $L = j_B - B - c_b$ where j_B total no. of joints, c_b is total number of fixed links and B total no. of moving links.

Now in starting of 21st century, drastic development of mechanism and manipulation in the field of robotics has been shown. There are several new parameters and structural formula related to the real world applications and implementation has been shown [73]-[74]. They have calculated dof of different mechanism and introduced new parameters for kinematics and structural analysis. New formula for number of independent loops were $L=c-B$, and simple structural formula for mechanism $\sum f_i = \lambda(c-B)$, where, $c = c_b + c_h + c_1$, c_1 .

The kinematic formulations of the mechanism or manipulator are evaluated through a number of task positions and their kinematics to find out the design equations. These formulas are having both structural and joint parameters as unknown. These design equation are mainly based on the kinematic analysis and different parameters. In this area of research during 20th century [46] developed basic theory of open loop serial chain and then this was utilized for different classifications of the structure. Thereafter, this theory was analysed by [47] for structural synthesis and kinematic analysis. This problem of structural synthesis has been done for the closed loop problem. This classification was made on the basis of number of moving or fixed links, closed loops and number of joints. Boden [57] has given spatial and planar configuration related to truss and later defined by number of closed loops. Kolchin [50] has presented the theory of new constraint i.e. passive constraint but that was not for identification of geometric conditions it was only for the general constraint problem of mechanism. The problem related to general constraint was first completed by Voinea et al. [54] through the rank of matrix and unknowns of the angular velocities. In 1963 Ozol has presented the topological properties of the mechanism.

The technique of configuration synthesis was based on graph theory to obtain kinematics chains and mechanism [52]-[63]. Now for the higher dof or complicated structures the kinematics analysis of spatial and planar case was done by [53]. Thereafter, resolving the classifications of structure is completed using the theory of dividing joints by [45]-[52]. In 20th century, [63] and [71] presented computer aided technique for the structural analysis of spatial manipulators. Later [73] introduced computer aided method for planar manipulator or mechanism and then loop formation for cancelling the isomorphism test was introduced by [68] and [72].

Table 2.2 Different mechanisms and mobility

SN	Authors	Equations	Remarks
1	Euler	$L_i = j - l^* + 1$	L_i represents total number of independent loops, l^* represents total number of links, j represents total number of joints
2	Chebyshev [40]	$3l_d - 2j - 1 = 0$ $0 < j - j_d < l^* + \frac{1}{2}l^*$ $j_d > l^* - 3 \quad l_d = n = l^* - 1$	First equation represents the planar mechanism with single dof, j_d represents the total moving joints and $l_d = n$ represents number of moving links
3	Sylvester [41]	$3l^* - 2j - 4 = 0$ $j = n - 1$	This equation represents the planar mechanism with single dof.
4	Grübler [42]	$M_o = 3l^* - 2j - 3$ $3l^* - 2j - 4 + q = 0$ $2l^* - j - 3 = 0$ $3l^* - 2j - 4 + q - C = 0$ $5H - 6l^* + 7 = 0$ or $M = 6(l^* - 1) - 5p_1$	M_o represents the dof of mechanisms. dof depends on the rank of functional determinant First equation is based on planar mechanism. Second Eq. represents the kinematic chain of revolute and prismatic joint. Third eqn. is for planar mechanism with only prismatic joint. Fourth eqn. Eq. is for revolute, cam and prismatic joints. Fifth eqn. represents the dof of spatial mechanism of helical joint.
5	Somov [43]	$l^* - (\lambda - 1)(v + 1) = 2$ $l^* + q + \sum K_u - (\lambda - 1)(v + 1)$ $M_o = (l^* - 1) + \sum f_i - j - 5L + q$ $l^* = 5v + 7, \lambda = 6,$ $v = L - 1, \sum K_u = j_p - 1$	First equation is based on both planar and spatial mechanism. Second one is also for plane and spatial mechanism where $M_o = 1$ Equation third is Somov's universal formula for structure where λ is the general parameter for constraint.

Table 2.2 Different mechanisms and mobility (Continued)

SN	Authors	Equations	Remarks
6	Gokhman [44]	$l(\lambda - 1) - S = 1$ $\sum f_i - \lambda L = 1$ $\lambda(j - L) - S = 1$	<p>First eqn. is for both planar and spatial mechanism, where $S = \sum (\lambda - i)f_i$ is the total no. of joint constraints</p> <p>Second one represents the mobility criterion</p> <p>Last eqn. is ultimate resolution of Euler eqn. using first and second eqn.</p>
7	Koeings [45]	$M = 6n - S$	Koeings also presented eqn. for the spatial mechanism like Gokhman's eqn.
8	Assur [46]	$3n - 2j = 0$	Assur presented eqn. for simple mechanism
9	Muller [47]	$(\lambda - 1)S_s - \lambda l^* + (\lambda + 1) = 0$ $M_o = \lambda n - (\lambda - 1)S_s$ <p>S_s is the number of screw pairs</p>	This eqn. represents the screw pair of the kinematic chain.
10	Malushev [48]	$M_o = 6(l^* - 1) - \sum_{i=1}^5 i p_i + q - n_v$ <p>p_i is the kinematic pairs with i class $i =$ number of joint constraint</p>	This eqn. is the combined approach of Somov and Malushev for mobility with n no. of links, where p_i represents the kinematic pair with no. of constraint i
11	Kutzbach [49]	$M_o = \lambda(l^* - j - 1) + \sum_{i=1}^j f_i$ $M_o = \lambda(l^* - 1) + \sum_{i=1}^j (\lambda - i)f_i$	Kutzbach has also given equation for universal configuration

Table 2.2 Different mechanisms and mobility (Continued)

SN	Authors	Equations	Remarks
12	Kolchin [50]	$M_o = 3(l^* - 1) - 2(P + R + K) - p_2$ <p>P is the number of prismatic pairs R is the number of revolute pairs</p>	This eqn. represents for planar mechanism where R is revolute, P is prismatic and K represents higher pair with pure slip and roll variables whereas, p_2 gives only for slipping and rolling higher pairs.
13	Artobolevskii [51]	$M_o = 6n - \sum_{i=1}^j S_j + \sum_{K=1}^L d_K + q$	This eqn. is also represents the universal mobility for different structure.
14	Dobrovolskii [52]	$M_o = \lambda n - \sum_{i=1}^{\lambda-1} (\lambda - i)p_i + q$ $\lambda = 2, \dots, 6$	Another mobility formula for different structure.
15	Moroshkin [53]	$M_o = \sum_i ip_i - r$ $M_o = \sum_i ip_i - \sum_{\lambda} \lambda p_{\lambda}$ <p>$i = 1, \dots, 5$, and $\lambda = 2, \dots, 6$ $L = j - n$</p>	<p>First eqn. represents the structural form of integral joints</p> <p>Second eqn. represents dof for variable constraints.</p>
16	Voinea and Atanasiu [54]	$M_o = \sum_i f_i - \sum_{K=1}^L r_K - j_p$	This eqn. represents dof for complex mechanism where r_K is rank for screw joints and $\sum_{i=1}^j f_i$ represents the total no. of dof for revolute, helical and prismatic joints.
17	Paul [55]	$L - j + l^* - 1 = 0$	Euler's formula for creating topological situation for planar kinematic chain
18	Rössner [56]	$M_o = \sum_{i=1}^j f_i - 6(j - l^* + 1)$	Similar to Euler's eqn. for mobility

Table 2.2 Different mechanisms and mobility (Continued)

SN	Authors	Equations	Remarks
19	Boden [57]	$M_o = \sum_{i=1}^j f_i - 6(j - l^* + 1) - 3(j - l^* + 1)$	dof eqn. for planar and spatial mechanism
20	Ozol [58]	$M_o = \sum_{i=1}^j f_i - 6L + q$ $M_o = \sum_{i=1}^j f_i - 3L + q$ $M_o = 2(l^* - 1) - j + q$ $M_o = j - 2L + q$	<p>First to third eqn. represents the mobility with variable and excessive constraint</p> <p>Last eqn. represents the mobility for cylindrical mechanism.</p>
21	Waldron [59]	$M_o = F - r$	Eqn. for mobility of closed loop mechanism.
22	Manolescu [60]	$M_o = \sum_{i=l^*+1}^5 (6-i)p_i - (6-d)L$	Eqn. for closed loop mechanism with elementary parameter
23	Bagci [61]	$M_o = 6(l^* - 1) - \sum_{i=1}^5 (6-i)f_i + \sum_{k=1}^L d_k + \sum q - \sum j_p$	Modified form of Artobolevskii's eqn. for mobility with new introduced parameter j_p
24	Antonescu [62]	$M_o = (6-d_a)(l^* - 1) - \sum_{i=1}^5 (i-d_a)p_i$	Similar to Dobrovolskii's eqn. for mobility with various motion coefficient
25	Freudenstein and Alizade [63]	$M_o = \sum_{i=1}^E m_i - \sum_{K=1}^L \lambda_K$ $M_o = \sum_{i=1}^j f_i - \sum_{K=1}^L \lambda_K$ $M_o = \sum_{i=1}^E m_i - \lambda L$ $M_o = \sum_{i=1}^j f_i - \lambda L$ $\lambda = 2,3,4,5,6$	Mobility eqn. for various conditions and parameter for spatial and planar mechanism.
26	Hunt [64]	$M_o = \lambda(l^* - j - 1) + \sum_{i=1}^j f_i$	This eqn. is the extended for of eqn. 25.

Table 2.2 Different mechanisms and mobility (Continued)

SN	Authors	Equations	Remarks
27	Herve [65]	$M_o = \lambda(l^* - 1) - \sum_{i=1}^j (\lambda - f_i)$	Mobility eqn. for the algebraic formula of the structure for the displacement set.
28	Gronowicz [66]	$M_o = \sum \lambda_k - \sum_{K=1}^{L-1} \sum_{j=K+1}^L F_{Kj}$	Eqn. of mobility for multi loop mechanism.
29	Davies [67]	$M_o = \sum_{i=1}^j f_i - r$	Eqn. for mobility similar to Moroshkin's eqn.
30	Agrawal and Rao [68]	$M_o = \sum_{K=1}^L \lambda_k - \sum_{K=1}^{L-1} \sum_{j=K+1}^L F_{Kj} + \sum_{i=1}^{N_1} \frac{1}{2} (\tilde{n}_i^2 + \tilde{n}_i - 2) F_{\tilde{n}_i} + \sum_{i=1}^{N_2} \frac{1}{2} (n_i^2 + 3n_i - 2) F_{n_i}$	Eqn. of general mechanism for the mobility with general constraint.
31	Dudita and Diaconescu [69]	$M_o = \sum_{i=1}^j f_i^e - \sum_{K=1}^L \lambda_K^e$ $M_o = \sum_{K=1}^L \lambda_K - \sum_j (L_{conij} - 1) f_{conij}^e$	Mobility eqn. for complex or elementary loop mechanism.
32	Angeles and Gosselin [70]	$M_o = \text{nullity}(J)$ $\text{nullity}(J) = d(v) - r(J)$	Eqn. of the multi-loop or closed kinematic chain using Jacobian matrix where J represents the Jacobian matrix of rank r(J) in v dimensional space
33	Alizade [71]	$L = j_B - B - c_b$ $M_o = \sum_{i=1}^E m_i - \lambda(j_B - B - c_b) + q - j_p$ $M_o = \sum_{i=1}^j f_i - \lambda(j_B - B - c_b) + q - j_p$ $\sum_{i=1}^j f_i = \lambda(j_B - B - c_b)$	Eqn. for no. of different loops and mobile platform to calculate its mobility's.

Table 2.2 Different mechanisms and mobility (Continued)

SN	Authors	Equations	Remarks
34	McCarthy [72]	$M_o = \lambda - \sum_{i=1}^{c_1} (\lambda - f_i)$	Eqn. represents the mobility of parallel mechanism
35	Huang and Li [73]	$M_o = (6-d)(l^* - j - 1) + \sum_{i=1}^j f_i + q$	Similar to Mccarthy's mobility formula for parallel mechanisms.
36	Alizade and Bayram [74]	$M_o = \sum_{i=1}^j f_i - \lambda(c - B)$ $\sum_{i=1}^j f_i = \lambda(c - B)$ $L = c - B, c = c_1 + c_b,$ $c_1 = j_B - 2c_b$	Mobility eqn. for simple and complex structural mechanisms.
37	Gogu [75]	$M_o = \sum_{i=1}^j f_i - \sum_{j=1}^l S_j + S_p$	Similar to Mccarthy, Alizade, and Bayram.
38	Alizade Bayram and Gezgin [76]	$M_o = (B - c)\lambda + \sum_{i=1}^j f_i + q - j_p$ $M_o = (\lambda + 3) + \sum_{i=1}^{c_1} (d_i - D) +$ $\sum_{i=1}^{c_1} (f_i - \lambda_1) + q - j_p$ $c = c_1 + c_b + c_h$	First eqn. represents for mobility of robot manipulator and second eqn. gives mobility for parallel Cartesian manipulator.

2.2.2 Conventional methods

It is well known that the three dimensional homogeneous transformation matrix broadly used in the robotics field. Homogeneous transformation matrix mostly deals in the field of mobile robot, industrial robot and computer graphics for motion analysis. On the other hand there are several conventional tools to find out the kinematic solutions of the robot manipulator.

Kanayama and Krahn [77] proposed a new “heterogeneous” two-dimensional (2-D) transformation group to solve motion analysis/planning problems in robotics. In the new method they used a 3X1 matrix to represent a transformation which is as capable as the homogeneous theory. This requires less memory space and less computation time as opposed to a 3X3 matrix in the homogeneous formulation and it does not have the rotational matrix inconsistency problem. This heterogeneous formulation has been

successfully implemented in the MML software system for the autonomous mobile robot Yamabico-11.

Paul and Zhang [78] presented homogeneous transformations based kinematic analysis of Manipulators with Spherical Wrists and described its position and orientation. They used proposed technique to obtain kinematic equations directly in a form suitable for computer implementation. The equations are numerically stable and are obtained almost automatically. The resulting equations involve the minimum number of mathematical operations.

Aspragathos and Dimitros [79] presented three methods for the formulation of the kinematic equations of robots with rigid links. The first and most common method in the robotics community is based on homogeneous matrix transformation, the second one is based on Lie algebra, and the third one on screw theory expressed via dual quaternion algebra. They compared these three methods for their use in the kinematic analysis of robot arms. They presented three analytic algorithms for the solution of the direct kinematic problem corresponding to each method. Finally, a comparative study on the computation and storage requirements for the three methods is worked out. However the application has not been done in higher DOF manipulators and it is applied to five DOF robots only.

De Xu [80] proposed an analytical solution for a 5-DOF manipulator to follow a given trajectory while keeping the orientation of one axis in the end-effector frame. They used homogeneous transformation matrix for forward kinematics and inverse kinematics of a 5-DOF manipulator. The singular problem is discussed after the forward kinematics is provided. For any given reachable position and orientation of the end-effector, the derived inverse kinematics will provide an accurate solution. In other words, there exists no singular problem for the 5-DOF manipulator, which has wide application areas such as welding, spraying, and painting. Experiment results verify the effectiveness of the methods developed in this paper.

Manocha and Canny [81] proposed an efficient algorithm for inverse kinematics solution of general 6-dof revolute manipulator with arbitrary geometry. When started mathematically, the problem reduces to solving a system of multivariate equations. They used properties of algebra and symbolic formulation for reducing the problem to solve univariate polynomial. However, the polynomial is expressed as a matrix determinant and its roots are computed by reducing to an Eigen value problem. These algorithms involve symbolic pre-processing, matrix computations and variety of other numerical techniques.

Lai and Menq [82] Proposed two algorithms, the degenerate axis and iterative methods for the motion control of manipulators with closed-form solutions in the neighbourhood of singularities. These two methods theoretically guarantee a robot's position accuracy. The degenerate axis method may not work well when a robot's orientation and location increments become finite. If a robot is moving with slow speed or the interpolation time is in the order of microsecond, the location and orientation increments are small. In this case, the degenerate axis method is favoured for it has less computation than that of the iterative method. Although it cannot be proved that the iterative scheme gives the required position accuracy and minimizes the orientation error, the results seem to show that this scheme converges to an acceptable solution. It is believed that the iterative method is the first of its kind to solve the singular motion control problem by using a robot's closed-form inverse kinematics. Simple computation for the iterative scheme makes it possible to be implemented in many industrial robots.

Pennock and Raghavan [83] proposed a numerical algorithm to solve the inverse kinematics of parallel robots based on numerical integration. Inverse kinematics algorithms based on numerical integration involve the drift phenomena of the solution; as a consequence, errors are generated when the end-effector location differs from that desired. The proposed algorithm associates a novel method to describe the differential kinematics with a simple numerical integration method. The methodology is presented in this paper and its exponential stability is proved. A numerical example and a real application are presented to outline its advantages.

Kucuk and Bingul [84] described forward and inverse kinematics transformations for an open kinematics chain based on the homogenous transformation. Then, geometric and algebraic approaches discussed with explanatory examples. Finally, the forward and inverse kinematics transformations are derived based on the quaternion modelling convention and are explained with the illustrative examples.

Walker [85] proposed the position of a manipulator expressed as either in joint coordinates or in Cartesian coordinates. A new algebra has been defined for the use in solving the forward and inverse kinematics problem of manipulators. The properties of the algebra are investigated and functions of an epsilon numbers are defined. The Ada language was used for illustration because of the ease in implementing the algebra and it is being used to solve the forward and inverse kinematics problems. However, the program actually used epsilon numbers and used the overloading feature of the Ada language to implement the epsilon algebra. By simply changing the order of the algebra, the resulting program can compute a time derivative of the end-effector's position when used to solve the forward kinematics problem and any time derivative of joint positions when used to solve the inverse kinematics problem.

Balkan et al. [86] presented inverse kinematic solutions analytically by manipulating the trigonometric equations directly without converting them necessarily into polynomial equations. Four different subgroups are selected for the demonstration of the inverse kinematic solution method. Two of these subgroups are examples to closed-form and semi-analytic inverse kinematic solutions for the most frequently seen kinematic structures among the industrial robots.

Lipkin [87] described the Denavit-Hartenberg conventions model chains of bodies connected by joints. Originally they were applied to single-loop chains but are now almost universally applied to open-loop serial chains such as robotic manipulators. Unfortunately there are several popular variations of the notation: the original, the distal variant, and the proximal variant. These three cases are compared for their application to serial robots. The proximal variate is advanced as the most notation ally transparent for the mechanical analysis of serial manipulators.

Ceccarelli and Ottaviano [88] described a kinematic design procedure to obtain closed-form formulation and/or numerical algorithms, which can be used not only for design purposes but even to investigate effects of design parameters on design characteristics and operation performance of manipulators. Usually, there is a distinction between open-chain serial manipulators and closed-chain parallel manipulators. This distinction is also considered as a constraint for the kinematic design of manipulators and in fact different procedures and formulation have been proposed to take into account the peculiar differences in their kinematic design. Nevertheless, recently, attempts have been made to formulate a unique view for kinematic design both of serial and parallel manipulators, mainly with an approach using optimization problems.

Low and Dubey [89] proposed two different approaches to the inverse- kinematics problem for a six-degree-of-freedom robot manipulator having three revolute joint axes intersecting at the wrist. One method uses three rotational generalized coordinates to describe the orientation of the body. The other method uses equivalent Euler parameters with one constraint equation. These two approaches have been incorporated into two different computer algorithms, and the results from each are compared on the basis of computational complexity, time simulation, singularity, etc. It was found that Euler parameters were less efficient than three rotational angles for solving the inverse-kinematics problem of the robot considered, and that the physical singularities caused by the robot mechanism could not be eliminated by using either of the two approaches.

Perez [90] proposed algorithms for computing constraints on the position of an object due to the presence of other objects. This problem arises in applications that require choosing how to arrange or how to move objects without collisions. They described the

approach based on characterizing 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 configurations forbidden to this object, due to the presence of other objects, can then be characterized as regions in the configuration space, called configuration space obstacles. The paper presents algorithms for computing these configuration space obstacles when the objects are polygons or polyhedral.

Singh and Claassens [91] proposed the inverse kinematics solution for the 7 Degrees of Freedom Barrett Whole Arm Manipulator with link offsets. The presence of link offsets gives rise to the possibility of the in-elbow & out-elbow poses for a given end-effector pose and is discussed. A parametric solution for all possible geometric poses is generated for a desired end-effector pose (position and orientation). The set of possible geometric poses are completely defined by three circles in the Cartesian space. A method of computing the joint variables for any geometric pose is presented. An analytical method of identifying a set of feasible poses for some joint angle constraints is also addressed.

Nielsen and Roth [92] proposed solution techniques of inverse kinematics using polynomial continuation, Gröbner bases, and elimination. They compared the results that have been obtained with these techniques in the solution of two basic problems, namely, the inverse kinematics for serial-chain manipulators, and the direct kinematics of in-parallel platform devices.

Xin et al. [93] proposed a simple effective method for inverse kinematics problem of general 6-dof revolute serial robot or forward kinematics problem of general 7-dof revolute single-loop mechanism based on a one-dimension searching algorithm. All the real solutions to inverse kinematics problems of the general 6-dof revolute serial robot or forward kinematics problems of the general 7-dof revolute single-loop mechanism can be obtained. They proposed following features of applied method: (1) using one-dimension searching algorithm, all the real inverse kinematic solutions are obtained and it has higher computing efficiency; and (2) compared with the algebraic method, it has evidently reduced the difficulty of deducing formulas. The principle of the new method can be generalized to kinematic analysis of parallel mechanisms.

Mavroidis et al. [94] proposed geometric design problem of R-R spatial manipulators with a new method that uses the DH parameters. They defined three end-effector positions and orientations using three 4 by 4 homogenous transformation matrices. The loop-closure geometric equations provide the required number of design equations. Polynomial Elimination techniques are used to solve these equations and obtain the manipulator DH parameters including DH parameters that describe the location of the

base frame with respect to an arbitrary reference frame and parameters associated with the end-effector. A sixth order polynomial is obtained in one of the design parameters. Novel method is applied to demonstrate that the two spatial R-R chains obtained as real solutions to the numerical example can form a four-bar Bennett mechanism. Finally, two special cases where the orientations of any two or all three precision points are identical are solved using the DH formulation.

Chen et al. [95] proposed formulation of a generic numerical inverse kinematics model and automatic generation of the model for arbitrary robot geometry, including serial and tree-typed geometries. Both revolute and prismatic types of joints are considered. The inverse kinematics is obtained through the differential kinematics equations based on the product-of-exponential POE formulas. The Newton Raphson iteration method is employed for solution. The automated model generation is accomplished by using the kinematic graph representation of a modular robot assembly configuration and the related accessibility matrix and path matrix. Examples of the inverse kinematics solutions for different types of modular robots are given to demonstrate the applicability and effectiveness of the proposed algorithm.

Rico et al. [96] proposed the application of Lie Algebra to the mobility analysis of kinematic chains. The instantaneous form of the mobility criterion presented here is based on the theory of subspaces and sub algebras of the Lie Algebra of the Euclidean group and their possible intersections. It is shown using this theory that certain results on mobility of over-constraint linkages derived previously using screw theory are not complete and accurate. The theory presented provides for a computational approach that would allow efficient automation of the new group theoretic mobility criterion.

Perez et al. [97] presented the simplest of the over-constrained linkages, the closed spatial RPRP linkage. They have used result in order to synthesize RPRP linkages with positive mobility and for a given shape of the screw system of relative displacements. In order to do so, they have stated the design equations using the Clifford algebra of dual quaternions [15]. The dual quaternion expression can be easily related to the screw system and it is also used to assign the magnitude to the screws in order to obtain the correspondence between the screw system and the trajectory of the end-effector. The design yields a single RPRP linkage.

Perez and McCarthy [98] proposed dual quaternion algebra based kinematic synthesis of constrained robotic system. They have proposed this method for one or more serial chain manipulator considering both prismatic and revolute joints. In this research they have used DH algorithm and successive screw displacement for determining the joint variables for the resolution of end effector position. Then dual quaternions are used to

define the transformation matrices obtained through DH algorithm to simplify the design formulations of different types of manipulators.

Radavellia et al. [99] proposed kinematic solution of 3-dof revolute manipulator using dual quaternion and they made comparison between DH algorithm and dual quaternion approach. In this work they have calculated position of end effector using homogeneous transformation matrix that is later compared with proposed method. They have performed the numerical robustness of adopted technique i.e. dual quaternion.

Serra and Gracia [100] proposed a new method for the description of positional dimensional synthesis of robot end effector. The proposed methodology of this work is based on rooted tree graph system wherein, the graph analysis is applied to determine exact position of end effector. They have presented many examples of tree topologies.

Krovi et al. [101] proposed design analysis and kinematics of single dof novel coupled serial chain manipulator. In this work they have presented dimensional synthesis for planar manipulator tasks, considering motions and torques of end effector. They have determined the kinematic and kinetostatic synthesis of planar CSC manipulator.

Lee et al. [102] proposed geometric design problem of 3-dof revolute serial manipulator using interval analysis method. They have applied DH algorithm for obtaining 4x4 homogeneous matrices which would later use for design analysis. In this research, five spatial positions and orientations of end effectors has been predefined to check for the accuracy of adopted technique.

Perez and McCarthy [103] proposed Clifford algebra for the serial coupled n-R 1-dof manipulator to obtained design equations and synthesis. They presented the relative kinematics of serial chain in the matrix exponential form. In this work the formulations of design equation using Clifford algebra are shown efficient for manipulation tasks. They have also presented the inverse kinematic solution of the proposed manipulator.

Hegedüs et al. [104] proposed factorization theory using motion polynomials over quaternion algebra for the solution of 6-dof revolute manipulator kinematics. In this work they proposed strategy for picking best solutions of the problem.

Zhang and Nelson [105] proposed kinematic design and optimization of serial spherical mechanism using genetic algorithm, In this work global manipulability and the uniformity of the mechanism and their workspace for synthesis has been analysed.

Müller [106] proposed generic properties of kinematic mapping for serial manipulator. Firstly they have presented the stability of the property for small changes in geometry of the considered mechanism and second one is concern with singularity analysis. In this work clear manifestation of motion spaces of each joint and classes of kinematic mapping is presented.

Mavroidis and Roth [107] presented a new method for the determination of uncertain configurations of general 6-dof revolute robot manipulator. In this work the proposed novel method for determining the uncertainty or redundancy is based on analytical formulations for the loop closure equations. In this formulation general 6-dof revolute manipulator is transformed into mR Configuration, and new structural parameters are defined.

Balkan et al. [108] presented a general method for the classification of 6-dof industrial manipulators based on the kinematic structure and their detail analyses of kinematic equations on the basis of classification are given. They have adopted the exponential rotation matrix algebra to find out the closed form solution of inverse kinematics of robot manipulator.

Özgören [109] proposed exponential rotation based matrix method for the kinematic analysis of screw and crank mechanism. They have presented the usefulness of the analytical tool for effective solution of kinematics for spatial mechanism involving displacement, singularity, velocity and acceleration.

Pennestri and Valentini [110] proposed dual algebra for the representation of various mechanical and mathematical entities such as screws, line vectors and wrenches. They have given different algorithms for the handling of these vector and matrices of dual number for the analysis kinematic of different mechanisms. They have also proposed the application of the derived algebra for the rigid body motion analysis.

Lee and Mavroidis [111] proposed polynomial continuation method for the analysis of geometric design problem of 3-dof revolute manipulator. They have developed the elimination method for 4 point precision geometric analysis of the manipulator. In this work, each precision point of the end effector has been considered spatial configuration. DH algorithm is used in this work for the formulation of the design equations.

Liang et al. [112] presented pose error analysis of SCARA manipulator using screw theory. They have presented the error produced by DH algorithm and compared the same with the output of the screw based analysis of the manipulation.

Zhuang et al. [113] proposed the linear solution of PUMA robot for the computation of transformations of coordinated from world coordinate to base coordinate. In this work, solution for locating the robot end effector with respect to a reference frame has been presented. They have also applied the quaternion algebra along with the homogeneous transformation matrix method.

Samer Yahya et al. [114] proposed a novel method for the solution of inverse kinematic of hyper redundant manipulator using geometric algebra. In this work, the joint angles are set to similar which makes facing of two or more joint axes impossible;

therefore it can avoid singularities. They have also presented workspace analysis of the proposed manipulator.

Cui et al. [115] proposed virtual model of an agricultural robot for fruit harvesting and their kinematics analysis using DH algorithm. In this work, the inverse kinematic is obtained using algebraic method and simulations are carried out using ADAMS.

Ahmed and Pechev [116] proposed pseudo-inverse based technique for the control of feedback inverse kinematics of Mitsubishi RV-1A a six degree of freedom robotic manipulator. In this work, kinematic analysis of 6-dof manipulator has been done on the basis of DH algorithm and later compared with damped least square inverse kinematics.

Wei et al. [117] proposed semi-analytic method for solving inverse kinematics of n-R robot manipulator that reduces the numerical method's margins related to accuracy. In this work, conformal geometric theory is used for the generation of general kinematic equation. Finally they have tested the proposed method in 6-dof revolute manipulator to prove the efficiency and quality of the solution.

Palacios [118] proposed several approach for the solution of inverse kinematic of 6-dof robot manipulators without considering explicit solution for the chosen manipulator. In this work, 16 different structure or configurations of the 6-dof manipulator has been presented and their classification on the basis of the structure. A complementary example is also presented for the inverse kinematic solution of 5-dof manipulator.

Muszynski [119] proposed a normal form approach for the solution of inverse kinematic of the ASEA IRB-6 robot manipulator. In this work two steps have been presented for the solution of inverse kinematics, firstly they have considered the hyperbolic normal form of the singular kinematics of the manipulator and then inversion algorithms is presented.

BHATTI et al. [120] proposed the problem of matching forward and inverse kinematic motion of 3-dimensional chain using pseudo-inverse Jacobian matrix method. This method is proposed for the solution of inverse kinematics of 3d-dimentional rig character for animations.

Herrera et al. [121] presented dual number representation for solving kinematics problem of rigid body, wherein robot manipulator has been considered for the kinematic analysis using dual number theory particularly serial manipulator. In this work, cylindrical, prismatic and rotational joints are used for the analysis of kinematics using the developed method.

Luo et al. [122] proposed a hyper-chaotic least square method for inverse kinematic solution of 6-dof revolute general manipulator. In this work all real solution of obtained nonlinear equations has been proposed and inverse displacement analysis of 6-dof

revolute manipulator is completed. These obtained nonlinear equations are basically formulated by using DH algorithm and they have presented the numerical example for the constrained equations.

Karpinska et al. [123] proposed approximation problem of Jacobian based inverse kinematic solution of 7-dof redundant manipulator. In this paper they have focused on Jacobian pseudo inverse using extended Jacobian algorithm specifically they have examined two methods, first method is referred to differential geometric and alternative method is based on minimization of approximation error using calculus of variations. .

Brandstotter et al. [124] proposed an efficient generic method for the solution of inverse kinematic of 6-dof serial manipulator. In this work they have mainly focused on DH algorithm considering seven geometric parameters.

Kofinas et al. [125] presented a complete forward and inverse kinematic analytical solution of Aldebaran NAO humanoid robot and their software implementation for real time on-board execution. In this work they have decomposed NAO robot into 5 independent structure of the robot such as two arms, head, and two legs, then DH algorithm is used for the kinematic resolutions.

Szkodny [126] presented all equations of forward and inverse kinematics of IRB-6 manipulator using matrix based method. In this work DH algorithms and homogeneous transformation matrices are used to formulate inverse and forward kinematics of IRB-6 robot manipulation.

Wang et al. [127] presented the geometric structure, particularly Lie group properties of the dual quaternion and the exponential form of the dual quaternion is derived. They have also presented the usefulness and application of the proposed model for kinematic analysis of robots.

Feng and Wan [128] presented blending algorithm for quaternion to dual quaternion representations of rigid body transformations. This work mainly focused on the character animation and kinematic analysis of the character using the dual quaternion and proposed method has been presented.

Gu and Luh [129] proposed dual number theory for representation of line transformation and their application to solve kinematic problem of robot manipulator. This work is mainly focused on an algorithm which pacts with the symbolic analysis of rotation and translation of links.

Wenz and Worn [130] proposed closed form solution of forward and inverse kinematics of 6-dof manipulator. In this work DH algorithm is used for derivation of nonlinear inverse kinematics equations and these kinematics equations are simplify using Groebner basis elimination method.

Neppalli et al. [131] proposed novel analytical method for inverse kinematic solution of multi section continuum manipulator. In this work, the kinematic of the mechanism is decomposed into some sub problems like solution of inverse kinematic for single trunk on the basis of known end points of trunk and then applying single section inverse kinematics to all section of the trunk. Finally, this approach computes final section kinematics of the proposed model of trunk.

Olunloyo et al. [132] proposed inverse and forward kinematic analysis of 5-dof robot manipulator to compare the accuracy and repeatability of the obtained solutions. In this work, DH algorithm is used for the derivation of kinematic of 2 link and 3 link manipulators using all algebraic equations derived from the kinematic transformations of the link.

Yildirim and Bayram [133] presented the mathematical modelling and kinematic analysis of industrial manipulator using Maple robotics toolbox. In this work position and orientation of the tool can be obtained by using DH algorithm and also for joint variables this method is capable of solving Jacobian and angular velocities.

Der et al. [134] proposed reduced deformation model based algorithm to solve inverse kinematics of animated character. A proposed algorithm provides intuitive and direct control of the reduced deformable models similar to a conventional inverse kinematic algorithm for the joint structure. They have presented the fully automatic pipeline transformations of controllable shapes with only few manipulations that reduce the mathematical complexity of the inverse kinematic of the mechanism.

Zoric et al. [135] proposed a quaternion approach for the modelling kinematic and dynamics of the rigid multi-body mechanism. In this work, regular Newton-Euler and Lagrange technique is sorted in the covariant form by applying Rodriguez approach and quaternion algebra that can be useful for calculation of kinematic and dynamics of any mechanism.

Calderon et al. [136] proposed trajectory planning and analytical inverse kinematic solution of 5-dof Parm robot manipulator. This work is based on the hybrid algorithm of analytical inverse kinematic and displacement error. Furthermore resolve motion rate control using Jacobian is used for the smooth motion of end effector. In this work they have used displacement error or Euclidean distance based inverse kinematic solution of 5-dof manipulator.

Ahmmad et al. [137] proposed inverse kinematic solution of 4-dof redundant manipulator and validated with experimental results. In this work, partition of the 4-dof manipulator into 2, 2-dof virtual sub-robot and then solved the inverse kinematic analytical for both sub-robots.

Fedák et al. [138] proposed kinematic and dynamic analysis of 6-dof robot manipulator. In this work 3D CAD model of robot manipulator is developed and later imported to the MATLAB Simulink environment. They have worked on the MATLAB sim-mechanics for the evaluation of kinematics and dynamics of the designed manipulator.

Gouasmi et al. [139] proposed kinematic analysis and trajectory planning for 2-dof and SACARA manipulator. They have used SolidWorks software for the modelling the manipulator later imported in MATLAB Simulink environment for simulations and motion analysis. The main task performed in this paper is comparison of two robot positions with the similar trajectory along with same time and establishing computer program for the kinematic and dynamic analysis.

Rehiara [140] proposed inverse kinematic solution of Adept three manipulator. Forward kinematics of the selected manipulator was calculated by DH-algorithm while inverse kinematic resolutions were completed by principle of cosines. A graphical simulations and calculations of robot kinematics have been presented by using LabVIEW.

Dahari and Tan [141] proposed forward and inverse kinematic solution for KUKA robot manipulator for the welding application. They have selected several welding spot to be performed by the manipulator. To do so they have used analytical method for solving inverse kinematics using DH-algorithm.

Soares et al. [142] proposed rhino manipulator kinematics and control using RobSim software. In this work they have focused on image capturing device for the position and orientation of the end effector. This method is developed in MATLAB presented simulations for the selected manipulator. In this platform a basic unit which is called primitives is used to simulate robot structure. Video capturing device is used for the vision guided manipulator experiments and image and positions are used for servoing.

Wang et al. [143] proposed inverse kinematic solution of general 6-dof revolute serial manipulator using Groebner bases method. They have reduced the complexity of the inverse kinematic polynomial equation using Groebner base method. From this, they have given maximum 16 solutions for the inverse kinematic and also concluded that this method can be easily implemented on nonlinear equations with the help of symbolic representations.

Gan et al. [144] proposed inverse kinematics of 7-dof robot manipulator using dual quaternion algebra. The considered manipulator configuration is serial 7-links with revolute joint in this work. They have used Dixon's resultant for input-output; expressed

in 6x6 determinant equated to zero, and also determined the angular displacement of the joint variables.

Chelnokov [145] proposed inverse kinematic solution of robot manipulator using bi-quaternion method. In this method screw system is considered for the coordinate frame representation.

2.2.3 Intelligent or soft computing approach

Conventional methods for kinematics analysis are more exhaustive and complex in nature as per literature survey; there are numerous conventional techniques as explained earlier such as analytical, algebraic, numerical, Jacobian matrix based, and geometric algebra. These methods generally yield nonlinear, time varying and uncertain equations for inverse kinematic. More over these equations does not provide single solution for the inverse kinematic problem whereas in case of forward kinematics always unique and single solution exists.

Because of the above-mentioned reasons, various authors adopted intelligent techniques to solve inverse kinematic. These intelligent techniques are artificial neural network, fuzzy logic, support vector machine, grey neural network, hybrid neural network etc. However, to find out the inverse kinematic solution of the given problem using above stated intelligent techniques, it is required to calculate forward kinematics of the mechanism which will be used to generate input for the intelligent system.

Artificial neural network (ANN) particularly MLP (multi-layered perceptron) neural network is generally used to learn forward as well as inverse kinematics equation of various configuration of the manipulator. This method is based on learning process of some standard data which rely on the workspace of the manipulator or mechanism. In case of ANN there are many ways of learning data such as supervised learning, unsupervised or combination of both. ANN follows the functional relationship between the input variables (Cartesian coordinates) and output variables (joint coordinates) based on the local revision of mapping between input and output. This concept is also a basis for fuzzy logic and hybrid intelligent techniques which leads to simple solution of inverse kinematic dropping the conventional complex mathematical formulae. The simulation and computation of inverse kinematics using intelligent techniques are predominantly useful were less computation cost is required, definitely for controlling in real time environment. If the configuration of manipulator as well as considering number of dof increases, then the conventional analytical methods will turn into more complex and difficult mathematics. There are numerous research has been done in the field of ANN, fuzzy logic and also for hybrid techniques.

Rodríguez et al. [146] proposed artificial neural adaptive interference system (ANFIS) and ANN based approach for the solution of the inverse kinematics of the 6-dof anthropomorphic manipulator which resembles the human upper limb. In this research they have used multi-layered perceptron (MLP) and ANFIS method for the inverse kinematic prediction in neuro-rehabilitation purpose under the assisted system. They have applied MLP and ANFIS training with Cartesian coordinates of the human upper limb for water serving and bottle picking application. Finally they evaluated the efficiency and quality of the adopted techniques.

Chiddarwar and Babu [147] proposed MLP and RBF neural network model for the solution of inverse kinematic of the 6-dof serial manipulator. In this work, a fusion approach of these ANN models is used with the forward kinematics of the manipulator. Forward kinematics equations are used to generate the data for training adopted models of ANN. They have proposed the Cartesian path to be followed by the manipulator end effector using the generated ANN inverse kinematic solution. KUKA 6-dof manipulator is tested with the obtained results wherein DH-algorithm is used to generate the input for the ANN models.

Koker [148] proposed inverse kinematic solution of the Stanford manipulator using neural network and genetic algorithm. In this work, Elman's neural network has been used and compared with genetic algorithm. A basic calculation for the input of the network has been carried out with the DH-algorithm. Three Elman's neural network models are trained with DH-algorithm output of kinematics. In case of genetic algorithm the fitness function is set to end effector position error based formula for the solution of joint angles.

Karlik and Aydın [149] proposed structured ANN approach for the inverse kinematic solution for 6-dof manipulator. In this work, they have used back-propagation algorithm for the training of the ANN model and input datasets were generated by using DH-algorithm. They have tried to find out the excellent ANN configuration for inverse kinematic resolution.

Hasan et al. [150] proposed adaptive learning plan of ANN for the solution of inverse kinematic of 6-dof manipulator. Moreover they have tried to resolve singularity and uncertainty problem of the adopted configuration of the manipulator. In this work ANN model have been trained using analytical solution of the adopted manipulator. Generated datasets using kinematics equations are used to trained and test the adopted model of ANN. They have concluded that the proposed model of ANN does not need to have previous information of the kinematics of the system that learns through the ANN model application.

Bocsi et al. [151] proposed inverse kinematic solution of 7-dof Barrett WAM using support vector machine. They have explained the learning problem of redundant manipulator using neural network based models. The major problem with the solution of inverse kinematic is non-unique in nature and generation of large datasets for input. Therefore they have proposed a suitable algorithm for learning the kinematics and applied to real world problem of 7-dof manipulator.

Hasan et al. [152] proposed ANN based solution of 6-dof manipulator to avoid singularity and uncertainty of the configuration. They have used input data for the training the ANN model from the experiments of the adopted model of robot using various sensors. They have designed the ANN network for one hidden layer and inputs were taken as coordinates of the end effector of robot manipulator. After training of neural network model they have tested it for real time application of the adopted manipulator with avoiding the singularity problem. Obtained results through their experiments shown their efficiency and quality.

Olaru et al. [153] proposed inverse kinematic solution of the didactical arm using neural network. In this work they have used two hidden layer and sigmoid transfer function for the training of the neural network. Mathematical modelling was created by using neural network and LabVIEW. They have done the experiments for selecting number of hidden neurons for training and better learning of the network to do so they have applied different number of neurons for evaluation for trajectory error generated by the end effector. All gained results were tested by basic kinematic through LabVIEW. Finally they obtained optimal sigmoid function with time delay and recurrent network.

Mayorga and Sanongboon [154] proposed neural network approach for inverse kinematic solution of the planar redundant manipulator and effective geometric singularity avoidance of the selected manipulator. Moreover they have presented some geometrical concept for the singularity avoidance and obstacle avoidance of the redundant manipulator. Finally they have presented the performance of the trained neural network for the stated problem.

Kalra and Prakash [155] proposed neuro-genetic approach for the resolution of inverse kinematics of planar manipulator. They have used massively parallel architecture of ANN for the solution of the stated problem. They have selected the MLP network and weights of the ANN model were optimized by real coded genetic algorithm so as to overcome the problem of backpropagation algorithm.

Bhattacharjee1 and Bhattacharjee [156] studied the problem of inverse kinematic solution using conventional method and therefor they applied ANN based approach for

the resolution of inverse kinematic of the manipulator. Firstly they have obtained the joint angles dataset of the end effector so as to use as input or training of ANN model. They have mainly focus on the obstacle avoidance of the manipulator using double hidden layer ANN model.

Martin et al. [157] proposed inverse kinematic learning of 3-dof planar and SCARA manipulator using neuro-controller. Furthermore, they have presented the some issues of neural network learning such as classical supervised learning scheme which generally converge in local optimum solution. Therefore they have applied neuro-evolution algorithm for the global optimum solution of the inverse kinematics of the selected manipulator. In this work DH-algorithm is used to generate the input data set for the neural network algorithm. They have reduced the drawback of the gradient descent learning of ANN model with the help of evolutionary algorithm.

Feng et al. [158] proposed novel neural network based approach for the solution of inverse kinematics of the PUMA 560 robot manipulator. In this work they have applied simple feed forward neural network to obtain the kinematic of the PUMA 560 manipulator and compared with the developed ELM (extreme learning machine) based neural network. They have used machine learning algorithm to overcome the problem of traditional gradient descent learning strategy.

Bingul et al. [159] proposed inverse kinematic solution of 6-dof revolute robot manipulator with offset wrist using ANN. Manipulator with offset wrist is considered because offset wrist based structure generally does not gives the exact solution using some traditional methods. Therefore they have adopted ANN model for the inverse kinematic solution. They have used DH-algorithm for the generation of input datasets of MLP model and later predicted solution will be used to compare with the traditional solution. They have presented the error occurred and the efficiency of the adopted technique.

Hasan et al. [160] proposed inverse kinematic solution of 6-dof robot manipulator using MLP neural network with different structures. In this work they have used different number of hidden layers for the prediction of the solution. In their first configuration or architecture of the MLP model they have used three inputs (X, Y and Z coordinated) and six outputs of the joint angles and in second experiment they have used four input i.e. Cartesian coordinates along with velocity and calculated outputs are six joint angles and their angular velocities.

Alsina and Gehlot [161] proposed a modular ANN based inverse kinematic solution for 4-dof SCARA manipulator. They have assigned each neural module in each link in order to find out the inverse kinematics. This approach of neural modules is connected

in global system for the updating of the inverse kinematic solution. In this work three layered neural network is used with sigmoid ADLINE transfer function. They have considered 3-dof and 4-dof manipulator for the simulation and verification of the solutions.

Onozato and Maeda [162] proposed MLP neural network based solution of 4-dof SCARA manipulator. They have used basic analytical approach for generating the input dataset for learning. A simultaneous perturbation technique is applied for the learning of network and calculated the inverse kinematic and dynamics of the manipulator.

Al-Khedher and Alshamasin [163] proposed neural network based control of SCARA manipulator and compared with the PD controller. In this work they have used DH algorithm for the evaluation of the inverse kinematic of the robot manipulator. A serial-parallel structure neural network is used for position control of all joint variables. They have used three layered neural network with back propagation supervised learning. They have also optimized the number of hidden layer to obtained better result. Later simulations are carried out in MATLAB Simulink.

Mayorga and Sanongboon [164] proposed neural network based approach for inverse kinematic solution and effective singularity avoidance of redundant manipulator. In this approach they have established some symbolizing matrices, expressing some geometrical ideas, so as to gain simple performance index for singularity avoidance. These methods of matrices are trained with neural network and finally computed the inverse kinematics.

Daachi and Benallegue [165] proposed neural network based adaptive controller for achieving end effector position of redundant manipulator. They have designed the controller in Cartesian space so as to overcome the problem of path and motion planning that is a well know problem of inverse kinematic. They have 3-dof redundant planar manipulator. The unidentified model of the scheme is approached by decomposed structural neural network. This approach is used to find adaptive stability and the algorithm is based on Lyapunov method with inherent properties of robot manipulators.

Howard and Zilouchian [166] proposed fuzzy logic based inverse kinematic solution of 3-dof robot manipulator. In this work hierarchical control based method is used for the controlling of robot manipulator. The mapping of Cartesian coordinate with the joint coordinate is established by fuzzy logic in order to evaluate each joint variable. The hierarchical control with fuzzy logic improves the robustness and also decreases the computational cost.

Kumar and Irshad [167] proposed neural network based solution for the inverse kinematic of 2-dof serial manipulator. They have used MLP neural network structure with unsupervised learning strategy. They have generated input datasets using forward kinematic equation of the manipulator. Back propagation algorithm is used for the training MLP neural network.

Oyama et al. [168] proposed novel modular neural network with expert system for the prediction of inverse kinematic of robot manipulator. In this method each expert estimates the continuous part of the function. The proposed method uses forward kinematic for the selection of experts. When the no. of considered expert increases the computation cost also increases for the inverse kinematics solution, without using any parallel computing system. They have used 7-dof redundant manipulator for the analysis of kinematics.

Tejomurtula and Kak [169] proposed structures neural network based inverse kinematic solution of planar and spatial manipulator. They have used MLP neural network with two hidden layers for training of the network. In this work backpropagation algorithm is used for 3-dof planar and spatial manipulator inverse kinematic resolution.

Kim and Lee [170] proposed inverse kinematic solution of redundant manipulator using Jacobian matrix and fuzzy logic methods. In this work motion rate resolving algorithm is used which is later improved by fuzzy logic. Furthermore, they have obtained rough solution of inverse kinematics based on gradient method which is later refined by fuzzy logic and extension principle.

Alavandar and Nigam [171] proposed inverse kinematic solution of 2-dof and 3-dof planar manipulator using adaptive neural fuzzy inference system (ANFIS). In this work, they have adopted Sugeno type fuzzy architecture and hybridized with simple neural network for the prediction of inverse kinematic of planar manipulator.

Kozalzewicz et al. [172] proposed inverse kinematic of 6-dof manipulator using partitioned neural network which is also known as parallel neural network. The selected architecture is collected of pre-processing layer and partitioned by modules containing devoted neurons. In this work they have used back propagation algorithm for the solution of inverse kinematic.

Kuroe et al. [173] proposed inverse kinematic prediction of 2-link robot manipulator using ANN. In this work they applied supervised learning theorem which is based on the Tellegen's theorem.

Jack et al. [174] proposed inverse kinematic solution of 3-dof manipulator using feed forward neural network technique. In this work they have selected three different configuration of neural network.

Aristidou and Lasenby [175] proposed inverse kinematic solution of various configuration of revolute manipulator using novel developed FABRIK (forward and backward reaching inverse kinematics) method. FABRIK evades the necessity of conventional rotational angle matrices.

Morten and Erleben [176] proposed inverse kinematic solution of animated character using projected-gradient method.

Zhang et al. [177] proposed dual neural network based kinematics and motion planning of redundant manipulator. In this work, linear vibrational inequalities (LVI) based and simplified LVI based dual neural network used for the problem resolution. To accomplish this drift-free condition is exploited in quadratic form.

Duguleana et al. [178] proposed neural network based kinematic solution of general 6-dof serial robot manipulator. In this work dual neural network with Q-learning reinforcement method is used for the solution of inverse kinematic and obstacle avoidance.

Daya et al. [179] proposed inverse kinematic solution of 2-dof planar manipulator using neural network. In this work neural network architecture consists of six sub neural network which is basically extended form of MLP neural network. Back propagation algorithm is applied for error minimization.

Xiulan et al. [180] proposed inverse kinematic solution of 2-dof planar manipulator using hybrid neural network. In this work, feed forward neural network is first optimized by particle swarm optimization (PSO) technique then used for inverse kinematic resolution. This worked is compared with the backpropagation evaluation of kinematics with PSO based ANN.

Aghajarian and Kiani [181] proposed inverse kinematic solution of PUMA 560 robot manipulator using adaptive neural fuzzy inference system (ANFIS). In this work MLP neural network is hybridized with fuzzy logic to obtain better result of inverse kinematic as compared to neural network.

Shen et al. [182] proposed inverse kinematic solution of 2-link planar manipulator using self-configuration fuzzy logic. In this work they have applied fuzzy logic first then self-configuration approach is introduced based on input-output pairs.

Kinoshita et al. [183] proposed inverse kinematic solution for 2-dof planar manipulator using MLP neural network. In this work forward propagation algorithm is used for the

estimation of output layer error. The adopted forward propagation rule is based on goal signal carried by Newton-like method, and then updating of weight is completed by regression coefficient.

Borboni [184] proposed inverse kinematic solution of simple SCARA manipulator using fuzzy logic technique. In this work they have explained several other algorithms like parallel chords algorithm, Newton-Raphson and Resconi-Faglia algorithms and compared with the fuzzy logic solutions of inverse kinematic.

Meshref and Vanlandingham [185] proposed forward and inverse kinematic solution of 3-dof robot manipulator using immune based neural network. In this work forward kinematic is completed by DH-algorithm which is later used as a input for the proposed immune based inverse kinematics solution.

Al-Mashhadany [186] proposed inverse kinematic solution for 6-dof manipulator using locally recurrent neural network with considered spherical wrist. The adopted method LRNN (locally recurrent neural network) is programmed in MATLAB and simulation has been completed in Simulink. In this work Levenberg-Marquardt based back propagation learning strategy is applied for high computation and for solution accuracy of inverse kinematic.

Asuni et al. [187] proposed inverse kinematic solution of PUMA robot manipulator using self-organizing neural network. In this work Visio-motor coordination is used for learning of neural network. This method is based on biological inspired model that imitates human brain power to create relationship between motor and sensory data with the help of learning process.

Yildirim and Eski [188] proposed inverse kinematic solution of PUMA 560 robot manipulator using neural network method. In this work they have applied feed forward neural network with different learning and weight updating algorithms. First they have considered the Online back propagation algorithm and then delta bar delta algorithm and finally they have applied quick propagation algorithm for the analysis of inverse kinematic of robot manipulator.

Zhang et al. [189] proposed inverse kinematic solution of MOTOMAN robot manipulator using neural network. In this work they have used radial basis function neural network (RBFNN) for the evaluation of inverse kinematics. The neural network system designed is multi input and single output (MISO) based technique.

Koker [190] proposed inverse kinematic solution of Stanford and PUMA 560 robot manipulators using neural network technique. In this work simulated annealing (SA) is applied along with the neural network to minimize the error of the joint variables. Three Elman's neural network model is used and trained with the help of SA algorithm.

Her et al. [191] proposed inverse kinematic solution of 2 and 4-dof planar robot manipulator using fuzzy logic together with the genetic algorithm. They have used triangular membership function for fuzzy logic and centre of gravity is used for the defuzzification. These parameters are later tuned by genetic algorithm for the surety of exact inverse kinematic solution.

Hua et al. [192] proposed inverse kinematic solution of PUMA 560 robot manipulator using wavelet neural network model. This method is working on multi-input multi output (MIMO) system. Neural network trained is completed by Levenberg-Marquardt algorithm.

Agarwal [193] proposed inverse kinematic solution of redundant manipulator using fuzzy c-means system. Novel developed fuzzy clustering method is generalized based on weighted scatter metrics and cluster metrics are developed for manipulator.

Qi and Li [194] proposed inverse kinematic solution of 6-dof robot manipulator using support vector machine with genetic algorithm. Support vector coefficient like kernel function, insensitive coefficient and penalty factors are tuned by GA.

Liu and Brown [195] proposed extended approach of fuzzy logic for the solution of inverse kinematics of robot manipulator. In this work they have used PUMA 560 robot manipulator for the implementation of proposed algorithm. The proposed algorithm is based on fuzzy trigonometry derivatives.

Martin and Emami [196] proposed real time neural fuzzy trajectory generation of EPSON robot manipulator for the rehabilitation purpose of patients with limb dysfunction.

Netto et al. [197] proposed inverse kinematic solution of hexapod robot leg using fuzzy system. In this work hexapod robot's leg consists of 3 revolute joint similar to another leg. The kinematic analysis is used to generate data for the black box of fuzzy and neural network, particularly forward kinematic is used to generate the training data set for fuzzy and neural network.

Song and Jung [198] proposed kinematic solution of 6-dof anthropomorphic robot manipulator using geometric algebra based method. In this work trajectory has been generated using fuzzy controller. In this work geometric inverse kinematic solution is used for the joint variable control of manipulator. The generated output from the adopted technique is later used as an input of fuzzy logic controller.

Crenganis et al. [199] proposed mathematical modelling of 7-dof human arm like manipulator kinematics. In this work both forward and inverse kinematic is presented and later compared with the (ANFIS) fuzzy logic solution of the kinematics. They have used MATLAB ANFIS toolbox for kinematic resolution.

Morishita and Tojo [200] proposed integer inverse kinematic solution of multi-joint robot manipulator using fuzzy logic based method. They have evaluated the efficiency of the adopted technique and tested it for trajectory generation and control application.

Neumann et al. [201] proposed inverse kinematic prediction based on neural network for humanoid robot ASIMO, in which they focused on bi-manual tool. Considered humanoid robot hand is highly redundant in this case and recurrent reservoir learning strategy has been implemented.

Hashim et al. [202] proposed manipulator positioning analysis using artificial intelligent techniques. In this work they have adopted three techniques namely fuzzy logic, genetic algorithm and neural network to solve inverse kinematics of 6-dof serial manipulator. Forward kinematic of serial manipulator has been taken as feedforward control on the other hand intelligence method resolves the inverse kinematics problem.

2.2.4 Optimization approach

Inverse kinematic closed form solutions for several configurations and simple structures are certain. Mathematical approaches are more complicated as per numerical, iterative or intelligent based methods and the obtained solution using these methods are not only configuration dependent but also matters to ambiguity of the manufacturing errors. Therefore, to overcome mathematical complexity and improve the efficiency of the solution, it is necessary to adopt engineering optimization methods. Optimization methods can be applied to solve inverse kinematics of manipulators and or general spatial mechanism. Basic numerical approaches like Newton-Raphson method can solve nonlinear kinematic formulae or another approach is predictor corrector type methods to assimilate differential kinematics formulae. But the major issues with the numerical method are that, when Jacobian matrix is ill conditioned or possess singularity then it does not yield a solution. Moreover, when the initial approximation is not accurate then the method becomes unbalanced even though initial approximation is good enough might not converge to optimum solution. Therefore optimization based algorithms are quite fruitful to solve inverse kinematic problem. Generally these approaches are more stable and often converge to global optimum point due to minimization problem. The key factor for optimization algorithms is to design objective function which might be complex in nature. On the other hand, metaheuristic algorithms generally based on the direct search method which generally do not need any gradient based information. In case of heuristic based algorithms local convergence rate is slow therefore some global optimization algorithms like GA, BBO, TLBO, ABC, ACO etc. can be gainfully used.

Nearchou [203] proposed inverse kinematic solution of redundant manipulator using modified genetic algorithm. They have implemented some assumptions; first they considered that the manipulator may be redundant and articulated. Then the second assumption is that the manipulator is in moving object of its workspace. And last assumption is that they are not considering dynamics of the manipulator. Thereafter, genetic algorithm is used in two different manners, first joint displacement ($\Delta\theta$) error minimization and the second approach is based on positional error of end effector.

Wang and Chen [204] proposed inverse kinematic solution of PUMA 560 robot using optimization method. In this work they have considered positional error and orientation error for robot manipulator. The proposed solution is based on cyclic coordinate descent (CCD) and Broyden-Fletcher-Shanno (BFS) technique. Total error is calculated based on the end-effectors initial and final displacement positions and relative angular displacement error.

Parker et al. [205] proposed inverse kinematic solution of 4-dof PUMA manipulator based on genetic algorithm. In this work they have considered two displacement minimization problems; first problem of minimization is end-effector displacement from initial position to desired position and the second approach is based on the relative joint rotation minimization. Both considered approach is solving together using genetic algorithm to find out the global solution.

Kim and Kim [206] proposed trajectory planning of 3-dof revolute manipulator using evolutionary algorithm. In this work they have first calculated optimal inverse kinematic of 3-dof redundant manipulator using Jacobian matrix method. The optimization objective function is selected on the basis of joint and end-effector displacement from initial Cartesian coordinate to desired location, then evolutionary algorithm is applied to find out optimal joint variable.

Piazzil and Visiolis et al. [207] proposed inverse kinematics solution and trajectory planning for D-joint robot manipulator based on deterministic global optimization based method. In this work they calculated inverse kinematic to find out the desired trajectory of 6-dof manipulator. They have applied interval analysis algorithm for the global optimization of the piecewise motion of joint variables.

Ahuactzin and Gupta [208] proposed inverse kinematic solution of redundant manipulator using novel developed global optimization algorithm. In this work they have used the developed algorithm for point to point movement of end effector and then calculated the displacement error using the proposed INVIKIN algorithm. The concept of the work is based on the Ariadne's Clew Algorithm (ACA) which is basically related to motion planning.

Chapelle and Bidaud [209] proposed inverse kinematic solution of PUMA robot manipulator using genetic programming. In this work, mathematical modelling is evolved using genetic programming through given direct kinematic equations. They have represented the evolutionary symbolic regression procedure for the inverse kinematics of GMF Arc Mate and PUMA manipulators.

Khatami and Sassani [210] proposed kinematic isotropy for the performance evaluation of the 2-dof manipulator, where Global isotropy Index has been used to measure of the above isotropy and depends on the entire workspace of the manipulator. Genetic algorithm is used to optimize the design parameter of the manipulator and the parameter is link length. Later, this approached is employed to optimize globally throughout the manipulator workspace.

Kalra et al. [211] proposed inverse kinematic solution of 2-dof articulated robot manipulator using real coded genetic algorithm. In this work they have used Euclidian distance norm for the optimization of joint variable of robot manipulator. Displacement error minimization objective function is subjected to joint angle constraint in this work, and basic steps of real coded genetic algorithm are recombination and mutation.

Korein and Badler [212] proposed inverse kinematic solution scheme of 3-dof redundant manipulator based on reach hierarchy method. In this work they have formulated inverse kinematic analytical and then using Lagrangian multipliers for making the problem with equality constraint. Thereafter they used numerical based optimization method for minimizing the proposed objective function.

Tabandeh et al. [213] proposed inverse kinematic solution of 3-dof PUMA manipulator for the major displacements propose. In this work they have adopted genetic algorithm with adaptive niching and clustering. Genetic algorithm's parameters are set by adaptive niching method which is later required the forward kinematic equations for the solution of inverse kinematic of adopted manipulator. Forward kinematic is simply calculated by standard analytical method. Thereafter for processing the output filtering and clustering is also added to the genetic algorithm.

He et al. [214] proposed inverse kinematic solution of 6-dof MOTOMAN robot manipulator for positioning of the end-effector. In this work they have adopted adaptive genetic algorithm for optimum placement of the end effector. There are several parameters like end-effector displacement error criteria, welding reachability index, motion stability index and dexterity index have been considered for making of objective function.

Rajpar et al. [215] proposed inverse kinematic and trajectory generation of humanoid arm manipulator using forward recursion with backward cycle computation method. In

this work DH-algorithm is used to formulate the forward kinematics of humanoid arm manipulator which is later used as an objective function for the optimization process. End effector displacement and the orientation error are completely used as objective function for this work.

Liu and Zhu [216] proposed inverse kinematic solution for 6-dof revolute manipulator using real time optimization algorithm. DH-algorithm is used to formulate the kinematics equations which are later reduced by the symbolic pre-processing. Later Eigen decomposition is used to extract roots from higher degree polynomial kinematic equations.

Jaryani [217] proposed inverse kinematic solution of 2-dof robot manipulator using virtual potential field method. In this work, a set of points between initial points to desired point is obtained by end-effector with different virtual potential field method. An optimum trajectory is created by using pattern search method which explains the power of the potential field method to optimize the value of generated objective function. In this work cubic splines are used to create a smooth trajectory joint space obtained through inverse kinematic equation. Finally the efficiency and effectiveness of the adopted method is presented through simulations.

Pham et al. [218] proposed inverse kinematic solution of 3-dof robot manipulator using Bee algorithm. In this work they have compared three different methods like evolutionary algorithm, neural network back propagation method and bee algorithm. Neural network structure is optimized by using bee algorithm to predict joint variables of the robot manipulator.

Albert et al. [219] proposed inverse kinematic solution of 3-dof revolute robot manipulator using real time genetic algorithm. In this work end-effector displacement from its initial point to desired point has been optimized using genetic algorithm. Genetic algorithm crossover selection is based on new method which is known as dynamic multi-layered chromosome (DMCC) to produce two offspring's. The GUI simulation has been verified with GA and DMCC.

Huang et al. [220] proposed inverse kinematic solution of 6-dof robot manipulator using immune genetic algorithm. In this work forward kinematic formulation is presented using DH-algorithms. End effector displacement error based fitness function is used for implementation of proposed algorithm and results obtained through adopted technique are compared with neural network back propagation algorithm.

Bailón et al. [221] proposed inverse kinematic solution for the 6-dof robot manipulator using genetic algorithm. In this work eight degree of polynomial equation is used to plan trajectory for the robot manipulator with the help of DH-algorithm. Genetic

algorithm is used for the energy optimization as well as trajectory optimization of robot manipulator.

Paramani [222] proposed inverse kinematic analysis of 12-dof robot manipulator compound numerical optimization method. In this work general analytical solution is fused with the numerical based method to solve inverse kinematic of 12-dof manipulator. The fusion approach is getting rid of with the problem of repeating value of numerical solution and generally that gives slow convergence. Therefore in this work combination of analytical and numerical solution for higher order polynomial function is made.

Lei-ping et al. [223] proposed inverse kinematic solution of 5-dof robot manipulator using genetic algorithm. In this work obstacle avoidance is major criteria, to avoid the obstacle it is required to calculate inverse kinematics of the manipulator. Then this kinematic equation is modelled as an end effector displacement error based fitness function of the genetic algorithm. MATLAB software is used to simulate the adopted problem.

Rubio et al. [224] proposed optimization of path planning of PUMA 560 robot manipulator using genetic algorithm. In this work several different criteria for fitness function have been taken to solve the path planning of the robot. A first criterion is displacement of end effector from its initial position to final or desired position, second criteria is based on its configuration. The genetic algorithm uses parallel populations with the migration for path planning.

Galicki [225] proposed inverse kinematic solution of mobile manipulator using penalty function based optimization method. This work presents solution on control feedback level which is subject to state equality and inequality constraint for the adopted manipulator. In this work Lyapunove stability constraint is used for the control trajectory generation via inverse kinematic solution.

Cavdar and Milani [226] proposed inverse kinematic solution of 6-dof PUMA manipulator using artificial bee colony algorithm. In this work DH-algorithm is used to formulate fitness function for the evaluation of end effector target position based on initial given position.

Lalo et al. [227] proposed inverse kinematic solution of 4-dof planar manipulator using liner programming method. In this work the main idea of generating the objective function is based on the minimizing the joint variables to reach the desired location. Later section deals with the singularity of adopted manipulator and they have also presented the formulations for the smooth trajectory generation for the 4-dof planar manipulator.

Bernal et al. [228] proposed metaheuristic algorithm application in robotics. In this work, ant colony optimization algorithm and genetic algorithm are used for path planning of robot manipulators end effector. The work is completed with natural selection and evolution, through two type of ants namely job and explorer. The basic parameter of genetic algorithm is hybridized with ant colony optimization to get global solution. .

Cubero [229] proposed inverse kinematic solution of serial manipulator using blind search method. In this work standard analytical solution for forward kinematics is require to prepare the fitness function even this can be accomplish with DH–algorithm.

Konietschke and Hirzinger [230] proposed inverse kinematic solution of highly redundant manipulator with combined optimization algorithm. In this work closed form solution of inverse kinematics of Justin robot is proposed and later it is combined with the proposed optimization algorithm for the global; solution. The proposed optimization algorithm is based on Levenberg-Marquardt criteria.

Zhang et al. [231] proposed inverse kinematic solution of 5-dof robot manipulator using hybrid genetic algorithm method. In this work mechanism and body frames are presented based on the DH-algorithm, which is later used to formulate the objective function. The objective function is based on the end-effectors initial position to the desired position displacement error minimization.

Huang et al. [232] proposed inverse kinematic solution of 7-dof spatial manipulator based on the particle swarm optimization (PSO) technique. In this work DH-algorithm is used to formulate the forward kinematic of the 7-dof manipulator which is later used to formulate the objective function for the particle swarm optimization technique. End effectors; initial position ad desired position based displacement error along with the orientation error is minimized using PSO.

Kumar et al. [233] proposed inverse kinematic solution of redundant manipulator using Lyapunov method. In this work, optimization approach to solve inverse kinematic problem which is converted into nonlinear problem solved by Lyapunov method. An improved energy based function is determined for the optimization.

Xu et al. [234] proposed inverse kinematic of the 4-dof redundant manipulator using two different optimization criteria. First optimization criteria is minimization of extra redundant dof and other criteria is based on total potential energy minimization of manipulator links. They have developed numerical optimization method for calculating the trajectory planning computation which is a bit more expensive. Therefore to overcome this computation cost a sequential quadratic programming and iterative Newton-Raphson method is used.

Mazhari and Kumar [235] proposed kinematics and dynamics solution of PUMA 560 robot manipulator using genetic algorithm, simulated annealing, and generalized pattern search methods. They have designed controller for PUMA manipulator using above adopted algorithms. Fine tuning is requiring for controller to achieve desired speed of simulations. MATLAB/Simulink software is used for simulations in this work.

Ramírez and Rubiano [236] proposed inverse kinematic solution of 3-dof revolute spatial manipulator using genetic algorithm. Forward kinematics formulation has been completed by using DH-algorithms and homogeneous matrix multiplication based method. Fitness function for the inverse kinematic solution is based on the end effectors initial and desired position error which is also known as Euclidean distance norm.

Feng et al. [237] proposed inverse kinematic solution of 3-dof general robot manipulator using Electromagnetism-like and modified Davidson-Fletcher-Powell (DFP) method. In this work DH-algorithm is used to formulate the forward kinematics of the 3-dof general robot manipulator. The objective function is based on the displacement error and orientation error of the end effector. The in total combination of the both error is used as a fitness function for the adopted technique. DFP method is hybridized using EM algorithm to get the best convergence rate.

Henten et al. [238] proposed inverse kinematic analysis of 7-link robot manipulator for the cucumber picking operation using analytical and numerical algorithm based methods. In this work standard DH-algorithm is used for the inverse kinematic solution of robot manipulator and later this analytical method is fused with the numerical analysis based algorithm to get the optimum solution of the robot manipulator.

Rokbani and Alimi [239] proposed inverse kinematic solution of 2-dof robot manipulator using particle swam optimization algorithm. In this work initial position and desired position error based objective function is used which is also known as the Euclidean distance norm for end effector. In this approach norm analytical solution of forward kinematic is presented which is later used in objective function of PSO algorithm.

Dutra [240] proposed inverse kinematic solution of 2-link planar manipulator using simulated annealing method. In this work standard analytical solution of forward kinematic is presented using forward kinematic equation the displacement based error minimization objective function is used for the simulated annealing approach.

Zhang et al. [241] proposed inverse kinematic solution of the serial dangerous articles disposal manipulator with multiple degrees of freedom using particle swarm gene optimization algorithm. In this work position and orientation error of end effector is used as a objective function for traditional PSO and modified PSGO method.

Rokbani and Alimi [242] proposed inverse kinematic solution of the biped robot leg using particle swarm optimization (PSO) method. In this work, two legs with 2-dof are considered for the optimization purpose. Moreover both leg manipulator's forward kinematic is calculated analytically to obtain fitness function for the PSO.

Luo and Wei [243] proposed kinematic analysis of 3-dof planar redundant manipulator using two different techniques like immune based and immune genetic algorithm methods. They have calculated the forward kinematic analytical for the path planning of robot manipulator. In later section immune and immune genetic algorithm is used to evaluate the efficiency and performance of the obtained solution.

Taylor et al. [244] proposed kinematics and dynamics of the 3-dof planar and spatial manipulator based on the complex optimization method. In this work they have used optimization algorithm for the evaluation of the trajectory planning and inverse kinematic modelling of the 3-dof manipulator. For trajectory planning cubic splines are used for the formulations. It has been concluded that the complex optimization algorithm is effective and performing better for path evaluations.

Števo et al. [245] proposed inverse kinematic solution of 6-dof ABB IRB 6400FHD robot manipulator using genetic algorithm. In this work forward kinematic equation are generated by using DH-algorithm which is later used to obtain fitness function for genetic algorithm. In this work displacement error of end-effector from point to point motion has been calculated through adopted method. Objective function containing three separate parts which are energy function, operation time and position accuracy to get combined fitness function for genetic algorithm.

2.3 Review analysis and outcomes

Focusing the attention on the manipulators configuration and their kinematics, the review depicts that inverse kinematics has been treated as the gold mine for robot designers. Numerous researchers have tried to develop inverse kinematic solution from late 80's until now with various approaches and for various configurations of robot manipulator. From the beginning robot manipulators are being used for various industrial applications like pick and place type work etc., so the major constraint was to find joint variables of the manipulator to reach the desired position with known object coordinate points. Now a day, robot manipulators applications are widened along with the industrial applications to perform in various filed like medical rehabilitation, under water applications, assembly task, agriculture, mining, space etc. along with the human interactions. From the literature review it can be summarized that to achieve desired position and orientation of the end-effector or tool along with manipulability, dexterity

and trajectory planning, the need of inverse and forward kinematics arises. Almost all reviewed articles indicated that the human arm is the key point of motivation and leads to design of robot manipulator. Now from design point of view it is required to calculate kinematics relationship of each joint variables so that the optimum design can be obtain. There may be numerous configurations or structures of mechanism or robot manipulators but the major properties of designing of any mechanism or manipulator are kinematic analysis, workspace analysis, anthropomorphic advent, manipulability, trajectory generation and control. It is also observed that concerning with the applications of the robot manipulator the explicit properties are always given importance. Working space and manipulability of robot manipulator increases when number of joint variables increases which generally cause more complex mathematical formulations for inverse kinematic resolutions and difficulty in control of the manipulator. Numerous designs and configurations of the robot manipulators are being used in many human environment as well as industrial applications.

The main aim of the literature survey is to explore different techniques and methodologies available to solve inverse kinematics of any configuration of robot manipulator. But it is perceived from literature review that DH-algorithm, homogeneous transformation matrix, analytical approach, algebraic approach and geometric approaches are mostly followed by various researches. Among all the developed methodologies the most frequently used approach is algebraic solution of inverse kinematic. This method covers conventional algebra along with quaternion, dual quaternion, quaternion vector pair, screw algebra, Clifford algebra and Lie algebra. DH-algorithm and associated parameters are the best way of representation rotation and translation of manipulator links and joints. The major drawback in case of conventional algebra is its complexities in modelling and obtaining appropriate solutions when the robot configuration is complex and has larger degree of freedoms. This problem of higher mathematical formulations was reduced by using quaternion and screw algebra. Thereafter few elimination methods for reducing the complexity of inverse kinematics formulations arise with their effective performance. In case of geometric algebra the major problem is when the first three joints of any mechanism or manipulator do not create any joint angle in between them, and then it does not give exact solution for the inverse kinematic problem. Moreover, the problem becomes unstable if the Jacobian matrix is in ill condition or suffering from singularity. Therefore, the conventional method are reliable but there is always mathematical complexity problem arises with the configurations and dof's of manipulator. To overcome these problem researchers adopted many intelligent techniques for soft computing domain such as artificial neural network technique, fuzzy logic, hybrid ANN etc. These methods do not require higher

mathematical programming and computation cost is also less. Apart from these, optimization approach like heuristic, metaheuristic, numerical based approach, etc. have shown their efficiency in solving inverse kinematic problem for any configuration of robot manipulator. However, there remains a scope to investigate further and work towards finding better solutions. Most of the optimization algorithms do not give global optimum point because of trapping in local optimum point. Therefore, it is important to work and to develop an algorithm so as to achieve global optimum for the fitness function.

2.4 Problem statement

The prime objective of the present research work is to develop and recommend an appropriate solution technique for the inverse kinematic problem of industrial robot manipulator with a view to obtain only fewer solutions that could be practically handled and used. Further the obtained solutions shown to be optimal and precise with respect to orientation and position. The developed technique should yield faster results so as to make it suitable for real time applications.

2.5 Scope of work

The development in the field of robots and is ever increasing adoption in industries has let to bring out many design and operational challenges. Researchers are invading large number of macro as well as micro problems to make the robot system as user-friendly as possible. Every single component of the robot technology has been, therefore, widened to provide research interest in multiple directions. With large number of robot manipulator configurations having their own complexity/ simplicity, the development of the operational codes has been an interesting and challenging area of research. Focusing on industrial robots in vogue, the present research work is envisaged with the following scope of the work.

The detail plan for the research work is given as:

- Based on the review and analysis of previous literature different configurations of industrial robot manipulator have been chosen. In order to include all typical configurations, the set of manipulators consists of rigid as well as semi flexible configurations, the degree of freedom ranging from 3 to 6.
- The kinematic modelling and analysis would use mathematical as well as intelligent heuristic, single and hybrid in order to find out their suitability in view of their modelling simplicity and solution efficiency. Since large numbers of such tools are available in the literature, the present work aims at only limited

to old tools and new tools. The old tools would be picked up on the basis of their performance in similar situations, whereas the new and hybrid tools would be chosen on the basis of their features that could match the character of the proposed problem.

- The very purpose of this work is limited to only developing a suitable methodology for solving the inverse kinematic problem with relative ease and by checking with existing tools and a few recently developed one including hybrid ones.

2.6 Summary

A broad study of literature reviews from all accessible sources and concerned sprightly or indirectly with the present part of work has been made. Some of the additional significant work has been extravagantly reviewed so as to expand and direction of the research in this area of work. Literature from the past till present time were explored and observed to find out the existence of present work scope for supporting the current work. A wide-ranging preparation and presentation has been covered throughout the completed work for the assistance of the readers.

Chapter 3

MATERIALS AND METHODS

3.1 Overview

In order to investigate and compare the inverse kinematic solution of the robot manipulator, it is required to select appropriate robot manipulator configuration. A robot manipulator can be considered as group of rigid links or bodies which are connected by specific joints. Joints may be revolute, prismatic, screw, universal or cylindrical etc. These joints provide relative movement in between the rigid bodies or links. First link is considered to be joined at the base of robot manipulator while the last link is free to move within the limit of workspace. In this work, some benchmark manipulators have been considered in such a way that the joint should possess 1-dof and joints are either revolute or prismatic. A revolute joint gives freedom to rotate about its axis, while prismatic joint offers joint to slide along the axis without any rotation. The selected benchmark configurations of robot manipulator have been described in the later section.

3.2 Materials

Robot manipulators are generally categorised according to their kinematic structure of open or closed chains. In chapter 1, different types of robot manipulators have been described. Despite of the types of robots, it is also required to make the classification on the basis of joint and links as explained earlier. The main focus of the research is primarily on industrial robot manipulators of which simplest configuration are 3-dof revolute robot. Considering the deployment of robots in industries for various tasks, it is apparent that robot manipulator with SCARA configuration and revolute robot with 6-dof are mostly used. Therefore, it has been planned to consider only these variety of robot manipulators, for abetting the proposed research work and deliberating on the

various issues around the problem. Therefore, some selected configurations of robot manipulators have been considered for the proposed kinematic analysis (see Tables 3.1).

Table 3.1 Configurations of robot manipulators

SN.	Kinematic Structure	Degree of freedom	Joint configuration	Kinematic motion
1	Serial	3	RRR	planar
3	Serial	4	RRPR	Spatial
4	Serial	5	RRRRR	Spatial
5	Serial	6	RRRRRR	Spatial

In this work both rigid and flexible type robot manipulators with serial structure are considered for the kinematic analysis. The serial robot manipulators are extensively used in industrial application due to the fact that they offer relatively large work envelope as compared to parallel robots with compact structure. This thesis contains seven different types of robot manipulator based on the configurations from Table 3.1. The selected manipulators are 3-dof revolute planar, 4-dof Adept One SCARA, 5-dof, Parm 2, 5-dof ASEA IRb-6, 6-dof PUMA 560, 6-dof, ABB IRB-1400 and 6-dof STAUBLI RX 160 L. 6-dof industrial manipulators are selected from Table 1.4 which are type A1, A2, B1 and C type of industrial robots. The considered manipulators are given as:

- (a) 3-dof revolute planar manipulator with RRR configuration
- (b) 4-dof Adept One SCARA manipulator with the joint configuration of RRPR
- (c) 5-dof Parm2 revolute manipulator with the joint configuration of RRRRR
- (d) 5-dof ASEA IRb-6 robot manipulator
- (e) 6-dof PUMA 560 manipulator with revolute configuration
- (f) 6-dof ABB IRB-1400 robot manipulator
- (g) 6-dof STAUBLI RX 160 L robot manipulator

The above described robot manipulator configurations are the foundation for the research work. From the last many decades researcher are working on these categories of robot manipulators as explained in previous chapter. One of the most fundamental and important problem for the positioning of the robot manipulator is kinematic analysis. Therefore, in this work different configurations of robot manipulators are selected for the kinematic analysis. For the kinematic analysis of robot manipulator one should start with the basic 3-dof revolute manipulator. On the other hand, 4-dof, 5-dof and 6-dof manipulators are mostly preferred in industrial applications due to its

high dexterity and large workspace. The detail descriptions of the selected materials are presented in the subsection.

3.2.1 Description of planar 3-dof revolute manipulator

A planar robot manipulator is can be made of serial chains with revolute or prismatic joints. Planar 3-dof revolute manipulator is basically constructed by three revolute joints. All the links or rigid bodies of a serial chain are constrained to rotate in same plane or parallel to each other. A planar manipulator can only have revolute or prismatic joints. Indeed the axes of all revolute joints should be perpendicular to the planar chain while the axes of prismatic joint should always parallel to the planar chain. Joint variables and parameters of 3-dof planar manipulator are given in Table 3.2. Main aim of this chapter is to provide details study of selected manipulators for the kinematic analysis and position of the end effector at the desired point. This section deals with the different types of planar manipulator and selection of appropriate planar manipulator for the kinematic analysis.

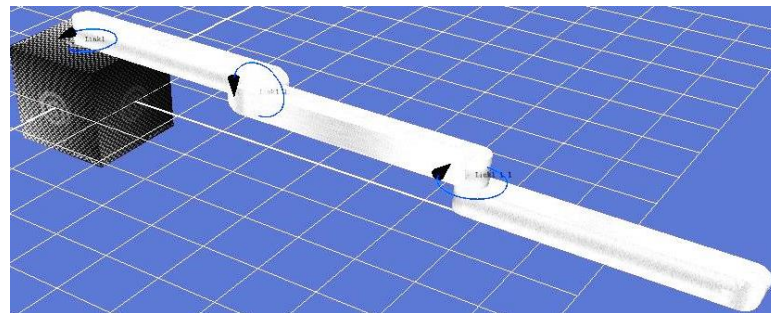


Figure 3.1 Model of 3-dof revolute manipulator

Table 3.2 Manipulator joint limits and kinematic parameters

Sl.	θ_i (degree)	d_i (mm)	a_i (mm)	α_i (degree)
1	$\theta_1 = \pm 180^0$	0	$a_1 = 100$	0
2	$\theta_2 = \pm 180^0$	0	$a_2 = 70$	0
3	$\theta_3 = \pm 180^0$	0	$a_3 = 50$	0

The mathematical modelling of higher dof or spatial manipulators is quite lengthy and time consuming. Planar manipulators are simple to figure kinematic relationship as well as for mathematical modelling. The planar manipulators examples represent the foundation for designing, kinematic analysis and for controlling purpose without consumption of time in mathematical expressions. However, this deals with the kinematic analysis of planar manipulator but the spatial description can also be

prolonged. We will start with the example of the planar 3-dof revolute manipulator as shown in Figure 3.1. There are many industrial manipulators available which resembles 3-dof revolute planar configuration. For example, swivel of shoulder, extension of elbow and pitch of Cincinnati Milacron T3 manipulator can be treated as 3-dof planar manipulator. Similarly, in case of SCARA manipulator without considering of prismatic joint will resemble the 3-dof revolute manipulator just to move end effector in up or down position. Thus, it is useful to consider 3-dof revolute planar manipulator for the inverse kinematic analysis.

The 3-dof revolute planar manipulator can be geometrically specified with the link lengths a_1 , a_2 and a_3 . These links length are basically variables which depend on the configuration of robot manipulator. The links lengths can be define in many ways but the precise way is the most distal link from distal joint axis to the end effector point or tool point. Other important variables are coordinate points of the end effector which represents the position and orientation of the end effector. The positions are defined as the coordinates (X and Y) while orientation can be define as ϕ angle. The overall variables (X, Y and ϕ) defines the pose (position and orientation) of the end effector. The proper definition of these variables and parameters can be found in the next chapter. The other possible configuration of planar manipulator can be R-P, P-P, and P-P-P. In this thesis 3-dof revolute planar manipulator is considered for the further kinematic analysis and the detail mathematical modelling of the manipulator is presented in next chapter.

3.2.2 Description of 4-dof SCARA manipulator

The second selected configuration for forward and inverse kinematic analysis is Adept One SCARA manipulator. The SCARA (Selective Compliant Assembly Robot Arm or Selective Compliant Articulated Robot Arm) has an RRPR structure. This manipulator having 4 joint axes consisting three revolute and one prismatic joint which is unlike from the spherical robot manipulator with different applications. The joints first, second and fourth are revolute and their joint is prismatic see Figure 3.2 for overview. The joint variable and related kinematic parameters for inverse kinematic solution are presented in Table 3.3. The joint motions of Adept One SCARA manipulator can be described as:

(a) Joint 1 motion

Joint 1 which is also known as shoulder swivel gives the freedom for rotation of inner link and the column and the range of the rotation is 300° .

(b) Joint 2 motion

Second joint is also referred as elbow joint which is pivot point in between inner link and outer link. The range of the motion is 294° . This joint is responsible for the lefty and righty configuration of the manipulator.

(c) Joint 3 motion

The third joint gives the vertical translation of the quill at the end and outer link with the standard stroke of 196mm (optional joint stroke may be vary up to 295mm).

(d) Joint 4 motion

The last joint is known as wrist joint which provides the rotation of the quill with the range limited to 554° . This joint motion is like human hand motion for unscrewing a bottle cap or tightening a bolt.

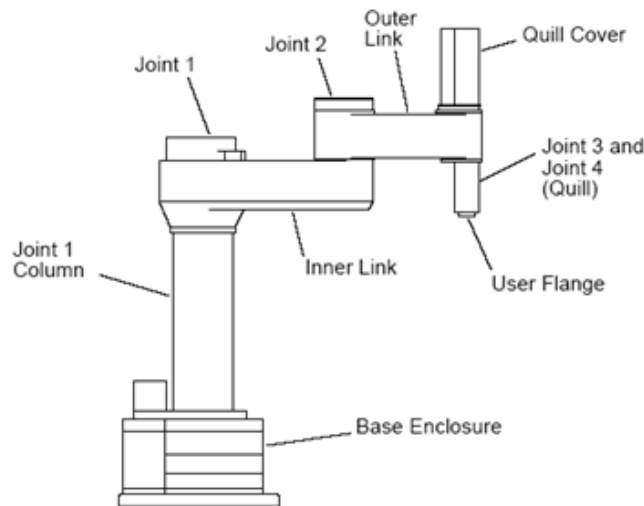


Figure 3.2 Structure of Adept One SCARA manipulator

Table 3.3 Manipulator joint limits and kinematic parameters

Sl.	θ_i (degree)	d_i (mm)	a_i (mm)	α_i (degree)
1	$\theta_1=\pm 120$	0	$a_1=250$	0
2	$\theta_2=\pm 130$	0	$a_2=150$	180
3	0	$d_3=150$	0	0
4	θ_4	$d_4=150$	0	0

This manipulator having one parallel shoulder, one elbow and rotatory wrist joints along with one linear vertical axis for translation wrist. These configurations of robot

manipulator are mostly used in light duty applications due to the high speed and precision of the manipulator. Common application areas are: electronic part assembling, printing of circuit boards, assembly of tiny parts of electromechanical device, assembling of disk drivers. The SCARA manipulators are very compact in design and work space is comparatively limited to less than 1000mm. But the payloads of the manipulators are ranged to 10-100kg. Therefore, in order to complete the kinematic analysis and performance, Adept One SCARA manipulator is selected for research.

3.2.3 Description of 5-dof revolute Pioneer2 manipulator

The third configuration considered for the kinematic analysis is Pioneer2 manipulator with 5 joint rotations. If the manipulator is redundant or having high dof, than conventional solution for inverse kinematic problem becomes more complicated. Therefore, considering newly developed Pioneer2 manipulator with 5 joint rotations for the kinematic analysis as shown in Figure 3.3. This manipulator is compact, low cost and lightweight for the use in research as well as in academic purpose. The actuation of the joint is driven by open loop servo motors and gripper which is attached to the end of the last link of manipulator.

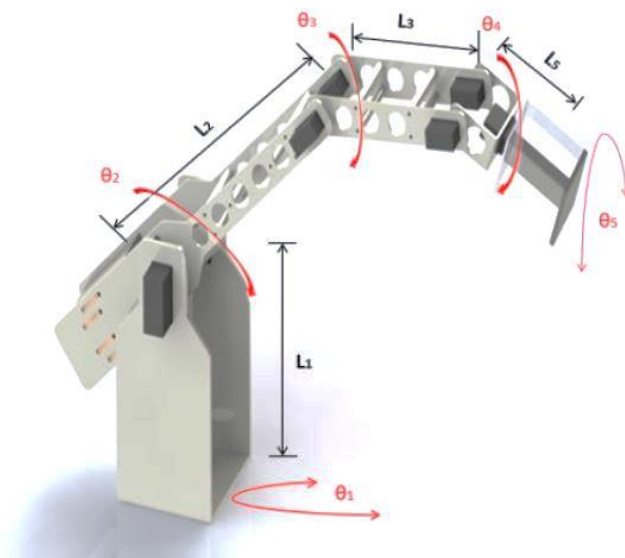


Figure 3.3 Structure of the Pioneer arm2

The major application of this robot is for grasping and manipulation of objects like soda cans up to the weight limit of 150grams within the workspace. Joint of the Parm2 are;

Joints rotations:

- base rotation
- shoulder rotation

- elbow rotation
- wrist rotation
- gripper mount
- gripper fingers

All joints are driven by servo motors except gripper fingers. The joint limits and parameters taken for the research has presented in Table 3.4.

Table 3.4 Parm2 manipulator joint limits and kinematic parameters.

Joints	θ_i (degree)	d_i (mm)	a_i (mm)	α_i (degree)
0	$\theta_1 = \pm 180^0$	$d_1 = 150$	$a_1 = 60$	-90
1	$\theta_2 = \pm 180^0$	0	$a_2 = 145$	0
2	$\theta_3 = \pm 180^0$	0	0	-90
3	$\theta_4 = \pm 180^0$	$d_2 = 125$	0	90
4	$\theta_5 = \pm 180^0$	0	0	-90
5	0	$d_3 = 130$	0	0

From Table 3.4 parameters and joint variables are listed and the ranges are presented. These parameters and joint variables are basis for the forward and inverse kinematic analysis of robot manipulator. Later using MATLAB programming the data sets for inverse kinematic solution will be used.

3.2.4 Description of 6-dof PUMA 560 manipulator

The fourth material for the kinematic analysis is PUMA 560 (Programmable Universal Machine for Assembly, or Programmable Universal Manipulation Arm) which is an industrial robot with six axis joints see Figure 3.4.

Table 3.5 Maximum limit of joint variables.

Joints	Limits (degree)
Waist	320
Shoulder	266
Elbow	284
Wrist pitch	200
Wrist roll	280
Wrist yaw	532

The end effector of the PUMA 560 robot is designed to operate a nominal load of 2.5kg with 0.1mm positional repeatability. The workspace of this manipulator is 0.92m from the centre axis to wrist centre and the maximum end effector velocity reaches 1m/s. All

six joints are actuated through the brushed DC servo motors. The joints limits and parameters are given in Table 3.5 and Table 3.6.

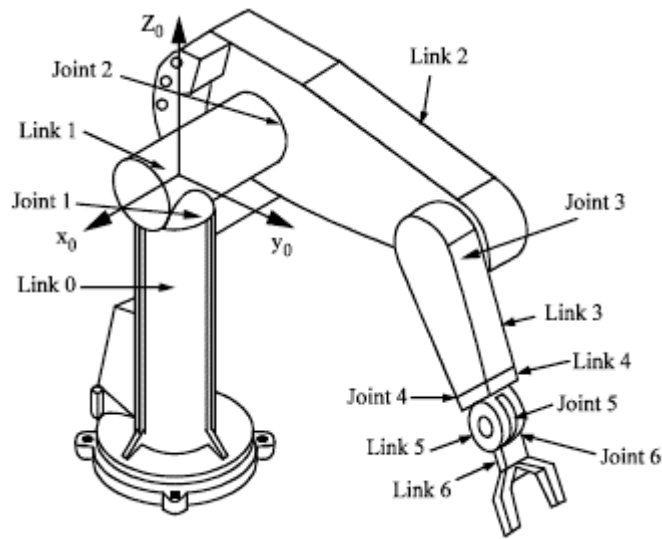


Figure 3.4 Structure of PUMA 560 robot manipulator [10]

Table 3.6 Joint variable and parameters of PUMA 560 robot

Joints	θ_i (degree)	d_i (m)	a_i (m)	α_i (degree)
0	$\theta_1 = \pm 160^0$	0	0	90
1	$\theta_2 = -225^0$ to $+45^0$	0	0	0
2	$\theta_3 = -45^0$ to $+225^0$	$d_3=0.1244$	$a_2=0.4318$	-90
3	$\theta_4 = \pm 110^0$	$d_4=0.4318$	$a_3=0.0203$	90
4	$\theta_5 = \pm 100^0$	0	0	-90
5	$\theta_6 = \pm 266^0$	0	0	0

PUMA 560 robots are most used in handling of small objects or parts in industrial due to its compact design, high speed ratio, repeatability and flexibility. Most complicated application or assembly of intricate parts can be done by PUMA 560 robot. For example PUMA 560 robot can be used for assembling of automotive panels, small electric motors, circuit board printings, appliances and so on. From Table 3.6 joint variables and parameters for DH-algorithms will be used to calculate the forward and inverse kinematic of PUMA 560 manipulator. Later the generated data sets will be input for the ANN models training and testing.

3.2.5 Description of 6-dof ABB IRb-1400 manipulator

The ABB IRB 1400 is 6-dof industrial robot which is specially desied for the manufacturing industries. The configuration of the manipulator is 6-dof revolute with

rigid structure see Table 1.4. Due to its open structure it is easily adopted for the flexible automation use and also flexible communication with external systems. This type of robot manipulator is known as anthropomorphic with 6-dof mechanism. The shoulder joint with roll and pitch motions moves the upper arm $\pm 170^\circ$ and $\pm 70^\circ$; the elbow joint with pitch actions drives the forearm $+70^\circ$ to -65° ; and the wrist roll and pitch rotations together with the tool-plate roll move the hand (see Figure 3.5). The joint limits and associated parameters are listed in Table 3.7 for ABB IRb-1400 manipulator.

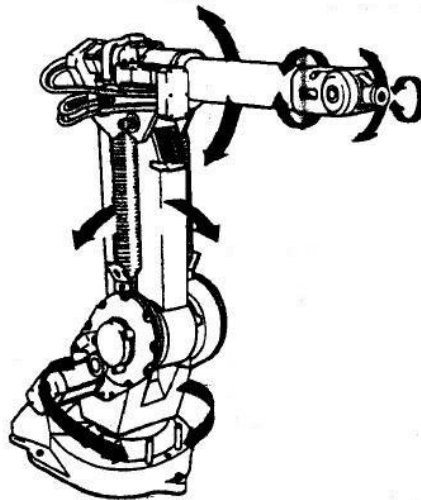


Figure 3.5 Configurations of ABB IRB 1400

Table 3.7 ABB IRB-1400 manipulator joint limits and kinematic parameters

Joints	θ_i (degree)	d_i (mm)	a_i (m)	α_i (degree)
1	$\theta_1 = \pm 170^\circ$	$d_1 = 0475$	0	0
2	$\theta_2 = \pm 70^\circ$	0	150	90
3	$\theta_3 = +70^\circ$ to -65°	0	600	0
4	$\theta_4 = \pm 150^\circ$	$d_4 = 720$	120	90
5	$\theta_5 = \pm 115^\circ$	0	0	-90
6	$\theta_6 = \pm 300^\circ$	$d_6 = 85$	0	90

This type of robot manipulator is mostly used for the arc welding process. The major advantages of this type of manipulator are its driveability, repeatability, accuracy with zero backlash and high resolution with nominal payloads. Apart from its technical advantages, it is commonly used in industries and research work. Therefore, this robot manipulator is selected for the forward and inverse kinematic analysis.

3.2.6 Description of 6-dof ASEA IRb-6 manipulator

The ASEA IRB 6 is 5-dof industrial robot manipulator which allows movement in 5-axis with maximum lifting capacity of 6 kg. This type of manipulator are commonly used in industries and research work. The configuration of the manipulator is 5-dof revolute with rigid structure see Table 1.4. The structure of the manipulator is rigid with maximum reach of 1114 mm. Due to its high dexterity it is accepted in industries for material handling, packaging, pick-n-place object, assembly etc. The basic model of this manipulator is presented in Figure 3.6.

The shoulder joint with roll and pitch motions moves the upper arm 90° to 130° and 50° to 130° ; the elbow joint with pitch actions drives the forearm -130° to -50° to -25° to -220° ; and the wrist roll and pitch rotations together with the tool-plate roll move the hand. The joint limits and associated parameters are listed in Table 3.8 for ASEA IRb-6 manipulator.

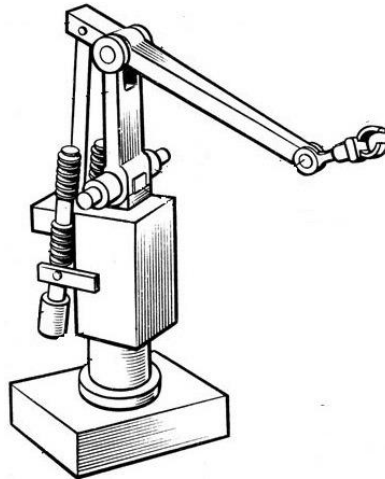


Figure 3.6 Configurations of ASEA IRb-6

Table 3.8 ASEA IRb-6 manipulator joint limits and kinematic parameters

Joints	θ_i (degree)	d_i (m)	a_i (m)	α_i (degree)
1	$\theta_1 = 90^{\circ} - 130^{\circ}$	$d_1 = 0.70$	0	90
2	$\theta_2 = 50^{\circ} - 130^{\circ}$	0	0.45	0
3	$\theta_3 = -130^{\circ}$ to -50°	0	0.67	-0
4	$\theta_4 = -25^{\circ}$ to -220°	0	0	90
5	$\theta_5 = 360^{\circ}$	$d_5 = 0.095$	0	0

This manipulator is basically designed to work on automated handling of grinding operation and later it became popular for many other applications such as material handling, packaging, assembling etc. The structure of this manipulator is compact and

rigid. This type of manipulator is also accepted for research work. Therefore, in this research work, ASEA IRb-6 robot manipulator is selected for the kinematic analysis.

3.2.7 Description of 6-dof STAUBLI RX160L manipulator

The Stäubli RX160L industrial robot manipulator is designed to perform many industrial applications such as material handling, welding, spraying, and assembling and also for research work. The structure of this manipulator is rigid with 6-axis of rotations. The main feature of this robot is high dexterity and flexibility in industrials applications. The Stäubli RX160L is resembles the human hand dexterity with 6-dof revolute joints.

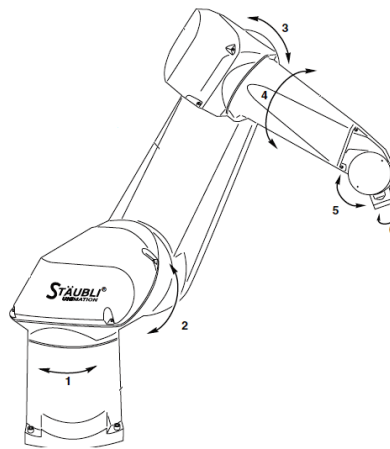


Figure 3.7 Configurations of STAUBLI RX160L

The major displacement joint angles are similar to PUMA, IRB-1400 but it allows more workspace as compared to other adopted manipulator. The maximum payload of this manipulator is 28 kg and nominal load is 14 kg. Maximum reach of this manipulator in between axis 1 to axis 6 is 2010mm which allows more work envelope. The basic model of this manipulator is presented in Figure 3.7 and joint variable with kinematic parameters are presented in Table 3.9.

Table 3.9 STAUBLI RX160L manipulator joint limits and kinematic parameters.

Joints	θ_i (degree)	d_i (m)	a_i (m)	α_i (degree)
1	$\theta_1 = \pm 180^0$	$d_1 = 0.3170$	0	-90
2	$\theta_2 = \pm 101.05^0$	0	0	90
3	$\theta_3 = \pm 180^0$	$d_3 = 0.4500$	0	-90
4	$\theta_4 = \pm 153.73^0$	0	0	90
5	$\theta_5 = \pm 270^0$	$d_5 = 0.4800$	0	-90
6	$\theta_6 = \pm 180^0$	0	0	90

3.3 Methods

The inverse kinematic robotics problem has been the focus of kinematic analysis for robot manipulators. In order to determine all possible formations to place the end effector of a robot manipulator at a particular point in space, one must compute the movements associated with each joint variable. In doing so, over the span of several decades, authors have faced the following difficulties:

- The complexity of the inverse kinematic robotics problem is determined by the geometry of the robot manipulator. Geometric solutions depends on the first three joints should be geometrically exist.
- Some calculations to solving the inverse kinematic problem cannot be computed in real-time.
- It is not always possible to obtained closed form or single solutions.
- It is also difficult to find reals solutions for some configuration of robot manipulator. Algebraic solutions; kinematic equations transform into higher order polynomial in tangent of half angle of joint angels and then all the roots of polynomials are numerically determined. (6-dof manipulator= 16th order polynomial)
- Numerical solutions- when the Jacobian matrix is singular (ill conditioned) or initial approximations is not accurate then there will be unstable solution

Main aim of this thesis is to find out the joint variables or inverse kinematic solutions for the selected benchmark manipulators. The major challenge to calculate inverse kinematic problem is, it follows the non-linear transcendental equation with complex mathematical formulations. The conventional approaches are time consuming as well as difficult to understand as explained earlier. Therefore, after considering all the stated problems the main objective is to select the appropriate method for the calculation of inverse kinematic problem. The adopted methods and steps for achieving objective have been planned as follows:

3.3.1 Conventional approach

In the present research work DH-algorithm, homogeneous transformation and quaternion vector based methods and their significance for the kinematic analysis have been considered. Mathematical modelling of the forward and inverse kinematic problem of rigid as well as semi-flexible robot with 3 to 6 joint axis is done. To reduce the methemathical complexities and computational time quaternion vector based kinematic formulations have been done for selected configurations of the robot

manipulator. The conventional kinematic equations of the open chain manipulator are transformed into consecutive quaternion transformations matrices and then articulated using quaternion. The Euler angle representation contains three angles which is not enough for regular representation and mostly trapped in singularity problem. Therefore, to overcome the problem of regular representation and for singularity avoidance quaternion algebra is much powerful. The number of mathematical operations can be reduced by quaternion vector based method. From the comparative results of homogeneous transformation methods with the quaternion based approach, mathematical operations are more in case of homogeneous transformation method. To maintain the accuracy of the obtained solution and reduce mathematical operations, quaternion based approach are much better. It can be clearly understood that the quaternion vector based method delivers a very effective and efficient tool as compared to other conventional approach. Further, this approach is cost effective due to its less mathematical operations. Comparing with the homogeneous transformation methods, it can be observed that quaternion method produces same results with less time consumption. Therefore, this method can be applied to any configuration of robot manipulator. This can be used as general tool for the kinematic solution of n-dof robot manipulator.

3.3.2 Intelligence based approaches

The second approach is based on the soft computing methods such as artificial neural network, fuzzy logic, hybrid fuzzy and hybrid ANN. These available tools have proven their efficiency to solve the non-linear and NP-hard problems. Since inverse kinematic solution yields number of alternate solutions, an appropriate iterative or intelligence based technique can be used. Forward kinematic solution of any configuration is producing exact solution. Therefore, using forward kinematic equations the input for the ANN models can be used to train the adopted network. Further trained network predicts the inverse kinematic solution of the selected configuration of the robot manipulator. Although there are many different neural networks have been tested on different configurations of the robot manipulator but the most frequent model is MLP neural network. In the present work three different networks is considered for the inverse kinematic solution. These adopted network models are not tested over the candidate manipulator under present investigation. The models of neural network which is used for the inverse kinematic solution are as follows:

- (1) Multi-layered Perceptron Neural Network (MLPNN)
- (2) Polynomial Pre-processor Neural Network (PPN)

(3) Pi-Sigma Neural Network

ANN based prediction of inverse kinematic solutions are later compared with the ANSFIS and hybrid ANNs. Similar to ANN models, ANFIS can be trained from the generated datasets using forward kinematic equations. The adaptability and learning ability increased using neural network into the fuzzy inference system. This method has already been applied in several different configurations of robot manipulators. Similar to the ANN models, ANFIS structure can be engaged to solve the nonlinear functions, NP-hard problems and can also predict the chaotic time series. The learning capability of neural networks is generally used to tune the parameters of the fuzzy logic. The learning algorithm provides the tuning of the membership function of a Sugeno type FIS (Fuzzy Inferencer System) using the input output training data. Therefore, ANFIS as well as ANN models with learning capability, adaptability, and handling of nonlinear problems which makes it suitable to solve the inverse kinematic problem.

For all selected configuration of robot manipulator FIS (Fuzzy inference system) structure are obtained and applied for prediction of the individual joint angles. Despite the advantages of the neural network and ANFIS approach for inverse kinematic resolution, a chief concern that often comes is about the convergence and stability of the solution. These networks training generally converged into the local optimum point. Therefore, neural network models can be hybridize with population based optimization algorithm to update the weight and bias of the network. The hybridization scheme has already been discussed in later chapter. The MLP neural network is most efficient and applied to many industrial manipulators. Therefore, in present work MLP neural network is hybridized with several optimization algorithms as well as comparison of gradient descent learning algorithms and appropriate scheme. After the application of the metaheuristic algorithms and trained neural network, is applied to find out the inverse kinematic solution of the robot manipulators. The adopted hybrid ANN models are as follows:

- (a) MLPPSO (Multi-layered perceptron particle swarm optimization)
- (b) MLPTLBO (Multi-layered perceptron teacher learner based optimization)
- (c) MLPGA (Multi-layered perceptron genetic algorithm)
- (d) MLPGWO (Multi-layered perceptron grey wolf optimizer)
- (e) MLPCIBO (Multi-layered perceptron crab intelligence based optimization)

Although there are many advantages of ANN and hybrid ANN that can be easily implemented for the inverse kinematic solution but important concern is computational

cost and convergence speed of the algorithm. ANN models with back propagation learning gives poor performance for the higher dof robot manipulators. The nonlinear functional relationship for higher dof problem become unstable and produces unacceptable error at the end of learning process.

3.3.3 Optimization algorithm approach

Population based optimization algorithms can be gainfully used to find out the inverse kinematic solution. The only requirement for the application of optimization algorithms is to develop the objective function for the concern manipulator. In chapter 6, objective function formulations are discussed in detail which can be further applied with minor modifications to any configuration of manipulator. Moreover, the objective function produces the candidate solution of each individual joint variable and that can be defined by the configuration vector of manipulator with number of point within the workspace limit. This method requires only the formulation of the forward kinematic equations of the robot manipulator and associated constant or parameters. This method provides flexibility to complete many task related to robot manipulator like design, kinematic analysis, synthesis of kinematic structures etc. On the other hand, for higher dof and complex task of robot manipulator, population based optimization algorithms can be used with the generic formulation of objective function. The optimization algorithms should be able to handle the problem of nonlinear, NP-hard and multimodal search problems. The different optimization algorithms are compared and used to calculate the inverse kinematic solutions are as follows:

- (1) Genetic Algorithm(GA)
- (2) Particle Swarm Optimization (PSO)
- (3) Teacher Learner Based Optimization (TLBO)
- (4) Grey Wolf Optimizer (GWO)
- (5) Crab intelligence based optimization (CIBO)

These stated algorithms are later compared with the novel developed CIBO algorithms. Many optimization algorithms require the number of control parameters setting and this increases the complexity of the adopted algorithm. The parameter associated with the algorithms can make the differences in the results like accuracy, convergence speed, efficiency, global optimum point and computational cost. Therefore, to avoid many parameter setting, novel effectual nature-inspired metaheuristic optimization technique grounded on crab behaviour is proposed (see chapter 6). The proposed Crab Intelligence Based Optimization (CIBO) technique is a population centered iterative

metaheuristic algorithm for D-dimensional and NP-hard problems. Besides using Jacobian matrix for the mapping of task space to the joint variable space, forward kinematic equations are used. Kinematic singularity is avoided using these formulations as compared to other conventional Jacobian matrix based methods. In general, proposed crab based algorithm gives generic solution of the inverse kinematic problem for some selected benchmark manipulators. But the proposed CIBO algorithm having some limitations like, it cannot apply for real time control and application for higher dof manipulator; it takes time to converge in single optimum point, etc. A concise plan of approach towards solution of the proposed problem is presented in Table 3.10. The table provides under investigation and the proposed tool(s) to be used during the research work.

Table 3.10 Adopted materials and methods

Methods	Materials		
Conventional approaches 1. HT 2. QA	Robots	Structures	Types
	3-dof revolute	Rigid(R-R-R)	Planar
	SCARA(4-dof)	Flexible(R-R-P-R)	SCARA
	Pioneer arm2(5-dof)	Rigid(R-R-R-R-R)	Spatial
	PUMA 560(6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-C
	ABB IRb-1400(6-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-A1
	ASEA IRb6 (5-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-A2
	STÄUBLI RX160 L(6-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-B2
Methods	Materials		
Intelligence approaches 1. MLPBP 2. ANFIS 3. MLPPSO 4. MLPGWO 5. PMLTLBO 6. MLPGA 7. MLPCIBO	Robots	Structures	Types
	3-dof revolute	Rigid(R-R-R)	Planar
	SCARA(4-dof)	Flexible(R-R-P-R)	SCARA
	Pioneer arm2(5-dof)	Rigid(R-R-R-R-R)	Spatial
	PUMA 560(6-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-C
	ABB IRb-1400(6-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-A1
	ASEA IRb6 (5-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-A2
	STÄUBLI RX160 L(6-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-B2
Methods	Materials		
Optimization approaches 1. PSO 2. GWO 3. TLBO 4. GA 5. CIBO	Robots	Structures	Types
	3-dof revolute	Rigid(R-R-R)	Planar
	SCARA(4-dof)	Flexible(R-R-P-R)	SCARA
	Pioneer arm2(5-dof)	Rigid(R-R-R-R-R)	Spatial
	PUMA 560(6-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-C
	ABB IRb-1400(6-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-A1
	ASEA IRb6 (5-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-A2
	STÄUBLI RX160 L(6-dof)	Rigid(R-R-R-R-R-R)	Spatial Type-B2
Where, HT- Homogeneous Transformation and QA- Quaternion Algebra			

3.4 Summary

This chapter presents the discussion of different materials and methods adopted for the kinematic analysis. The main purpose of this chapter is to avail the detail description of adopted material for kinematic analysis and different methods to achieve the objective of the thesis. The detailed derivation of inverse kinematic solution has been given in next chapter. In the result chapter inverse kinematic solution for adopted models of manipulator has been tabularised and comparison on the basis of mathematical complexity is made over other adopted method.

Chapter 4

MATHEMATICAL MODELLING AND KINEMATIC ANALYSIS

4.1 Overview

The conventional solution approach of kinematics is important in various fields of recent trend and modern technology, extending through computer graphics (e.g. animation character analysis) to expansion of space manipulation and simulators. All these fields of applications are fundamentally required to evaluate both orientation and position of the Cartesian coordinates of end effector and joint variables of robot manipulator. To evaluate the position and orientation of end effector and its joint variables one can adopt homogeneous transformation matrix method. This method is the conventional tool to describe the kinematic relationship of joint and links. Moreover, this method of representation is used from many decades for tracing the end effector position of the robot manipulator. On the other hand, it is extremely redundant for the representation of 6-dof of a system. The redundancy generally consumes more computational cost and more storage space. This is also related to the problem of mathematical operations which generally creates more complexity. Therefore, many alternative methods for the representation of non-inertial coordinates and inertial coordinates have been introduced. The proposed method should always be less complex and computationally efficient for the representation of mechanism and transformation of the system.

Keeping all in mind, alternative techniques like Epsilon algebra, quaternion and dual quaternion, Euler angle, screw transformation, exponential rotation matrix and lie algebras are required to overcome the problem of inverse kinematic, for better understanding of representations of same and deducing the mathematical operations and computational cost to ensure fast and responsive system in real environment. [89],

proposed two different approaches to the inverse- kinematics problem for a six-degree-of-freedom robot manipulator having three revolute joint axes intersecting at the wrist. One method uses three rotational generalized coordinates to describe the orientation of the body. The other method uses equivalent Euler parameters with one constraint equation. These two approaches have been incorporated into two different computer algorithms, and the results from each are compared on the basis of computational complexity, time simulation, singularity, etc. It was found that Euler parameters were less efficient than three rotational angles for solving the inverse-kinematics problem of the robot considered, and that the physical singularities caused by the robot mechanism could not be eliminated by using either of the two approaches.

[85], proposed the position of a manipulator expressed as either in joint coordinates or in Cartesian coordinates. A new algebra has been defined for the use in solving the forward and inverse kinematics problem of manipulators. The properties of the algebra are investigated and functions of an epsilon numbers are defined. The Ada language was used for illustration because of the ease in implementing the algebra and it is being used to solve the forward and inverse kinematics problems. However, the program actually used epsilon numbers and used the overloading feature of the Ada language to implement the epsilon algebra. By simply changing the order of the algebra, the resulting program can compute a time derivative of the end-effector's position when used-to solve the forward kinematics problem and any time derivative of joint positions when used to solve the inverse kinematics problem.

[111], proposed polynomial continuation method for the analysis of geometric design problem of 3-dof revolute manipulator. They have developed the elimination method for 4 point precision geometric analysis of the manipulator. In this work, each precision point of the end effector has been considered spatial configuration. DH algorithm is used in this work for the formulation of the design equations. [143], proposed solution techniques of inverse kinematics using polynomial continuation, Gröbner bases, and elimination. They compared the results that have been obtained with these techniques in the solution of two basic problems, namely, the inverse kinematics for serial-chain manipulators, and the direct kinematics of in-parallel platform devices.

[98], Proposed dual quaternion algebra based kinematic synthesis of constrained robotic system. They have proposed this method for one or more serial chain manipulator considering both prismatic and revolute joints. In this research they have used DH algorithm and successive screw displacement for determining the joint variables for the resolution of end effector position. Then dual quaternions are used to define the transformation matrices obtained through DH algorithm to simplify the design formulations of different types of manipulators. [108], presented a general method for

the classification of 6-dof industrial manipulators based on the kinematic structure and their detail analyses of kinematic equations on the basis of classification are given. They have adopted the exponential rotation matrix algebra to find out the closed form solution of inverse kinematics of robot manipulator. [112], presented pose error analysis of SCARA manipulator using screw theory. They have presented the error produced by DH algorithm and compared the same with the output of the screw based analysis of the manipulation.

From the discussed literature related to different methods of representations and kinematic analysis it can be understood that homogeneous matrix with DH-algorithm method is the well-known conventional method. Therefore the above explained method can be the benchmark method for the comparison of other alternative methods with respect to the efficiency and quality of the solution.

Therefore, from abovementioned techniques and from the previous literature review quaternion, dual quaternion, screw, exponential rotation matrix and Lie algebra are the methods which expansively used for the kinematic analysis of manipulators. But still detail description and the deep theory behind the representation of these methods are not very much clear to most of the researcher. Therefore, further detail derivation of quaternion algebra and its application without making it hectic to the readers are provided in this section. On the other hand, brief descriptions of other methods are also presented.

4.2 Representation methods and kinematics

Kinematics can be understood with the system of links or chain connected with joints to create relative motion without analysing the torque/forces or sources of the motion. Analytical study of the motion of robot link with respect to one fixed coordinate or base coordinate system with function of time could be understood as a robot kinematics. The kinematics of robot link also provides the study of its higher derivatives like velocity, jerk, acceleration etc.

4.2.1 Kinematic variables and parameters

A kinematic chain consists of kinematic pair of links which may be connected by revolute or prismatic joints subjected to rotational or translational degree of freedom. As explained in the literature there exist many approaches for the mathematical representation of kinematic chain. The major differences of these methods are the attachment of coordinate frames. Therefore Denavit-Hartenberg parameters [246], are commonly used. Homogeneous transformation matrix based methods are better for placement of coordinate's frames to the links and joint variables. The method consists of four scalars

which are known as DH parameters of kinematic chain. These scalars are used to define the geometry of link and relative displacement of joint. This method of representations reduces the mathematical/arithmetical operations for the kinematic description.

In the Figure 4.1 the position and orientation of the axis of joint can be determined with respect to the base coordinate X, Y and Z with minimum four parameters. To accomplish this, common normal OP between axis of joint and Z axis of the base frame has been drawn. Therefore the magnitude of common normal is representing length a , which is located from the d offset distance of Z axis from the origin of base frame to the point O. θ is the angle between OA which is parallel to x-axis with common normal OP. This angle represents the rotation about the z-axis which is measure in x-axis to the direction of common normal. Angle α represents the rotation of joint axis with PQ which is parallel to z-axis and measured in direction of z-axis. These four scalar a , θ , α and d are the parameters of Denavit Hartenberg parameters to represents the position of the axis of any joint in Cartesian coordinate system. In the later section detail discussion about these four parameters are given.

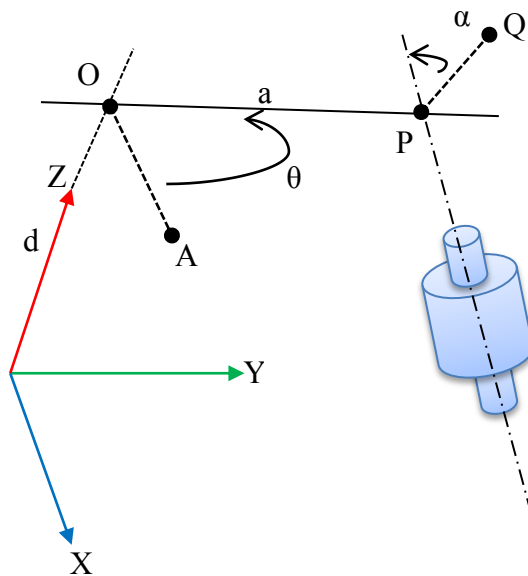


Figure 4.1 Position and direction of a cylindrical joint in a Cartesian coordinate frame

4.2.2 DH-Parameters

Now let us observe all characteristic properties of scalar parameters of DH method for modelling of considered kinematic pair in Figure 4.2. Standard method of representation has been followed without altering the concern properties of kinematic pair.

From Figure 4.2 link $i-1$ connected by cylindrical joint with link i , and $i+1$ link is consecutive link with same joint i . The attached coordinate frame with link i is

orientated in such a way that the Z_i axis is aligned with consecutive link $i+1$ and X_i -axis is aligned with common normal in between i and $i+1$. Base coordinate frame is situated at the intersection of common normal with $i+1$ axis. And the last coordinate Y_i will be placed as per right hand rule which is $y_i = z_i \times x_i$.

Therefore, from Table 4.1, DH- parameters can be defined as with considered geometry and orientation of associated links are as follows:

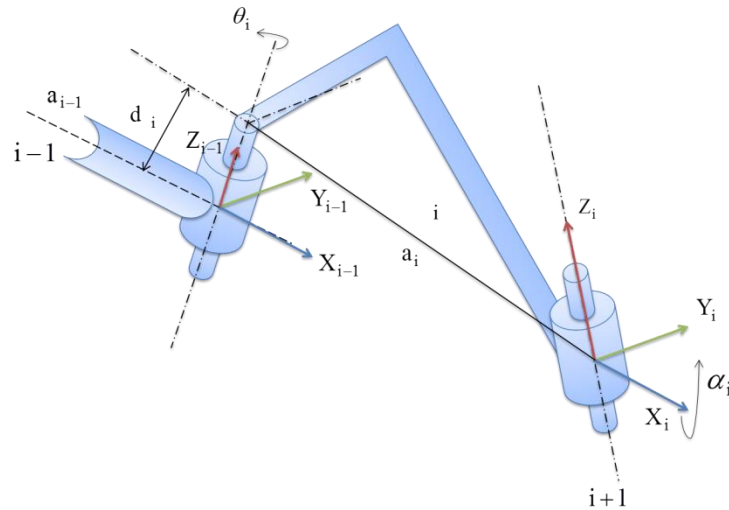


Figure 4.2 kinematic pair and DH parameters

Table 4.1 DH parameters

θ_i : Joint rotation parameter, which can be described as the angle of rotation of links i and $i-1$ which is measured from X_{i-1} to X_i about the Z_{i-1} .

a_i : Link length parameter, can be represented as length of common normal of links $i+1$ and i , measured in the direction of X_i , i.e. axis i to $i+1$.

d_i : Link offset parameter; can be described as the distance between common normal and the coordinate X_{i-1} or distance between start points of a_i in the direction of Z_{i-1} with the origin of coordinate frame.

α_i : Twist angle parameter, can be described as the inclination angle between the axes of links measured in X_i direction with Z_{i-1} to Z_i .

These parameters describes the complete geometry of kinematic pair, if the joint is revolute then θ_i , d will be only variables and rest of the parameters will be constant while in case of prismatic joint d_i will be the variable and similarly other parameters will be constant. From Figure 4.3, the coordinate frame $X_{i-1}, Y_{i-1}, Z_{i-1}$ is over imposed with frame of i joint so that the distance or offset length d_i can be described, and the

rotation angle θ_i from Y_{i-1} with i joint can be understand. The Z_{i-1} axis of coordinate frame is parallel to the imposed consecutive joint $i+1$, which gives the common normal and can be described as link length parameter a_i . This link length is perpendicular to axes $i+1$ and i and creates the twist angle α_i about axes i and $i+1$.

Therefore after description of DH parameters, mathematical expression of position of coordinate frames and imposed frame can be gives by homogeneous transformation matrix $A_{i-1,i}$, which is successive product of homogeneous transformation matrices $B_{i-1,i}$, and $C_{i-1,i}$, describing all DH parameters. Therefore, DH matrices can be gives as,

$$A_{i-1,i} = B_{i-1,i} * C_{i-1,i} \quad (4.1)$$

$$B_{i-1,i} = \begin{bmatrix} \cos \theta_i & -\sin \theta_i & 0 & 0 \\ \sin \theta_i & \cos \theta_i & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.2)$$

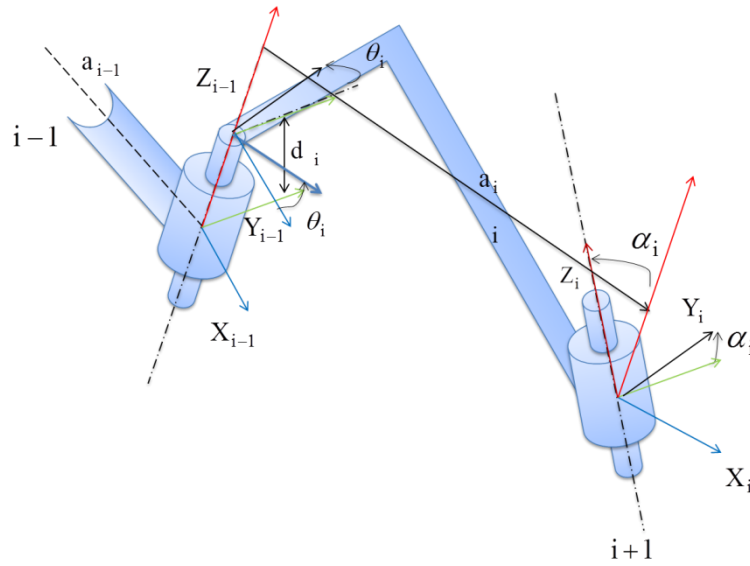


Figure 4.3 Denavit-Hartenberg parameters for successive translation and rotation of links

$$C_{i-1,i} = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & \cos \alpha_i & -\sin \alpha_i & 0 \\ 0 & \sin \alpha_i & \cos \alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.3)$$

From equation $A_{i-1,i}$ can be given as,

$$A_{i-1,i} = \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cos \alpha_i & \sin \theta_i \sin \alpha_i & a_i \cos \theta_i \\ \sin \theta_i & \cos \theta_i \cos \alpha_i & -\cos \theta_i \sin \alpha_i & a_i \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.4)$$

The above $A_{i-1,i}$ matrix can be used for any kinematics chain which contains revolute or prismatic joint for the position and orientation analysis. But in case if the joint axes i and $i+1$ are parallel then α_i will be zero and the matrix will be given as follows:

$$A_{i-1,i} = \begin{bmatrix} \cos \theta_i & -\sin \theta_i & 0 & a_i \cos \theta_i \\ \sin \theta_i & \cos \theta_i & 0 & a_i \sin \theta_i \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_i^{i-1} = A_1 A_2 A_3 A_4 A_5 A_6 \dots A_n = \begin{bmatrix} n_x & o_x & a_x & X \\ n_y & o_y & a_y & Y \\ n_z & o_z & a_z & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.5)$$

4.2.3 DH-algorithm for frame assignment

In the DH algorithm, a base coordinate frame X_0, Y_0, Z_0 is attached to fixed based of non-moving link and local coordinated will be fixed at each joint of moving links. The connected links $i-1$ to i , where $i=1,2,3\dots n$. Therefore, the basic steps of DH-algorithm for frame assignment are follows [251]:

Step 1 Base frame X_0, Y_0, Z_0 typically attached to the fixed body at the origin in such a way that axis of rotation should coincident with the Z_0 axis, while X_0 will be places arbitrarily directed towards the perpendicular of the rotation axis or it can be understand with the forward reaching direction of manipulator. Using right hand coordinate rule, the last Y_0 axis can be placed i.e. $Y_0 = Z_0 \times X_0$

Step 2 Following the second step the subsequent second joint i , rotation axis will be placed in the axis Z_{i-1} , which goes to coordinates $X_{i-1}, Y_{i-1}, Z_{i-1}$. The second coordinate frame origin will be placed on the i -th joint axis at the end of the common normal away from the joint axis $i-1$ to the joint axis i . But in case if the joint axes i and $i-1$ are parallel and joint type is revolute then the origin of the frame will be simply imposed to second joint axis confirming that $d_i=0$. otherwise in case of prismatic joint the origin of frame can be places arbitrarily along the joint axis i . Final condition of intersection of i and $i-1$, the frame will be positioned at the point of intersection.

Step 3 For moving link $i-1$, axis X_{i-1} where $i=2,3,4,..n$, will be directed towards the common normal axes of joint i and $i-1$ from $i-1$ to i . If the joint axes i and $i-1$ intersect, then axis X_{i-1} will be perpendicular to the intersecting plane and can be directed towards arbitrarily perpendicular axis. The rotation angle θ_i , will be chosen by normal direction of Z_{i-1} axis, which is basically represented between the X_{i-1} and X_i through rotation axis Z_{i-1} . Therefore third axis Y_{i-1} can be evaluated similarly with right hand coordinate rule $Y_{i-1} = Z_{i-1} \times X_{i-1}$.

Step 4 Now the placement of manipulator end effector coordinate frame X_e, Y_e, Z_e will be on the reference point of the gripper. Z_e axis will be directed anywhere in the orthogonal plane of X_e , similar to step three, X_e will be aligned with common normal of $Z_{e_{i-1}}$ and Z_{e_i} . But in case of revolute joint axis of last joint, Z_e will be considered as parallel to the previous joint axis. The last axis will be given as right hand coordinate rule $X_e = Z_e \times Y_e$.

Step 5 Finally after assignment of all coordinate frames for all links $i=1, 2, 3, \dots, n$, DH parameters can be evaluated and can be written in tabular form given in the next section and pictorial view is presented in Figure 4.4 and 4.5.

4.2.4 Mathematical modelling of 3-dof revolute manipulator

The mathematical modeling of forward and inverse kinematics of robot manipulator using homogeneous transformation matrix method with DH parameters is presented. The purpose of this application is to introduce to robot kinematics, and the concepts related to both open and closed kinematics chains. The Inverse Kinematics is the opposite problem as compared to the forward kinematics, forward kinematics gives the exact solution but in case of inverse kinematics it gives multiple solutions. The set of joint variables when added that give rise to a particular end effectors or tool piece pose. Figure 4.4 (a) shows the basic joint configuration of 3-dof revolute planar manipulator and Figure 4.4 (b) represents the model of Cincinnati Milacron T3 and used as 3-dof planar manipulator. Figure 4.5 shows the simulation of 3-dof revolute planar manipulator using DH procedure. Position and orientation of the end effectors can be written in terms of the joint coordinates in the following way,

Table 4.2 DH-parameters for 3-dof revolute manipulator

Sl.	θ_i (degree)	d_i (mm)	a_i (mm)	α_i (degree)
1	θ_1	0	a_1	0
2	θ_2	0	a_2	0
3	θ_3	0	a_3	0

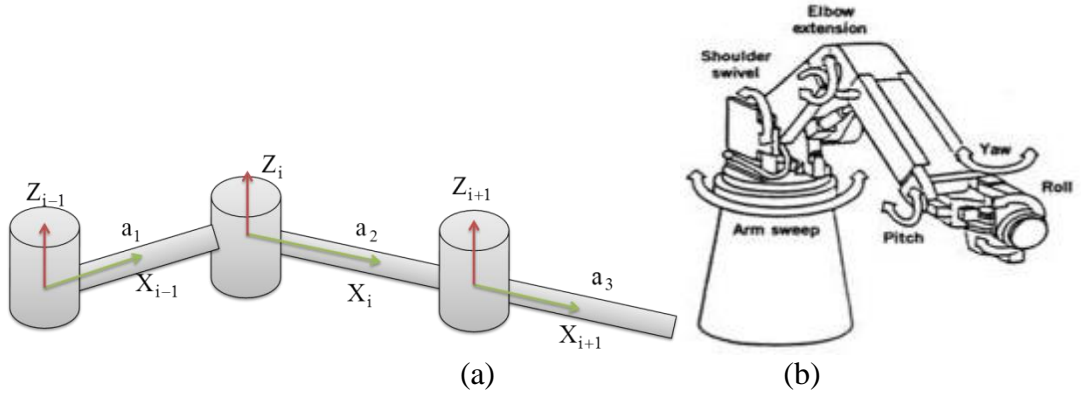


Figure 4.4 Planar 3-dof revolute manipulator

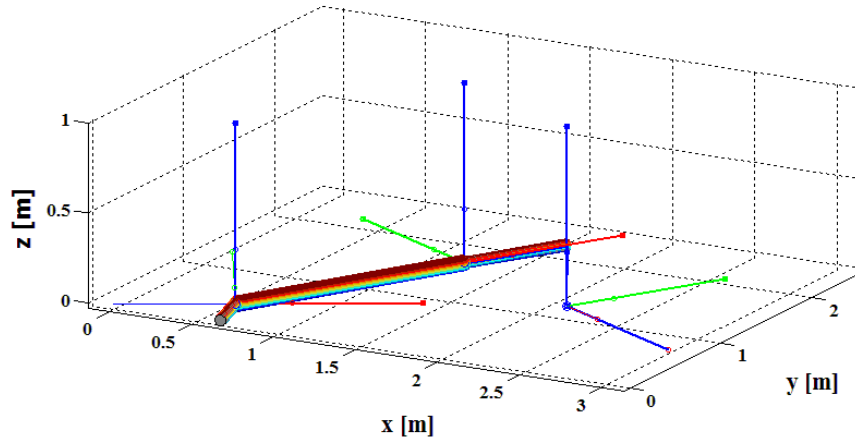


Figure 4.5 Coordinate frames of 3-dof revolute manipulator

Transformation matrix will be given by equation (4.4)

$$A_{i-1,i} = \begin{bmatrix} \cos \theta_i & -\sin \theta_i & 0 & a_i \cos \theta_i \\ \sin \theta_i & \cos \theta_i & 0 & a_i \sin \theta_i \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{i-1,i} = \begin{bmatrix} c_{123} & -s_{123} & 0 & a_1 c_1 + a_2 c_{12} + a_3 c_{123} \\ s_{123} & c_{123} & 0 & a_1 s_1 + a_2 s_{12} + a_3 s_{123} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.6)$$

where, $c_1 = \cos \theta_1$, $s_1 = \sin \theta_1$, $c_{12} = \cos(\theta_1 + \theta_2)$, $s_{12} = \sin(\theta_1 + \theta_2)$, $c_{123} = \cos(\theta_1 + \theta_2 + \theta_3)$ and $s_{123} = \sin(\theta_1 + \theta_2 + \theta_3)$

Therefore forward kinematics is given by,

$$X = a_1 c_1 + a_2 c_{12} + a_3 c_{123} \quad (4.7)$$

$$Y = a_1 s_1 + a_2 s_{12} + a_3 s_{123} \quad (4.8)$$

$$\phi = \theta_1 + \theta_2 + \theta_3 \quad (4.9)$$

ϕ represents orientation of the end effector. All the angles have been measured counter clockwise and the link lengths are assumed to be positive going from one joint axis to the immediately distal joint axis. However, to find the joint coordinates for a given set of end effectors coordinates (x, y, ϕ) ; one needs to solve the nonlinear equations for θ_1, θ_2 and θ_3 .

Inverse kinematics,

$$\theta_2 = a \tan 2(s_2, c_2) = a \tan 2\left(\pm \sqrt{1 - (c_2)^2}, \frac{x^2 + y^2 - a_1^2 - a_2^2}{2a_1 a_2}\right) \quad (4.10)$$

$$\theta_1 = a \tan 2(y, x) - a \tan 2(k_2, k_1) \quad (4.11)$$

Where,

$$k_2 = a_1 + a_2 \cos \theta_2 \text{ and } k_1 = a_2 s_2$$

$$\theta_3 = \phi - \theta_1 - \theta_2 \quad (4.12)$$

4.2.5 Mathematical modelling of 4-dof SCARA manipulator

The Denavit-Hartenberg (DH) notation and methodology are used in this section to derive the kinematics of robot manipulator. The coordinate frame assignment and the DH parameters are depicted in Figure 4.5, and listed in Table 4.3 respectively, where O_1 represents the local coordinate frames at the five joints respectively, O_4 represents the local coordinate frame at the end-effector, where θ_i represents rotation about the Z-axis, α_i rotation about the X-axis, transition along the Z-axis, and transition along the X-axis.

Table 4.3 The DH Parameters

Sl.	θ_i (degree)	d_i (mm)	a_i (mm)	α_i (degree)
1	$\theta_1 = \pm 120$	0	$a_1 = 250$	0
2	$\theta_2 = \pm 130$	0	$a_2 = 150$	180
3	0	$d_3 = 150$	0	0
4	θ_4	$d_4 = 150$	0	0

The transformation matrix A_i between two neighbouring frames O_{i-1} and O_i is expressed in equation (4.1) as,

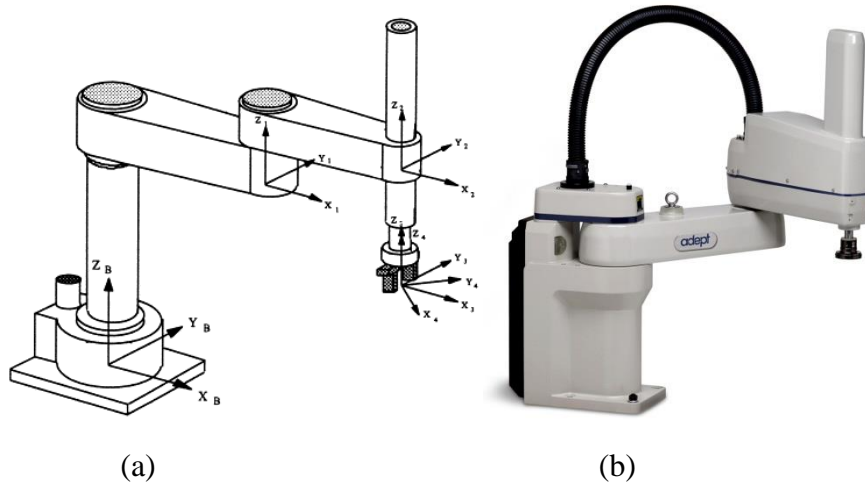


Figure 4.6 DH frames of the SCARA robot

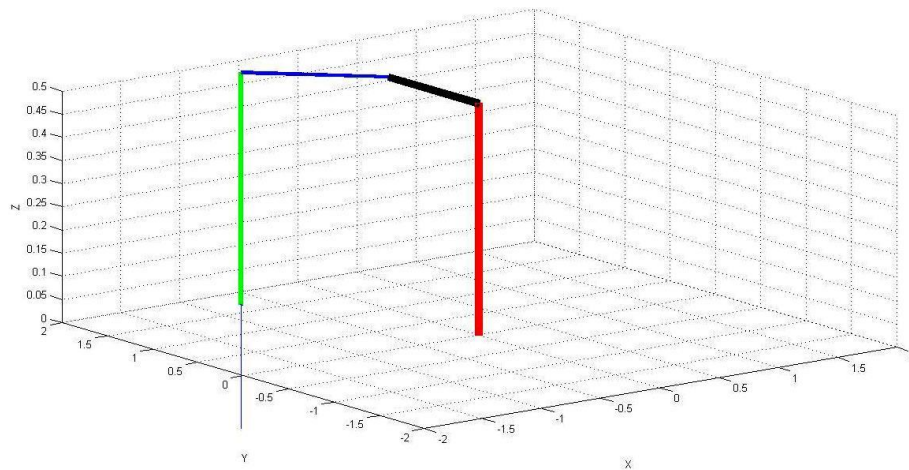


Figure 4.7 Structure of SCARA manipulator through MATLAB

By substituting the DH parameters in Table 4.3 into equation (4.3), the individual transformation matrices A_1 to A_4 can be obtained and the general transformation matrix from the first joint to the last joint of the manipulator can be derived by multiplying all the individual transformation matrices (0T_4) and final configuration of SCARA is shown in Figure 4.7.

$${}^0T_4 = A_1 A_2 A_3 A_4 = \begin{bmatrix} n_x & o_x & a_x & X \\ n_y & o_y & a_y & Y \\ n_z & o_z & a_z & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.13)$$

Where (X, Y, Z) represents the position and $\{(n_x, n_y, n_z), (o_x, o_y, o_z), \text{ and } (a_x, a_y, a_z)\}$ represents the orientation of the end-effector. The orientation and position of the end-

effector can be calculated in terms of joint angles and the DH parameters of the manipulator are shown in following matrix as:

$$\begin{bmatrix} c_{124} & -s_{124} & 0 & a_1c_1 + a_2c_{12} \\ s_{124} & c_{124} & 0 & a_1s_1 + a_2s_{12} \\ 0 & 0 & 1 & -d_3 - d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.14)$$

$$\theta_2 = \tan^{-1}\left(\frac{s_2}{c_2}\right) = \tan^{-1}\left[\frac{\pm 2a_1a_2\sqrt{1-c_2^2}}{p_x^2 + p_y^2 - a_1^2 - a_2^2}\right] \quad (4.15)$$

$$\theta_1 = \tan^{-1}\left(\frac{s_1}{c_1}\right) = \tan^{-1}\left[\frac{(a_1 + a_2c_2)p_y - a_2s_2p_x}{(a_1 + a_2c_2)p_x + a_2s_2p_y}\right] \quad (4.16)$$

$$d_3 = -p_z - d_4 \quad (4.17)$$

$$\theta_4 = \tan^{-1}\left[\frac{-n_x s_{12} + n_y c_{12}}{n_x c_{12} + n_y s_{12}}\right] \quad (4.18)$$

It is obvious from the representation given in equations (4.15) through (4.18) that there exist multiple solutions to the inverse kinematics problem. The above derivations with various conditions being taken into account provide a complete analytical solution to inverse kinematics of arm. So to know which solution holds good to study the inverse kinematics, all joints variables are obtained and compared using forward kinematics solution. This process is been applied for θ_1, θ_2, d_3 and θ_4 , to choose the correct solution, all the four sets of possible solutions (joint angles) calculated.

4.2.6 Mathematical modelling of 5-dof revolute manipulator

Similarly Denavit-Hartenberg (DH) algorithm can be used to find out the end effector position and orientation. DH parameters and associated values for 5-dof revolute manipulator have given in Table 4.4 and assigned to coordinate frames are shown in Figure 4.8 and 4.9.

Table 4.4 The DH parameters

Frame	θ_i (degree)	d_i (mm)	a_i (mm)	α_i (degree)
0	θ_1	$d_1= 150$	$a_1= 60$	-90
1	θ_2	0	$a_2= 145$	0
2	$-90 + \theta_3$	0	0	-90
3	θ_4	$d_2= 125$	0	90
4	θ_5	0	0	-90
5	0	$d_3= 130$	0	0

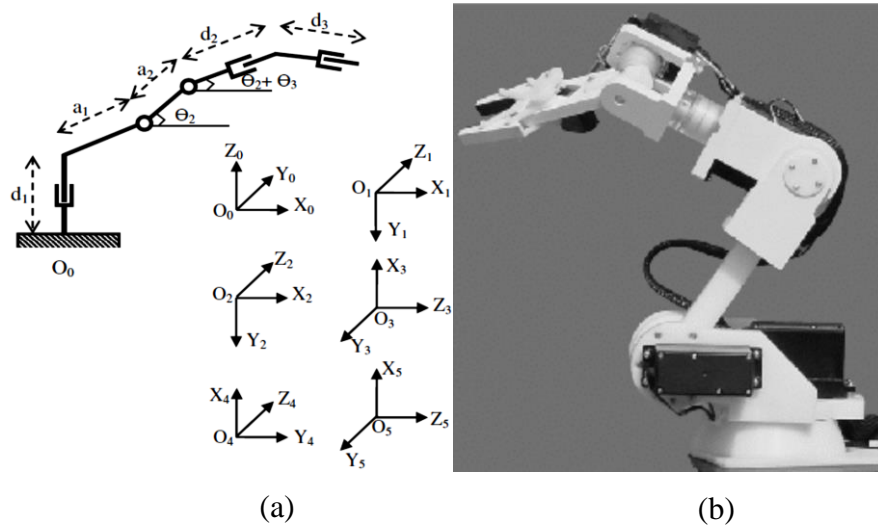


Figure 4.8 Model and coordinate frames of the manipulator

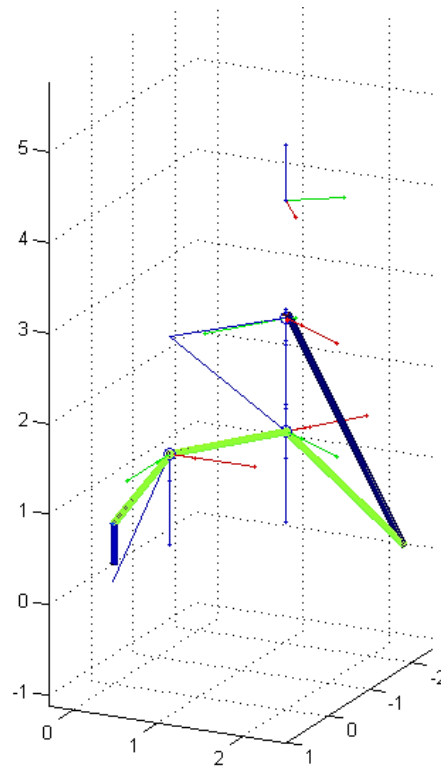


Figure 4.9 Configuration of 5-dof revolute manipulator

By substituting the DH parameters in Table 4.4 into equation (4.3), the individual transformation matrices A_1 to A_6 can be obtained and the general transformation matrix from the first joint to the last joint of the manipulator can be derived by multiplying all the individual transformation matrices given in equation (4.19) The orientation and position of the end-effector can be calculated in terms of joint angles and the DH parameters of the manipulator are shown in following matrix as:

$$\begin{bmatrix}
c_1s_{23}c_4c_5 & & -c_1s_{23}c_4s_5 & -d_3c_1s_{23}c_4s_5 \\
+s_1s_4c_5 & -c_1s_{23}s_4 & -s_1s_4s_5 & -d_3s_1s_4s_5 + d_3c_1c_{23}c_5 \\
+c_1c_{23}s_5 & +s_1c_4 & +c_1c_{23}c_5 & +d_2c_1c_{23} + a_2c_1c_2 + a_1c_1 \\
\\
s_1s_{23}c_4c_5 & & -s_1s_{23}c_4s_5 & -d_3s_1s_{23}c_4s_5 + \\
-c_1s_4c_5 & -s_1s_{23}s_4 & +c_1s_4s_5 & d_3c_1s_4s_5 + d_3s_1c_{23}c_5 \\
+s_1c_{23}s_5 & -c_1c_4 & +s_1c_{23}c_5 & +d_2s_1c_{23} + a_2s_1c_2 + a_1s_1 \\
\\
c_{23}c_4c_5 & & -c_{23}c_4s_5 & -d_3c_{23}c_4s_5 \\
-s_{23}s_5 & -c_{23}s_4 & -s_{23}c_5 & -d_3s_{23}c_5 - d_2s_{23} \\
& & & -a_2s_2 + d_1 \\
0 & 0 & 0 & 1
\end{bmatrix} \quad (4.19)$$

From equation (4.19), we can get positional equations

$$\mathbf{X} = -d_3c_1s_{23}c_4s_5 - d_3s_1s_4s_5 + d_3c_1c_{23}c_5 + d_2c_1c_{23} + a_2c_1c_2 + a_1c_1 \quad (4.20)$$

$$\mathbf{Y} = -d_3s_1s_{23}c_4s_5 + d_3c_1s_4s_5 + d_3s_1c_{23}c_5 + d_2s_1c_{23} + a_2s_1c_2 + a_1s_1 \quad (4.21)$$

$$\mathbf{Z} = -d_3c_{23}c_4s_5 - d_3s_{23}c_5 - d_2s_{23}c_5 - d_2s_{23} - a_2s_2 + d_1 \quad (4.22)$$

$$n_x = c_1s_{23}c_4c_5 + s_1s_4c_5 + c_1c_{23}s_5 \quad (4.23)$$

$$n_y = s_1s_{23}c_4c_5 - c_1s_4c_5 + s_1c_{23}s_5 \quad (4.24)$$

$$n_z = c_{23}c_4c_5 - s_{23}s_5 \quad (4.25)$$

$$o_x = -c_1s_{23}s_4 + s_1c_4 \quad (4.26)$$

$$o_y = -s_1s_{23}s_4 - c_1c_4 \quad (4.27)$$

$$o_z = -c_{23}s_4 \quad (4.28)$$

$$a_x = -c_1s_{23}c_4s_5 - s_1s_4c_5 + c_1c_{23}c_5 \quad (4.29)$$

$$a_y = -s_1s_{23}c_4s_5 + c_1s_4s_5 + s_1c_{23}c_5 \quad (4.30)$$

$$a_z = -c_{23}c_4s_5 - s_{23}c_5 \quad (4.31)$$

The position and orientation of end effector can be obtained from equations (4.19) through (4.30). These equations provide the forward kinematic solution of robot manipulator. As we know the complexity of the above equation can lead to more mathematical complexity for derivation of inverse kinematics, due to its successive mathematical operations. Therefore, it is required to make some techniques to solve these equations for inverse kinematic derivation of the manipulator.

Using equations (4.19) and (4.28),

$$X - d_3 a_x = c_1(d_2 c_{23} + a_2 c_2 + a_1) \quad (4.32)$$

Similarly by using equations (4.22) and (4.33),

$$Y - d_3 a_y = s_1(d_2 c_{23} + a_2 c_2 + a_1) \quad (4.33)$$

It can be understand that the θ_2 and θ_3 joint angles are totally dependent on the position of end effector so it can be fixed as well as it generally creates more effect on the entire system. In case if $(d_2 c_{23} + a_2 c_2 + a_1) \neq 0$ then $X - d_3 a_x$ and $Y - d_3 a_y$ will not be equals to zero. If it is more than zero then θ_1 will be given by,

$$\theta_1 = a \tan 2(Y - d_3 a_y, X - d_3 a_x) \quad (4.34)$$

Otherwise,

$$\theta_1 = a \tan 2(d_3 a_y - Y, d_3 a_x - X) \quad (4.35)$$

Now for the derivation of θ_2 and θ_3 , equations (4.32) and (4.33) can be manipulated as,

$$d_2 c_{23} + a_2 c_2 = (X - d_3 a_x) / c_1 - a_1 \quad (4.36)$$

$$d_2 c_{23} + a_2 c_2 = (Y - d_3 a_y) / s_1 - a_1 \quad (4.37)$$

Now using equations (4.20) and (4.31),

$$Z - d_3 a_z = -d_2 s_{23} - a_2 s_2 + d_1 \quad (4.38)$$

Now considering (4.36) and (4.37),

Let

$$r = (X - d_3 a_x) / c_1 - a_1 \quad (4.39)$$

and

$$r_z = -d_2 s_{23} - a_2 s_2 + d_1 \quad (4.40)$$

Squaring and adding the equations (4.39) and (4.40),

$$d_2^2 + 2a_2 d_2 (c_2 c_{23} + s_2 s_{23}) + a_2^2 = r^2 + r_z^2 \quad (4.41)$$

Solving the terms $c_2 c_{23} + s_2 s_{23}$ in the above equation (4.41), we get

$$\begin{aligned} (c_2 c_{23} + s_2 s_{23}) \cos \theta_3 &= -\cos(\theta_3 - \pi) \\ &= \cos(-\theta_3) = -\cos(\pi - \theta_3) \end{aligned}$$

Therefore, θ_3 gives many possible solutions,

$$\theta_3 = \pm \arccos \left(\frac{a^2 + r_z^2 - a_2^2 - d_2^2}{2a_2 d_2} \right) \quad (4.42)$$

Or,

$$\theta_3 = \pm \left[\pi - \arccos \left(\frac{a^2 - d_2^2 - r^2 + r_z^2}{2a_2 d_2} \right) \right] \quad (4.43)$$

Rewriting equation (4.38) for the solution of θ_2 ,

$$d_2 s_{23} = B_1 - a_2 s_2 \quad (4.44)$$

where, $d_2 a_z - p_z + d_1 = B_1$

Considering the equations (4.36) and (4.37), equation (4.45) is derived as,

$$d_2 c_{23} + a_2 c_2 = \pm \sqrt{(-a_x d_3 + X)^2 + (-a_y d_3 + Y)^2} \quad (4.45)$$

Let $B_2 = \pm \sqrt{(-a_x d_3 + X)^2 + (-a_y d_3 + Y)^2}$,

so equation (4.46) can be rewritten as,

$$d_2 c_{23} = B_2 - a_2 c_2 \quad (4.46)$$

For solution of B_1, B_2 , rearranging equation (4.42), (4.43)

$$B_1 = (d_2 c_3 + a_2) s_2 + (d_2 s_3) c_2 \quad (4.47)$$

$$B_2 = (d_2 c_3 + a_2) c_2 - (d_2 s_3) s_2 \quad (4.48)$$

Diving both side of (4.47) and (4.48), by $\sqrt{B_1^2 + B_2^2}$, equation (4.49) and (4.50) is derived as,

$$\cos \theta * \sin \theta_2 + \sin \theta * \cos \theta_2 = \frac{B_1}{\sqrt{B_1^2 + B_2^2}} \quad (4.49)$$

$$\cos \theta * \sin \theta_2 - \sin \theta * \cos \theta_2 = \frac{B_2}{\sqrt{B_1^2 + B_2^2}} \quad (4.50)$$

where, $\cos \theta = \frac{(d_2 c_3 + a_2)}{\sqrt{B_1^2 + B_2^2}}$ and $\sin \theta = \frac{(d_2 s_3)}{\sqrt{B_1^2 + B_2^2}}$

The equation (4.48) and (4.49) are rewritten as,

And,

$$\sin(\theta + \theta_2) = \frac{B_1}{\sqrt{B_1^2 + B_2^2}} \quad (4.51)$$

$$\cos(\theta + \theta_2) = \frac{B_2}{\sqrt{B_1^2 + B_2^2}} \quad (4.52)$$

Therefore, $\theta + \theta_2 = \text{atan2}(B_1, B_2) + 2\pi$ and $\theta = \pm \text{acos} \frac{(d_2 c_3 + a_2)}{\sqrt{B_1^2 + B_2^2}}$,

It is clear that θ could be in $[0, \pi]$ or $[-\pi, 0]$. The range of will depend on the range of θ_3 . Therefore, if $0 \leq \theta_3 \leq \pi$, then $s_3 > 0$ and $\sin(\theta) < 0$, thus $0 \leq \theta \leq \pi$. Then θ_2 can be derived as:

$$\theta_2 = \text{atan2}(B_1, B_2) - \text{acos} \frac{(d_2 c_3 + a_2)}{\sqrt{B_1^2 + B_2^2}} + 2\pi \quad (4.53)$$

Otherwise, if $-\pi < \theta_3 < 0$, then $s_3 < 0$ and $\sin(\theta) < 0$, thus $-\pi < \theta < 0$. Then the next possible solution for θ_2 is as:

$$\theta_2 = \text{atan2}(B_1, B_2) + \text{acos} \frac{(d_2 c_3 + a_2)}{\sqrt{B_1^2 + B_2^2}} + 2\pi \quad (4.54)$$

Now that θ_1, θ_2 and θ_3 are known, the solutions for θ_4 and θ_5 can be found by using the remaining forward kinematics equations. Considering equation (4.28), the value of

$$s_4 = -\frac{o_z}{c_{23}}, \text{ when } c_{23} \neq 0 \quad (4.55)$$

Similarly from equation (4.26) and (4.27), the possible solution for c_4 is derived as:

$$c_4 = \frac{(o_x - c_1 s_{23} o_z / c_{23})}{s_1} \quad (4.56)$$

And again

$$c_4 = \frac{-(o_y - s_1 s_{23} o_z / c_{23})}{c_1} \quad (4.57)$$

Using equation (4.56) and (4.57) for small value of c_1 , the solution for θ_4 is

$$\theta_4 = a \tan 2 \left(-\frac{o_z}{c_{23}}, \frac{(o_x - c_1 s_{23} o_z / c_{23})}{s_1} \right) \quad (4.58)$$

Otherwise for small s_1 ,

$$\theta_4 = a \tan 2 \left(\frac{-o_z}{c_{23}}, \frac{-(o_y - s_1 s_{23} o_z / c_{23})}{c_1} \right) \quad (4.59)$$

Now for solution of θ_5 , considering equation (4.25), the value of

$$c_5 = \frac{n_z + s_{23} s_5}{c_{23} c_4} \quad (4.60)$$

Similarly the value of s_5 is derived by using equation (4.31) i.e.,

$$s_5 = -\frac{a_z + s_{23} c_5}{c_{23} c_4} \quad (4.61)$$

Using equation (4.57) in (4.56) and vice versa, the term c_5 and s_5 is rewritten as:

$$c_5 = \frac{n_z c_{23} c_4 - s_{23} a_z}{c_{23}^2 c_4^2 + s_{23}^2} \quad \text{And} \quad s_5 = -\frac{(a_z c_{23} c_4 + s_{23} n_z)}{c_{23}^2 c_4^2 + s_{23}^2}$$

Now using this above derivation of c_5 and s_5 , θ_5 is derived as follows:

$$\theta_5 = a \tan 2 \{ -(a_z c_{23} c_4 + s_{23} n_z), (n_z c_{23} c_4 - s_{23} a_z) \} \quad (4.62)$$

As per the inverse kinematic solution of 5-dof revolute manipulator, it can be understand similar to SCARA solution, exist multiple solution while in case of forward kinematics it provides unique solution. So to know which solution is giving better results for all joint variables are evaluated using MATLAB and compared the obtain solution in the result chapter.

4.2.7 Mathematical modelling of PUMA 560 robot manipulator

DH parameters and associated values for PUMA 560 manipulator have given in Table 4.5 and assigned coordinate frames are shown in Figure 4.10,

Table 4.5 The DH parameters

Frame	θ_i (degree)	d_i (m)	a_i (m)	α_i (degree)
0	θ_1	0	0	0
1	θ_2	0	0	-90
2	θ_3	$d_3=0.1244$	$a_2=0.4318$	0
3	θ_4	$d_4=0.4318$	$a_3=0.0203$	-90
4	θ_5	0	0	90
5	θ_6	0	0	-90

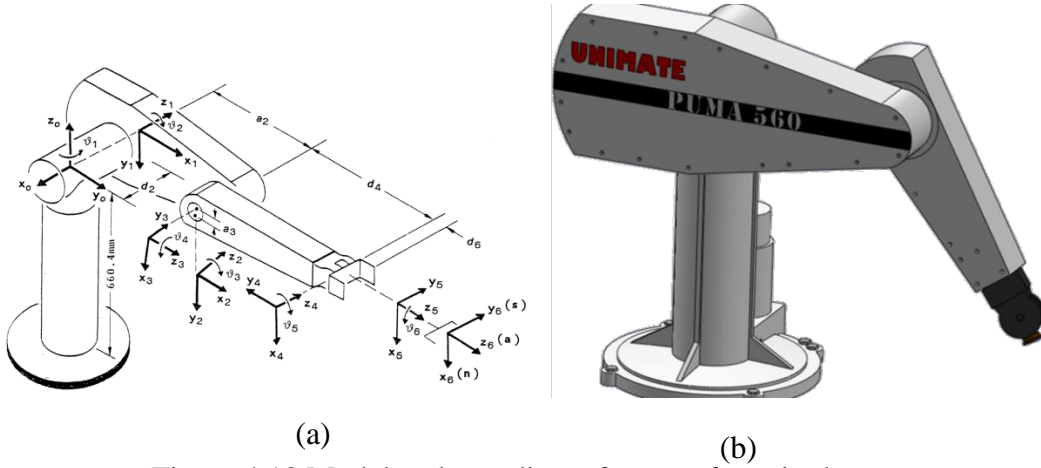


Figure 4.10 Model and coordinate frames of manipulator

Forward kinematics of the PUMA 560 robot can be given from the transformation matrix as:

$$X = c_1(d_4s_{23} + a_3c_{23} + a_2c_2) - s_1d_3 \quad (4.63)$$

$$Y = s_1(d_4s_{23} + a_3c_{23} + a_2c_2) + c_1d_3 \quad (4.64)$$

$$Z = -(-d_4c_{23} + a_3s_{23} + a_2s_2) \quad (4.65)$$

$$a_x = c_1(c_{23}c_4s_5 + s_{23}c_5) - s_1s_4s_5 \quad (4.66)$$

$$a_y = s_1(c_{23}c_4s_5 + s_{23}c_5) + c_1c_4c_5 \quad (4.67)$$

$$a_z = -s_{23}c_4s_5 + c_{23}c_5 \quad (4.68)$$

$$o_x = c_1[-c_{23}(c_4c_5s_6 + s_4c_6) + s_{23}s_5s_6] + s_1[-s_4c_5s_6 + c_4c_6] \quad (4.69)$$

$$o_y = s_1[-c_{23}(c_4c_5s_6 + s_4c_6) + s_{23}s_5s_6] + c_1[-s_4c_5s_6 + c_4c_6] \quad (4.70)$$

$$o_z = s_{23}(c_4c_5s_6 + s_4c_6) + c_{23}s_5s_6 \quad (4.71)$$

$$n_x = c_1[c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6] - s_1[s_4c_5c_6 + c_4s_6] \quad (4.72)$$

$$n_y = s_1[c_{23}(c_4c_5s_6 - s_4s_6) - s_{23}s_5c_6] + c_1[s_4c_5c_6 + c_4s_6] \quad (4.73)$$

$$n_z = s_{23}(c_4c_5c_6 - s_4s_6) - c_{23}s_5c_6 \quad (4.74)$$

Using forward kinematic equations (4.63) through (4.74), inverse kinematic of PUMA 560 manipulator can be derived as below,

$$\theta_1 = a \tan 2(\pm\sqrt{X^2 + Y^2 - d_3^2}, d_3) - a \tan 2(X, Y) \quad (4.75)$$

$$\theta_2 = a \tan 2(-Z, \pm\sqrt{X^2 + Y^2 - d_3^2}) - a \tan 2(\pm\sqrt{a_3^2 + d_4^2 - b_2^2}, b_2 + a_2) \quad (4.76)$$

where, $b_2 = \frac{p^2 - a_2^2 - a_3^2 - d_3^2 - d_4^2}{2a_2}$, $p = \sqrt{P_x^2 + P_y^2 + P_z^2}$, θ_2 can also be expressed in other form:

$$\theta_2 = a \tan 2(\pm\sqrt{X^2 + Y^2 - d_3^2}, Z - a \tan 2(\pm\sqrt{a_3^2 + d_4^2 - b_2^2}, b_2 + a_2)) - \frac{\pi}{2} \quad (4.77)$$

$$\theta_2 = a \tan 2(\pm\sqrt{a_3^2 + d_4^2 - b_2^2}, b_2) - a \tan 2(d_4, a_3) \quad (4.78)$$

We can separate the arm and wrist if the manipulator has spherical wrist. Therefore rotation matrix for arm can be given by:

$$R_A = \begin{bmatrix} c_1c_{23} & s_1 & -c_1s_{23} \\ s_1c_{23} & -c_1 & -s_1s_{23} \\ -s_{23} & 0 & -c_{23} \end{bmatrix} \quad (4.79)$$

Position matrix for arm can be given by:

$$P_A = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} c_1(c_{23}a_3 - s_{23}d_4 + c_2a_2) - s_1d_3 \\ s_1(c_{23}a_3 - s_{23}d_4 + c_2a_2) + c_1d_3 \\ -s_{23}a_3 - c_{23}d_4 - s_2a_2 \end{bmatrix} \quad (4.80)$$

Now general equation for spherical wrist can be evaluated from mapping of z-y-z Euler angle into given rotation matrix:

$$Z-Y-Z(\theta_4, -\theta_5, \theta_6) = G \quad (4.81)$$

Where, $G = R_A^T R$

Therefore we can evaluate elements of matrix G from equation (4.24),

$$g_{1i} = (c_1c_{23})r_{1i} + (s_1c_{23})r_{2i} - s_{23}r_{3i} \quad (4.82)$$

$$g_{2i} = s_1r_{1i} - c_1r_{2i} \quad (4.83)$$

$$g_{3i} = -(c_1s_{23})r_{1i} - (s_1s_{23})r_{2i} - c_{23}r_{3i} \quad (4.84)$$

Therefore,

$$\theta_4 = \begin{cases} -\pi - a \tan 2(g_{23}, g_{13}), & \text{if } \sigma_3 < 0 \\ a \tan 2(g_{23}, g_{13}), & \text{if } \sigma_3 > 0 \end{cases} \quad (4.85)$$

Where, $\sigma_3 = \pm \sqrt{g_{31}^2 + g_{32}^2}$

$$\theta_5 = -a \tan 2(\sigma_3, g_{33}) \quad (4.86)$$

$$\theta_6 = \begin{cases} -a \tan 2(g_{32}, g_{31}), & \text{if } \sigma_3 < 0 \\ \pi - a \tan 2(g_{32}, g_{31}), & \text{if } \sigma_3 > 0 \end{cases} \quad (4.87)$$

Similar to previous derivation of forward kinematic, equations (4.63) through (4.74) can be implemented for positioning of end effector with known joint variables. Thereafter inverse kinematics solution can be found using equations (4.75), (4.76), (4.77), (4.78), (4.85), (4.86) and (4.87).

4.3 Quaternion algebra kinematics

There have been tremendous work completed in the field of kinematics and recently after development of quaternion algebra some identities are added to quaternion for enhancing the efficiency and quality of results. Clifford developed dual number concept using quaternion algebra and named it dual quaternion algebra which is power full mathematical tool for design, synthesis and for computer graphics applications. This method is widely used in the field of robot kinematics using few more entities like screw displacement, exponential rotation matrix etc. which is used to represent position and orientation of mechanism. The most important advantage of quaternion algebra reduces the mathematical operations for kinematics analysis as well as gives the singularity free analysis. Therefore, it yields numerically stable equations for the kinematic and synthesis of mechanism. On the basis of application quaternion algebra can be treated as powerful analytical tool for calculation the transformations of mechanism and their representation. However, quaternions are not that much popular in the field of robot kinematics and dynamics due to the difficulty of interpretation in 3D space. Therefore to overcome this problem, the quaternion treatments for real numbers using linear algebra and matrices is proposed. In this work two operators related to real quaternion, are determined and formulated. These operators are used to translate quaternion into the matrix which is easier to understand and for applications. Quaternions are basically extensions of complex number having four fractals, with one real number with following some rule three imaginary values. This is also known as 4-dimensional components.

4.3.1 Mathematical background

In this section mathematical background of quaternion algebra is presented and its application for the derivation of forward and inverse kinematics is discussed. Quaternion can be used for both rotation and translation of a point, line, etc. with references to base coordinate system without use of homogeneous transformation matrix. Interpolation of the sequence of rotations and translations are quite easy in quaternion as compared to Euler angles. It generally lies in isotropic space that is generalization of sphere surface topology. A brief discussion about the quaternion mathematics is described in this segment for evaluation of references and to give important background for mathematical derivation of inverse kinematic of robot manipulator.

Quaternion algebra implemented by Hamilton, has shown their potential in various fields like differential geometry, design, analysis and synthesis of manipulators and mechanism, simulations etc. In quaternion algebra having four dimensions and each dimension consists of four different scalar numbers, in which one is real number and rest are imaginary dimensions. This three imaginary components having value of $i = \sqrt{-1}$ and all are mutually orthogonal to each other, and can be represented as i, j and k . therefore quaternion can be represented as;

$$\begin{aligned} h &= r + ix + jy + kz \\ h &= (r, x, y, z) \\ h &= (r, v) \end{aligned} \tag{4.88}$$

where r is the scalar component of h , and $v=\{x,y,z\}$ form the vector part, in which $r \in \mathbb{R}, x, y, z \in \mathbb{R}^3$ and i, j, k are mutually orthogonal imaginary units, whose composition rule can be stated concisely as follows,

$$1=(1,0,0,0), i=(0,1,0,0), j=(0,0,1,0), k=(0,0,0,1)$$

where multiplication of imaginary values can be explained as:

$$i^2 = j^2 = k^2 = ijk = -1 \text{ and } \begin{aligned} ij &= k, jk = i, ki = j \\ ji &= -k, kj = -i, ik = -j \end{aligned}$$

a) Conjugate of quaternion

In this case magnitude will be same but the sign of imaginary parts will be changes therefore from equation (4.89), conjugate is as follows;

$$\text{conj}(h) = r - ix - jy - kz \tag{4.89}$$

$\text{conj}(h)$ can also be represented as h' .

b) Magnitude of quaternion

Magnitude of quaternion can be explained as,

$$h = r + ix + jy + kz = \sqrt{r^2 + x^2 + y^2 + z^2} \quad (4.90)$$

c) Norm

Norm for quaternion can be explained as

$$\|h\| = \sqrt{h * h'} = \sqrt{r^2 + x^2 + y^2 + z^2} \quad (4.91)$$

d) Quaternion inverse

Quaternion inverse can be calculated as ratio of conjugate quaternion to its magnitude,

$$h^{-1} = \frac{h'}{(h * h')} \quad (4.92)$$

4.3.2 Quaternion rotation and translation

As it is clear from the above discussion that the quaternion deals with four dimensional spaces so it is quite difficult to explain it physically it can be understood with quantity that represents a rotation as show in Figure 4.11. Now the rotation of a point in a space can be explained from equation (4.93).

$$h = \cos\left(\frac{\theta}{2}\right) + i * \sin\left(\frac{\theta}{2}\right) + j * \sin\left(\frac{\theta}{2}\right) + k * \sin\left(\frac{\theta}{2}\right) \quad (4.93)$$

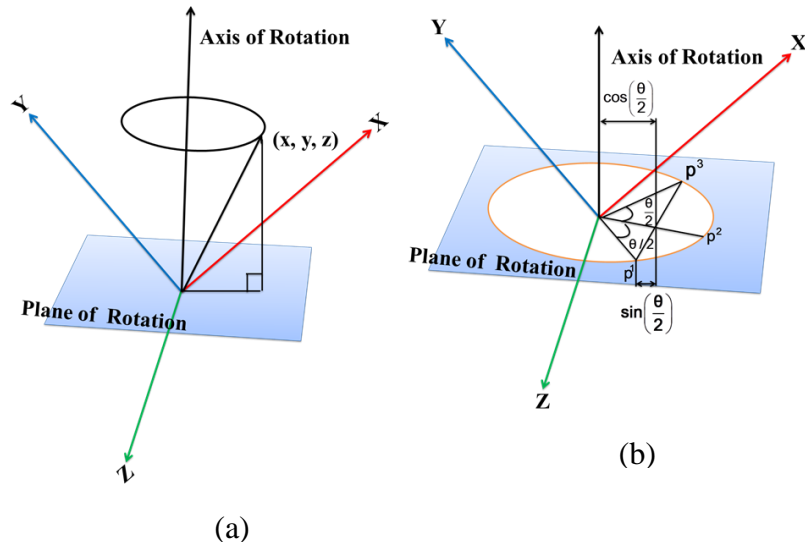


Figure 4.11 Representation of rotation

In 4 dimensional space it is quite difficult to imagine 4th axis therefore in Figure 4.11 (a), a point around the rotation axis (X, Y, Z), that is unit distance from the origin and tracing a plane of circle. When the circle is projected to the rotation plane there is point p1 rotating by angle θ to point p3 which is passing by mid-point p2. Therefore p1 point

is transforming to p^3 following by straight line makes $\cos(\theta/2)$ and $\sin(\theta/2)$. From Figure 4.11 (b) p^1 is the point vector representing initial position and p^3 is the point vector final condition to be transformed. Therefore, two quaternions can be represented on the basis of above discussed concept. If there is subsequent rotation of two quaternions h_1 and h_2 then the composite rotations $h_1 * h_2$ can be given by equation (4.94) as,

$$\begin{aligned} p^3 &= h_2 * (h_1 * p^1 * h_1^{-1}) * h_2^{-1} \\ &= (h_2 * h_1) * p^1 * (h_1^{-1} * h_2^{-1}) \\ &= (h_2 * h_1) * p^1 * (h_2^{-1} * h_1^{-1}) \end{aligned} \quad (4.94)$$

Now pure translations t_r can be done by quaternion operator that is given below,

$$t_r = h + p^1 \quad (4.95)$$

Quaternion transform can be given by,

$$p^2 = h * p^1 * h^{-1} \quad (4.96)$$

Finally, an equivalent expression for the inverse of a quaternion-vector pairs can be written as,

$$H^{-1} = [\langle h^{-1} \rangle, \langle -h^{-1} \otimes P \otimes h \rangle] \quad (4.97)$$

Where, $-h^{-1} \otimes P \otimes h = -P + [-2s(v \times (-P)) + 2v \times (v \times (-P))]$

where it is implied that the product of any two terms in the above expressions is indeed a quaternion product, which is defined in the most general form for two quaternions $h_1 = (r_1, v_1)$, and $h_2 = (r_2, v_2)$ as

$$h_1 \otimes h_2 = [r_1 r_2 - v_1 \bullet v_2 \quad r_1 v_2 + r_2 v_1 + v_2 \times v_1] \quad (4.98)$$

where $v_1 \bullet v_2$ and $v_2 \times v_1$ denote the familiar dot and cross products respectively, between the three-dimensional vectors v_1 and v_2 . Obviously, quaternion multiplication is not commutative, since the vector cross product is not. The set of elements $\{\pm 1, \pm i, \pm j, \pm k\}$ form a group (known as the quaternion group) of order 8 under multiplication.

Similarly the quaternion multiplication for two point vector transformation can be calculated as

$$H_1 \otimes H_2 = (h_1, P^1) \otimes (h_2, P^2) = h_1 * h_2, h_1 * P^2 * h_1^{-1} + P^1 \quad (4.99)$$

Where, $h_1 * P^2 * h_1^{-1} = P^2 + 2r_1(v_1 \times P^2) + 2v_1 \times (v_1 \times P^2)$

4.3.3 Kinematic solution of SCARA manipulator using quaternion

SCARA manipulator model and coordinate frames attached to it is shown in Figure 4.12. Where θ_1 , θ_2 , d_3 and θ_4 represents the joint variables of revolute and prismatic joints and a_1 , a_2 are links lengths.

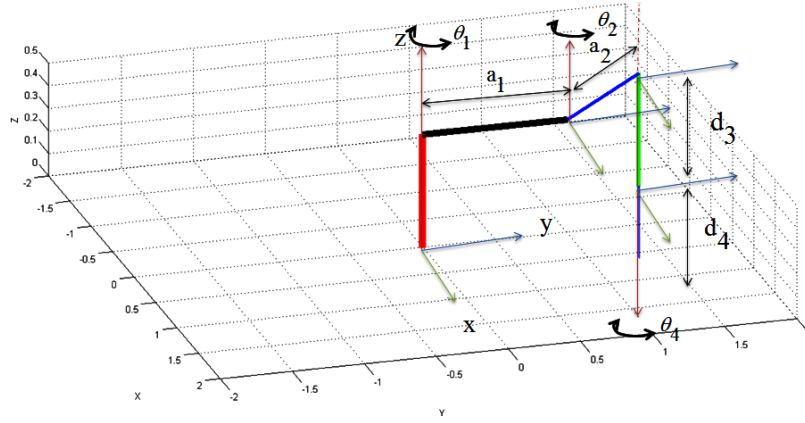


Figure 4.12 SCARA manipulator

Quaternion for each joint variable can be calculated from eqn. (4.99)

$$H_1 = [\langle \overline{C}_1 + \overline{S}_1 \hat{k} \rangle, \langle a_1 C_1 \hat{i} + a_1 S_1 \hat{j} \rangle] \quad (4.10)$$

$$H_2 = [\langle \overline{C}_2 + \overline{S}_2 \hat{k} \rangle, \langle a_2 C_2 \hat{i} + a_2 S_2 \hat{j} \rangle] \quad (4.101)$$

$$H_3 = [\langle 1, 0 \rangle, \langle -d_3 \hat{k} \rangle] \quad (4.102)$$

$$H_4 = [\langle \overline{C}_4 + \overline{S}_4 \hat{k} \rangle, \langle -d_4 \hat{k} \rangle] \quad (4.103)$$

Therefore inverse quaternion for each joint can be calculated by using equation (4.97),

$$H_1^{-1} = [\langle \overline{C}_1 - \overline{S}_1 \hat{k} \rangle, \langle -a_1 \hat{i} \rangle] \quad (4.104)$$

$$H_2^{-1} = [\langle \overline{C}_2 - \overline{S}_2 \hat{k} \rangle, \langle -a_2 \hat{i} \rangle] \quad (4.105)$$

$$H_3^{-1} = [\langle 1, 0 \rangle, \langle d_3 \hat{k} \rangle] \quad (4.106)$$

$$H_4^{-1} = [\langle \overline{C}_4 - \overline{S}_4 \hat{k} \rangle, \langle d_4 \hat{k} \rangle] \quad (4.107)$$

Now calculating quaternion vector products using equation (4.99) and (4.108)

$$Q_i = H_i \otimes H_{i+1} \dots \dots H_n \quad (4.108)$$

Where in case of SCARA, n=4. Therefore from equation (4.108) individual quaternions can be calculated as,

$$Q_4 = H_4 = [\langle \overline{C}_4 + \overline{S}_4 \hat{k} \rangle, \langle -d_4 \hat{k} \rangle] \quad (4.109)$$

$$Q_3 = H_3 \otimes Q_4 = [\langle \overline{C}_4 + \overline{S}_4 \hat{k} \rangle, \langle (-d_3 - d_4) \hat{k} \rangle] \quad (4.110)$$

From equation (4.108), $Q_2 = H_2 \otimes M_3$

Where multiplication of dual quaternion $H_1 \otimes H_2$ can be calculated using equation (4.108)

$$Q_2 = H_2 \otimes Q_3 = [\langle \overline{C}_2 + \overline{S}_2 \hat{k} \rangle, \langle a_2 C_2 \hat{i} + a_2 S_2 \hat{j} \rangle] \otimes [\langle \overline{C}_4 + \overline{S}_4 \hat{k} \rangle, \langle (-d_3 - d_4) \hat{k} \rangle]$$

Therefore,

$$Q_2 = [\langle \overline{C}_{24} + \overline{S}_{24} \hat{k}, \langle a_2 C_2 \hat{i} + a_2 S_2 \hat{j} + (-d_3 - d_4) \hat{k} \rangle] \quad (4.111)$$

Now calculating Q_1 from equation (4.108),

$$Q_1 = H_1 \otimes Q_2 = [\langle \overline{C}_1 + \overline{S}_1 \hat{k}, \langle a_1 C_1 \hat{i} + a_1 S_1 \hat{j} \rangle] \otimes [\langle \overline{C}_{24} + \overline{S}_{24} \hat{k}, \langle a_2 C_2 \hat{i} + a_2 S_2 \hat{j} + (-d_3 - d_4) \hat{k} \rangle]$$

Therefore Q_1 is expressed in equations (4.112),

$$Q_1 = [\langle \overline{C}_{124} + \overline{S}_{124} \hat{k}, \langle (a_1 C_1 + a_2 C_{12}) \hat{i} + (a_1 S_1 + a_2 S_{12}) \hat{j} + (-d_3 - d_4) \hat{k} \rangle] \quad (4.112)$$

Now calculating quaternion vector pairs using equation (4.113)

$$O_{j+1} = H_j^{-1} \otimes O_j \quad (4.113)$$

To solve the inverse kinematics problem, the transformation quaternion of end effector of robot manipulator can be defined as

$$[R_{be}, T_{be}] = O_1 = [\langle w + a \hat{i} + b \hat{j} + c \hat{k}, \langle X \hat{i} + Y \hat{j} + Z \hat{k} \rangle] \quad (4.114)$$

Now using equations (4.113) and (4.114), O_2 will be given by,

$$\begin{aligned} O_2 &= H_1^{-1} \otimes O_1 \\ O_2 &= [\langle \overline{C}_1 - \overline{S}_1 \hat{k}, \langle -a_1 \hat{i} \rangle] \otimes [\langle w + a \hat{i} + b \hat{j} + c \hat{k}, \langle X \hat{i} + Y \hat{j} + Z \hat{k} \rangle] \\ O_2 &= [\langle (w \overline{C}_1 + c \overline{S}_1) + (a \overline{C}_1 + b \overline{S}_1) \hat{i} + (b \overline{C}_1 - a \overline{S}_1) \hat{j} + (c \overline{C}_1 - w \overline{S}_1) \hat{k}, \\ &\quad \langle (X \overline{C}_1 - a_1 + Y \overline{S}_1) \hat{i} + (Y \overline{C}_1 - X \overline{S}_1) \hat{j} + Z \hat{k} \rangle] \end{aligned} \quad (4.115)$$

$$O_3 = H_2^{-1} \otimes O_2$$

$$O_3 = [\langle o_{31} + s_{32} \hat{i} + o_{33} \hat{j} + o_{34} \hat{k}, \langle o_{35} \hat{i} + o_{36} \hat{j} + o_{37} \hat{k} \rangle] \quad (4.116)$$

Where,

$$o_{31} = \overline{C}_2 (w \overline{C}_1 + c \overline{S}_1) + c (\overline{C}_2 \overline{S}_2) - w (\overline{S}_2 \overline{S}_1)$$

$$o_{32} = \overline{C}_2 (a \overline{C}_1 + b \overline{S}_1) + \overline{S}_2 (b \overline{C}_1 - a \overline{S}_1)$$

$$o_{33} = \overline{C}_2 (b \overline{C}_1 - a \overline{S}_1) - \overline{S}_2 (a \overline{C}_1 + b \overline{S}_1)$$

$$o_{34} = \overline{C}_2 (c \overline{C}_1 - w \overline{S}_1) - \overline{S}_2 (w \overline{C}_1 + c \overline{S}_1)$$

$$o_{35} = -Z S_2 + X C_1 C_2 + Y S_1 C_2 - a_1 C_2$$

$$o_{36} = Z S_1 + Y C_1$$

$$o_{37} = X C_1 S_2 - Y S_1 S_2 + a_1 S_2 + Z C_2 + a_2$$

Now,

$$O_4 = H_3^{-1} \otimes O_3$$

$$O_4 = [\langle o_{41} + o_{42} \hat{i} + o_{43} \hat{j} + o_{44} \hat{k}, \langle o_{45} \hat{i} + o_{46} \hat{j} + p_z \hat{k} \rangle] \quad (4.117)$$

Where, $o_{41} = o_{31}, o_{42} = o_{32}, o_{43} = o_{33}, o_{44} = o_{34}, o_{45} = o_{35}, o_{46} = o_{36}$

Now all the joint variables can be calculated by equating quaternion vector products and quaternion vector pairs i.e. Q_1, Q_2 and Q_3 to O_1, O_2 and O_3 respectively.

$$p_x = a_1 C_1 + a_2 C_{12} \quad (4.118)$$

$$p_y = a_1 S_1 + a_2 S_{12} \quad (4.119)$$

Therefore,

$$\theta_2 = \tan^{-1} \frac{\pm 2a_1 a_2 \sqrt{1 - C_2^2}}{p_x^2 + p_y^2 - a_1^2 - a_2^2} \quad (4.120)$$

$$\theta_1 = a \tan\left(\frac{-p_x}{p_y}\right) \pm a \tan\left(\frac{\sqrt{p_x^2 + p_y^2 - (a_2 S_2)^2}}{a_2 S_2}\right) \quad (4.121)$$

$$d_3 = -p_z - d_4 \quad (4.122)$$

We know that there is no translation in fourth joint of SCARA robot it only gives orientation so we can equate the scalar and vector part of quaternion vector product and quaternion vector pair i.e. Q_1, Q_2 and Q_3 to O_1, O_2 and O_3 respectively.

From equations (4.115), (4.116) and (4.117),

$$\bar{C}_4 = \bar{C}_2 w \bar{C}_1 + c \bar{C}_2 \bar{S}_1 + c \bar{S}_2 \bar{C}_1 - w \bar{S}_2 \bar{S}_1$$

$$-\bar{S}_4 = \bar{C}_2 c \bar{C}_1 - w \bar{C}_2 \bar{S}_1 - w \bar{S}_2 \bar{C}_1 - c \bar{S}_2 \bar{S}_1$$

$$\bar{C}_{124} = w$$

$$\bar{S}_{124} = c$$

θ_4 can be given as,

$$\theta_4 = \tan^{-1}\left(\frac{-\bar{C}_{124} \bar{S}_{12} + \bar{S}_{124} \bar{C}_{12}}{\bar{C}_{124} \bar{C}_{12} + \bar{S}_{124} \bar{S}_{12}}\right) \quad (4.123)$$

4.3.4 Kinematic solution of 5-dof revolute manipulator kinematics

The configuration and base coordinate frame attachment of 5-dof revolute manipulator is given in Figure 4.16. Where $\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5 joint angles for articulated arm and d_1, d_2 and d_3 are the link offset. a_1 , and a_2 represents link lengths.

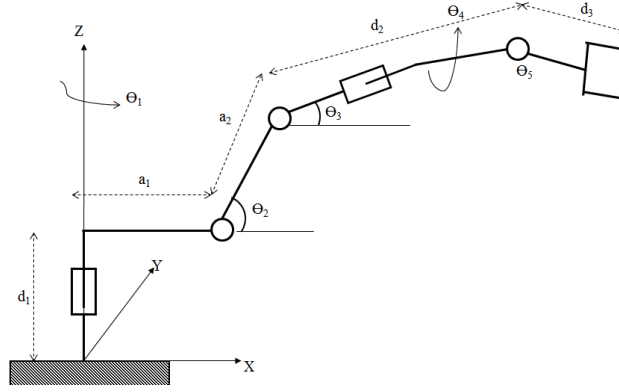


Figure 4.13 Base frame and model of 5-dof revolute manipulator

Now quaternion for successive transformation of each joint can be calculated from the equation (4.99) as follows,

$$H_1 = [\langle \overline{C}_1 + \overline{S}_1 \hat{k} \rangle, \langle a_1 C_1 \hat{i} + a_1 S_1 \hat{j} + d_1 \hat{k} \rangle] \quad (4.124)$$

$$H_2 = [\langle \overline{C}_2 + \overline{S}_2 \hat{j} \rangle, \langle -a_2 S_2 \hat{i} - a_2 C_2 \hat{k} \rangle] \quad (4.125)$$

$$H_3 = [\langle \overline{C}_3 + \overline{S}_3 \hat{j} \rangle, \langle -d_4 S_3 \hat{i} - d_4 C_3 \hat{k} \rangle] \quad (4.126)$$

$$H_4 = [\langle \overline{C}_4 + \overline{S}_4 \hat{i} \rangle, \langle d_4 \hat{i} \rangle] \quad (4.127)$$

$$H_5 = [\langle \overline{C}_5 + \overline{S}_5 \hat{j} \rangle, \langle -d_6 S_5 \hat{i} - d_6 C_5 \hat{k} \rangle] \quad (4.128)$$

Inverse of a dual quaternion can be calculated by equation (4.108),

$$H_1^{-1} = [\langle \overline{C}_1 - \overline{S}_1 \hat{k} \rangle, \langle -a_1 \hat{i} \rangle] \quad (4.129)$$

$$H_2^{-1} = [\langle \overline{C}_2 - \overline{S}_2 \hat{j} \rangle, \langle a_2 \hat{k} \rangle] \quad (4.130)$$

$$H_3^{-1} = [\langle \overline{C}_3 - \overline{S}_3 \hat{j} \rangle, \langle d_4 \hat{k} \rangle] \quad (4.131)$$

$$H_4^{-1} = [\langle \overline{C}_4 - \overline{S}_4 \hat{i} \rangle, \langle -d_4 \hat{i} \rangle] \quad (4.132)$$

$$H_5^{-1} = [\langle \overline{C}_5 - \overline{S}_5 \hat{j} \rangle, \langle d_6 \hat{k} \rangle] \quad (4.133)$$

$$Q_i = H_i \otimes H_{i+1} \dots \dots H_n \quad (4.134)$$

Where in case of 5-dof revolute manipulator arm $n=5$. Now calculating quaternion vector products using equation (4.108)

$$Q_5 = H_5 = [\langle \overline{C}_5 + \overline{S}_5 \hat{j} \rangle, \langle -d_6 S_5 \hat{i} - d_6 C_5 \hat{k} \rangle] \quad (4.135)$$

$$Q_4 = H_4 \otimes Q_5 = [\langle \overline{C}_4 + \overline{S}_4 \hat{i} \rangle, \langle d_4 \hat{i} \rangle] \otimes [\langle \overline{C}_5 + \overline{S}_5 \hat{j} \rangle, \langle -d_6 S_5 \hat{i} - d_6 C_5 \hat{k} \rangle] \quad (4.136)$$

$$Q_4 = [\langle \overline{C}_4 \overline{C}_5 + \overline{S}_4 \overline{C}_5 \hat{i} + \overline{C}_4 \overline{S}_5 \hat{j} + \overline{S}_4 \overline{S}_5 \hat{k} \rangle, \langle (d_4 - d_6 S_5) \hat{i} + d_6 C_5 S_4 \hat{j} - d_6 C_4 C_5 \hat{k} \rangle] \quad (4.137)$$

$$Q_3 = H_3 \otimes Q_4 = [\langle \overline{C_3} + \overline{S_3} \hat{j}, \langle -d_4 S_3 \hat{i} - d_4 C_3 \hat{k} \rangle] \otimes \quad (4.138)$$

$$[\langle \overline{C_4 C_5} + \overline{S_4 C_5} \hat{i} + \overline{C_4 S_5} \hat{j} + \overline{S_4 S_5} \hat{k} \rangle, \langle (d_4 - d_6 S_5) \hat{i} + d_6 C_5 S_4 \hat{j} - d_6 C_4 C_5 \hat{k} \rangle]$$

$$Q_3 = [\langle \overline{C_4 C_{3+5}} + \overline{S_4 C_{3-5}} \hat{i} + \overline{C_4 S_{3+5}} \hat{j} - \overline{S_4 S_{3+5}} \hat{k} \rangle, \quad (4.139)$$

$$\langle (d_4 + d_4 - d_6 C_4 C_5 S_3 - d_4 C_3 - d_6 S_5 C_3 - d_4 S_3) \hat{i} +$$

$$(d_6 C_5 S_4 \hat{j} + (-d_4 S_3 + d_6 S_5 S_3 - d_6 C_4 C_5 C_3 - d_4 \hat{C}_3) \hat{k} \rangle]$$

$$Q_2 = H_2 \otimes Q_3 = [\langle \overline{C_2} + \overline{S_2} \hat{j}, \langle -a_2 S_2 \hat{i} - a_2 C_2 \hat{k} \rangle] \otimes \quad (4.140)$$

$$[\langle \overline{C_4 C_{3+5}} + \overline{S_4 C_{3-5}} \hat{i} + \overline{C_4 S_{3+5}} \hat{j} - \overline{S_4 S_{3+5}} \hat{k} \rangle,$$

$$\langle (d_4 + d_4 - d_6 C_4 C_5 S_3 - d_4 C_3 - d_6 S_5 C_3 - d_4 S_3) \hat{i} +$$

$$(d_6 C_5 S_4 \hat{j} + (-d_4 S_3 + d_6 S_5 S_3 - d_6 C_4 C_5 C_3 - d_4 \hat{C}_3) \hat{k} \rangle]$$

Therefore,

$$Q_2 = [\langle (\overline{C_2 C_4 C_{3+5}} - \overline{S_2 C_4 S_{3+5}}) + (\overline{C_2 S_4 C_{3-5}} - \overline{S_2 S_4 S_{3+5}}) \hat{i} + (\overline{C_2 C_4 S_{3+5}} + \overline{S_2 C_4 C_{3+5}}) \hat{j} + \quad (4.141)$$

$$(\overline{-C_2 S_4 S_{3+5}} - \overline{S_2 S_4 S_{3-5}}) \hat{k} \rangle,$$

$$\langle (-a_2 S_2 + d_4 C_2 + d_4 C_2 - d_4 C_{2-3} + d_6 S_5 C_{2+3} - d_4 S_{2+3} - d_6 C_4 C_5 S_{2+3} - d_4 S_3) \hat{i} +$$

$$(d_6 C_5 S_4 \hat{j} + (-a_2 C_2 - d_4 S_2 - d_4 S_2 + d_6 C_4 C_5 C_{2+3} + d_4 S_{2-3} + d_6 S_5 S_{2+3} + d_4 \hat{C}_{2-3}) \hat{k} \rangle]$$

$$Q_1 = H_1 \otimes Q_2 = [\langle \overline{C_1} + \overline{S_1} \hat{k}, \langle a_1 C_1 \hat{i} + a_1 S_1 \hat{j} + d_1 \hat{k} \rangle] \otimes \quad (4.142)$$

$$[\langle (\overline{C_2 C_4 C_{3+5}} - \overline{S_2 C_4 S_{3+5}}) + (\overline{C_2 S_4 C_{3-5}} - \overline{S_2 S_4 S_{3+5}}) \hat{i}$$

$$+ (\overline{C_2 C_4 S_{3+5}} + \overline{S_2 C_4 C_{3+5}}) \hat{j} + (\overline{-C_2 S_4 S_{3+5}} - \overline{S_2 S_4 S_{3-5}}) \hat{k} \rangle,$$

$$\langle (-a_2 S_2 + d_4 C_2 + d_4 C_2 - d_4 C_{2-3} + d_6 S_5 C_{2+3} - d_4 S_{2+3}$$

$$- d_6 C_4 C_5 S_{2+3} - d_4 S_3) \hat{i} + (d_6 C_5 S_4 \hat{j} +$$

$$(-a_2 C_2 - d_4 S_2 - d_4 S_2 + d_6 C_4 C_5 C_{2+3} + d_4 S_{2-3} + d_6 S_5 S_{2+3} + d_4 \hat{C}_{2-3}) \hat{k} \rangle]$$

Therefore,

$$Q_1 = [\langle (\overline{C_1 C_4 C_{2+3+5}} + \overline{S_1 S_4 S_{2+3-5}}) + (\overline{C_1 S_4 C_{2+3-5}} - \overline{S_1 C_4 S_{2+3+5}}) \hat{i} + \quad (4.143)$$

$$(\overline{C_1 C_4 S_{2+3+5}} + \overline{S_1 S_4 C_{2+3-5}}) \hat{j} +$$

$$(\overline{S_1 C_4 C_{2+3+5}} - \overline{C_1 S_4 S_{2+3-5}}) \hat{k} \rangle,$$

$$\langle (a_1 C_1 - a_2 S_2 + d_4 C_2 C_1 + d_4 C_2 C_1 - d_4 C_{2-3} C_1 + d_6 S_5 C_{2+3} C_1 - d_4 S_{2+3} C_1$$

$$+ d_6 C_4 C_5 S_{2+3} C_1 - d_6 C_5 S_4 S_1) \hat{i} +$$

$$(d_6 C_5 S_4 + d_6 C_5 S_4 C_1 - a_2 S_2 + a_2 S_2 C_1 - a_2 S_2 S_1 + d_4 C_2 S_1 + d_4 C_2 S_1 -$$

$$d_4 C_{2-3} S_1 + d_6 S_5 C_{2+3} S_1 - d_4 S_{2+3} S_1 - d_6 C_4 C_5 S_{2+3} S_1) \hat{j} +$$

$$(d_1 - a_2 C_2 - d_4 S_2 - d_4 S_2 + d_6 C_4 C_5 C_{2+3} + d_4 S_{2-3}$$

$$+ d_6 S_5 S_{2+3} + d_4 \hat{C}_{2-3}) \hat{k} \rangle]$$

Now calculating vector pair of quaternion using equation (4.144), to solve the inverse kinematics problem, the transformation quaternion of end effector of robot manipulator can be defined as

$$[\mathbf{R}_{bc}, \mathbf{T}_{bc}] = \mathbf{O}_1 = [\langle w + a\hat{i} + b\hat{j} + c\hat{k} \rangle, \langle X\hat{i} + Y\hat{j} + Z\hat{k} \rangle] \quad (4.144)$$

Now using equations (4.108) and (4.144), \mathbf{O}_2 will be given by,

$$\mathbf{O}_2 = \mathbf{H}_1^{-1} \otimes \mathbf{O}_1$$

$$\mathbf{O}_2 = [\langle \overline{C}_1 - \overline{S}_1 \hat{k} \rangle, \langle -a_1 \hat{i} \rangle] \otimes [\langle w + a\hat{i} + b\hat{j} + c\hat{k} \rangle, \langle X\hat{i} + Y\hat{j} + Z\hat{k} \rangle]$$

$$\begin{aligned} \mathbf{O}_2 = & [\langle (o_{21}) + (o_{22})\hat{i} + (o_{23})\hat{j} + (o_{24})\hat{k} \rangle, \\ & \langle (o_{25})\hat{i} + (o_{26})\hat{j} + o_{27}\hat{k} \rangle] \end{aligned} \quad (4.145)$$

where, $o_{21} = w\overline{C}_1 + c\overline{S}_1$

$$o_{22} = a\overline{C}_1 + b\overline{S}_1$$

$$o_{23} = b\overline{C}_1 - a\overline{S}_1$$

$$o_{24} = c\overline{C}_1 - w\overline{S}_1$$

$$o_{25} = X\overline{C}_1 - a_1 + Y\overline{S}_1$$

$$o_{26} = Y\overline{C}_1 - X\overline{S}_1$$

$$o_{27} = Z$$

Now,

$$\mathbf{O}_3 = \mathbf{H}_2^{-1} \otimes \mathbf{O}_2$$

$$\mathbf{O}_3 = [\langle o_{31} + o_{32}\hat{i} + o_{33}\hat{j} + o_{34}\hat{k} \rangle, \langle o_{35}\hat{i} + o_{36}\hat{j} + p_z\hat{k} \rangle] \quad (4.146)$$

Where,

$$o_{31} = \overline{C}_2 o_{21} + c(\overline{C}_2 \overline{S}_2) - w(\overline{S}_2 \overline{S}_1)$$

$$o_{32} = \overline{C}_2 o_{22} + \overline{S}_2 o_{23}$$

$$o_{33} = \overline{C}_2 o_{23} - \overline{S}_2 o_{22}$$

$$o_{34} = \overline{C}_2 o_{24} - \overline{S}_2 o_{21}$$

$$o_{35} = -ZS_2 + XC_1 C_2 + YS_1 C_2 - a_1 C_2$$

$$o_{36} = YC_1 - XS_1$$

$$o_{37} = a_2 - ZC_2 + XC_1 S_2 + YS_1 S_2 - a_1 S_2$$

$$\mathbf{O}_4 = [\langle o_{41} + o_{42}\hat{i} + o_{43}\hat{j} + o_{44}\hat{k} \rangle, \langle o_{45}\hat{i} + o_{47}\hat{j} + o_{47}\hat{k} \rangle] \quad (4.147)$$

$$o_{41} = \overline{C}_{2+3} o_{21} + \overline{S}_{2+3} o_{23}$$

$$o_{42} = \overline{C}_{2+3} o_{22} - \overline{S}_{2+3} o_{24}$$

$$o_{43} = \overline{C}_{2+3} o_{23} - \overline{S}_{2+3} o_{21}$$

$$o_{44} = \overline{C}_{2+3} o_{24} - \overline{S}_{2+3} o_{22}$$

$$o_{45} = -ZS_2 - a_2S_3 - XC_1S_2S_3 - YS_1S_2S_3 + a_1S_2S_3 - \\ ZC_2S_3 + Z - ZC_3 + XC_1C_2C_3 + XS_1C_2C_3 - a_1C_2C_3$$

$$o_{46} = YC_1 - XS_1$$

$$o_{47} = -ZS_2S_3 + a_2C_3 + XC_1C_2S_3 + YS_1C_2S_3 - \\ a_1C_2S_3 + ZC_2C_3 + XC_1S_2C_3 + YS_1S_2C_3 - a_2S_2C_3$$

Therefore, all the joint variables can be calculated by equating quaternion vector products and quaternion vector pairs i.e. Q_1 , Q_2 and Q_3 to O_1 , O_2 and O_3 respectively.

$$a_1C_1 - a_2S_2 + d_4C_2C_1 + d_4C_2C_1 - d_4C_{2-3}C_1 + \\ d_6S_5C_{2+3}C_1 - d_4S_{2+3}C_1 + d_6C_4C_5S_{2+3}C_1 - d_6C_5S_4S_1 = X \quad (4.148)$$

Form equation (4.148),

$$u_x = -a_2S_2 - d_6C_5S_4S_1 \quad (4.149)$$

$$a_1C_1 + d_4C_2C_1 + d_4C_2C_1 - d_4C_{2-3}C_1 + d_6S_5C_{2+3}C_1 - d_4S_{2+3}C_1 + d_6C_4C_5S_{2+3}C_1 + u_x = X$$

$$a_1C_1 + d_4C_2C_1 + d_4C_2C_1 - d_4C_{2-3}C_1 + d_6S_5C_{2+3}C_1 - d_4S_{2+3}C_1 + d_6C_4C_5S_{2+3}C_1 = X - u_x$$

$$C_1(a_1 + d_4C_2 + d_4C_2 - d_4C_{2-3} + d_6S_5C_{2+3} - d_4S_{2+3} + d_6C_4C_5S_{2+3}) = X - u_x$$

$$C_1 = \frac{X - u_x}{(a_1 + d_4C_2 + d_4C_2 - d_4C_{2-3} + d_6S_5C_{2+3} - d_4S_{2+3} + d_6C_4C_5S_{2+3})} \quad (4.150)$$

And

$$d_6C_5S_4 + d_6C_5S_4C_1 - a_2S_2 + a_2S_2C_1 - a_2S_2S_1 + \\ d_4C_2S_1 + d_4C_2S_1 - d_4C_{2-3}S_1 + d_6S_5C_{2+3}S_1 - d_4S_{2+3}S_1 \\ - d_6C_4C_5S_{2+3}S_1 = Y \quad (4.151)$$

From equation (4.151)

$$u_y = d_6C_5S_4 + d_6C_5S_4C_1 - a_2S_2 + a_2S_2C_1$$

$$- a_2S_2S_1 + d_4C_2S_1 + d_4C_2S_1 - d_4C_{2-3}S_1 + d_6S_5C_{2+3}S_1 - d_4S_{2+3}S_1 - d_6C_4C_5S_{2+3}S_1 + u_y = Y$$

$$S_1(-a_2S_2 + d_4C_2 + d_4C_2 - d_4C_{2-3} + d_6S_5C_{2+3} - d_4S_{2+3} - d_6C_4C_5S_{2+3}) = Y - u_y$$

$$S_1 = \frac{Y - u_y}{(-a_2S_2 + d_4C_2 + d_4C_2 - d_4C_{2-3} + d_6S_5C_{2+3} - d_4S_{2+3} - d_6C_4C_5S_{2+3})} \quad (4.152)$$

From equations (4.150) and (4.152)

$$\Rightarrow \tan \theta_1 = \frac{Y - u_y}{\frac{(-a_2S_2 + d_4C_2 + d_4C_2 - d_4C_{2-3} + d_6S_5C_{2+3} - d_4S_{2+3} - d_6C_4C_5S_{2+3})}{(a_1 + d_4C_2 + d_4C_2 - d_4C_{2-3} + d_6S_5C_{2+3} - d_4S_{2+3} + d_6C_4C_5S_{2+3})}}$$

$$\Rightarrow \tan \theta_1 = \frac{Y - u_y}{(a_1 + d_4C_2 + d_4C_2 - d_4C_{2-3} + d_6S_5C_{2+3} - d_4S_{2+3} + d_6C_4C_5S_{2+3})} \times \\ \frac{(-a_2S_2 + d_4C_2 + d_4C_2 - d_4C_{2-3} + d_6S_5C_{2+3} - d_4S_{2+3} - d_6C_4C_5S_{2+3})}{X - u_x}$$

$$\Rightarrow \theta_1 = a \tan \left[\frac{Y - u_y}{X - u_x} \times \frac{(a_1 + d_4 C_2 + d_4 C_2 - d_4 C_{2-3} + d_6 S_5 C_{2+3} - d_4 S_{2+3} + d_6 C_4 C_5 S_{2+3})}{(-a_2 S_2 + d_4 C_2 + d_4 C_2 - d_4 C_{2-3} + d_6 S_5 C_{2+3} - d_4 S_{2+3} - d_6 C_4 C_5 S_{2+3})} \right] \quad (4.153)$$

Now for theta 2

$$d_1 - a_2 C_2 - d_4 S_2 - d_4 S_2 + d_6 C_4 C_5 C_{2+3} + d_4 S_{2-3} + d_6 S_5 S_{2+3} + d_4 C_{2-3} = Z \quad (4.154)$$

$$v_x = d_1 - a_2 C_2 + d_6 C_4 C_5 C_{2+3} + d_4 S_{2-3} + d_6 S_5 S_{2+3} + d_4 C_{2-3} \quad (4.155)$$

$$d_4 S_2 + d_4 S_2 - v_x = Z \quad (4.156)$$

$$d_4 S_2 + d_4 S_2 = Z + v_x \quad (4.157)$$

$$S_2 = \left(\frac{Z + v_x}{d_4} \right) \quad (4.158)$$

As we know that

$$\sin \theta = a \Rightarrow a \tan 2 \left[a, \mu \sqrt{1 - a^2} \right]$$

Therefore using equations (4.154)-(4.158)

$$\theta_2 = a \tan 2 \left[\left(\frac{Z + v_x}{d_4} \right), \mu \sqrt{1 - \left\{ \left(\frac{Z + v_x}{d_4} \right) \right\}^2} \right] \quad (4.159)$$

Similarly,

$$-ZS_2 + XC_1 C_2 + YS_1 C_2 - a_1 C_2 = d_4 + d_4 - d_6 C_4 C_5 S_3 - d_4 C_3 - d_6 S_5 C_3 - d_4 S_3 \quad (4.160)$$

$$v_y = d_4 + d_4 - d_6 C_4 C_5 S_3 - d_4 S_3 \quad (4.161)$$

$$-ZS_2 + XC_1 C_2 + YS_1 C_2 - a_1 C_2 = -d_4 C_3 - d_6 S_5 C_3 + v_y \quad (4.162)$$

$$-ZS_2 + XC_1 C_2 + YS_1 C_2 - a_1 C_2 - v_y = -d_4 C_3 - d_6 S_5 C_3 \quad (4.163)$$

$$-ZS_2 + XC_1 C_2 + YS_1 C_2 - a_1 C_2 - v_y = (-d_4 - d_6 S_5) C_3 \quad (4.164)$$

$$\left(\frac{-ZS_2 + XC_1 C_2 + YS_1 C_2 - a_1 C_2 - v_y}{(-d_4 - d_6 S_5)} \right) = C_3 \quad (4.165)$$

Therefor theta 3 using equations (4.160)-(4.165),

$$\theta_3 = a \tan 2 \left[\mu \sqrt{1 - \left\{ \left(\frac{-XS_2 + XC_1 C_2 + YS_1 C_2 - a_1 C_2 - v_y}{(-d_4 - d_6 S_5)} \right) \right\}^2}, \left(\frac{-ZS_2 + XC_1 C_2 + YS_1 C_2 - a_1 C_2 - v_y}{(-d_4 - d_6 S_5)} \right) \right] \quad (4.166)$$

Similarly for theta4 and theta 5

$$d_4 - d_6 S_5 = -ZS_2 - a_2 S_3 - XC_1 S_2 S_3 - YS_1 S_2 S_3 + a_1 S_2 S_3 - ZC_2 S_3 + Z - ZC_3 + XC_1 C_2 C_3 + YS_1 C_2 C_3 - a_1 C_2 C_3$$

$$\begin{aligned}
-d_6 S_5 &= -ZS_2 - a_2 S_3 - XC_1 S_2 S_3 - YS_1 S_2 S_3 + a_1 S_2 S_3 - ZC_2 S_3 + Z - ZC_3 + XC_1 C_2 C_3 + \\
&YS_1 C_2 C_3 - a_1 C_2 C_3 - d_4 \\
S_5 &= \left(\frac{-ZS_2 - a_2 S_3 - XC_1 S_2 S_3 - YS_1 S_2 S_3 + a_1 S_2 S_3 - ZC_2 S_3 + Z - ZC_3 + XC_1 C_2 C_3 + YS_1 C_2 C_3 - a_1 C_2 C_3 - d_4}{-d_6} \right)
\end{aligned} \tag{4.167}$$

Therefore from equation (4.167) theta5 will be

$$\theta_5 = a \tan 2 \left[\begin{array}{l} \left(\frac{-XS_2 - a_2 S_3 - XC_1 S_2 S_3 - YS_1 S_2 S_3 + a_1 S_2 S_3 - ZC_2 S_3 + Z - ZC_3 + XC_1 C_2 C_3 + YS_1 C_2 C_3 - a_1 C_2 C_3 - d_4}{-d_6} \right), \\ \mu \sqrt{1 - \left\{ \left(\frac{-ZS_2 - a_2 S_3 - XC_1 S_2 S_3 - YS_1 S_2 S_3 + a_1 S_2 S_3 - ZC_2 S_3 + Z - ZC_3 + XC_1 C_2 C_3 + YS_1 C_2 C_3 - a_1 C_2 C_3 - d_4}{-d_6} \right)^2 \right\}} \end{array} \right] \tag{4.168}$$

$$\begin{aligned}
-d_6 C_4 C_5 &= -ZS_2 S_3 + a_2 C_3 + XC_1 C_2 S_3 + YS_1 C_2 S_3 - \\
&a_1 C_2 S_3 + ZC_2 C_3 + XC_1 S_2 C_3 + YS_1 S_2 C_3 - a_2 S_2 C_3 \\
C_4 &= \left(\frac{-ZS_2 S_3 + a_2 C_3 + XC_1 C_2 S_3 + YS_1 C_2 S_3 - a_1 C_2 S_3 + ZC_2 C_3 + XC_1 S_2 C_3 + YS_1 S_2 C_3 - a_2 S_2 C_3}{-d_6 C_5} \right)
\end{aligned} \tag{4.169}$$

Theta 4 will be given by using equation (4.169),

$$\theta_4 = a \tan 2 \left[\begin{array}{l} \mu \sqrt{1 - \left\{ \left(\frac{-ZS_2 S_3 + a_2 C_3 + XC_1 C_2 S_3 + YS_1 C_2 S_3 - a_1 C_2 S_3 + ZC_2 C_3 + XC_1 S_2 C_3 + YS_1 S_2 C_3 - a_2 S_2 C_3}{-d_6 C_5} \right)^2 \right\}} \\ \left(\frac{-ZS_2 S_3 + a_2 C_3 + XC_1 C_2 S_3 + YS_1 C_2 S_3 - a_1 C_2 S_3 + ZC_2 C_3 + XC_1 S_2 C_3 + YS_1 S_2 C_3 - a_2 S_2 C_3}{-d_6 C_5} \right) \end{array} \right] \tag{4.170}$$

4.3.5 Kinematic solution of PUMA 560 manipulator using quaternion

PUMA 560 manipulator model and coordinate frames attached to it is shown in Figure 4.15. Where $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5,$ and θ_6 represents the joint variables of revolute type joints and a_2, a_3 are links lengths and $d_2, d_3,$ and d_4 are link offsets.

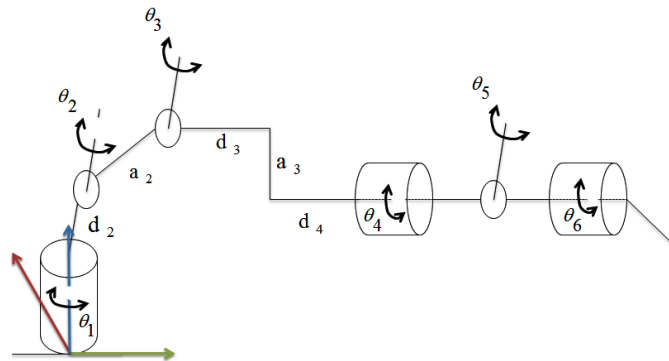


Figure 4.14 PUMA 560 manipulator model

Quaternion of each joint variables can be calculated using equation (4.99), that is similar process like SCARA.

$$H_1 = [\langle \bar{C}_1 + \bar{S}_1 \hat{k} \rangle, \langle d_2 C_1 \hat{i} + d_2 S_1 \hat{j} \rangle] \quad (4.171)$$

$$H_2 = [\langle \bar{C}_2 + \bar{S}_2 \hat{j} \rangle, \langle -a_2 S_2 \hat{i} - a_2 C_2 \hat{k} \rangle] \quad (4.172)$$

$$H_3 = [\langle \bar{C}_3 + \bar{S}_3 \hat{j} \rangle, \langle -a_3 S_3 \hat{i} - d_3 \hat{j} - a_3 C_3 \hat{k} \rangle] \quad (4.173)$$

$$H_4 = [\langle \bar{C}_4 + \bar{S}_4 \hat{i} \rangle, \langle d_4 \hat{i} \rangle] \quad (4.174)$$

$$H_5 = [\langle \bar{C}_5 + \bar{S}_5 \hat{j} \rangle, \langle 0,0,0 \rangle] \quad (4.175)$$

$$H_6 = [\langle \bar{C}_6 + \bar{S}_6 \hat{i} \rangle, \langle 0,0,0 \rangle] \quad (4.176)$$

Now calculating inverse of quaternion using equation (4.108),

$$H_1^{-1} = [\langle \bar{C}_1 - \bar{S}_1 \hat{k} \rangle, \langle -d_2 \hat{i} \rangle] \quad (4.177)$$

$$H_2^{-1} = [\langle \bar{C}_2 - \bar{S}_2 \hat{j} \rangle, \langle a_2 \hat{k} \rangle] \quad (4.178)$$

$$H_3^{-1} = [\langle \bar{C}_3 - \bar{S}_3 \hat{j} \rangle, \langle d_3 \hat{j} \rangle] \quad (4.179)$$

$$H_4^{-1} = [\langle \bar{C}_4 - \bar{S}_4 \hat{i} \rangle, \langle -d_4 \hat{i} \rangle] \quad (4.181)$$

$$H_5^{-1} = [\langle \bar{C}_5 - \bar{S}_5 \hat{j} \rangle, \langle 0,0,0 \rangle] \quad (4.182)$$

$$H_6^{-1} = [\langle \bar{C}_6 - \bar{S}_6 \hat{i} \rangle, \langle 0,0,0 \rangle] \quad (4.183)$$

Now calculating quaternion vector products using equation (4.99) and (4.108)

$$Q_i = H_i \otimes H_{i+1} \dots H_n$$

$$Q_6 = H_6 = [\langle \bar{C}_2 + \bar{S}_2 \hat{j} \rangle, \langle -a_2 S_2 \hat{i} - a_2 C_2 \hat{k} \rangle] \quad (4.184)$$

$$Q_5 = H_5 \otimes Q_6 = [\langle \bar{C}_5 \bar{C}_6 + \bar{C}_5 \bar{S}_6 \hat{i} + \bar{S}_5 \bar{C}_6 \hat{j} + \bar{S}_5 \bar{S}_6 \hat{k} \rangle] \quad (4.185)$$

$$Q_4 = H_4 \otimes Q_5 = [\langle \bar{C}_5 \bar{C}_{4+6} + \bar{C}_5 \bar{S}_{4+6} \hat{i} + \bar{S}_5 \bar{C}_{4+6} \hat{j} + \bar{S}_5 \bar{S}_{4+6} \hat{k} \rangle, \langle d_4 \hat{i} \rangle] \quad (4.186)$$

$$Q_3 = H_3 \otimes Q_4 = [\langle (\bar{C}_3 \bar{C}_5 \bar{C}_{4+6} - \bar{S}_3 \bar{S}_5 \bar{C}_{4+6}) + (\bar{C}_3 \bar{C}_5 \bar{S}_{4+6} + \bar{S}_3 \bar{S}_5 \bar{S}_{4+6}) \hat{i} + (\bar{C}_3 \bar{S}_5 \bar{C}_{4+6} + \bar{S}_3 \bar{C}_5 \bar{C}_{4+6}) \hat{j} + (\bar{C}_3 \bar{S}_5 \bar{S}_{4+6} - \bar{S}_3 \bar{C}_5 \bar{S}_{4+6}) \hat{k} \rangle, \langle (d_4 C_3 - a_3 S_3) \hat{i} - d_3 \hat{j} - (d_4 S_3 + a_3 C_3) \hat{k} \rangle] \quad (4.187)$$

$$Q_2 = H_2 \otimes Q_3 = [\langle A_2 + B_2 \hat{i} + C_2 \hat{j} + D_2 \hat{k} \rangle, \langle E_2 \hat{i} - F_2 \hat{j} - G_2 \hat{k} \rangle] \quad (4.188)$$

Where,

$$A_2 = (\bar{C}_2\bar{C}_3\bar{C}_5\bar{C}_{4+6} - \bar{C}_2\bar{S}_3\bar{S}_5\bar{C}_{4+6} - \bar{S}_2\bar{C}_3\bar{S}_5\bar{C}_{4+6} - \bar{S}_2\bar{S}_3\bar{S}_5\bar{C}_{4+6})$$

$$B_2 = (\bar{C}_2\bar{C}_3\bar{C}_5\bar{S}_{4+6} + \bar{C}_2\bar{S}_3\bar{S}_5\bar{S}_{4+6} + \bar{S}_2\bar{C}_3\bar{S}_5\bar{S}_{4+6} - \bar{S}_2\bar{S}_3\bar{C}_5\bar{S}_{4+6})\hat{i}$$

$$C_2 = (\bar{C}_2\bar{C}_3\bar{S}_5\bar{C}_{4+6} + \bar{C}_2\bar{S}_3\bar{C}_5\bar{C}_{4+6} + \bar{S}_2\bar{C}_3\bar{C}_5\bar{C}_{4+6} - \bar{S}_2\bar{S}_3\bar{S}_5\bar{C}_{4+6})\hat{j}$$

$$D_2 = (\bar{C}_2\bar{C}_3\bar{S}_5\bar{S}_{4+6} - \bar{C}_2\bar{S}_3\bar{C}_5\bar{S}_{4+6} - \bar{S}_2\bar{C}_3\bar{C}_5\bar{S}_{4+6} - \bar{S}_2\bar{S}_3\bar{S}_5\bar{S}_{4+6})\hat{k}$$

$$E_2 = (-a_2S_2 - d_4C_{23} - a_3S_{23})\hat{i}$$

$$F_2 = -d_3\hat{j}$$

$$G_2 = (-a_2C_2 - d_4S_{23} + a_3C_{23})\hat{k}$$

Now,

$$Q_1 = H_1 \otimes Q_2 = [< A^* + B^*\hat{i} + C^*\hat{j} + D^*\hat{k} >, < E^*\hat{i} - F^*\hat{j} - G^*\hat{k} >] \quad (4.189)$$

Where,

$$A^* = \bar{C}_1A_2 - \bar{S}_1D_2$$

$$B^* = \bar{C}_1B_2 - \bar{S}_1C_2$$

$$C^* = \bar{C}_1C_2 + \bar{S}_1B_2$$

$$D^* = \bar{C}_1D_2 + \bar{S}_1A_2$$

$$E^* = (d_2C_1 + d_3S_1 - a_2S_2C_1 - d_4C_{2+3}C_1 - a_3S_{2+3}C_1)\hat{i}$$

$$F^* = (d_2S_1 - d_3C_1 - a_2S_2S_1 - d_4C_{2+3}S_1 - a_3S_{2+3}S_1)\hat{j}$$

$$K^* = (-a_2C_2 + a_3C_{2+3} - d_4S_{2+3})\hat{k}$$

Now calculating vector pair of quaternion using equation (4.190), to solve the inverse kinematics problem, the transformation quaternion of end effector of robot manipulator can be defined as

$$[R_{be}, T_{be}] = O_1 = [< w + a\hat{i} + b\hat{j} + c\hat{k} >, < X\hat{i} + Y\hat{j} + Z\hat{k} >] \quad (4.190)$$

Now using equations (4.108) and (4.190), O_2 will be given by,

$$O_2 = H_1^{-1} \otimes O_1$$

$$O_2 = [< \bar{C}_1 - \bar{S}_1\hat{k} >, < -a_1\hat{i} >] \otimes [< w + a\hat{i} + b\hat{j} + c\hat{k} >, < X\hat{i} + Y\hat{j} + Z\hat{k} >]$$

$$\begin{aligned} O_2 = [< (o_{21}) + (o_{22})\hat{i} + (o_{23})\hat{j} + (o_{24})\hat{k} >, \\ < (X\bar{C}_1 - a_1 + Y\bar{S}_1)\hat{i} + (Y\bar{C}_1 - X\bar{S}_1)\hat{j} + Z\hat{k} >] \end{aligned} \quad (4.191)$$

where, $o_{21} = w\bar{C}_1 + c\bar{S}_1$

$$o_{22} = a\overline{C_1} + b\overline{S_1}$$

$$o_{23} = b\overline{C_1} - a\overline{S_1}$$

$$o_{24} = c\overline{C_1} - w\overline{S_1}$$

Now,

$$O_3 = H_2^{-1} \otimes O_2$$

$$O_3 = [\langle o_{31} + o_{32} \hat{i} + o_{33} \hat{j} + o_{34} \hat{k} \rangle, \langle o_{35} \hat{i} + o_{36} \hat{j} + p_z \hat{k} \rangle] \quad (4.192)$$

Where,

$$o_{31} = \overline{C_2}o_{21} + c(\overline{C_2}\overline{S_2}) - w(\overline{S_2}\overline{S_1})$$

$$o_{32} = \overline{C_2}o_{22} + \overline{S_2}o_{23}$$

$$o_{33} = \overline{C_2}o_{23} - \overline{S_2}o_{22}$$

$$o_{34} = \overline{C_2}o_{24} - \overline{S_2}o_{21}$$

$$o_{35} = XC_{12} - a_2 + YS_{12} - a_1C_2$$

$$o_{36} = YC_{12} - XS_{12} + a_1S_2$$

Therefore, all the joint variables can be calculated by equating quaternion vector products and quaternion vector pairs i.e. Q_1 , Q_2 and Q_3 to O_1 , O_2 and O_3 respectively.

$$\theta_1 = a \tan 2 \left(\frac{-X \pm \sqrt{X^2 + Y^2 - d_3^2}}{d_3 + Y} \right) \quad (4.193)$$

$$\theta_2 = a \tan 2 \left(\frac{-a_2 - a_3C_3 + d_4S_3 \pm \sqrt{a_2^2 + a_3^2 + d_4^2 + 2a_3(a_3C_3 - d_4S_3) - Z_2^2}}{Z - (a_3S_3 + d_4C_3)} \right) \quad (4.194)$$

$$\theta_3 = a \tan 2 \left(\frac{-d_4 \pm \sqrt{a_3^2 + d_4^2 - k_2^2}}{k + a_3} \right) \quad (4.195)$$

Where,

$$k = \frac{X_2^2 + Y_3^2 + Z_2^2 - d_3^2 - a_2^2 - a_3^2 - d_4^2}{2a_2}$$

Similarly for θ_4 , θ_5 and θ_6 ,

$$O_4 = [\langle o_{41} + o_{42} \hat{i} + o_{43} \hat{j} + o_{44} \hat{k} \rangle, \langle o_{45} \hat{i} + o_{47} \hat{j} + o_{47} \hat{k} \rangle] \quad (4.196)$$

$$o_{41} = \overline{C_{2+3}}o_{21} + \overline{S_{2+3}}o_{23}$$

$$o_{42} = \overline{C_{2+3}}o_{22} - \overline{S_{2+3}}o_{24}$$

$$o_{43} = \overline{C_{2+3}}o_{23} - \overline{S_{2+3}}o_{21}$$

$$o_{44} = \overline{C}_{2+3}o_{24} - \overline{S}_{2+3}o_{22}$$

$$o_{45} = \overline{C}_{2+3}o_{35} - \overline{S}_{2+3}o_{37}$$

$$o_{47} = \overline{C}_{2+3}o_{37} + \overline{S}_{2+3}o_{35}$$

$$\theta_4 = a \tan 2\left(\frac{o_{44}}{o_{41}}\right) - a \tan 2\left(\frac{\pm o_{42}}{\pm o_{43}}\right) \quad (4.197)$$

$$\theta_5 = a \tan 2\left(\frac{\pm \sqrt{o_{42}^2 + o_{43}^2}}{+ \sqrt{o_{41}^2 + o_{44}^2}}\right) \quad (4.198)$$

$$\theta_6 = a \tan 2\left(\frac{o_{44}}{o_{41}}\right) + a \tan 2\left(\frac{\pm o_{42}}{\pm o_{43}}\right) \quad (4.199)$$

4.3.6 Kinematic solution of ABB IRb-1400 manipulator using quaternion

The base coordinate frames and configuration of the 6-dof ABB IRB-1400 robot manipulator is presented in Figure 4.15. Where θ_i represents the joint variables of revolute type joints and a_i are links lengths and d_i are link offsets. The base frame is fixed rotation is fixed for all joint rotations.

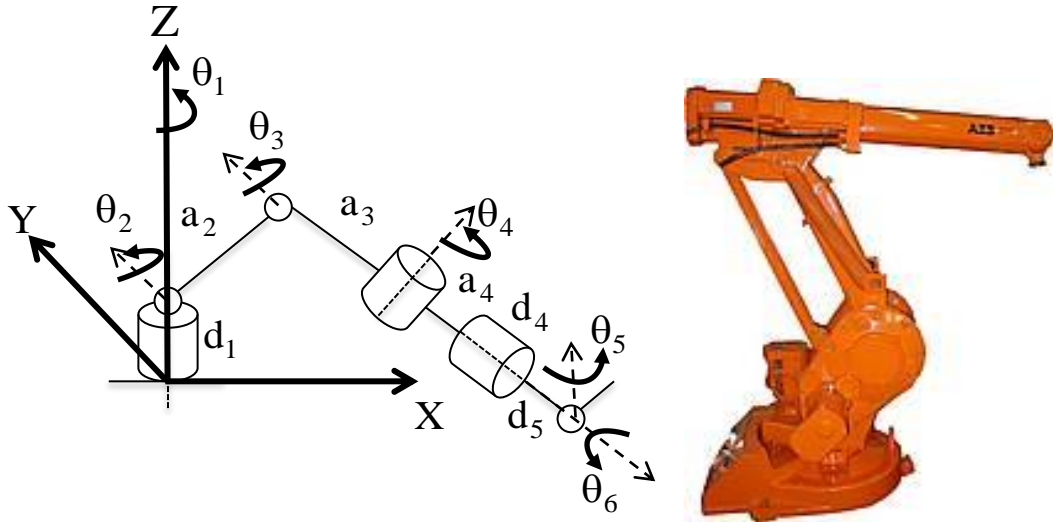


Figure 4.15 Configuration and model of ABB IRB-1440 robot manipulator

The quaternion vector of each joint can be calculated by equation 4.98 and 4.99.

$$H_1 = [\langle \overline{C}_1 + \overline{S}_1 \hat{k} \rangle, \langle a_2 C_1 \hat{i} + a_2 S_1 \hat{j} + d_1 \hat{k} \rangle] \quad (4.200)$$

$$H_2 = [\langle \overline{C}_2 + \overline{S}_2 \hat{j} \rangle, \langle a_3 C_2 \hat{i} + a_3 S_2 \hat{k} \rangle] \quad (4.201)$$

$$H_3 = [\langle \overline{C}_3 + \overline{S}_3 \hat{j} \rangle, \langle -a_4 S_3 \hat{i} - a_4 C_3 \hat{k} \rangle] \quad (4.202)$$

$$H_4 = [\langle \overline{C}_4 - \overline{S}_4 \hat{k} \rangle, \langle -d_4 \hat{k} \rangle] \quad (4.203)$$

$$H_5 = [\langle \overline{C}_5 - \overline{S}_5 \hat{j}, \langle -d_5 \hat{k} \rangle] \quad (4.204)$$

$$H_6 = [\langle \overline{C}_6 - \overline{S}_6 \hat{k}, \langle 0,0,0 \rangle] \quad (4.205)$$

Inverse of a dual quaternion can be calculated by equation (4.97),

$$H_1^{-1} = [\langle \overline{C}_1 - \overline{S}_1 \hat{k}, \langle -a_1 \hat{i} - d_1 \hat{k} \rangle] \quad (4.206)$$

$$H_2^{-1} = [\langle \overline{C}_2 - \overline{S}_2 \hat{j}, \langle -a_2 \hat{k} \rangle] \quad (4.207)$$

$$H_3^{-1} = [\langle \overline{C}_3 - \overline{S}_3 \hat{j}, \langle -a_3 \hat{k} \rangle] \quad (4.208)$$

$$H_4^{-1} = [\langle \overline{C}_4 + \overline{S}_4 \hat{k}, \langle 0,0,d_4 \hat{k} \rangle] \quad (4.209)$$

$$H_5^{-1} = [\langle \overline{C}_5 + \overline{S}_5 \hat{j}, \langle 0,0,d_5 \hat{k} \rangle] \quad (4.210)$$

$$H_6^{-1} = [\langle \overline{C}_6 + \overline{S}_6 \hat{k}, \langle 0,0,0 \rangle] \quad (4.211)$$

Similar to previous work, end effector position can be formulated as,

$$X = a_1 c_1 - a_2 c_1 s_2 - a_3 c_1 s_{23} + d_4 c_1 c_{23} + ((-c_1 s_{23} c_4 + s_1 s_4) s_5 + c_1 c_{23} c_5) d_6 \quad (4.212)$$

$$Y = a_1 s_1 - a_2 s_1 s_2 - a_3 s_1 s_{23} + d_4 s_1 c_{23} + ((-s_1 s_{23} c_4 + c_1 s_4) s_5 + s_1 c_{23} c_5) d_6 \quad (4.213)$$

$$Z = d_1 + a_2 c_2 + a_3 c_{23} + d_4 s_{23} + d_6 (c_{23} c_4 s_5 + s_{23} c_5) \quad (4.214)$$

Similar to the kinematic solution of PUMA manipulator, forward kinematics can be calculated for STAUBLI RX 160L. Therefore, all the joint variables can be calculated by equating quaternion vector products and quaternion vector pairs i.e. Q_1 , Q_2 and Q_3 to O_1 , O_2 and O_3 respectively.

$$\theta_1 = A \tan 2(Y, X) \quad (4.215)$$

if $\theta_2 \geq \pi - \theta_2$, then,

$$\theta_1 = \pi + A \tan 2(Y, X) \quad (4.216)$$

$$\theta_2 = A \tan 2(S_2, C_2) \quad (4.217)$$

where,

$$s_2 = \frac{-(a_2 + (c_{23} c_4 c_5 - s_{23} s_5) c_6 - c_{23} s_4 s_6) s_3} {a_2^2 + ((c_{23} c_4 c_5 - s_{23} s_5) c_6 - c_{23} s_4 s_6)^2} + 2a_2 (c_{23} c_4 c_5 - s_{23} s_5) c_6 - c_{23} s_4 s_6) s_3$$

$$c_2 = \frac{(a_2 + (c_{23} c_4 c_5 - s_{23} s_5) c_6 - c_{23} s_4 s_6) s_3} {a_2^2 + ((c_{23} c_4 c_5 - s_{23} s_5) c_6 - c_{23} s_4 s_6)^2} + 2a_2 (c_{23} c_4 c_5 - s_{23} s_5) c_6 - c_{23} s_4 s_6) s_3$$

$$A = a_2 c_2 + (c_{23} c_4 c_5 - s_{23} s_5) c_6 - c_{23} s_4 s_6) c_{23}$$

$$B = a_2 s_2 + (c_{23} c_4 c_5 - s_{23} s_5) c_6 - c_{23} s_4 s_6) s_{23}$$

$$\theta_3 = A \tan 2(S'_3, C'_3) - A \tan 2(a_3 / d_4) \quad (4.218)$$

where,

$$S_3' = \frac{A^2 + B^2 - a_2^2 - a_x^2}{2a_2(c_{23}c_4c_5 - s_{23}s_5)c_6 - c_{23}s_4s_6}$$

$$C_3' = \pm\sqrt{1 - s_3'^2}$$

$$\theta_4 = A \tan 2(c_{23}c_4s_5 + s_{23}c_5, (c_{23}c_4c_5 - s_{23}s_5)c_6 - c_{23}s_4s_6)) \quad (4.219)$$

$$\theta_5 = A \tan 2(\sqrt{(c_{23}c_4c_5 - s_{23}s_5)c_6 - c_{23}s_4s_6})^2 + (c_{23}c_4s_5 + s_{23}c_5)^2}, \\ -(-c_{23}c_4c_5 + s_{23}s_5)s_6 - c_{23}s_4c_6) \quad (4.220)$$

$$\theta_6 = A \tan 2(-((s_1s_{23}c_4 + c_1s_4)c_5 + s_1c_{23}s_5)s_6 + (s_1s_{23}s_4 - c_1c_4)c_6, \\ ((-s_1s_{23}c_4 - c_1s_4)c_5 - s_1c_{23}s_5)c_6 + (s_1s_{23}s_4 - c_1c_4)s_6) \quad (4.221)$$

4.3.7 Kinematic solution of STAUBLI RX160L manipulator using quaternion

The coordinate frames and configuration of the 6-dof STAUBLI RX160L robot manipulator is presented in Figure 4.16. Where θ_i represents the joint variables of revolute type joints and a_i are links lengths and d_i are link offsets. The base frame is fixed rotation is fixed for all joint rotations.

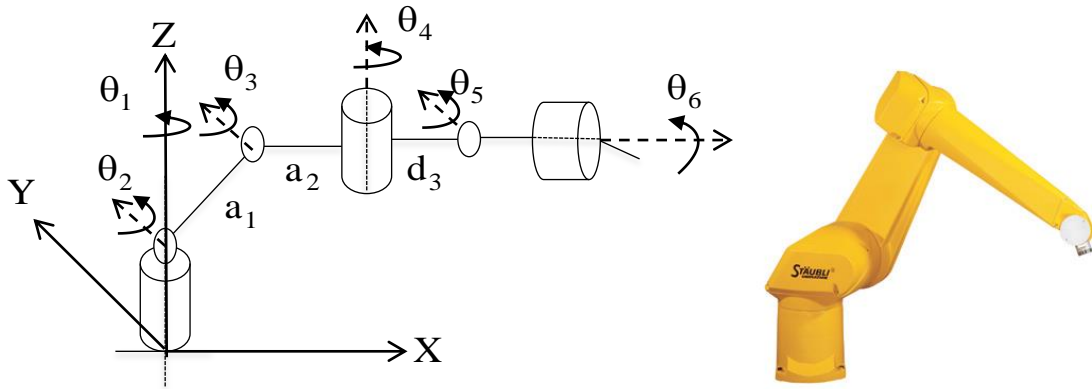


Figure 4.16 Coordinate frame and model of STAUBLI RX160L robot manipulator

The quaternion vector of each joint can be calculated by equation 4.98 and 4.99.

$$H_1 = [\langle \overline{C}_1 + \overline{S}_1 \hat{k} \rangle, \langle a_1 C_1 \hat{i} + a_1 S_1 \hat{j} \rangle] \quad (4.222)$$

$$H_2 = [\langle \overline{C}_2 + \overline{S}_2 \hat{j} \rangle, \langle a_2 C_2 \hat{i} + a_2 S_2 \hat{k} \rangle] \quad (4.223)$$

$$H_3 = [\langle \overline{C}_3 + \overline{S}_3 \hat{j} \rangle, \langle -d_3 S_3 \hat{i} - d_3 C_3 \hat{k} \rangle] \quad (4.224)$$

$$H_4 = [\langle \overline{C}_4 - \overline{S}_4 \hat{k} \rangle, \langle 0, 0, 0 \rangle] \quad (4.225)$$

$$H_5 = [\langle \overline{C_5} - \overline{S_5} \hat{j} \rangle, \langle 0,0,0 \rangle] \quad (4.226)$$

$$H_6 = [\langle \overline{C_6} - \overline{S_6} \hat{k} \rangle, \langle 0,0,0 \rangle] \quad (4.227)$$

Inverse of a dual quaternion can be calculated by equation (4.98),

$$H_1^{-1} = [\langle \overline{C_1} - \overline{S_1} \hat{k} \rangle, \langle -a_1 \hat{i} \rangle] \quad (4.228)$$

$$H_2^{-1} = [\langle \overline{C_2} - \overline{S_2} \hat{j} \rangle, \langle -a_2 \hat{k} \rangle] \quad (4.229)$$

$$H_3^{-1} = [\langle \overline{C_3} - \overline{S_3} \hat{j} \rangle, \langle -d_3 \hat{k} \rangle] \quad (4.230)$$

$$H_4^{-1} = [\langle \overline{C_4} + \overline{S_4} \hat{k} \rangle, \langle 0,0,0 \rangle] \quad (4.231)$$

$$H_5^{-1} = [\langle \overline{C_5} + \overline{S_5} \hat{j} \rangle, \langle 0,0,0 \rangle] \quad (4.232)$$

$$H_6^{-1} = [\langle \overline{C_6} + \overline{S_6} \hat{k} \rangle, \langle 0,0,0 \rangle] \quad (4.233)$$

Similar to the kinematic solution of PUMA manipulator, forward kinematics can be calculated for STAUBLI RX 160L. Therefore, all the joint variables can be calculated by equating quaternion vector products and quaternion vector pairs i.e. Q_1 , Q_2 and Q_3 to O_1 , O_2 and O_3 respectively.

$$\theta_1 = a \tan 2 \left(\frac{-X \pm \sqrt{X^2 + Y^2 - d_3^2}}{d_3 + Y} \right) \quad (4.234)$$

$$\theta_2 = a \tan 2 \left(\frac{-a_1 - a_2 C_3 + d_3 S_3 \pm \sqrt{a_1^2 + a_2^2 + d_3^2 + 2a_2(a_2 C_3 - d_3 S_3) - Z^2}}{Z - (a_2 S_3 + d_3 C_3)} \right) \quad (4.235)$$

$$\theta_3 = a \tan 2 \left(\frac{-d_3 \pm \sqrt{a_2^2 + d_3^2 - k^2}}{k + a_2} \right) \quad (4.236)$$

where,

$$k = \frac{X_2^2 + Y_3^2 + Z_2^2 - a_1^2 - a_2^2 - d_3^2}{2a_1}$$

similarly for θ_4 , θ_5 and θ_6 ,

$$O_4 = [\langle o_{41} + o_{42} \hat{i} + o_{43} \hat{j} + o_{44} \hat{k} \rangle, \langle o_{45} \hat{i} + o_{47} \hat{j} + o_{47} \hat{k} \rangle] \quad (4.237)$$

$$o_{41} = \overline{C_{2+3}} o_{21} + \overline{S_{2+3}} o_{23}$$

$$o_{42} = \overline{C_{2+3}} o_{22} - \overline{S_{2+3}} o_{24}$$

$$o_{43} = \overline{C_{2+3}} o_{23} - \overline{S_{2+3}} o_{21}$$

$$o_{44} = \overline{C_{2+3}} o_{24} - \overline{S_{2+3}} o_{22}$$

$$o_{45} = \overline{C}_{2+3}o_{35} - \overline{S}_{2+3}o_{37}$$

$$o_{47} = \overline{C}_{2+3}o_{37} + \overline{S}_{2+3}o_{35}$$

$$\theta_4 = a \tan 2 \left(-\frac{o_{44}}{o_{41}} \right) + a \tan 2 \left(\frac{o_{42}}{o_{43}} \right) \quad (4.238)$$

$$\theta_5 = a \tan 2 \left(\pm \sqrt{o_{42}^2 + o_{43}^2}, \pm \sqrt{o_{41}^2 + o_{44}^2} \right) \quad (4.239)$$

$$\theta_6 = a \tan 2 \left(-\frac{o_{44}}{o_{41}} \right) - a \tan 2 \left(\frac{o_{42}}{o_{43}} \right) \quad (4.240)$$

4.3.8 Kinematic solution of ASEA IRb-6 manipulator using quaternion

The base coordinate frames and configuration of the 5-dof ASEA IRb-6 robot manipulator is presented in Figure 4.17. Where θ_i represents the joint variables of revolute type joints and a_i are links lengths and d_i are link offsets. The base frame is fixed rotation is fixed for all joint rotations.

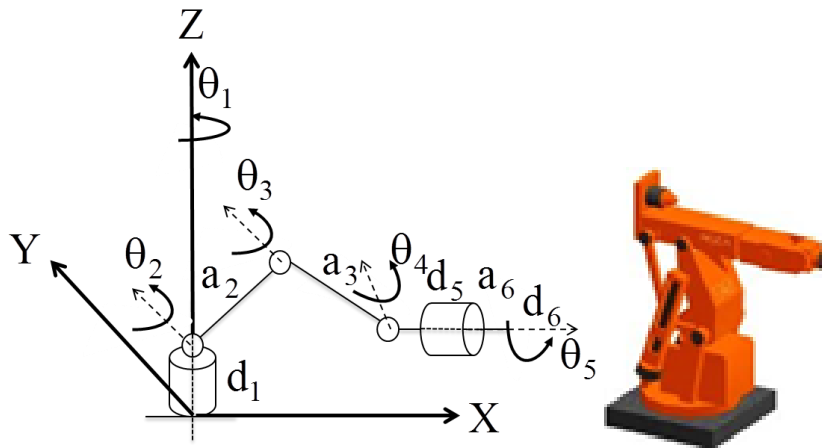


Figure 4.17 Coordinate frame and model of ASEA IRb-6 robot manipulator

The quaternion vector of each joint can be calculated by equation 4.98 and 4.99.

$$H_1 = [\langle \overline{C}_1 + \overline{S}_1 \hat{k} \rangle, \langle a_2 C_1 \hat{i} + a_2 S_1 \hat{j} + d_1 \hat{k} \rangle] \quad (4.241)$$

$$H_2 = [\langle \overline{C}_2 + \overline{S}_2 \hat{j} \rangle, \langle a_3 C_2 \hat{i} + a_3 S_2 \hat{k} \rangle] \quad (4.242)$$

$$H_3 = [\langle \overline{C}_3 + \overline{S}_3 \hat{j} \rangle, \langle -d_5 S_3 \hat{i} - d_5 C_3 \hat{k} \rangle] \quad (4.243)$$

$$H_4 = [\langle \overline{C}_4 - \overline{S}_4 \hat{j} \rangle, \langle 0,0,0 \rangle] \quad (4.244)$$

$$H_5 = [\langle \overline{C}_5 - \overline{S}_5 \hat{i} \rangle, \langle 0,0,0 \rangle] \quad (4.245)$$

Inverse of a dual quaternion can be calculated by equation (4.108),

$$H_1^{-1} = [\langle \overline{C}_1 - \overline{S}_1 \hat{k} \rangle, \langle -a_2 \hat{i} - d_1 \hat{k} \rangle] \quad (4.246)$$

$$H_2^{-1} = [\langle \overline{C}_2 - \overline{S}_2 \hat{j} \rangle, \langle -a_2 \hat{k} \rangle] \quad (4.247)$$

$$H_3^{-1} = [\langle \overline{C}_3 - \overline{S}_3 \hat{j} \rangle, \langle -d_5 \hat{k} \rangle] \quad (4.248)$$

$$H_4^{-1} = [\langle \overline{C}_4 + \overline{S}_4 \hat{k} \rangle, \langle 0,0,0 \rangle] \quad (4.249)$$

$$H_5^{-1} = [\langle \overline{C}_5 + \overline{S}_5 \hat{j} \rangle, \langle 0,0,0 \rangle] \quad (4.250)$$

Quaternion vector products can be calculated by using equation (4.99) and (4.108). Q_1 , Q_2 and Q_3 can be calculated using equation above, therefore forward kinematic equation can be given as,

$$X = a_2 S_1 S_2 - a_3 S_1 C_{23} - d_5 S_1 C_{234} + a_6 (S_1 S_{234} C_5 + C_1 S_5) - d_6 S_1 C_{234} \quad (4.251)$$

$$Y = -a_2 C_1 S_2 + a_3 C_1 C_{23} + d_5 C_1 C_{234} + a_6 (-C_1 S_{234} C_5 + S_1 S_5) + d_6 C_1 C_{234} \quad (4.252)$$

$$Z = d_1 + a_2 C_2 + a_3 S_{23} + d_5 S_{234} + a_6 C_{234} C_5 + d_6 S_{234} \quad (4.253)$$

Similar to previous work inverse kinematics can be derived using the equations and equating Q_1 , Q_2 and Q_3 to O_1 , O_2 and O_3 respectively.

$$\theta_1 = a \tan 2 \left(\frac{-X}{Y} \right) \quad (4.254)$$

$$\theta_2 = a \tan 2 \left(\frac{S_2}{C_2} \right) \quad (4.255)$$

$$\text{Where, } S_2 = \frac{Ba_3 C_3 - A(a_3 S_3 + a_2)}{a_3^2 C_3^2 + (a_3 S_3 + a_2)^2}, \quad C_2 = \frac{Aa_3 C_3 + B(a_3 S_3 + a_2)}{a_3^2 C_3^2 + (a_3 S_3 + a_2)^2}$$

$$A = -S_1 X + C_1 Y + d_5 S_1 C_\phi S_\phi - d_5 C_1 S_\phi S_\phi$$

$$B = Z - d_1 - d_5 C_1 C_\phi$$

$$\theta_3 = a \tan 2 \left(\frac{S_3}{C_3} \right) \quad (4.256)$$

$$\text{where, } S_3 = \frac{A^2 + B^2 - (a_2^2 + a_3^2)}{2a_2 a_3}, \quad C_3 = \sqrt{1 - S_3^2}$$

$$C = a \tan 2 \left(\frac{S_1 S_2 X - C_1 S_2 Y + C_2 Z - d_1 C_2 - a_2 - a_3 S_3}{-S_1 C_2 X + C_1 C_2 Y + S_2 Z - d_1 S_2 - a_3 C_3} \right)$$

$$\theta_4 = C - \theta_3 \quad (4.257)$$

$$\theta_5 = a \tan 2 \left(\frac{C_1 (C_\phi C_\alpha C_\beta - S_\phi S_\beta) + S_1 (-S_\phi C_\alpha C_\beta + C_\phi S_\beta)}{C_1 (-C_\phi C_\alpha S_\beta - S_\phi C_\beta) + S_1 (-S_\phi C_\alpha S_\beta + C_\phi C_\beta)} \right) \quad (4.258)$$

where, ϕ , α and β are the orientation of the end effector.

4.4 Summary

This chapter delivers the basis of conventional methods for modelling the different configurations of robot manipulator in terms of kinematics. The chief purpose of this chapter is to provide brief discussion of DH-algorithm and homogeneous matrix method for representation of rotation and translation of manipulator link. In the later section quaternion application for forward and inverse kinematic solution has been given to show the efficiency and easiness of the method. Therefore, inverse kinematic solution of 4-dof (Adept One SCARA), 5-dof (Parm2) , 5-dof (ASEA IRb-6), 6-dof (PUMA 560), 6-dof (ABB IRB-1400) and 6-dof (STAUBLI RX 160 L) revolute manipulators without Euler wrist are solved mathematically using quaternion vector based method. The adopted method is compact and efficient tool for representation of transformations of end effector. The detailed derivation of inverse kinematic solution has been provided to show the mathematical complexity of homogeneous matrix based solution of robot manipulator over quaternion. In chapter 7 inverse kinematic solutions for adopted manipulator has been tabulated and comparison on the basis of mathematical complexity is made over other conventional based method.

Chapter 5

INTELLIGENT TECHNIQUES FOR INVERSE KINEMATIC SOLUTION

5.1 Overview

Cognitive process of learning and using it for decision making in case of hard to understand processes has been well appreciated by the community researchers. Now a days, human beings are grasping the intelligence from the nature and are trying to implement into the machine. The purpose is to retrieve end effector position of a robot manipulator, which can work in uncertain and cluttered environment on the basis of knowledge or information so as to learn complex nonlinear functions from outside information without the use of mathematical structures or any geometry. The intelligent methods mimic the cognition and consciousness in many aspects like they can learn from the experience or previous training then it can be universalize to that domain for testing, basic concept is the mapping of input output variables faster than conventional methods so as to reduce the computational cost. So the motivation is to reduce the computational cost and consequently increase the speed for robust control. On the other hand, inverse kinematic mapping for any configuration of robot manipulator can be analytically done but the process will be long and slow for real time control.

As explained in previous chapter the inverse kinematic solution of robot manipulator is difficult if following the conventional methods. The difficulty arises due to fact that inverse kinematic equations are not true function and gives multiple solutions. In addition, input-output mapping of inverse kinematics problem is non-linear and tendency of the solution is qualitatively differs when end effector position changes within the workspace. On the other hand, conventional methods yields efficient solution of inverse kinematics but suffer some drawbacks like complex structure of manipulator or higher dof can be time consuming and mathematically difficult to obtain results,

singularities occurs in some cases etc. Therefore, considering overall complexity of inverse kinematic solution and search for efficient intelligent techniques like artificial neural network (ANN), fuzzy logic, ANFIS and hybrid neural network will be fruitful. ANNs are extensively adopted technique to solve inverse kinematics problem and generally offers an alternative approach to handle complex, NP-hard and ill-conditioned problems. ANN models can acquire previous knowledge or information from examples and are able to tackle noisy and inadequate data and to learn non-linear problems. Once the adopted neural network models are trained then it can perform prediction of output with higher computational speed. These models are appropriate in modelling and implementation of system with complex mappings. A detail introduction of different adopted models of ANN has been presented in this chapter.

However, ANN is quite adaptive to the system and does not requires higher level of programing but apart from this it has some drawback like selection of ANN architecture, numerical computation for weight updating (i.e. Gradient descent learning, Levenberg-Marquardt based back propagation learning etc.), etc. In contrast above discussed nature of ANN models, it is required to set some rules for fuzzy logic to avail the advantages of interpretability and transparency of the method. Fuzzy logic requires the prior knowledge of the problem and based on the experience of expert decision that makes use of linguistic information on the basis of hit and trial method. Therefore, from last decades, fuzzy logic becomes an alternative method over conventional techniques for nonlinear inverse kinematic solutions. The main idea behind this algorithm is if-then logic which is inherent to expert decision. However, this algorithm is based on trial and error logic therefore it can be fruitfully merged with ANN models. Fuzzy logic has different membership function which is fixed and might be arbitrarily. And the shape of the function relies on few parameters and this can be optimized using ANN back propagation rule. This method is known as adaptive neural-fuzzy inference system (ANFIS). Therefore, hybridization of ANN with fuzzy can give benefits of both method. However, the major drawback of ANFIS is stuck in local optimum point. Therefore to overcome this problem the wise decision is to adopt some metaheuristic algorithm for the optimization of weight and bias of ANN models. Therefore, in this chapter hybrid ANN models are developed to overcome the problem of ANN and ANFIS with the hybridization strategy. Detail discussion of ANN models, ANFIS and hybrid ANN has been presented in the later section.

5.2 Application of ANN models

ANN models like MLP, PPN, Pi-NN etc. generally used to learn joint angles of robot manipulator and the data sets are generally generated through some conventional

methods like DH-algorithm, homogeneous transformation matrix, algebraic methods etc. Forward kinematic equations are mostly used to train the neural network models whereas in this chapter both forward and inverse kinematics equations are used to train the neural network models. The method of learning is based on the standard data which generally rely on the workspace of the manipulator. The learning can be completed by supervised, unsupervised or both. ANN monitors the input-output relationship between Cartesian coordinate and joint variables based on the mapping of data. Inverse kinematics is a transformation of a world coordinate frame (X, Y, and Z) to a link coordinate frame ($\theta_1, \theta_2, \dots, \theta_n$). This transformation can be performed on input/output work that uses an unknown transfer function. A simple strategy for input-output mapping is shown in Figure 5.1 (a) and (b) which are feed forward and back propagation for error minimization strategy of ANN models.

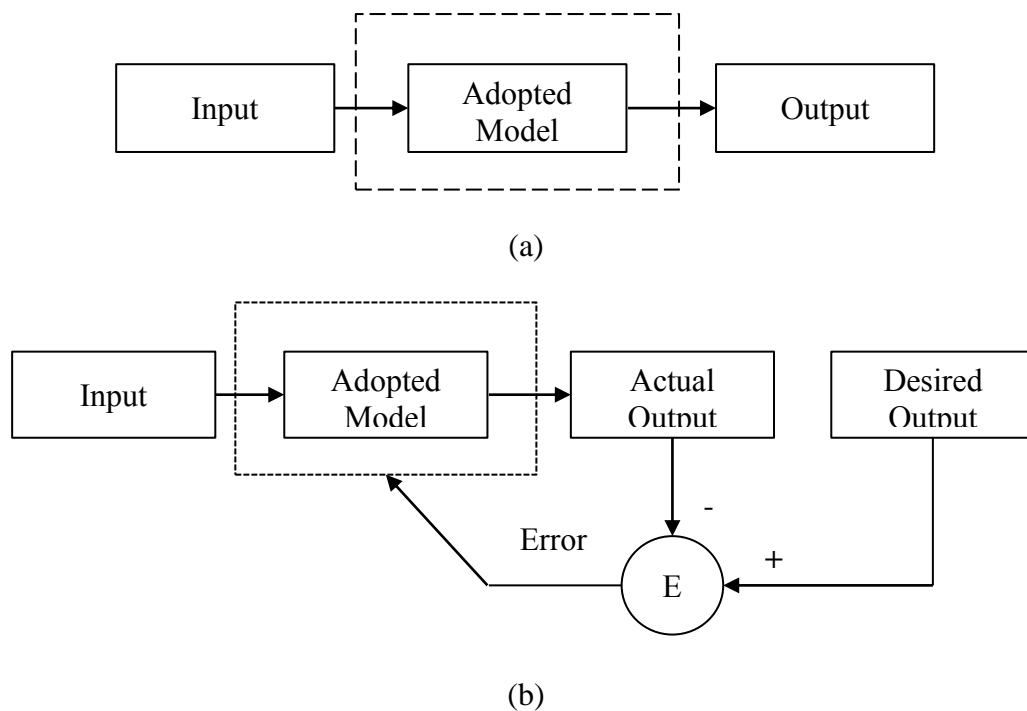


Figure 5.1 (a) Feed forward and (b) back propagation strategy

In chapter multi-layered neural network, polynomial pre-processor neural network and Pi-neural network models are presented. Brief discussions of these adopted models are given in the next section.

5.2.1 Multi-layered perceptron neural network (MLP)

It is well known that neural networks have the better ability than other techniques to solve various complex problems. MLP neural network's neuron is a simple work element, and has a local memory. A neuron takes a multi-dimensional input, and then

delivers it to the other neurons according to their weights. This gives a scalar result at the output of a neuron. The transfer function of an MLP, acting on the local memory, uses a learning rule to produce a relationship between the input and output. For the activation input, a time function is needed.

We propose the solution using a multi-layered perceptron with back-propagation algorithm for training. The network is then trained with data for a number of end effector positions expressed in Cartesian co-ordinates and the corresponding joint angles. The data consist of the different configurations available for the arm. The different poses of the arm are then used to train a three-layer, fully connected back-propagation model (Figure 5.2). This result in two sets of weights for each manipulator arm after the training session was over. A block diagram of the proposed work is shown in Figure 5.2. The signals, o_{jn} , are presented to a hidden layer neuron in the network via the input neurons. Each of the signals from the input neurons is multiplied by the value of the weights of the connection, w_{ij} , between the respective input neurons and the hidden neuron.

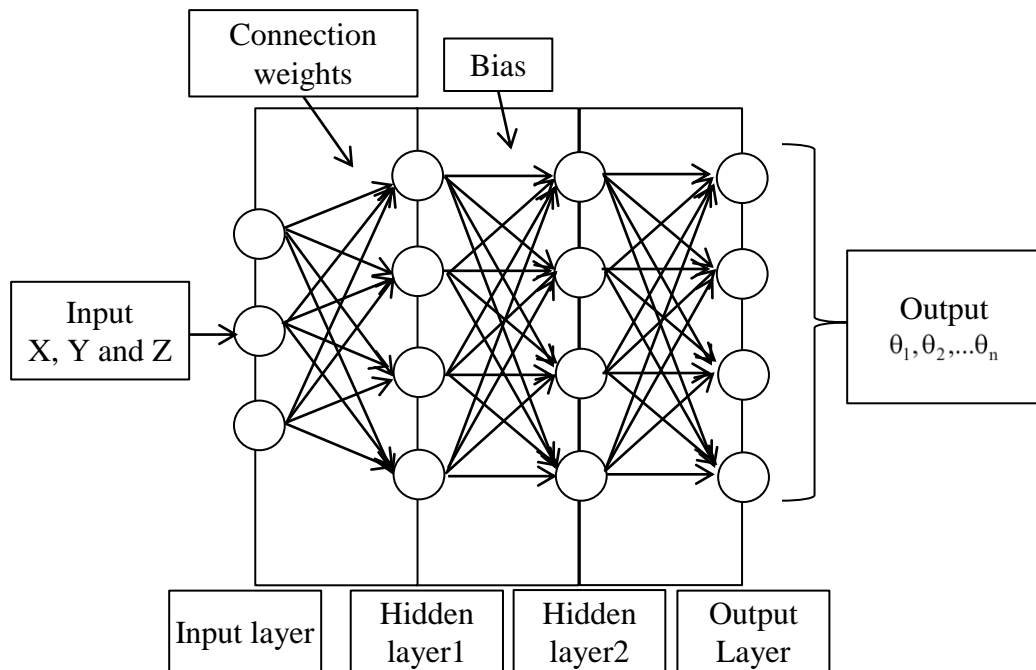


Figure 5.2 Multi-layered perceptron neural network structure

A neural network is a massively parallel-distributed processor as shown in Figure 5.2 that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the human brain in two respects; the knowledge is acquired by the network through a learning process, and interneuron connection strengths known as synaptic weights are used to store the knowledge [247].

Training is the process of modifying the connection weights in some orderly fashion using a suitable learning method. The network uses a learning mode, in which an input is presented to the network along with the desired output and the weights are adjusted so that the network attempts to produce the desired output. Weights after training contain meaningful information whereas before training they are random and have no meaning [247]. Therefore flow chart of MLP neural network is presented in Figure 5.3 and basic steps are as follows:

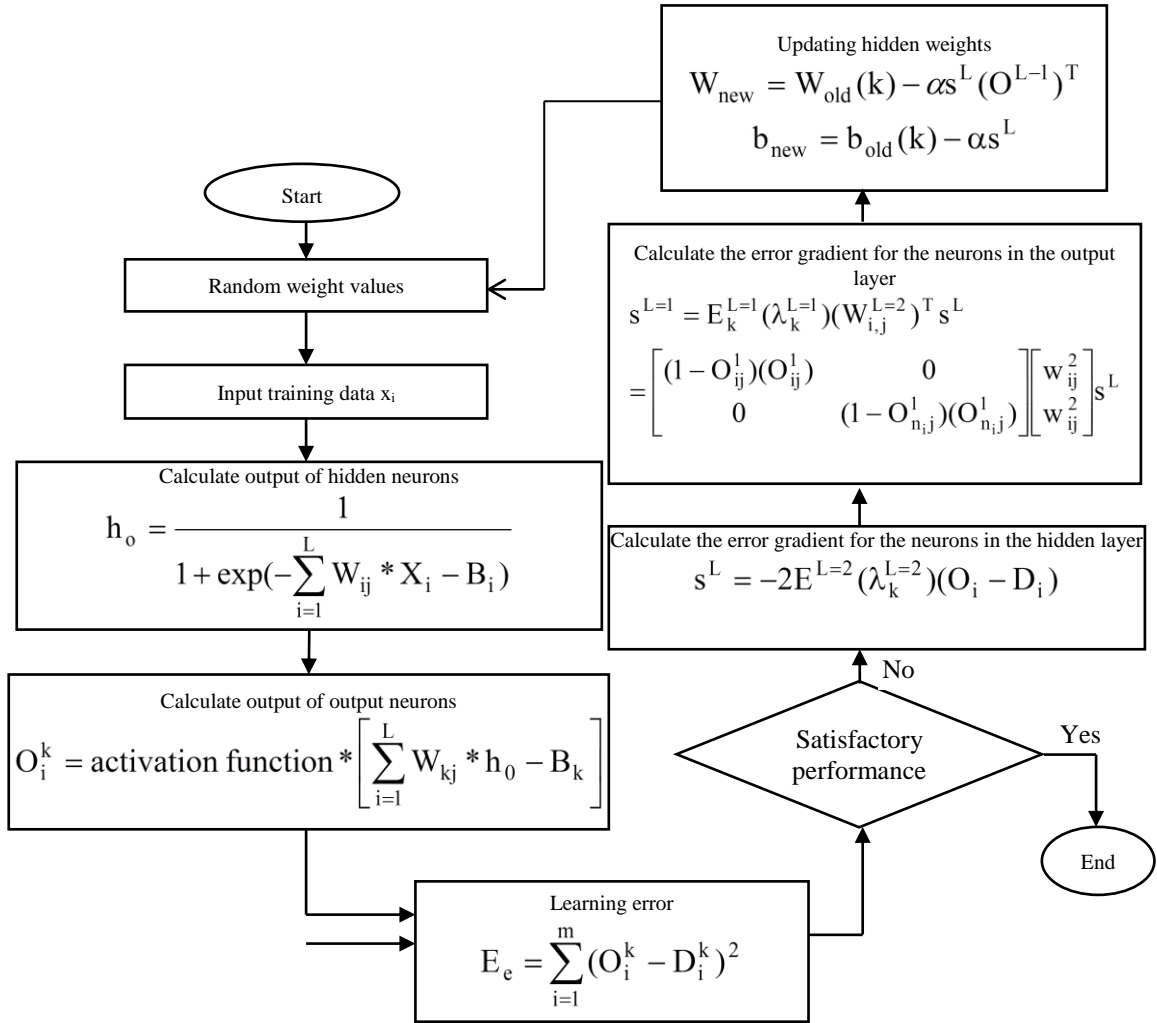


Figure 5.3 Flow chart for MLPBP

- Step 1** Selection of hidden layers (L) and total number of hidden neurons (n_e , $e=1,2,3,\dots,L-1$) with error tolerance 0.
- Step 2** Selection of weight vectors (W_{ij}^L) on the basis of random number generator, where $i=1,2,\dots,n_e$ and $L=2$.
- Step 3** Initialization of weights W_{ij}^L with random number in between 0-1 and given as, $W_{ij}^L \Leftrightarrow [0,1]$

Step 4 Calculation of output of neurons as hidden input, $n_o = \sum_{i=1}^L W_{ij} * X_i + B_i$,

$j=1,2,\dots,L$. Calculation of output of hidden neurons as,

$$h_o = \frac{1}{1 + \exp(-\sum_{i=1}^L W_{ij} * X_i - B_i)}$$
 and output of output neurons can be given by,

$$O_i^k = \text{activation function} * \left[\sum_{i=1}^L W_{kj} * h_o - B_k \right], k=1,2,\dots,m.$$

Step 5 Error estimation of output layer neurons as, $E_k = \sum_{i=1}^k (O_i^k - D_i^k)^2$.

Step 6 If the output of neuron is similar to desired output then end else choose next step

Step 7 Gradient calculation of hidden and output neurons can be given as

$$\lambda_k = E_k * O_i^k * (1 - O_i^k) \text{ and } \lambda_i^h = h_o * (1 - h_o) * \sum_{k=1}^L (\lambda_k * W_{ki})$$

Step 8 Sensitivity of hidden and output layers will be given by, for output layer=
 $s^L = -2E^{L=2}(\lambda_k^{L=2})(O_i - D_i)$ and

$$s^{L=1} = E_k^{L=1}(\lambda_k^{L=1})(W_{i,j}^{L=2})^T s^L = \begin{bmatrix} (1 - O_{ij}^1)(O_{ij}^1) & 0 \\ 0 & (1 - O_{n_i,j}^1)(O_{n_i,j}^1) \end{bmatrix} \begin{bmatrix} w_{ij}^2 \\ w_{ij}^2 \end{bmatrix} s^L$$

Step 9 Updating of weight, $W_{\text{new}} = W_{\text{old}}(k) - \alpha s^L (O^{L-1})^T$ and $b_{\text{new}} = b_{\text{old}}(k) - \alpha s^L$
 where $L=1,2,\dots,L-1$

Step 10 Evaluation of termination criteria if error \leq termination (Ψ) then go to step 11 else step 3.

Step 11 Network is available for testing.

The network uses a learning mode, in which an input is presented to the network along with the desired output and the weights are adjusted so that the network attempts to produce the desired output. Weights after training contain meaningful information whereas before training they are random and have no meaning.

Net input of hidden neurons (for L inputs) =

$$n_o = \sum_{i=1}^L W_{ij} * X_i + B_i \quad (5.1)$$

The output, n_o of a hidden neuron as a function of its net input is described in equation (5.1). The sigmoid function is:

$$h_o = \frac{1}{1 + \exp(-\sum_{i=1}^L W_{ij} * X_i - B_i)} \quad (5.2)$$

Once the outputs of the hidden layer neurons have been calculated, the net input to each output layer is calculated in a similar manner as in equation (5.2). After calculation of

output of output neurons comparison between desired value and network output is made on the basis of mean square error as given in equation (5.3),

$$E_e = \sum_{i=1}^m (O_i^k - D_i^k)^2 \quad (5.3)$$

If the obtained mean square error is zero then algorithm stops otherwise it goes to the error gradient calculation of hidden neuron using the formula as show in equation (5.4),

$$s^L = -2E^{L=2} (\lambda_k^{L=2})(O_i - D_i) \quad (5.4)$$

Further error gradient calculation of output layer can be given as,

$$\begin{aligned} s^{L=1} &= E_k^{L=1} (\lambda_k^{L=1})(W_{i,j}^{L=2})^T s^L \\ &= \begin{bmatrix} (1-O_{ij}^1)(O_{ij}^1) & 0 \\ 0 & (1-O_{n,j}^1)(O_{n,j}^1) \end{bmatrix} \begin{bmatrix} w_{ij}^2 \\ w_{ij}^2 \end{bmatrix} s^L \end{aligned} \quad (5.5)$$

The weight and bias updating can be performed according to equation (5.6).

$$\begin{aligned} W_{new} &= W_{old}(k) - \alpha s^L (O^{L-1})^T \\ b_{new} &= b_{old}(k) - \alpha s^L \end{aligned} \quad (5.6)$$

The main aim of this overall training process of MLP network is to minimize the mean square error of the particular adopted network architecture. Convergence of network can be tuned with the parameters α and λ . In this work, two hidden layers are considered throughout the research with three inputs X, Y and Z, while output is depending on the configuration of the robot manipulator.

5.2.2 Polynomial pre-processor neural network

Polynomial pre-processor neural network model having distinguished property of summation of all inputs as compared to MLP network it follows the Weierstrass approximation theorem that states "Any function which is continuous in a closed interval can be uniformly approximated within any prescribed tolerance over that interval by some polynomial". Figure 5.4 depicts a PPN network where X, Y and Z are the inputs pattern given by,

$$X = [x_1, x_2, x_3 \dots x_m]^T \quad (5.7)$$

For instant considering 2D input pattern $X = [x_1 \ x_2]$, to explain the Weierstrass approximation theorem wherein polynomial is order of 2, therefore the function of decision can be written as

$$D(X) = W' X^* \quad (5.8)$$

Where, $W' = [w_0, w_{x_1}, w_{x_2}, \dots, w_{x_1^2}, w_{x_1 x_2}, w_{x_2^2}]^T$ and $X^* = [1, x_1, x_2, x_1^2, x_1 x_2, x_2^2]^T$

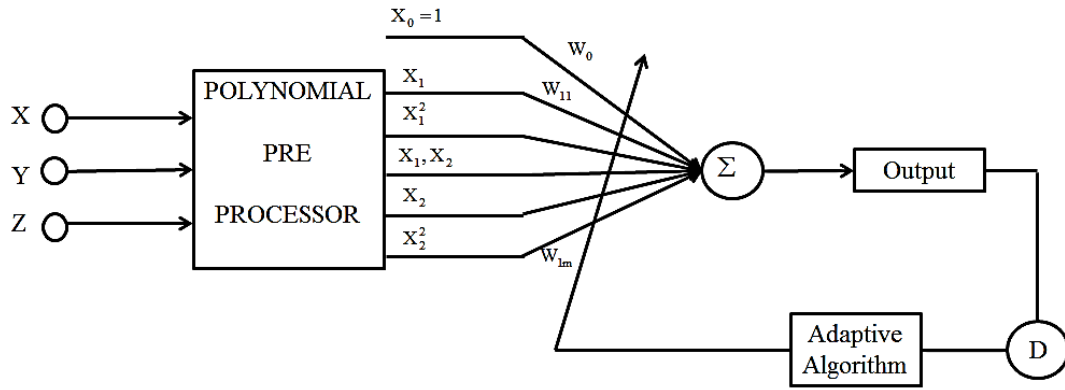


Figure 5.4 Polynomial perceptron network

Now for m-dimensional pattern of input can be formulized using general quadratic case with considering all combination of X elements,

$$D(X) = \sum_{j=1}^m w_{jj}x_{jj}^2 + \sum_{j=1}^{m-1} \sum_{k=j+1}^m w_{jk}x_jx_k + \sum_{j=1}^M w_jx_j + w_0 = W^T X^* \quad (5.9)$$

For the m-dimensional case, the number of coefficients in a function of r^{th} degrees is given by

$$N_{m,r} = {}^{m+r}C_r = \frac{(m+r)!}{m!r!} \quad (5.10)$$

The input pattern X to the PPN at time n is the channel output vector X (n). This is then converted into $X^*(n)$ by passing it into a polynomial pre-processor. The weighted sum of the components of $X^*(n)$ is passed through a nonlinear function sigmoid and pure linear function to produce the output as shown in Figure 5.4.

5.2.3 Pi-Sigma neural network

PSNN (Pi-Sigma Neural Network) is also a feed forward or multi layered neural network consisting of one hidden layer. The major different of PSNN is summing units of hidden layer and product unit of output layer as compared to MLPNN. The weights of input and hidden layer can be obtained during training process of network while hidden layer to output layer weights are fixed to one.

This network uses two different activation functions at hidden layer linear activation function and at output layer non-linear activation function. Therefore pi-sigma network evaluates the summing production of input layer and corresponding weights and passes through nonlinear activation function. This concept of one hidden layer with two activation functions drastically minimizes the total training time for the network. The pi-sigma network structure is presented in Figure 5.5.

Moreover, summing product layer of pi-sigma network provides higher dimension capabilities through the expansion of input dimension into higher dimensional space therefore it can easily split nonlinear separable class to linear separable class. Finally this network is capable of providing the nonlinear decision with better classification of higher dimension data than the normal network.

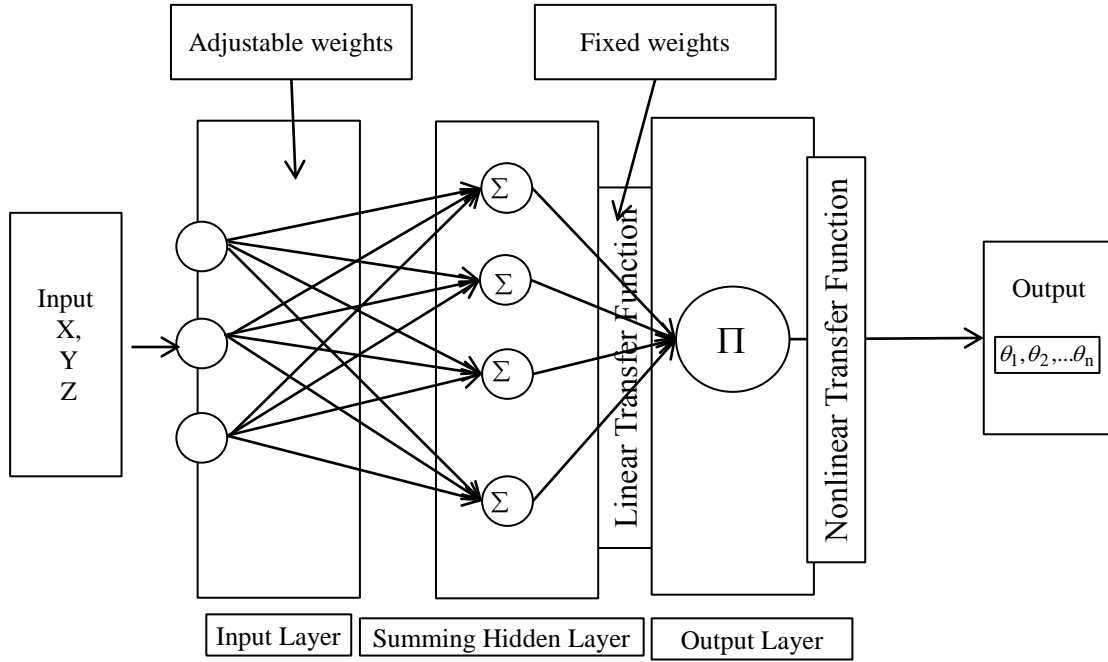


Figure 5.5 Polynomial perceptron network

Now consider pi-sigma network with n number of inputs, with n_h number of hidden neurons and one output neuron. n_h , defines the order of p-sigma network that is $n \times n_h$ considering all summing units are related to n weights. The output of the network will be given by the product of the output of n_h hidden units which passes through the nonlinear activation function; therefore it can be given as,

$$O = \delta \left(\prod_{k=1}^{n_h} h_k \right) \quad (5.11)$$

Where δ a nonlinear activation functions and h_k is the output of k th hidden layer neurons which is then calculated by summing the products of all inputs (x, y, z) with the corresponding weight (W_{ij}) between i th input and k th hidden unit. Therefore output of hidden layer will be given by:

$$h_k = \sum_{i=1}^n (W_{ik} X_i) \quad (5.12)$$

5.3 Application of adaptive neural-fuzzy inference system (ANFIS)

Adaptive Neural-Fuzzy Inference System (ANFIS) developed by Roger Jang []. ANFIS is a hybridization of neural network and fuzzy logic methods. This is basically type of a feed forward neural network which involves fuzzy inference system through the structure of neural network and their neurons. It gives the learning ability of neural network to fuzzy inference system. The method is mainly developed for the evaluations of nonlinear functions that generally identifies nonlinear elements on line for control system design and predicts chaotic time series.

On the other hand, (FIS) fuzzy inference system is most popular computing method which is based on the fuzzy set theory wherein if-then rule and fuzzy reasoning is mainly focused. It is evident from the literature review that FIS having large application areas such as control system, classification of data, decision analysis, system of experts, prediction of time series, robotics, image processing and recognition. The architecture of the FIS consists of three fundamental elements: a rule element, that covers the selection of appropriate fuzzy rules; database, that gives the relationship of membership function with the established fuzzy rules; then finally reasoning components, which gives the appropriate inference method of adopted rules and provides facts to develop reasonable output or conclusion. This can take either fuzzy input or crisp value, but produced outputs are almost fuzzy sets. But sometimes it is required to have crisp value, especially where FIS is used for controller. Therefore, defuzzification is required to decode the crisp value whichever best represent fuzzy set.

Therefore FIS with neural network is used to update the parameters of neural network and can perform mapping of input to output data through appropriate learning algorithm. This process of tuning gives the optimize parameter of neural network. ANFIS structure is consists of five different layers such as fuzzy layer, normalized layer, product layer, defuzzy layer, and summation layer. Basic structure of the ANFIS is given in Figure 5.6, in which fixed node is given by circle and adjustable node is given by square. Suppose if there is two inputs x and y with one output z then ANFIS can be used as a first order Sugeno FIS. There are many fuzzy systems like Sugeno, Mamdani etc., but most popular and widely used system is Sugeno model due to its high interpretability and computational efficiency with default optimal and adaptive tools.

Therefore first order Sugeno fuzzy rule can be expressed as,

$$\text{First rule: If } x \text{ is } A_1 \text{ and } y \text{ is } B_1, \text{ then } Z_1 = p_1x + q_1y + r_1 \quad (5.13)$$

$$\text{Second rule: If } x \text{ is } A_2 \text{ and } y \text{ is } B_2, \text{ then } Z_2 = p_2x + q_2y + r_2 \quad (5.14)$$

Where, A_i and B_i are fuzzy sets and p_i, q_i and r_i are parameters which is assigned during training process. From Figure 5.7 ANFIS structure consists all five layers. Now output node will be defined by,

$$O_i^1 = \mu_{A_i}(x), \quad i = 1,2 \quad (5.15)$$

$$O_i^1 = \mu_{B_i}(y), \quad i = 3,4$$

where $\mu_{A_i}(x)$ and $\mu_{B_i}(y)$ can hold any membership function (MF). For example, in this work widely used membership function i.e. Gaussian MF is used throughout the work.

$$\text{gaussmf}(A, B, C) = e^{-\frac{(A-c)^2}{2B^2}} \quad (5.16)$$

where C_i, B_i are the parameters which changes shape of MF. Second layer nodes are represented by Π which is fixed.

$$O_i^2 = \omega_i = \mu_{A_i}(x)\mu_{B_i}(y), \quad i = 1,2 \quad (5.17)$$

Each node output represents the firing strength of a rule.

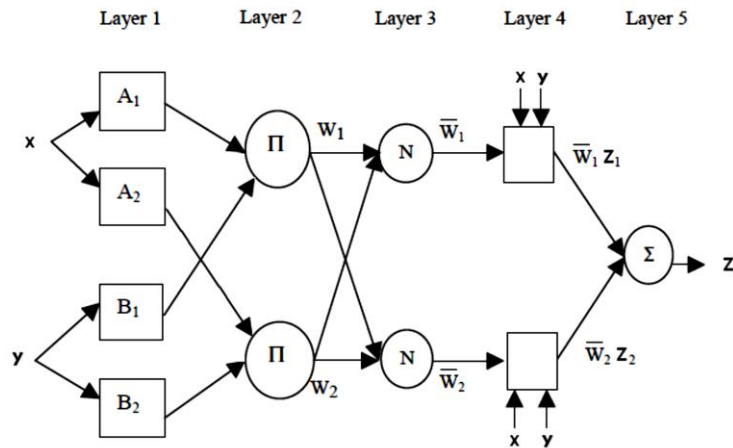


Figure 5.6 Architecture of ANFIS

Third layer fixed nodes are represented by N . In this layer, the average is calculated based on weights taken from fuzzy rules:

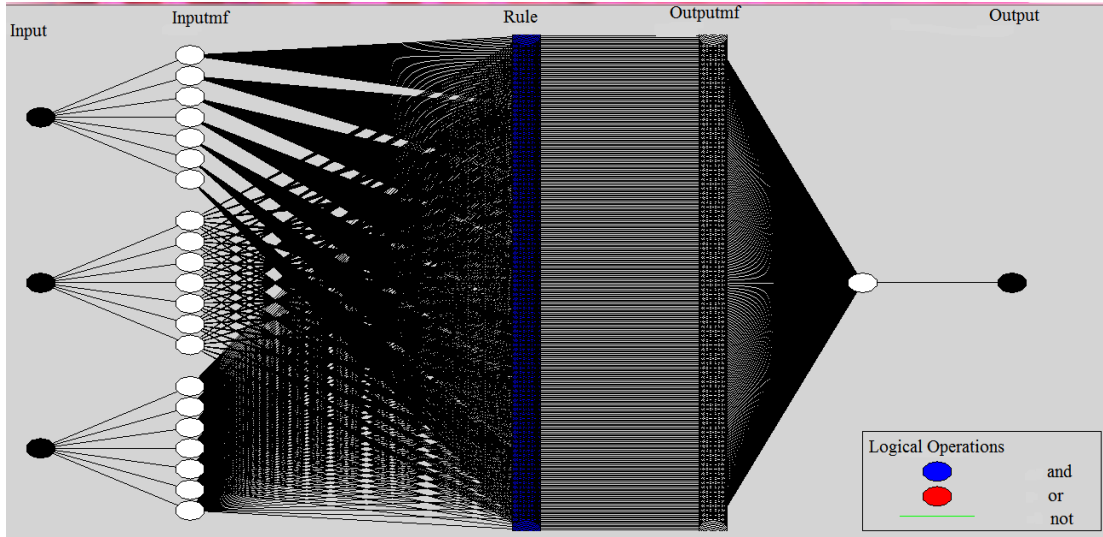


Figure 5.7 Training of ANFIS structure

$$O_i^3 = \bar{\omega}_i = \frac{\omega_i}{\omega_1 + \omega_2}, \quad i = 1, 2 \quad (5.18)$$

Where ω_i are normalized firing strengths. Every i th node in the fourth layer is an adaptive node given by following node function,

$$O_i^4 = \bar{\omega}_i z_i = \bar{\omega}_i (p_i x + q_i y + r_i), \quad i = 1, 2 \quad (5.19)$$

The parameters (ω_i , p_i , q_i and r_i) of this layer are consequent parameters. For the fifth layer fixed node is given is Σ that calculates all output as summation of all inputs by,

$$O_i^5 = \sum_{i=1}^2 \bar{\omega}_i z_i = \frac{\omega_1 z_1 + \omega_2 z_2}{\omega_1 + \omega_2} \quad (5.20)$$

5.3.1 Learning algorithm

In ANFIS there is forward learning process which is based on least square method and backward learning is given by gradient descent learning process. If the premise parameters are fixed then the output of the ANFIS can be given as,

$$z = \frac{\omega_1}{\omega_1 + \omega_2} z_1 + \frac{\omega_2}{\omega_1 + \omega_2} z_2 \quad (5.21)$$

Replacing Eq. (5.19) into Eq. (5.21) gives,

$$z = \bar{\omega}_1 z_1 + \bar{\omega}_2 z_2 \quad (5.22)$$

Replacing the fuzzy if-then rules into Eq. (5.22), it becomes:

$$z = \bar{\omega}_1 (p_1 x + q_1 y + r_1) + \bar{\omega}_2 (p_2 x + q_2 y + r_2) \quad (5.23)$$

After rearrangement, the output can be written as a linear combination of the consequent parameters:

$$z = (\overline{\omega_1 x})p_1 + (\overline{\omega_1 y})q_1 + (\overline{\omega_1})r_1 + (\overline{\omega_2 x})p_2 + (\overline{\omega_2 y})q_2 + (\overline{\omega_2})r_2 \quad (5.24)$$

Least square method is used to calculate the optimal value of the consequent parameters. When premise and consequent both parameters are adaptive, then it develops higher search space and this leads to solve convergence of training process. Therefore hybrid learning with back propagation is used to solve convergence problem. This hybrid learning reduces the search space dimension. At the learning stage, both premise and consequent parameters are properly tuned till the desired output achieved by FIS. Figure 5.6 represents the training process of ANFIS which is done by using MATLAB ToolBox of `anfisedit` command. In this work different ANFIS structure with first order Sugeno fuzzy system is considered for various considered joint variables of robot manipulator. Where input is considered as the end effector positions (X, Y and Z) and data sets were generated by forward and inverse kinematic equations.

5.4 Hybridization of ANN with metaheuristic algorithms

After introduction of simple neural network with the wide application of feed forward neural network with back propagation algorithm as well as multi-layered perceptron network yields many troubles for training of an algorithm. Back propagation algorithm is generally direct search method with weight updating rule to ensure the minimization of the error. However, there are many key points, which ensure the algorithm not definite for the comprehensively useful for many applications. One of the major key point of this algorithm is learning rate parameter which is strictly require to tune properly else it creates fluctuation as well as more computational time for training. On the other hand, weights updating leads to long training time for the specific application of the algorithm. Furthermore, back propagation algorithm ultimately gives slow convergence rate if the number of hidden layer increased due to its weight updating rule. The most important point is the learning algorithms such as gradient descent learning of back propagation algorithm which is generally complex and also contains various local minimum points. Therefore, this algorithm mostly gets stuck into local minima, which make it utterly dependent on weight updating and initial settings. Therefore, hybridization of ANNs can be done in many ways to overcome all stated problems. The categorization of the hybridization of ANN can be explained as follows:

1. Architecture optimization
2. Weight and bias optimization
3. Learning rate and momentum parameter optimization

In case of optimal architecture design, the number of hidden layers is the key factor for designing the architecture. To find out the best structure for specific problem training

algorithm apart from gradient descent learning an optimization algorithm is used. The architecture is dependent on the neurons connections, no. of hidden layers and hidden nodes of neural network. Many researches have been done in this field for elementary solutions. The structure of the neural network model can be given by upper bound method. But in case of boundary it may only provide basic idea about the structure but in case of high nonlinear functions and highly dynamic nature can cause the network to go beyond the requirement. Therefore it can be used as approximations of the structure optimization. There are few determinations for the designing of systematic architecture such as constructive and pruning algorithms. Constructive method initially assume the neural network with minimum nodes and then start adding nodes and links until to get optimum structure while in case of pruning method it assumes the large network which proceed with pruning off the nodes and links form the network to get best structure. These algorithms are also trapped in local optimum structure because of the non-differentiable space, complex and multi-model structure. Therefore these algorithms are also facing the similar problem like back propagation algorithms.

Hence the second case i.e. weight and bias optimization is more promising and stable method to optimize the neural network for better training than optimizing of architecture. In case of weight and bias optimization algorithms the architecture is constant before the training of the neural network model. The training algorithms can be application of any metaheuristic algorithms which make sure the global optimum point for the specific problem. Therefore the main aim of the training algorithm is to find an appropriate connection weight and bias to reduce overall error. Therefore global optimization algorithms like, PSO, WDO, GSA, Evolutionary algorithms, GWO, TLBO, BBO, ABC, ACO etc. are quite healthy to use for the training and finding out the optimum weight and bias for the neural network. The common factor for all global optimization algorithms is population based stochastic method and can easily avoid local optimum points to get best solution. Moreover, these algorithms can applied to any model of neural network with different number of activation functions.

In this work, PSO, GA, GWO, CIBO, TLBO etc. algorithms are applied using weight and bias based optimization criteria and multi-layered perceptron neural network (MLP) is used throughout the research. The hybrid ANN can be called as MLPPSO (multi-layered perceptron particle swam optimization), MLPGA, MLPGWO etc. Therefore to design proper algorithm objective function or fitness function is most important factor for optimization. In the later section mean square error based objective function formulation is presented. Now hybridization of metaheuristic or population based algorithms method can be start with the introduction of the adopted algorithms with the

specific model of neural network. Therefore the basic of population based stochastic algorithms are explained below.

5.4.1 Particle swarm optimization

PSO is a population-based optimization algorithm imprinted from the simulation of social behaviour of bird flocking. The population comprises of the number of particles (candidate solution) which flies in search space to find the out global optimum point. Initial approximation of particles for position and velocity in search space is randomly chosen as shown in Figure 5.8. Each individual flies in the search space with specific velocity and carrying position, simultaneously each particle update its own velocity and position based on the best experience of its own and the social population [248]. The basic steps with mathematical modelling of Particle Swarm Optimization Algorithm are shown in flowchart:

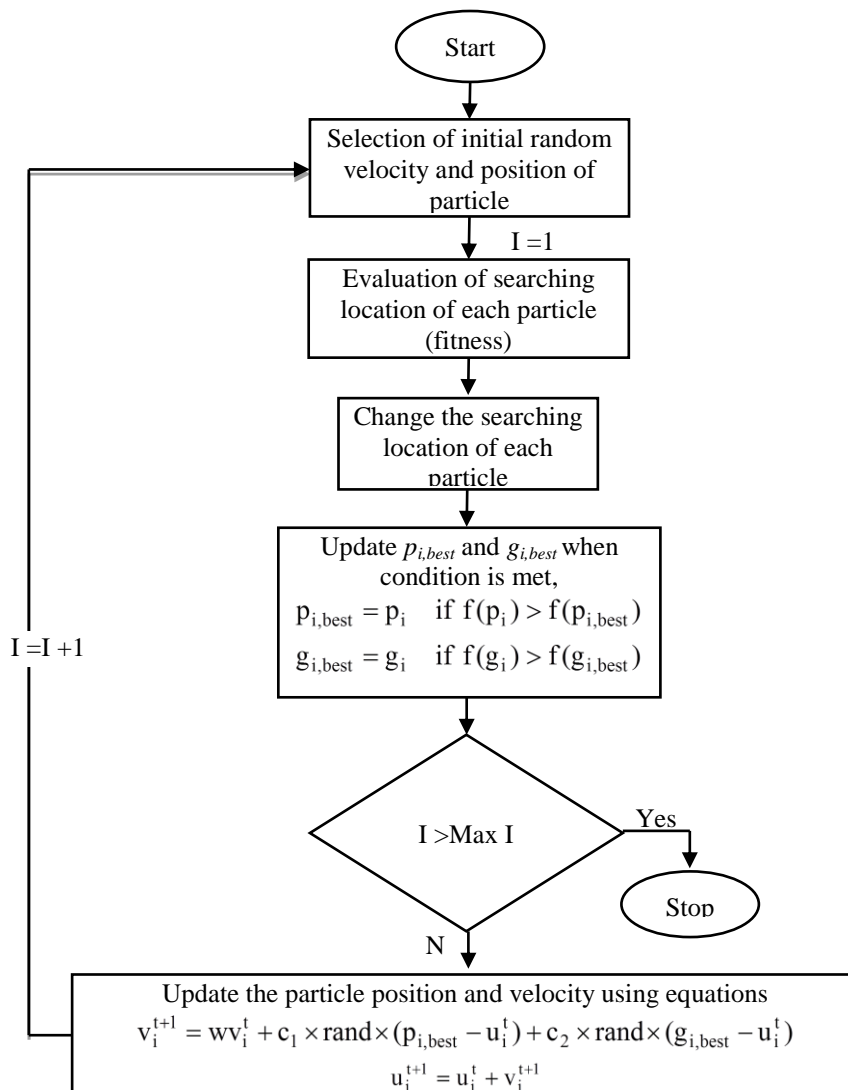


Figure 5.8 Flow chart for PSO

5.4.2 Teaching learning based optimization (TLBO)

Teaching-Learning-Based Optimization (TLBO) is population based algorithm works on the effect of impact of a teacher on learners. In this algorithm population is considered as group of student where each student either learns from teacher called as teacher phase and they also gain some knowledge from other classmates or students that are learners phase. Output will be in terms of results or grades. In these algorithms different subjects for learner resembles the variables and learner results is equivalent to the fitness function for any problem and finally the teacher will be considered as the best solution achieved so far. There are several other population based methods have been successfully implemented and shown efficiency. The details about this algorithm can be found on reference. [249].

5.4.3 Objective function for training MLP

Analytical solution of the inverse kinematics problem is highly non-linear and mathematically complex in nature. An ANN model does not require higher mathematical calculations and complex computing program. ANN requires initial selection of weight, which is vigorous to yield local optima, convergence speed and training time for the network. As we know that the bias and weight for each neuron directly affect the output vector of neural network. Generally, weight is randomly selected in the range of 0 to 1, after activation function weight of each neuron adjusted for the next iteration. The heuristic optimization algorithm optimizes the weights of the neural networks. When certain termination criteria are met, or a maximum number of iterations are reached, the iterations cease. From the previous research hybrid optimization, algorithm started evolving with high and remarkable advances in their performances, [250]-[251]. These techniques produce better outflow from local optimum and testified to being more operative than the standard method. All these approaches yield better results when neuron weight is adjusted. In this work, optimized weight and bias for each neuron using various metaheuristic algorithms are used for the training of MLP network. For the training of network, it is important to have all connection weights and biases in order to minimize the mean square error.

5.4.4 Objective function

From [24], in each epoch of learning, the output of each hidden node is calculated from equation (5.35).

$$f(n_k) = \frac{1}{(1 + \exp(-(\sum_{i=1}^n w_{ij} \cdot x_i - b_j)))}, \quad j = 1, 2, 3, \dots, h \quad (5.35)$$

Where $n_k = \sum_{i=1}^n w_{ij} \cdot x_i - b_j$, n is the number of the input nodes, w_{ij} is the connection weight from the i th node in the input layer to the j th node in the hidden layer, b_j is the bias (threshold) of the j th hidden node, and x_i is the i th input. After calculating outputs of the output nodes from equation (5.36).

$$o_k = \sum_{i=1}^h w_{kj} \cdot f(n_k) - b_k, \quad k = 1, 2, \dots, m \quad (5.36)$$

Where, w_{kj} is the connection weight from the j th hidden node to the k th output node and b_k is the bias (threshold) of the k th output node.

Finally, the learning error E (fitness function) is calculated from equation (5.37-5.38).

$$E_k = \sum_{i=1}^m (o_i^k - y_i^k)^2 \quad (5.37)$$

$$E = \sum_{k=1}^q \left(\frac{E_k}{q} \right) \quad (5.38)$$

Where, q is the number of training samples, y_i^k is the desired output of the i th input unit when the k th training sample is used, and o_i^k is the actual output of the i th input unit when the k th training sample is used. Fitness function can be calculated from equation (5.39). Where the number of input nodes is equal to n , the number of hidden nodes is equal to h , and the number of output nodes is m . Therefore, the fitness function of the i^{th} training sample can be defined as follows:

$$\text{Fitness}(X_i) = E(X_i) \quad (5.39)$$

5.4.5 Weight and bias optimization scheme

To represent weights and biases it is required to indicate the encoding strategy after defining the fitness function for hybrid ANN, [252]-[253] From the literatures, there are three encoding strategies for representing the weights and biases. First strategy is vector method in which every agent is encoded as a vector. For training MLP each agent encoded as a vector to represent all weights and biases for the MLP structure (see Figure 5.10). The optimization of weight and bias using PSO as shown in Figure 5.10 can be implemented for all other optimization algorithms. In matrix encoding, each agent is encoded as a matrix. In case of binary encoding, agents are encoded as strings of binary bits. From the literatures [250]-[251], in case of vector encoding strategy, the encoding is simple, but after calculation of output of MLP, it is required to decode each particle into weight matrix, therefor decoding process becomes complicated. Vector encoding strategy is generally used in the function optimization field. In case of matrix encoding strategy, the decoding is simple for weight matrix but the encoding is difficult

for neural networks with complex structures. This method is very suitable for the training processes of neural networks because the encoding strategy makes it easy to execute decoding for neural networks. In the last strategy, each particle should represent in the binary form, so encoding and decoding becomes complicated for the complex network structure. An example of this encoding strategy for the MLP has given in Figure 5.9.

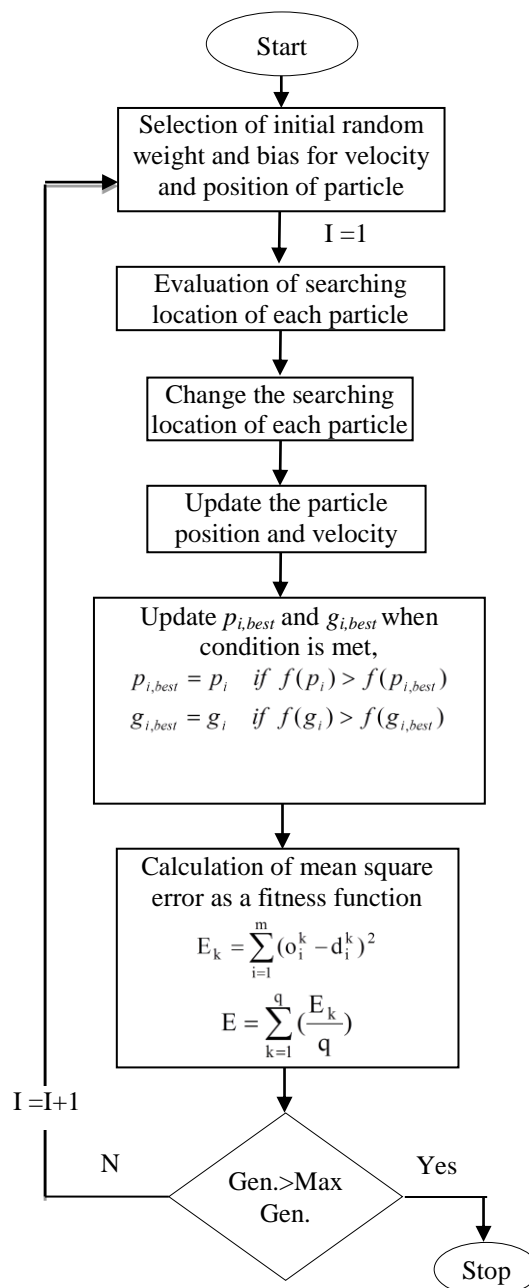


Figure 5.9 Flow chart for MLPSO

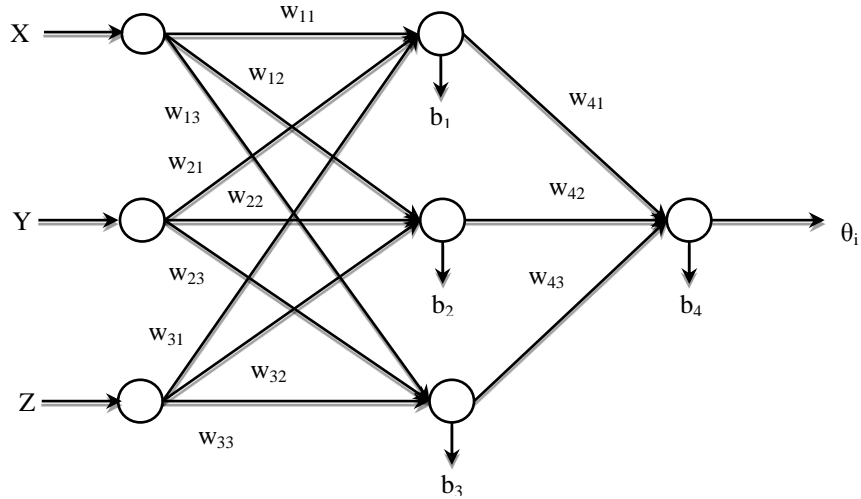


Figure 5.10 MLP network with structure 3-3-1

$$\text{search_agents}(:, :, i) = [W_1, B_1, W_2, B_2]$$

$$W_1 = \begin{bmatrix} w_{11} & w_{21} & w_{31} \\ w_{12} & w_{22} & w_{32} \\ w_{13} & w_{23} & w_{33} \end{bmatrix}, W_2 = \begin{bmatrix} w_{41} \\ w_{42} \\ w_{43} \end{bmatrix}, B_1 = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}, B_2 = [b_4]$$

Where W_1 is the hidden layer weight matrix, B_1 is the hidden layer bias matrix, W_2 is the output layer weight matrix, W_2' is the transpose of W_2 , and B_2 is the hidden layer bias matrix.

5.5 Summary

This chapter delivers the basics of artificial neural network technique and their hybridization scheme with metaheuristic optimization algorithms. Furthermore, different types of multi-layered perceptron network, their learning abilities are discussed. Moreover, it is also covered the combination of evolutionary algorithms with MLP neural network as well as comparison of gradient descent learning algorithms and appropriate scheme. In current scenario hybridization of ANNs with metaheuristic algorithms are popular and reaching to the advanced stage of the soft computing techniques to handle non-linear, NP-Hard problems, complex mathematics and noisy problems. Therefore in the later section, few different type of metaheuristic algorithms such as PSO, GA, GWO, and CIBO is discussed here which is later used to obtain the optimized weight and bias of the adopted neural network model. After the application of the metaheuristic algorithms and trained neural network, is applied to find out the inverse kinematic solution of the robot manipulators. Different types of configuration of the robot manipulators have been taken for the kinematic analysis. The results obtained out of all these models are presented in chapter 7.

Therefore, inverse kinematic solution of various configurations of the manipulators with and without Euler wrist are solved computationally using trained hybrid MLP neural network. The adopted method is compact and efficient tool for kinematic analysis. In the result chapter inverse kinematic solution for adopted manipulator has been tabularised and comparison on the basis of mathematical complexity is made over other conventional based method.

Chapter 6

OPTIMIZATION APPROACH FOR INVERSE KINEMATIC SOLUTION

6.1 Overview

Optimization is the method which yields best solution of a problem having number of variables and alternatives. From the definition, it includes the phenomenon or some biological concept in our daily life that inspires to minimize the energy, computational cost, mathematical operations, time, etc. and maximises efficiency, profits, power etc. with the help of some direct and indirect parameters. For example, computation of inverse kinematics problem of robot manipulator with the direct relation of considered torque, energy and time to be minimized to get the desired position. In this example joint variables can be calculated after optimization of the position error, torque, energy etc.

Therefore in broad sense, the major constituents of the optimization methods can be recognize as its objective function which is generally a quantitative expression of the system to be optimized and then the number of unknown parameters or set of variables that is required proper setting to yield optimum value, finally the number of constraints which gives the complete objective function for the concern domain. These three constituents is the basis to solve any optimization problem and their objective function (fitness function) formulations. On the other hand, the major objectives for optimizing of any function would be the convergence of the solution. Furthermore, optimization algorithms should always be flexible to manage various problem such as nonlinear, NP-hard, discrete, multi-objectives, multi-modals etc. Most important property of any optimization algorithms is to avoid the local optimum point. Considering an equality and inequality constraints problems, objective function can be defined,

Minimise $F(\theta)$

Subject to:

$$\begin{aligned} G_1(\theta) &= a_1 \\ G_2(\theta) &= a_2 \\ &\cdot \\ G_n(\theta) &= a_n \\ \\ H_1(\theta) &= b_1 \\ H_2(\theta) &= b_2 \\ &\cdot \\ H_l(\theta) &= b_l \end{aligned}$$

Further these constraints $G_n(\theta)$ and $H_l(\theta)$ can be handle by Lagrangian formulations by equation (6.1) as,

$$L(\theta, \lambda, \mu) = F(\theta) + \sum_{i=1}^n \lambda_i (a_i - G_i(\theta)) + \sum_{j=1}^l \mu_j (b_j - H_j(\theta)) \quad (6.1)$$

Suppose that $\theta^* = \theta_1^*, \theta_2^*, \dots, \theta_n^*$ is responsible for the maximization of the objective function (1) which is subjected to the constraints $G_i(\theta) = a_i$ and $H_j(\theta) = b_j$ where $i=1,2,\dots,n$ and $j=1,2,\dots,l$ then the vectors $\delta G_n(\theta^*)$ and $\delta H_l(\theta^*)$ will be linearly independent to the problem. On the other hand, there may the Lagrangian vectors λ^* and μ^* will be given by equation (6.2) as,

$$\delta F(\theta^*) - \sum_{i=1}^n \lambda_i^* \delta G_i(\theta^*) - \sum_{j=1}^l \mu_j^* \delta H_j(\theta^*) = 0 \quad (6.2)$$

$\mu_j^* (H_j(\theta^*) - b_j) = 0$ will be complementarity and $\mu_j^* \geq 0$.

These above mentioned conditions for the optimality and complementarity is known as basic concept of the Kuhn-Tucker. Therefore the constraint optimization problem can easily handle with these concepts to make the objective function unconstrained. The optimal solution of the θ^* can be either minimum or maximum depends on the considered problem and the solution may be local, global optimum or near optimal. Further the optimization problem can also be categorized as the considered objective function may be linear or non-linear following algebraic, polynomials or transcendental etc. formulations. It can also be based on constraint with integer or mixed integer, and also the problem may be the numeric or symbolic. Therefore the optimization is depending on the real world problem which can be formulated by above considered cases.

In case of iterative optimization, initial approximation of the solution accelerates the process by consequently updating the current solution with the old solutions until it obtained the optimal point. The method is basically a manifold but it requires the attributes of objective function. On the other hand the conventional methods gradient based searching process is the key point to obtain the local optimum point, which means the objective function is differentiable and the gradient of the function can be evaluated, then optimal solution yields with descent direction search with each gradient point. These methods are known as line search and some other conventional methods are steepest-descent, quasi-Newton, Newton, Non-linear conjugate methods etc. The major advantages of the above mentions methods are its local search ability, convergence of the solution for unconstrained problems, wherein accurate solution with the help of gradient based method is easy and computational cost is less. However, the fitness function (objective function) should be unimodal and can be differentiable for two steps. The problem with the method is non-smoothness and noisy solution of the objective function if it cannot be explain by algebraic or analytical formulations. On the other hand, zeroth order method does not require the higher derivatives and gradient based approximation. The interesting point in this method is the deficiency of the wide assumptions like continuity and differentiability of the function is not important. Few examples of the methods are direct search, particle swarm based optimization, bacteria foraging, evolutionary algorithms etc.

The direct search methods are also known as heuristic based algorithms which contains the test and generation of the strategy. Wherein, every individual solution for the function is compared and evaluated so as to find the best solution with the constant observation of the improvement. There are two strategies for selection or sampling namely stochastic and deterministic. Stochastic search can be understand with the random search in the current dimension while in case of deterministic search a fixed or predefined coordinate of search for local best solution is known. Random walk is the examples for stochastic search process and pattern search, simplex method are deterministic methods for local optimization process. Due to its random variable dependencies its gives slow convergence while derivative based method performs faster. If the numbers of local optimum points are more than one then poor approximation which is combined with the greedy search could be stuck at sub-optimal point. Subsequently, initial approximation for the algorithms is less important if one considers the effective selection of the search space.

Therefore, population based algorithms gives solution to the initial approximation problem with the help of selection of the objective function that acts as indirect local optimizers. Furthermore, it is also required to find the exact number of initial

approximations for the objective function so as to find global optimum. In this regard, evolutionary algorithms can be fruitful to adopt wherein it helps not only for the population generation of the candidate solution even use parallel local optimizers. On the other hand, exploration and exploitation abilities are also help to find the global points. The overview of this chapter is concerned with the brief introduction of the conventional and biological inspired algorithms, with the three major concepts of the optimization algorithms. The above introduced classical methods on the basis of the conceptual frame will be later used to explain the evolutionary algorithms, metaheuristic algorithms and swarm based algorithms etc. for the evaluation of the inverse kinematic problem of robot manipulators.

6.2 Metaheuristic algorithms

In the last few years, metaheuristic algorithms have been extensively used for resolving various complicated optimization problem. Nature is playing key role for developing many optimization algorithms for example artificial bee colony algorithm, firefly algorithm, ant colony optimization etc. We are always attracting by tiny or large organisms like diminutive invertebrate, charismatic vertebrates, birds, primates, bees, and ants etc. which are often the source of inspiration for many researchers [252]. These organisms provide the most delicate systems for exploring nature and answering fundamental scientific questions.

Comparatively metaheuristic algorithms are more appropriate and dominant than the other analytical methods which are based on conventional mathematics and derivatives. Metaheuristic algorithms commonly have two elementary features like intensification and diversification. Intensification normally offers local search near to existing current best solutions whereas diversification offers efficient exploration of search space, mostly based on random numbers [253], [254]. Metaheuristic algorithms are widely used because they provide global solution keeping the aim of faster solution, solution of lengthy problems and obtaining robust techniques. Metaheuristic algorithms can find proximate optimum solutions at a sound computational cost which doesn't assure feasibility or optimality of the obtained solution, on the other hand in most of the cases researchers are keen to see the closeness to optimal and feasibility of the solution. [255]-[256].

The nature is infinite and there is no limit for the source of inspiration for example previously developed algorithms are inspired from ants, bees, fireflies, bacteria, music, habitats, frogs etc. There are many nature-inspired optimization algorithms have appeared for example the Genetic Algorithm (GA) [257], which mimics the genetic process of biological organism. The concept of GA came from the Darwin's principle

"survival of the fittest", which describes the evolution of population on the basis of natural selection. In this algorithm each individual represented by gene and the combination of gene creates chromosome which is ultimately yields solution. These chromosomes are recombining using crossover and mutation. This behaviour leads to global solution for the objective function [258]. In the process of evolution of natural things is mostly the source where selection process of the concern organisms in population keeping the best fitted to the environment is always adapted. This provides the most prominent optimization algorithms. As per Darwin theory, evolution mainly concern with the interaction of the physical mechanism of selection, reproduction, mutation, and competition with other species or organisms. In this theory, each individual are compulsorily need to compete the physical process for the survival, and this will lead to find the best or selection of survivals with better genetic character for the concerned environment so as to produce offspring or reproduction.

Evolutionary algorithms EAs are metaheuristic algorithm based on the population of the individual solution that evolves by selection, mutation and reproduction of best fit in the population. The major advantage of these EAs algorithms compare to conventional method, conventional methods relies on the local memory of one point in each step of iteration which leads to local optimization process whereas population based metaheuristic methods uses parallel search mechanism with the major ability of exploration and exploitation. The major application fields of these algorithms are mostly in research and industries, mostly in robotics, machine deign, control application, image processing, modelling, signal processing etc. The basic pseudo code for evolution algorithms can be given as,

Evolutionary algorithm

1. Initialization of population
2. evaluation of each individual
3. **While** termination criteria met **do**
4. selection of parents
5. recombination of parents
6. mutation yields offspring
7. evaluation of new individuals
8. selection for next generation
9. **end while**

In the above pseudo code considering the first example of minimization of $F(\theta)$ function optimization where initialization can be done within the feasible search space. Initial population can be selected randomly within the search space. Once initialized, population will go through the iteration and selection of each individual best solution until it converges for e.g. function threshold, no. of generations, etc. This each iteration mechanism gives information encoded in the current population so as to achieve new

trial generations, e.g. mutation and recombination. Selection process gives the solutions to replace the trial population within the current population so as to determine the next generation. The whole process of optimization of each individual solution within the current and trial population are assigned with the objective function value that signifies the closeness to minimum value. Moreover, this objective function evaluation helps the overall search process to find out which individual can be used for reproduction and even survive for the more generations.

Therefore, convergence of the solution would be depending on exploration and exploitation ability of the search process within certain regions. In evolutionary algorithms, iterative process gives the ability to explore the new regions within the search space while selection is responsible for the exploitation of individual which would be carrying the information's to ensure the next generation to be completed. Finally, evolutionary algorithms can be categorized as, Genetic algorithms (GAs), Genetic programming (GP), Evolutionary Programming (EP) and Evolution strategies (ESs).

The bio-logically inspired metaheuristic algorithms mimic the best feature of the nature which could turn into better efficiency as compared to other conventional algorithms. More often, these approaches are selecting the fittest value which has evolved by natural selection. Bio-inspired techniques may be categorized into: (a) Bacterial foraging algorithms (b) Evolutionary algorithms, (c) Swarm intelligence based algorithm [259]. Genetic algorithm is the key factor for the establishment of evolutionally algorithms because it satisfies the principle of "survival of fittest" given by Darwin. This classification covers genetic programming (GP), differential evolution (DE), evolutionary strategy (ES) and biogeography based optimization (BBO), but also other. These are also population based metaheuristic algorithms working with some form of the Darwin's principle [259].

A swarm intelligence based algorithm anticipates specific operations, interactions and sharing information with other particles. These operations can be social and cognitive, due to their social behaviour and knowledge sharing habits turns into intelligence, which can be further known as swarm intelligence. Their cognitive and social behaviour yields global results [218].

Another category of population based metaheuristic search algorithm is bacteria foraging algorithm. [260]-[261]. The most well-known types of the bacterial foraging algorithms are computing systems of microbial interactions and communications and rule-based bacterial modelling. Basic concept of these nature inspired population based algorithms are, dimension of the search space, number of individuals, basic related parameters, stopping criteria, number of evaluations etc. Each individual signifies a

resolution for the function optimization problem. In every generation a set of new solutions are obtained and then best solution are kept in memory to produce new set solution this process end when it reaches to certain termination criteria [262].

Major drawback of these algorithms is the number of individuals which share information and this may cause hurdle to yield best or global solution. On the basis of inaccurate or insufficient information they may be converged in the local optimum point because the searching space or dimension of the problem may not discovered adequately. Moreover, similar individuals don't yields different solution that can also be drawback of the algorithm when the function having many local optimum points.

6.2.1 Genetic algorithms (GAs) representation

Genetic algorithm was first developed by J. Holland based on the artificial behaviour of natural system. GA's are encoded with the binary strings of 0's and 1's and it can be represented as genes of biological or natural systems. These genes are certain sequence of the chromosomes and determine the behavioural and physical characteristics of an organism in the environment. In the same way any evolutionary algorithms can be defined as two separate search spaces in which genes represents the variables to be optimized. Physical or behaviour parameter of the system can be represented by the solution space while the encoding with the genes gives the representation space. These physical parameters are known as phenotype and gene encoding is genotype of the system. Since genotype influence the individual solutions in the representation space while evaluation is accomplished in the solution space, therefore it is required to complete encoding and decoding of the variables from the solution to representation space. Moreover, the variables or parameters can also be represented as d-dimensional arrays, where each individual is either binary or real valued, that is $\theta \in [0, 1]^d$ or $\theta \in \mathcal{R}^d$ respectively.

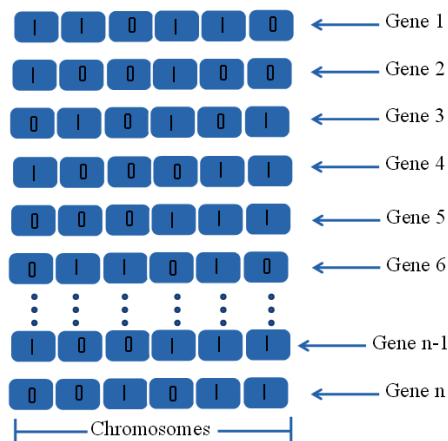


Figure 6.1 Binary representations of genes

In genetic algorithm binary value is most often used for the representation of genes and the chromosomes, whereas in case of mutation based evolutionary programming and evolutionary strategies are real valued representation. Figure 6.1 represents a chromosome with n genes all are coded as six-bit binary words. Design variables and control parameters for both algorithms are encoded in single array. In case of evolutionary strategies each individual solution can be represented equation (6.3) as,

$$S = (\theta, \sigma, \alpha) \quad (6.3)$$

where, $\theta \in \mathfrak{R}^n$ is the design vector, and σ, α are the evolution strategy parameters. Parameter σ belongs to the vector space of standard deviation which modifies the amplitude during mutation of θ and can be given as $\sigma \in \mathfrak{R}^{n_\sigma}$ ($n_\sigma \in \{1, 2, \dots, n\}$). Similarly α is a set of rotation angles that gives the axes of orientation for the mutation in the search space topology and can be represented as $n_\alpha \in (n_\alpha \in \{0, \Lambda (2n - n_\alpha)(n_\alpha - 1)/2\}) [\dots\dots]$.

Similarly, evolutionary programming each candidate is represented as design vector and vector of variance υ and can be given equation (6.4) as as,

$$S = (\theta, \upsilon) = (\theta_1, \Lambda \theta_n, \upsilon_1, \Lambda \upsilon_n) \quad (6.4)$$

where, $\theta \in \mathfrak{R}^n$ is real valued parameter and $\upsilon \in \mathfrak{R}^{n_\upsilon}$ is real positive variance.

(a) Initialization

In most of the metaheuristic algorithms the decision for the initial approximation is play crucial role to reach the optimum point. Since the optimization process is absolutely based on the initial approximations therefore it is required to ensure the convenient procedure for random sampling of the initial approximations. Initialization gives the hint to build the candidate solutions by sampling of the feasible search space. In most of the cases, random number generation is used to sample the initial guessing so as to ensure the high diversity in the initial point. Instead, if prior information about the optimum point is available, then this information will be fruitful to use for the initialization process.

In genetic algorithm, initialization process is done with the random sampling of the $d \cdot \theta$ times the binary value $\{0, 1\}$. In case of evolutionary strategies, initialization process is made through the mutation upon which a starting point is selected randomly or defined by user, and small standard deviation value is suggested. Finally, evolutionary programming uses the uniform random distribution for the initialization of design vector and variances.

(b) Recombination

Recombination process can be understood with the mechanism of involving of two or more parents which may be sexual, asexual or panmictic to produce new offspring's. It can be represented as $R : S^p \rightarrow S^q$ and this can be understood with the sexual or apomictic gene operator where $2 \leq p \leq \theta$. The above mechanism of recombination imitates the biological process to generate new individual solutions or offspring by sharing genetic information that are imprinted in all individuals of the parents.

In genetic algorithm, this recombination process is generally based on the selection of chromosome in each individual randomly where p represents the no. of chromosome for selection and it can generate by the crossover probability $p_c = [0, 1]$. This probability value is compared and measured with the simple random number $r = [0, 1]$. If the $r \leq p_c$ then random crossover of chromosome in the bit string can be selected for the next generation of offspring. After crossover of the selected bit other data will be swapped to create children chromosome with the replacement of the random crossover point. If $r \geq p_c$ then the parent chromosomes can be duplicated as shown in Figure 6.2.

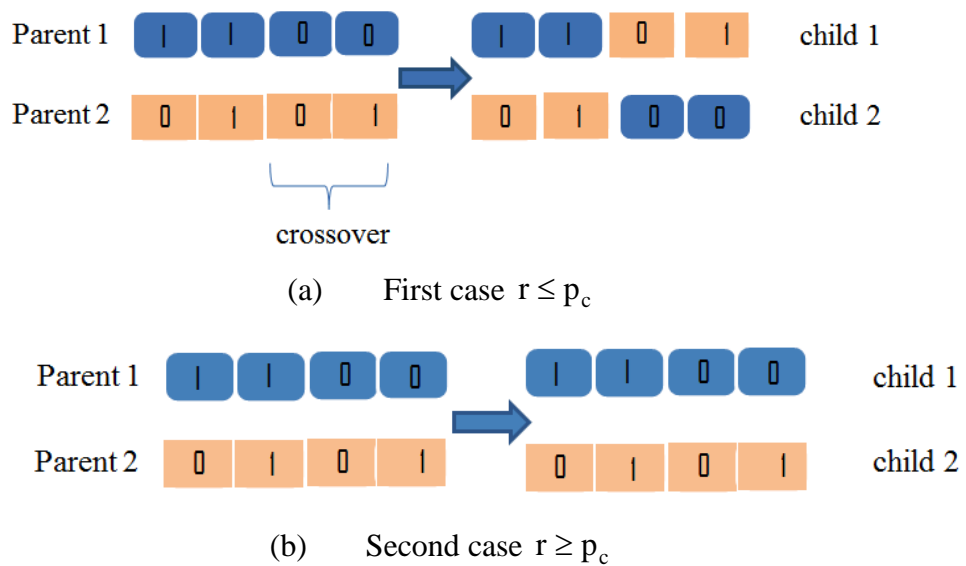


Figure 6.2 Examples for simple crossover with two different cases.

On the other hand, recombination in ESs can be sexual or panmictic for the generation of new offspring with considered random parents. Consequently, sexual recombination will be on the pair basis where $p=2$, for each new offspring's. In case of panmictic recombination one parent will be constant and another will be randomly selected from the parent population ($p = \theta$) for each individual offspring. Recombination can be intermediate or discrete, discrete recombination is random selection of the each

component of the offspring and intermediate calculates the arithmetic mean of the each component of the offspring.

(c) Mutation

Mutation can understand with the mechanism similar to the recombination process besides sexual or panmictic operators it works on asexual operator and can be represented as $M : S \rightarrow S$. This gives the small random changes into the gene coding for each individual. Mutation operator basically works on the population multiplicity with the addition of small perturbations on the individuals with further ability of exploration of new regions within the search space. It also helps to overcome the problem of trapping in local minima.

Genetic mutation is similar to recombination process apart from inverting the value of random bits of chromosomes. Correspondingly, one point crossover, mutation is generated by some activation of mutation probability $p_m = [0, 1]$. This mutation probability will then be compared with the uniformly randomly generated number $r = [0, 1]$ such that if $r \leq p_m$ then bit will be inverted otherwise it will be unchanged (Figure 6.3).

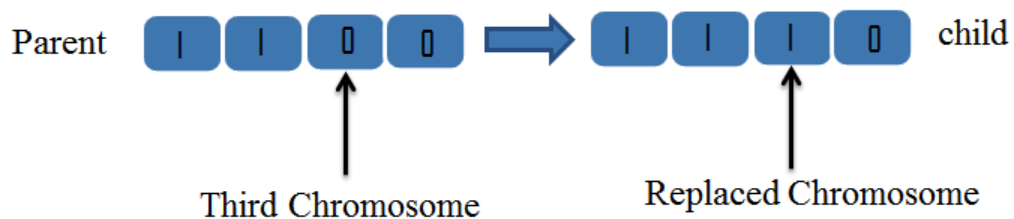


Figure 6.3 Mutation in genetic algorithm

(d) Selection

As we know that recombination and mutation gives the ability of exploration, whereas selection is mainly responsible for the exploiting the candidate solution with the advancement of the next generations. Since the selection exploits the favourable points in the search space, the fitness of each individual must be measured in the population. To accomplish the most promising area in the search space, it is required to define the objective or fitness function which confirms the closeness of the solution towards the optimal value. Let us assume the fitness function f to elaborate the selection procedure in genetic algorithm. Therefore the probability of selection can be given for each chromosome $s_i, i = 1, \Lambda \theta$, in the population equation (6.5) as,

$$p_s = \frac{f(s_i)}{\sum_{k=1}^{\theta} f(s_k)} \quad (6.5)$$

Where, θ represents the population size. The most common type of selection is roulette wheel selection procedure which is partitioned into θ times and the size of the each partitioned is proportional to selection probability of each individual. New population can be generated by spinning the roulette wheel θ times and in every spin random selection of the chromosome is done from the current generation (Figure 6.4). Therefore higher selection probability is leads towards the generation of new individuals in the population.

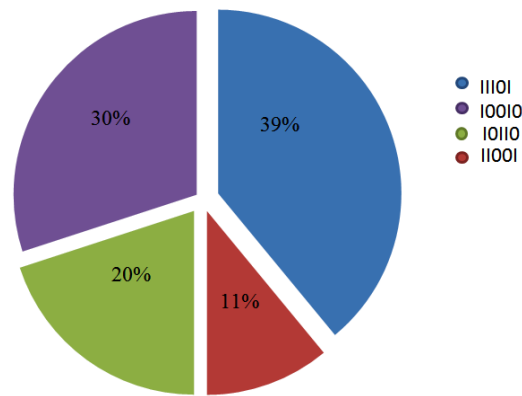


Figure 6.4 Roulette wheel selections. The selection probability for all four chromosomes is 0.11, 0.20, 0.30 and 0.39.

6.2.2 Particle swarm optimization

Kennedy et al. [248], proposed an efficient evolutionary algorithm which is based on swarm intelligent which is known as Particle swarm optimization (PSO). It is a population-based optimization algorithm imprinted from the simulation of social behaviour of bird flocking. Here in this algorithm population comprises of the number of particles (candidate solution) which flies in search space to find the out global optimum point. Each individual flies in the search space with specific velocity and carrying position, simultaneously each individual update its own velocity and position based on the best experience of its own and the social population [248]. The basic steps and mathematical modelling of Particle Swarm Optimization Algorithm has been discussed in previous chapter. In this chapter optimization algorithm will be used to evaluate the joint variables of various configuration of manipulator.

6.2.3 Grey wolf optimization algorithm

In this algorithm leadership of grey wolves are arranged hierarchal namely alpha, beta, delta and omega with the main purpose of hunting, encircling of victim, looking for victim, attack on victim. These strategies give intelligent and social behaviour of the grey wolf for the arrangement of their food source with the minimum labour. The starting steps of this algorithm are (a) finding of food source (i.e. victim), (b) chasing the victim and (c) approaching the victim. Thereafter confirmation of victim or food source, grey wolves encircles and harasses the victim so as to not lose the food source which is later finished with the killing. Searching for the victim represents the exploration ability of the wolves for the development of the algorithm and exploitation can be understood with the hunting of victim. Therefore the main theme of the algorithm is to updating the searching agents of their positions and calculation of fitness for all agents [263].

Different parameters of GWO are initialization of alpha, beta and delta, max iterations, searching agents, neighbourhood site selection and termination criteria. After initialization GWO follows certain steps such as,

- (1) Tracking, chasing and approaching the prey
- (2) pursuing the prey then enclosing and harassing the prey till it quite the movements
- (3) killing the prey

Mathematical modelling of wolves behaviour can be given as the best fitness value will be considered as alpha, then second and last best can be named as beta and delta. Other individual solutions can be considered as omega. Now encircling of the grey wolves can be calculated as equation (6.6)-(6.7),

$$\vec{D} = |\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)| \quad (6.6)$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \quad (6.7)$$

Where, t represents old iteration and $t+1$ is new iteration, \vec{X}_p is the position vector of victim, \vec{A} , \vec{C} are the coefficient vector, \vec{X} is the position vector of the grey wolf. Now the corresponding vectors can be calculated equation (6.8)-(6.9) as,

$$\vec{A} = 2\vec{a} * \vec{r}_1 - \vec{a} \quad (6.8)$$

$$\vec{C} = 2\vec{r}_2 \quad (6.9)$$

where \vec{r}_1, \vec{r}_2 are random number vector of $[0, 1]$ and \vec{a} decreases linearly from 2 to 0 through the complete iteration.

Now hunting behaviour of the grey wolves can be mathematically describes as, alpha wolf is the best individual together with the beta and delta which search for the potential location of the prey or this can be understand with the optimum location. Therefore keeping the positions of these wolves can be considered as best location and can keep in memory so as to update the old position with the comparison of memory. Therefore the concerned mathematical formulas can be given equations (6.10)-(6.12) as,

$$D_{\alpha} = |C_1 * X_{\alpha} - X|, D_{\beta} = |C_2 * X_{\beta} - X|, D_{\delta} = |C_3 * X_{\delta} - X| \quad (6.10)$$

$$X_1 = (X_{\alpha} - A_1) * D_{\alpha}, X_2 = (X_{\beta} - A_2) * D_{\beta}, X_3 = (X_{\delta} - A_3) * D_{\delta} \quad (6.11)$$

$$X(t+1) = \frac{X_1 + X_2 + X_3}{3} \quad (6.12)$$

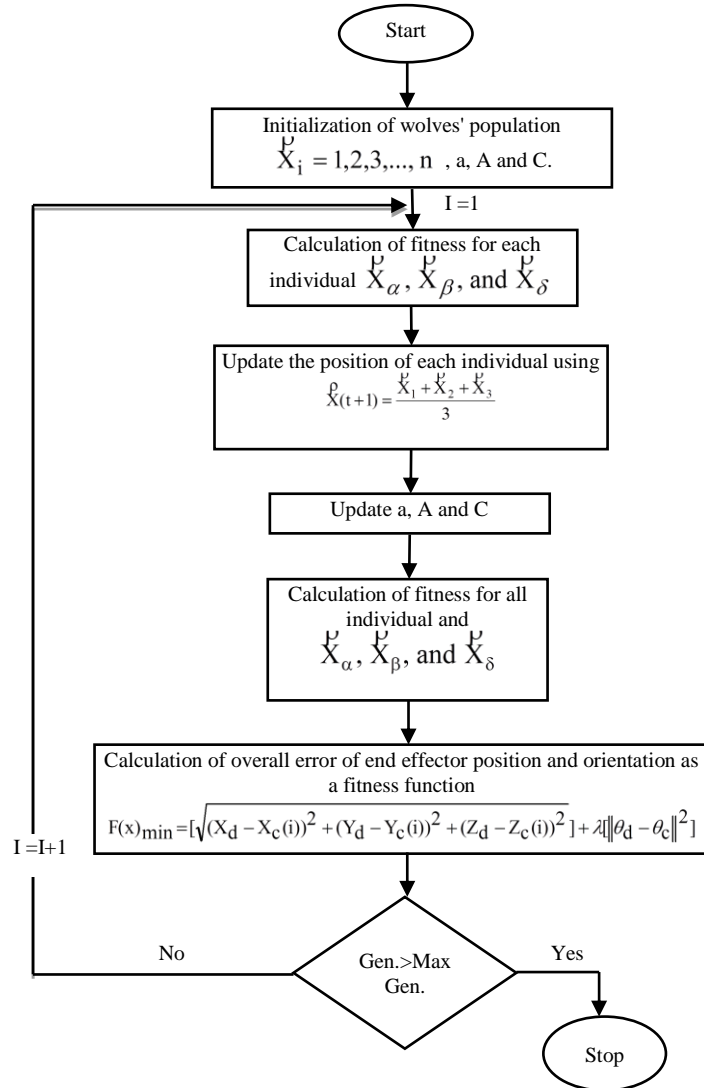


Figure 6.5 Flow chart for grey wolf optimizer

Therefore, the positions of alpha, beta and delta can be calculated from the above formula and the final position will be given by random place within the radius of search diameter. Alpha, beta and delta behaves like leader to get the position of the victim and according to this other wolves update their position randomly around the victim. Flow chart of GWO is presented in Figure 6.5 for the corresponding fitness function of overall error minimization of the end effector position and orientation.

Another efficient and famous metaheuristic algorithm based on population of bees is artificial bee colony (ABC) algorithm. ABC is also population based optimization technique like PSO which resembles the intellectual performance of honey bee swarm. The honey bee society comprises of three groups namely employed, onlookers and scouts. Onlookers bees gives the hint for the food source which later discover through employed bees and then scout bees search for new sources. In this system, the location of food source signifies a potential solution of the concern problem and nectar amount of food source resembles to the fitness of the related solution [254].

Teaching-Learning-Based Optimization (TLBO) is population based algorithm works on the effect of impact of a teacher on learners. In this algorithm population is considered as group of student where each student either learns from teacher called as teacher phase and they also gain some knowledge from other classmates or students that are learners phase. Output will be in terms of results or grades. In these algorithms different subjects for learner resembles the variables and learner results is equivalent to the fitness function for any problem and finally the teacher will be considered as the best solution achieved so far. There are several other population based methods have been successfully implemented and shown efficiency [249]. However, it is not always necessary that every algorithm can solve complex problem and provides best solution in fact it was mathematically proved by Wolpert et al. [264].

6.3 Development of novel metaheuristic optimization algorithm

This section introduces a different nature inspired algorithm, called Crab Intelligence based optimization (CIBO), for optimizing various unimodal, multimodal, separable, non- separable problems and for inverse kinematics solution of robot manipulators. The CIBO algorithm is based on the swarm, crossing and shell selection behavior of the crabs. Each crab represents the individual or candidate solution of the problem and fitness evaluation can be done by shell selection behavior of the crab. When all crab occupies the shell then it can be understand with the convergence of the solution which is evaluated by position vector of each crab. The best position will be kept in memory

and shorted to best fit value for the next search. This process of searching stops when it reaches to maximum iterations.

6.3.1 Crab intelligence based optimization algorithm

Novel effectual nature-inspired metaheuristic optimization technique grounded on crab behavior is proposed in this paper. The proposed Crab Intelligence Based Optimization (CIBO) technique is a population centered iterative metaheuristic algorithm for D-dimensional and NP-hard problems. The population of swarm represents the group of crabs which have social behavior as well as interact with their relatives and neighbors. Population of small group's moves over a D-dimensional search space collectively behaves like swarm. In this work positional vector of each individual which permits mutual movements of other individuals within the swarm is introduced. This algorithm considers three parts of crab behavior analysis: the first part is swarm behavior of crab, second part is related to crossing behavior and the third part is shell selection or recognition behavior of crabs. The mathematical modeling of the algorithm and the source of inspiration of the CIBO algorithm are explained in detail. In this work, the efficiency of the suggested algorithm with diverse individualities has been tested and then compared its performance with well-known population based metaheuristic optimization algorithms. In the later section this algorithm has been applied for inverse kinematics solution for 5R robot manipulator.

Most of the bio-inspired processes can be inferred in terms of computational cost. Social behavior indicates intelligence on crab which can be foundation for inspiration. Crabs have Intelligence for surviving the predators, looking for the right path for grabbing food and finally shelter for their safety.

Various studies of crab's life which could be perfectly suitable to develop an optimization algorithm have been done. The different behavioral studies of crabs are: (a) Swarm behavior, (b) Foraging behavior, (c) Predator Protection, (d) Shell selection (Recognition behavior) and (e) Crossing behavior. Among above mentioned behaviors only three of them namely swarm behavior, shell selection and crossing behavior of crabs have considered in this paper. Predator protection and foraging behavior has not been considered in this research which could be part of future work. Few species of crabs (e.g. *Mictyris guinotae*) populate on flat lagoons and form massive groups of several hundreds and sometimes hundreds of thousands of crabs. It has been observed that crabs show searching and swarm behavior as per biological experiments [265]. A front group of their swarm is driven by inherent turbulence that causes each individual always changing their position in entire search area. This inherent turbulence helps to find out local search points. Swarm behavior gives potential to cross water pools and

avoidance area wherein a single individual or group of individuals never tries to cross avoidance area; however, a huge swarm enters the water and crosses a lagoon without reluctance. In the swarm crossing prevention or avoidance area consists of forward facing and submissive tail. Backward or submissive tail simply keeps an eye on forward group. It has been assumed in here that there are two types of neighborhoods first one is optimistic interactive and another for observing and succeeding flock-mates. It has been observed that the swarm or group of swarm can mingle with their relatives or even with non-relatives due to their diffusion mechanism. Mostly crabs spend their whole mature lives on land, but to reproduce they choose sea and into it they discharge their developing larvae. The main reason of swarm behavior is collectively defend against predators, or to come together to eat stamped food resources.

The biological organisms interacted due to sharing or extracting relevant information from the environment. Environment produces various physical or chemical signals which could be extracted by organism through their evolved sensory mechanisms [266]. Terrestrial and aquatic organisms have chemical, vision and tactile sensors and among these sensors chemical sensors play crucial role to extract ecological information. These chemical sensors produce signals for presence of predators, convenience of food resources, and status of companions and availability of shell [267]-[269].



From the previous experiments it has been shown that chemical recognitions are the mediator for the behavioral study of crabs and other crustaceans. Most of the species gives attentions to adaptive behavior when exposed to odors to recognize the availability of shell [267]-[269]. It has also been observed that *P. longicarpus* spends more time investigation an empty shell.

Many species of crabs are dependent on shell produced by gastropods for their protection. They generally does not interfere on living gastropods shell, rather they compete with each other for gastropods shells that die by other organism or other means. The most important behavior of crabs is they continually search for new shells due to their body growth and for getting higher quality of shell than their current shell. When they leave the current shell other crabs occupies vacated shell. For better understanding of shell selection behavior we have followed some previous research of vacancy chain [270] - [271]. Synchronous and asynchronous, these are two distinguish category of shell selection which differ in their behavioral and ecological cost and benefits as shown in Table 6.1, this study gives social and alone search in straight divergence to shell relations comprising each individual for single shell selection. Synchronous vacancy chains arise when many crab stands in queue in front of shell. When bigger size crab occupies the vacant shell, others will wait as per their descendent order of their size. On the other hand, asynchronous vacancy chain will be occupied by

individual without making queue or any social interaction with others. In both cases, if shell quality is too low or damaged all individuals will discard the shell.

Based on these evidences, our objective was learning from the mechanism that underlie for each individuals as chemical recognition behavior for the selection of appropriate shell. Later we observed that whether the crabs are able to classify two different shells or target on the bases of their size, rank and shell quality. This behavior yields better clue for developing the algorithm.

Table 6.1 Shell selection

 <p style="text-align: center;">Asynchronous</p>	 <p style="text-align: center;">Synchronous</p>
Low potential for finding an optimal shell	Greater potential for finding an optimal shell
Easily reversible shell switching	Greater potential to get stranded in a sub-optimal shell
No risk of injury for predators	Predator competition requires time and energy; creates risk of injury
Decreased vulnerability to predators	Large crab aggregations

Recently H. Murakami et al [272], conducted an experiment with *Mictyris guinotae* for the swarm behavior and invading avoidance area. Through numerous experiments and field study of these crabs, they examined following observations of swarming behavior:

- 1) Moving swarm in the tideland has inherent turbulence and different velocities in each individual.
- 2) If the swarm faces water pool or avoidance area they do not enter into the pool until and unless they have dense population.
- 3) Each individual follows their predecessor.

So these above stated observations like crossing behavior of the swarm has been adopted for the development of the algorithm.

6.3.2 Methodology of CIBO algorithm

Social behavior specifies intelligence on crab which can be origin for inspiration. Crab intelligence for avoiding dangerous water pool and finding their shelter for the protection are the main aim of the development of the algorithm. As we are now well familiar with the swarm behavior of the crab but there are many other behavior like

foraging, predator protection, olfactory behavior etc. Besides their behavioral analysis some assumptions has been established which are as follows:

- 1) Swarm may be moving on shore or inside water confirming that the size and initial distance between them satisfy the minimum distance criteria.
- 2) It is important for swarm to cross water pool or avoidance area to get the shell.
- 3) Every shore or tideland contains unknown number of shell.
- 4) Shell acquisition may be synchronous or asynchronous depends on swarm.
- 5) Shell design parameters like volume, weight and geometry etc. have not considered.

6.3.3 Mathematical modeling

This section describes the mathematical modelling of swarm and shell selection behavior of crabs. In this model swarm having N -individuals moving in D -dimensional space. Where $N = \{1,2,3,4,\dots,N \max\}$. Boundary condition belongs to search space D . The location of i -th crab at the p -th step is given by equation (6.13),

$$L_{ip} = (x_1, x_2) \quad (6.13)$$

Where $x_1, x_2 \in D$ and $i \in I = \{1,2,3,\dots,M\}$

Position vector for each i -th individual at p -th step $P_v(i, p, v)$ with $v \in V = \{0,1,\dots,V-1\}$ and $P_v(i, p, v) \leq 1$. If $v=0$, the position vector $P_v(i, p, 0)$, will be present position vector and can be exemplified by equation (6.14),

$$P_v(i, p, 0) = z\{(R \cos \theta_{i,p}), (R \sin \theta_{i,p})\} \quad (6.14)$$

Where z is integer and R is the length of current position vector from origin. If $v \neq 0$ the vector will be defined by random number α $[0, 1]$ and angular random value δ $[-4\pi, 4\pi]$ by equation (6.15) as,

$$P_v(i, p, v) = z\{(R\alpha \cos(\theta_{i,p} + \delta)), (R\alpha \sin(\theta_{i,p} + \delta))\} \quad (6.15)$$

Position vector of shell can be obtained by equation (6.16),

$$S_{pos} = \rho_1 L_{i,p} + \rho_2 P_v(i, p, v) \quad (6.16)$$

where $\rho_1, \rho_2 \in [0,1]$, are positional constant. As we know that each crab interacting and sharing information for crossing and shell selection. Here we can define the suitability of shell on the basis of their size and distance. Suitability index for p -th position vector (x_1, x_2) , $(x_1, x_2) \in D$ would be given by equation (6.17) as,

$$\begin{aligned}
f(x_1, x_2, p) &\Rightarrow L_{ip} \neq (x_1, x_2) \\
\text{where } i &\in I = (1, 2, 3, \dots, M) \\
\text{and } v &\in V = (0, 1, 2, \dots, V-1)
\end{aligned} \tag{6.17}$$

Now setting up memory for updating the position vector (x_1, x_2) at the p -th position by equation (6.18) as,

$$m(x_1, x_2, p) = 0 \tag{6.18}$$

Updating the position of individuals may be synchronous and asynchronous will be based condition given below:

Condition 1:

Size of individuals $C_{i, \text{size}}$ of the i -th individual is based on random number γ $[0, 1]$.

$$C_{i, \text{size}} = \gamma$$

Condition 2:

Minimum number of individual in a single swarm will always be more than 5.

$$C_{i, \text{min}} \leq 5$$

Size of the i -th shell will be

$$S_{i, \text{size}} \geq \gamma$$

Condition 3:

Minimum distance between crabs $C_{i, \text{min_dist}}$ is also based on random value μ $[0, 1]$.

$$C_{i, \text{min_dist}} = \mu$$

Condition 4:

If number of i -th individual will be more than the number of shell present on shore then the shell selection will be on the basis of muscular power of crab M_p .

$$M_p = [0, 1]$$

The next position of shell for the i -th individual will be given by equation (6.19),

$$L_{i, p+1} = S_{\text{pos}, j} \tag{6.19}$$

Where j satisfies the condition for $v \in V = (0, 1, 2, \dots, V-1)$ by equation (6.20) as,

$$L_{i, p+1} \geq f(x_1, x_2, p) \tag{6.20}$$

Now the updated position will be given by equation (6.21),

$$P_{\text{new}} = \{(x_1, x_2) \in D, L_{i,p+1} = (x_1, x_2)_{\text{new}}\} \quad (6.21)$$

Now we can set updated memory by equation (6.22),

$$m(x_1, x_2, p) = P_{\text{new}} \chi \{(x_1, x_2) \in D, L_{i,p+1} = (x_1, x_2)_{\text{new}}\} \quad (6.22)$$

Now we can implement the end criteria for global point by equation (6.23),

$$C_i = S_i \quad (6.23)$$

Where C_i is number of individual and S_i is number of shell present, $i = (1, 2, \dots, M)$.

A. Hypotheses

This section first gives some basic and important definitions for validating the CIBO algorithms then in later we introduce the basic steps of algorithm.

Hypothesis 1:

A swarm $S \in \langle C_{i,\text{min}}^z \mid C_{i,\text{min_dist}}^z \rangle$ is a vector of z integers that signifies the feasibility of result and dependent upon random value generator.

Hypothesis 2:

Size of each individual $C_{i,\text{size}}$ and size of target or shell $S_{i,\text{size}}$ are two dependent parameters and represented by random value generator.

Hypothesis 3:

A position vector $S_{\text{pos}} \rightarrow R$ of a swarm is a measure of the fitness of the solution. Where R is set of real number.

Hypothesis 4:

Proportionality of $C_{i,\text{size}}$ and $S_{i,\text{size}}$ represents the global searching ability.

Hypothesis 5:

Muscular power of all individual can be defined by the random number generator $M_p = [0,1]$.

6.3.4 CIBO algorithm

This section elaborates the basic phases of the proposed algorithm. As shown in Fig. 1 there is N-number of swarm containing n-number of individuals that represents the search diameter and minimum distance in between them. Therefore, by generating huge search space, the swarm could cross the avoidance area and can reach to the suitable target. The shell is identified when swarm of crabs found local optimum. Identification of local optimum is based on minimum difference of past positions of crab.

In nature crabs are fighting for the better shell and the best shell will be occupied by stronger one. Once they reached to the better shell they test it for suitability and if it not then starts searching for next point (switching from local search to global search). In this approach each crab is initialized with value of his size (best found solution shell); when crab finds shell they test it for better size. If fitness value of local optimum – size of shell is better than size of crab it will occupy. The testing of shell and leaving it for smaller crabs is causing some delay where some crabs can escape the swarm. Because crabs cannot switch to global search alone they need minimum size of swarm. If enough crabs leave the shell and others are already testing for the size they can make global search, respectively migrate to other part of space and leave others behind.

It has been observed through various researches for different organisms that they spontaneously invade from avoidance area with the inspiration of rich food sources [265]-[272]. This behavior demonstrates the power of their neural processing. Another important assumption has been made that is swarm needs to cross the water pool or avoidance area and after crossing, there will be chances of getting their shells. This mentioned assumption is helping to find out global optimum point on the entire search space.

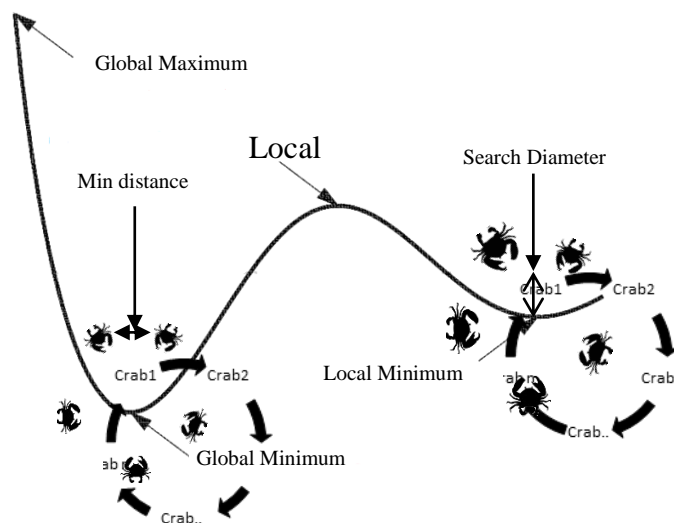


Figure 6.6 Representation of swarm and searching behaviour

These crabs have greater ability of recognition of better quality, size and emptiness of shell because of their chances of survival and protection from predation as explained earlier. In Fig. 6.7 represented the basic steps of the algorithm.

CIBO algorithm can be described briefly as:

- 1) Initialization of CIBO parameters like swarm considering N-individuals moving in D-dimensional space, boundary condition belongs to search space D, the initial location of each individual, diameter of local search, initial distance between crabs and initial swarm size.
- 2) Initialization of minimum number of individuals as per condition 2, compared with random value generations and random value for minimum distance and size of each individual as described in definition 1 if not then go to step 1.
- 3) Recognition of empty shell near the search space if not go to 2.
- 4) Fitness evaluation of each individual for their current position vector as explained in definition 2 and evaluation of suitability of shell as per definition 4.
- 5) Calculations of size of all individual.
- 6) Shell selection: comparing the size of all individual with the suitability of shell if individual size is not less than shell size then go to step 2.
- 7) Taking the shell if all individual satisfies the condition for selection of shell then the decision will be based on muscular power of each individual generated randomly as per condition 3.
- 8) Updating the value of position vector of individual.
- 9) If all individual occupies the shell then stop else go to step 2.

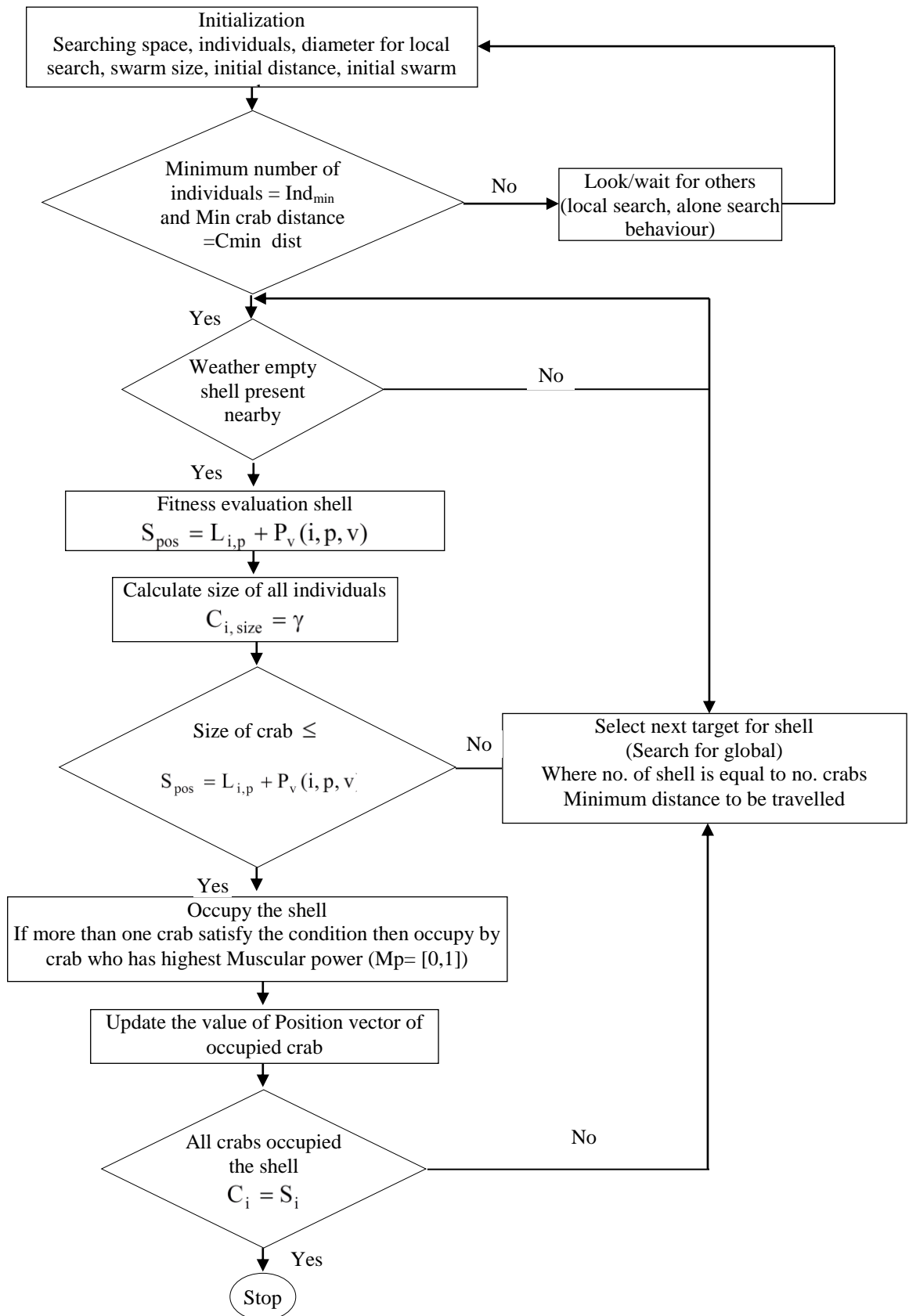


Figure 6.7 Flow chart for CIBO algorithm

6.4 Implementation for solving inverse kinematics

Inverse kinematics of any robot manipulator can generally be defined as finding out the joint angles for specified Cartesian position as well as orientation of an end effector and opposite of this, determining position and orientation of an end effector for given joint variables is known as forward kinematics. Forward kinematic having unique solution but in case of inverse kinematics it does not provide any closed form solution thus it is require to have some suitable technique to solve inverse kinematics of robot manipulator. In this section adopted algorithms are applied to find out the inverse kinematics of different configurations of robot manipulator. However, there are many optimization algorithms that can be fruitfully used to produce the desired results, but most of the population based algorithm does not have an ability to search for global optimum, and it gives slow convergence rate. On the other hand, developed optimization algorithm will be used to overcome the problem of Jacobian and other numerical based methods for inverse kinematic solution. Further mathematical modelling of objective function is discussed in detail to avail the inverse kinematic solution.

6.4.1 Mathematical modelling of objective function

Any Optimization algorithms which are capable of solving various multimodal functions can be implemented to find out the inverse kinematic solutions. In this section a general model of objective function is introduced for further implementation of optimization algorithm. In chapter 3, various configuration of robot manipulator has been considered for inverse kinematic solution. Now let's assume N-dof serial robot manipulator having θ joint configuration. Therefore, position and orientation of the end effector can be gives by equation (6.24) as,

$$P_e = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} \text{ or } P_e = \begin{bmatrix} n_x & o_x & a_x & X \\ n_y & o_y & a_y & Y \\ n_z & o_z & a_z & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.24)$$

Where R represents the orientation of the end effector $R \in SO^n$ and end effector position $P \in \mathcal{R}^n$ these are relative to the fixed reference frame at the base of manipulator.

Now the current position P_c of the end effector can be calculated from the equation (1), and for the given desired position P_d , considering this problem to find out minimum one feasible joint variable which gives the position of end effector at the target position coordinate. On the other hand, it is quite difficult to find the closed form solution like

conventional inverse kinematic derivation therefore without loss of the generality optimization approach is necessary.

Now optimization approach to solve inverse kinematic problem can be solve considering the error E between the current position and desired position of the end effector. This error is known as Euclidean distance norm and condition of optimization of objective function is when $P_c = P_d$. Therefore, for the specific pose P with corresponding joint variables θ^* with the major aim to minimize error E can be considered solution of inverse kinematic problem.

Above discussed error E defines two different search areas for the optimization process; (1) pose coordinates of the end effector and (2) feasible joint variables in the configuration space. Therefore the first point can be known as representation space or exploration and second point defines the solution space for the metaheuristic algorithms. Now these spaces can be mapped together to form a function in such a way that feasible joint variables can be transform onto pose coordinates. Finally the transformation between the forward kinematics and inverse kinematics could be the function for the evaluation of inverse kinematic problem similar to conventional method. Forward kinematic of the manipulator will be useful for the generation of objective function in terms of current position with unknown joint variables.

Further, previously defined error E will measure the difference between the current end effector pose with respect to the given goal. But to reach the desired position, will not only helpful for obtaining the joint variables. Hence, the end effector coordinate with relating to position will be known as position error E^P and on the second side desired orientation of the end effector could be helpful to reach to the exact point is known as orientation error E^O . Therefore, total error E will be the function of position error as well as orientation error is given by equation (6.25),

$$E = E^P + \mu E^O \quad (6.25)$$

where μ is constant weighting factor and can be defined as random number in between 0 to 1, the individual errors are $E^P \in \mathfrak{R}$ and $E^O \in \mathfrak{R}$. The weighting factor can also be set to zero that will yield the particular task requirements.

Finally objective function formulation for the inverse kinematic solution can be given by equation (6.26) as,

$$\begin{aligned} & \min E \\ & \theta \in q \\ & \text{sub. to } P_c = FK(\theta) \end{aligned} \quad (6.26)$$

$$q = \{\theta \mid g(\theta) \geq 0 \wedge h(\theta) \geq 0\}$$

Where g and h represents the limit of the joint variables of robot manipulator. The constraint g and h gives the maximum and minimum value of the joint variables so as to obtain a specific workspace. Hence these constraints can be formulated as equation (6.27),

$$\begin{aligned} g(\theta) &= \theta - \theta_{\text{lower}} \\ h(\theta) &= \theta_{\text{higher}} - \theta \end{aligned} \quad (6.27)$$

where θ_{lower} and θ_{higher} represents the lower and higher limits of the joint variables.

$$\theta_1 \in [\theta_{1\text{min}}, \theta_{1\text{max}}]$$

$$\theta_2 \in [\theta_{2\text{min}}, \theta_{2\text{max}}]$$

M

$$\theta_n \in [\theta_{n\text{min}}, \theta_{n\text{max}}]$$

The above general formulations of objective function for the solution of inverse kinematic problem provides direct solution with respect to joint variables. In this concept manipulator singularities are also avoided as compared to conventional methods and the objective function can easily be modified as per the task requirements.

6.4.2 Position based error

To optimize the joint angle of rotation of robot manipulator, one describes a fitness function that is composed of the difference between current position of and effector to the desired position that is known as positional function and second approach is to calculate joint angle error or orientation error. Manipulator accuracy can be measured by its ability to reach to the desired position within the workspace. Therefore, the distance between the current position and desired position should be zero so as to achieve higher accuracy. The difference between the desired positions to target position is known as position based error (P^E) as shown in Figure 6.8. To resolve the problem of inverse kinematic the position based error will be defined by the distance norm (Euclidean distance) as given in equation (6.28),

$$E^P = \|P^E\| = \|P_d - P_c\| \quad (6.28)$$

where $\|\bullet\|$ represents the Euclidean distance norm function \mathcal{R}^n .

The norm function defines the specific scalar metric of vector space elements, and can be defined in many ways. Commonly used norm on \mathcal{R}^n is above defined Euclidean distance norm or l_2 -norm and this norm can be given by equation (6.29),

$$\|P^E\|_2 = \sqrt{\sum_{i=1}^n |P_i^E|^2} \quad (6.29)$$

The second important norm is Manhattan norm or l_1 -norm, and can be defined as given in equation (6.30),

$$\|P^E\|_1 = \sum_{i=1}^n |P_i^E| \quad (6.30)$$

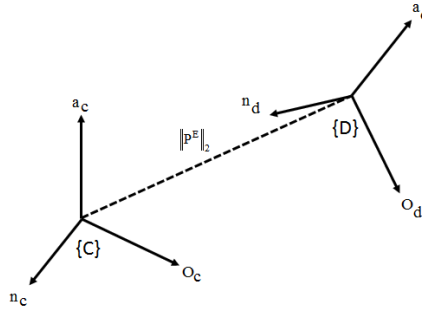


Figure 6.8 Position based error (P^E)

The l_1 -norm is widely used for the robot pose estimation under various levels of noise contagion in the sensory output and proves to be more robust and accurate as compared to other norm. Similarly, l_2 -norm is also used for the estimation of robot pose but it provides better convergence speed and mostly adopted for the inverse kinematic optimization. Therefore, in this work l_2 -norm is adopted throughout the dissertation.

6.4.3 Orientation based error

Orientation of the end effector of robot manipulator can be represented by the most common Euler angles method. Orientation between two different orthogonal Cartesian coordinate system xyz and uvw is generally described by the rotation matrix R , which is parameterized by Euler angles α , β and γ . Nonetheless, orientation angles can be obtained from the Euler angles representation but they undergo the problem of singularities. Therefore, to avoid the problem of singularity it has to be replaced with the Quaternion vector method. Quaternion vector method has already been discussed in chapter 3 which gives stable and compact representation of the kinematics representation. Hence using the formulations of quaternion vector method, orientation error could be formulized.

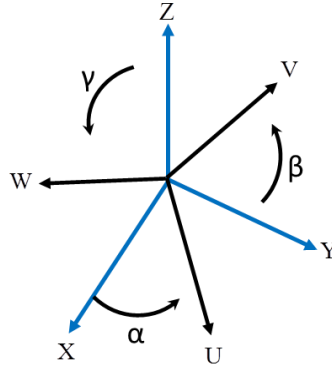


Figure 6.9 Orientation angle between two frames

Now the orientation error can be resolved using the current frame difference with the desired frame. Frame XYZ and UVW represents two different frames that are fixed at the same origin as shown in Figure 6.9. Therefore, the current frame XYZ should rotate to align with frame UVW so as to formulate for the orientation error R^E . Now the orientation error can be formulized on the basis of current frame rotation to desired frame. Orientation error can be given by equation (6.31) as,

$$R^E = R_d R_c^{-1} = R_d R_c^T \quad (6.31)$$

Where R_d rotation matrix of desired frame and R_c is the rotation matrix of current frame. Rotation matrix R^E defines the required rotation to obtain the end effect position to the desired coordinate within the workspace. It is required to find out the scalar function which can express the orientation of end effector error. As per Euler angles representation theorem, any orientation $\mathfrak{R} \in SO(n)$ will be equivalent to a rotation of fixed axis $k \in \mathfrak{R}^n$ through an angle $\psi \in [0, 2\pi]$. Therefore, to find out the equivalent axis representation quaternion vector method can be used. Now the quaternion expression can be given by equation (6.32) as,

$$H^e = R_{\text{rotation}}(\psi^e, k^e) = \left(\cos\left(\frac{\psi^e}{2}\right), \sin\left(\frac{\psi^e}{2}\right) k^e \right) \quad (6.32)$$

where, $s = \cos\left(\frac{\psi^e}{2}\right)$ is scalar component and $v = \sin\left(\frac{\psi^e}{2}\right) k^e$ is vector part of the quaternion method. ψ^e represents the angle of error and k^e is rotation axis. ψ^e represents the absolute error angle between the R_d and R_c . Therefore, Euler rotation matrix R^E can convert into the quaternion representation as given in equation (6.33),

$$s = \frac{1}{2} \sqrt{n_x^e + o_y^e + a_z^e + 1} \quad (6.33)$$

where n_x^e , o_y^e and a_z^e are the elements of rotation matrix R^E . Therefore from equation (6.32) and (6.33) orientation error can be given as equation (6.34),

$$E^O = 2 \cos^{-1} \left(\frac{1}{2} \sqrt{n_x^e + o_y^e + a_z^e + 1} \right) \quad (6.34)$$

6.5 Solution scheme of inverse kinematics problem

In this chapter, numerical optimization contexts based on metaheuristic algorithms are proposed to resolve inverse kinematic problem. Different configurations of the robot manipulator with several metaheuristic algorithms based solution is proposed in the next chapter. The major advantage of the implementation of optimization algorithms are compact and accurate solution of the inverse kinematic problem. Compared to the conventional methods like damped least square or Jacobian based methods it does not suffer singularity.

Metaheuristic algorithms are generally population based methods which are capable of solving various multimodal functions. The fitness or objective function can be given by the total error of the manipulator pose. Each individual represents the joint variable with in the population. The optimum set of joint variables can be obtained by using optimization approaches. In case of inverse kinematics problem, multiple solutions exist for the single position of the end effector so it is required to find out the best set of joint angle in order to minimize whole movement of manipulator.

As we know that objective function (pose error) is defined on the task space while the joint variables are the subset of configuration space of the manipulator. Therefore, objective function will be evaluated on the basis of forward kinematic mapping and total error obtained by the adopted method.

6.6 Summary

Inverse kinematics of any robot manipulator can generally be defined as finding out the joint angles for specified Cartesian position as well as orientation of an end effector and opposite of this, determining position and orientation of an end effector for given joint variables is known as forward kinematics. Forward kinematic having unique solution but in case of inverse kinematics it does not provide any closed form or unique solution thus it is require to have some suitable technique to resolve the problem for any configuration of robot manipulator. Therefore optimization based algorithms are quite fruitful to solve inverse kinematic problem. Generally these approaches are more stable and often converge to global optimum point due to minimization problem. Moreover, selection of appropriate optimization technique leads to global optimum results.

This chapter provides the basics of metaheuristic optimization algorithms and different types of metaheuristic optimization algorithms are described. Novel effectual nature-inspired metaheuristic optimization technique grounded on crab behaviour is proposed in this chapter. The proposed Crab Intelligence Based Optimization (CIBO) technique is a population centered iterative metaheuristic algorithm for D-dimensional and NP-hard problems. A general formulation of objective function for the inverse kinematic solution is derived and later is used for various configurations of the robot manipulator. In the derived objective function, optimization algorithms are used to minimize the end effector pose error for the given task. Besides using Jacobian matrix for the mapping of task space to the joint variable space, forward kinematic equations are used. Kinematic singularity is avoided using these formulations as compared to other conventional Jacobian matrix based methods. Different types of configuration of the robot manipulators have been taken for the kinematic analysis.

Therefore, inverse kinematic solution of various configurations of the manipulators with Euler wrist are solved computationally using several populations based metaheuristic algorithms. The adopted method is compact and efficient tool for kinematic analysis. In the result chapter inverse kinematic solution for adopted manipulator has been tabularised and comparison on the basis of mathematical complexity is made over other conventional based method.

Chapter 7

RESULTS AND DISCUSSIONS

7.1 Overview

The previous chapters have been essentially devoted to understanding the background of the research topic, various configurations of industrial robot manipulators for kinematic analysis, modelling of the inverse kinematic problem and solution techniques for selected robot manipulators. Although, the topic of inverse kinematic of robot manipulators is an intensively research topic, various new techniques are attempted by various researchers in order to make the solution method easier and/or faster depending upon the requirements. The method could be a single tool based or hybrid one. The requirements could be to simplify the solution method, obtain a precise result, or to make it suitable for the real time applications.

Modelling of the inverse kinematic problem using derived mathematical tools has been done in chapter 4. These mathematical modelling of kinematics is used to generate input data set for the training of ANN models and hybrid ANN's. The training schemes of the adopted ANN models have been discussed in chapter 5. In chapter 6, different optimization algorithms and their application to solve inverse kinematic problem is discussed in details. The formulation of objective function for the solution of inverse kinematic problem based on positional as well as orientation based error has been discussed in detail. Therefore, current chapter is summarized under the following objectives.

1. Inverse kinematic solution of the selected manipulators using conventional mathematical tools such as homogeneous transformation matrix and quaternion algebra is presented.
2. Inverse kinematic solution using ANN models and obtained results are presented.

3. Results of some selected manipulators using ANFIS models are presented and comparisons have been made with hybrid ANN models.
4. Results of inverse kinematic solution through adopted optimization algorithms and their comparative analysis is presented.

To have an incorporated perspective of the research work done and do legitimate examination, all the outcomes have been gathered and are introduced in this chapter.

7.2 Inverse kinematics solution using ANN

Typically, industrial manipulators are designed to perform various tasks in spatial as well as in planar space. The end effector/tool of manipulator is programmed to follow a specific trajectory to execute the desired task within the workspace. The control over each joints and links of the manipulator is required to reach to the desired position along with the control of end effector for explicit orientation and position within prescribed limit. Therefore, kinematic relationship of the joints and links plays crucial role to obtained desired position and orientation of the end effector. In case of forward kinematic analysis, joint variables are known which helps end-effector to reach at desired location while inverse kinematic requires orientation as well as position of the object within the workspace. In chapter 3, DH-algorithms and homogeneous matrix based methods are used for the kinematic derivation for various configurations of manipulator. Inverse kinematic solution of 4-dof SCARA, 6-dof PUMA and 5-dof manipulators are described without use of Euler wrist using quaternion algebra.

In this section ANN models like MLP, PPN, and Pi-Sigma NN are used to learn joint angles of robot manipulator and the data sets are generated through conventional methods like DH-algorithm, homogeneous transformation matrix, quaternion algebra method. Forward kinematic equations from chapter 3 are used to train the neural network models whereas in this chapter both forward and inverse kinematics equations are used to trained the neural network models. ANN monitors the input-output relationship between Cartesian coordinate and joint variables based on the mapping of data. Inverse kinematics is a transformation of a world coordinate frame (X, Y, and Z) to a link coordinate frame $(\theta_1, \theta_2, \dots, \theta_n)$. This transformation can be performed on input/output work that uses an unknown transfer function.

In this section results are produced for the manipulators starting from 3-dof planar to 7-dof anthropomorphic manipulator using conventional forward kinematic equations as well as ANN based models. In this chapter inverse kinematic solution for adopted manipulator has been tabularized and comparison on the basis of mathematical complexity is made over other conventional based method. MATLAB software is used

for the calculation of inverse kinematic solution of selected manipulators as presented in chapter 3. Further results of ANN presented in the following sections.

7.2.1 Result for 3-dof planar revolute manipulator using ANN

The proposed robot manipulator model is considered as 3-dof planar manipulator (see Figure 3.1). The length of the each links are $a_1 = 10$, $a_2 = 7$ and $a_3 = 5$. Let θ_1 is the angle between based and first link similarly θ_2 and θ_3 makes angle with second and third arm. Considering all joint angles limits 0 to 180 degrees.

Now using forward kinematic equation from chapter 4, training data for MLP, PPN and Pi-Sigma network has been obtained through MATLAB program. The generated sample data sets for of training adopted neural network models are given in Table 7.1.

Table 7.1 Position of end effector and joint variables

SN.	X	Y	θ_1	θ_2	θ_3
1	4.7023	9.4277	46.7776	152.3456	-139.1232
2	3.7386	10.8467	47.8992	157.9094	-145.8087
3	10.6173	0.2456	25.905	118.869	-84.774
4	5.8569	2.6275	32.9457	121.5943	-94.54
5	10.4862	6.0982	46.0634	148.5189	-134.5823
6	10.2722	4.9191	41.5963	141.1135	-122.7098
7	3.6785	4.4251	36.7329	124.6804	-101.4133
8	0.888	0.8301	27.8623	103.661	-71.5232
9	0.7141	9.8974	43.9037	140.343	-124.2467
10	8.2878	5.2346	40.8674	138.7558	-119.6233

Three different models have been taken for the validation of the results and the models are: MLP (Multi-layer perceptron), PPN (Polynomial perceptron network) and Pi-sigma network which are considered for the analysis of inverse kinematics problem, simulation studies are carried out by using MATLAB. A set of 1000 data sets were first generated as per the formula for this the input parameter X and Y coordinates. These data sets were basis for the training and evaluation or testing the ANN models. Out of the sets of 1000 data points, 900 were used as training data and 100 were used for testing for ANN models.

Back-propagation algorithm was used for training the network and for updating the desired weights. In this work epoch based training method was applied. The comparisons of desired and predicted value of joint angles of MLP model for 100 epochs have been represented in Figure 7.1 through 7.5. Where, Figure 7.1 represent the 2-3-2-3 configuration of network in which (a), (b) and (c) depicts the predicted angles respectively. Another two different configurations have been taken for the testing of network shown in Figure 7.2 and Figure 7.3.

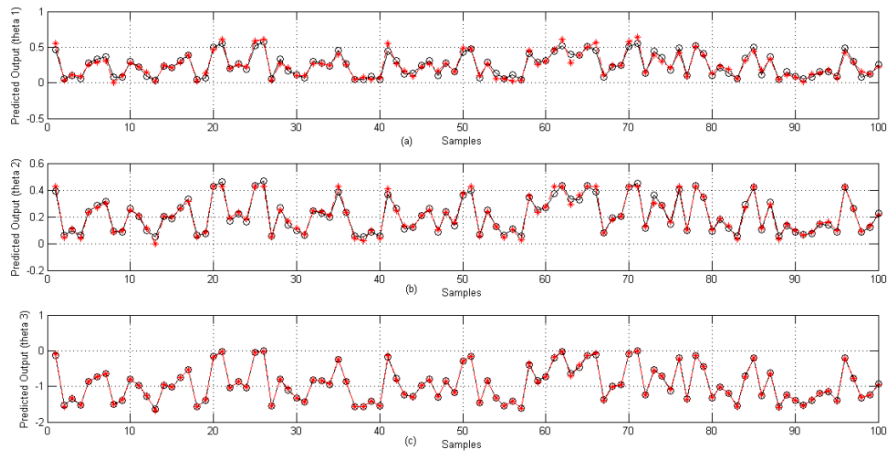


Figure 7.1 Comparison of desired and predicted value of joint angles for 2-3-2-3 configuration using MLP model

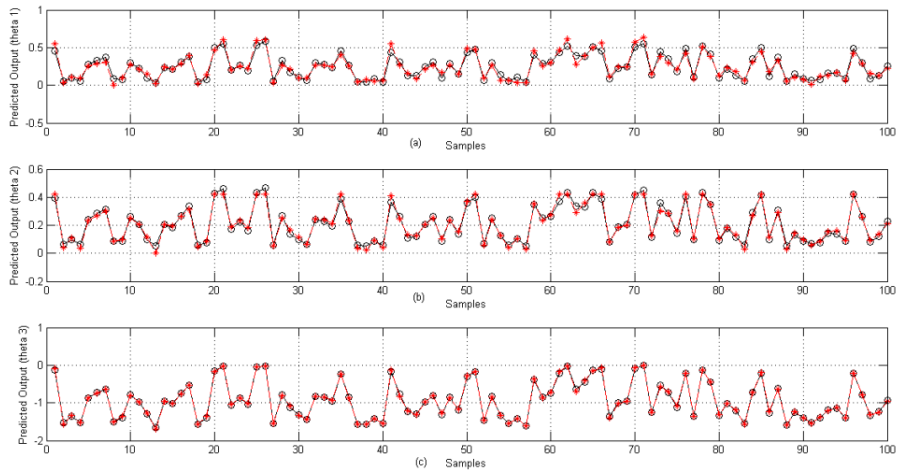


Figure 7.2 Comparison of desired and predicted value of joint angles for 2-4-4-3 configuration using MLP model

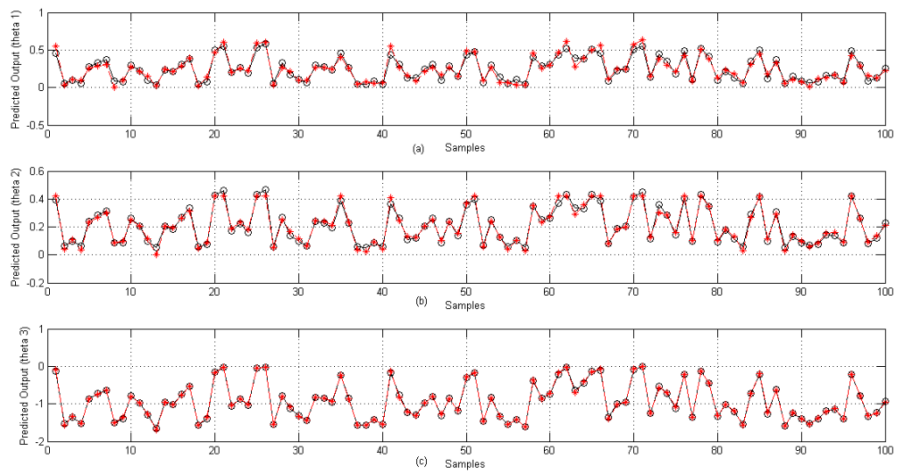


Figure 7.3 Comparison of desired and predicted value of joint angles for 2-5-5-3 configuration using MLP model

To test the stability of the MLP two other models i.e. PPN and Pi-sigma network models have been studied. Figure 7.4 (a), (b) and (c) shows the mean square error of joint angles using PPN network which gives relative poor result as compared to MLP model and in case of Pi-sigma network the obtained result is also poor to compare with MLP as shown in Figure 7.5 (a), (b) and (c).

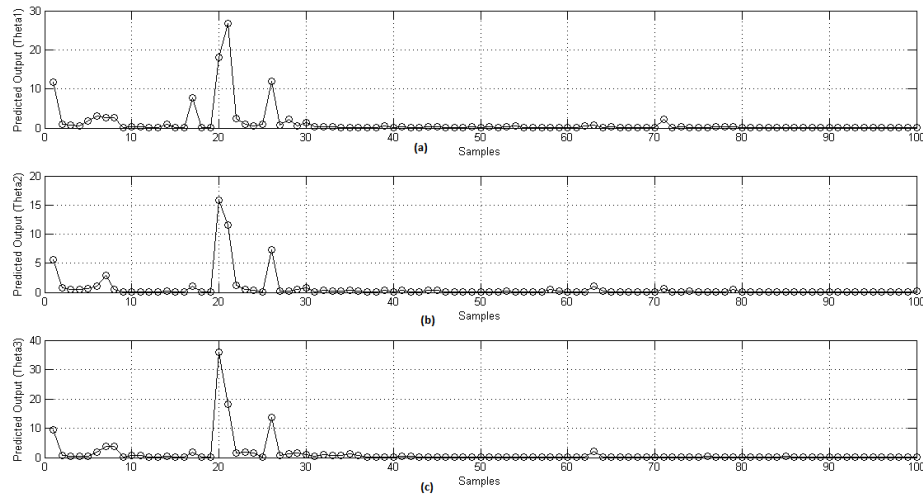


Figure 7.4 Mean square error for joint angles using PPN model

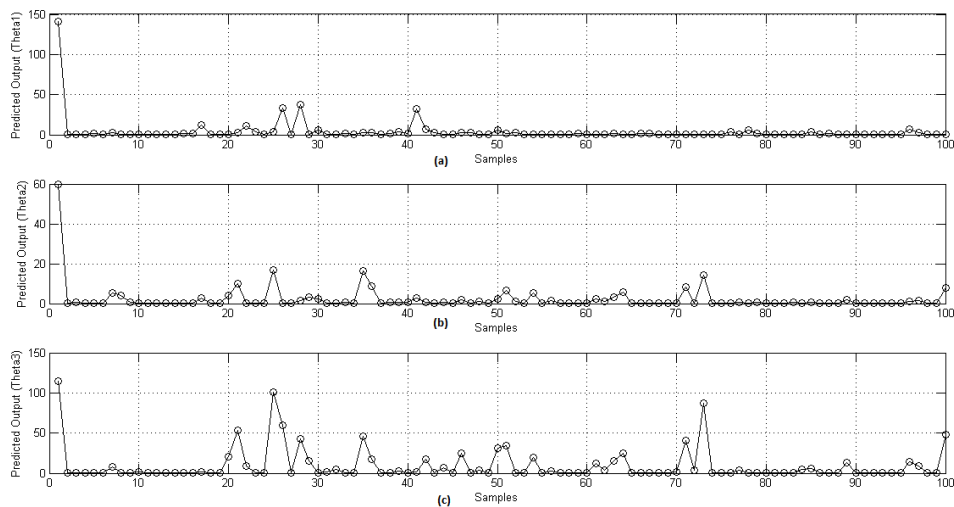


Figure 7.5 Mean square error for joint angles using Pi-sigma network model

7.2.2 Results of 4-dof SCARA manipulator

To validate the adopted MLP neural network, the proposed work is performed on the MATLAB Neural Networks Toolbox. The training data sets were generated by using forward kinematic equations from chapter 4. A set of 1000 data was first generated as per the formula for the input parameter X, Y and Z coordinates. These data sets were the beginning for the training, evaluation and testing the adopted MLP neural network

model. Out of the sets of 1000 data, 900 were used as training data and 100 were used for testing for MLP as shown in Table 7.2. The following parameters were taken,

Table 7.2 Configuration of MLPNN

Sl. Parameters	Values taken
1 Learning rate	0.99
2 Momentum parameter	0.01
3 Number of epochs	10000
4 Number of hidden layers	2
5 Number of inputs	3
6 Number of output	4
7 Target datasets	1000
8 Testing datasets	900
9 Training datasets	100

Table 7.3 Comparison between analytical solution and MLPNN solution

S.N.	Joints variables through analytical method				Joints variables through MLPNN method			
	θ_1	θ_2	d_3	θ_4	θ_1	θ_2	d_3	θ_4
1	-120	-130	0	-91.256	-113.119	-120.409	0.936	-12.364
2	-109.759	-19.739	0.150	10.360	-110.011	10.341	0.911	11.111
3	19.5195	-129.479	0.300	90.720	12.89	30.270	0.91	87.194
4	104.273	95.219	0.450	-101.081	-102.779	103.196	1.965	-107.274
5	91.039	100.95	0.600	129.441	-11.653	-100.118	3.059	99.393
6	-102.798	-18.698	0.750	51.801	102.522	-12.03	2.197	-56.701
7	-51.558	-28.438	0.900	-63.162	-12.383	-29.955	1.374	107.479
8	40.318	-1.178	1.051	52.522	42.239	-1.870	2.578	69.129
9	-63.071	27.917	1.201	-14.882	-62.089	29.784	3.785	-11.553
.
.
.
.
10	-78.837	66.657	1.351	-102.24	61.935	69.697	2.961	-108.831
11	99.597	-101.397	1.501	23.603	91.777	-109.611	3.059	29.041
12	-10.357	-17.137	1.651	114.964	-11.619	-29.525	0.036	113.256
13	117.117	26.876	1.801	-126.323	60.460	31.441	1.891	-101.481
14	16.876	53.616	1.952	152.684	-25.303	69.359	3.694	159.788
15	1.636	102.356	2.102	-34.045	2.1486	109.2791	2.552	-10.154
16	11.396	-106.096	2.252	151.405	30.998	-10.200	0.524	53.517
17	-25.156	25.835	2.402	119.765	-21.852	52.124	0.604	-91.757
18	-122.915	19.575	2.552	-81.126	-120.711	39.049	2.782	-81.748
19	-106.675	-91.315	2.702	-15.486	-91.577	-8.975	2.073	-10.548
20	-10.435	-100.055	2.852	100.846	-11.447	-110.902	3.454	100.664

Similar to previous work, back-propagation algorithm was used for training the network and for updating the desired weights. The formulation of the MLPNN model is a generalized one and it can be used for the solution of forward and inverse kinematics

problem of manipulator of any configuration. However, a specific configuration has been considered in the present work only to illustrate the applicability of the method and the quality of the solution vis-à-vis other alternatives methods. Table 7.3 gives the data for position of joints determined through analytical solution and that obtained from MLPNN model.

Table 7.4 Regression analysis

	Regression Sl. coefficient (r)	Mean square error	Epoch number	Resolution through adept one robot with smart controller user's guide	Resolution through MLPNN
1	0.99824	0.0076	2632	0.00078 ⁰	0.000778 ⁰
2	0.99519	0.00471	10000	0.00312 ⁰	0.003104 ⁰
3	0.99972	0.00028	10000	0.0033mm	0.003299mm
4	0.99928	0.00072	10000	0.047 ⁰	0.046966 ⁰

This is consistent with our title that it is a good approach to train the ANN with a good representative set of fixed targets positions instead of variable target positions for the learning process that will introduce noise in the cost function and may result in poor convergence.

The mean square curves, shown in Figure 7.6 through Figure 7.9 exhibit the proper description of the mean square error of trained network. As shown in result, the used solution method the chance of selecting the output, which has the least error in the system. Hence, the solution can be obtained with less error as shown in Figures 7.6 through 7.9 for the best validation performance of the obtained data with the desired data.

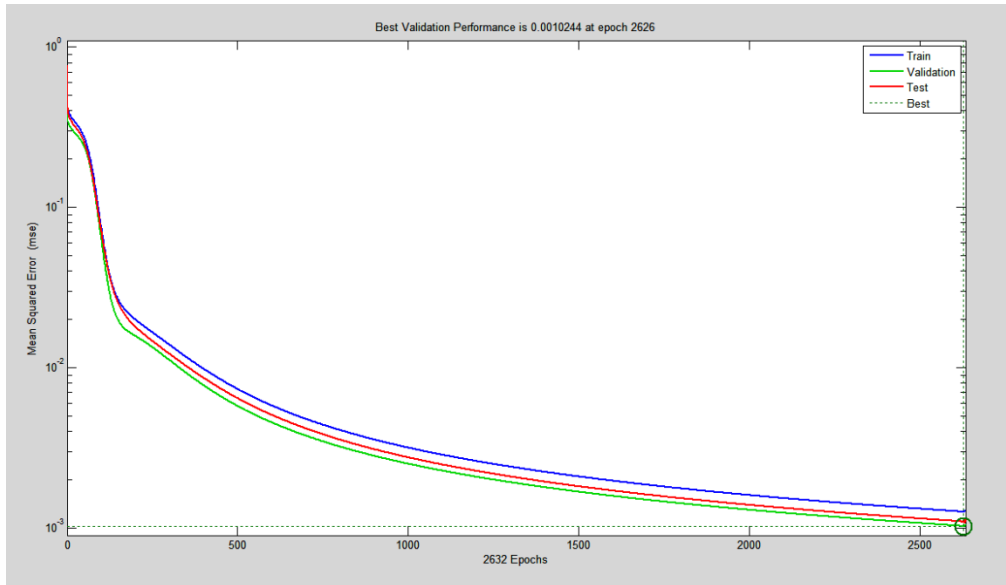


Figure 7.6 Mean square error for θ_1

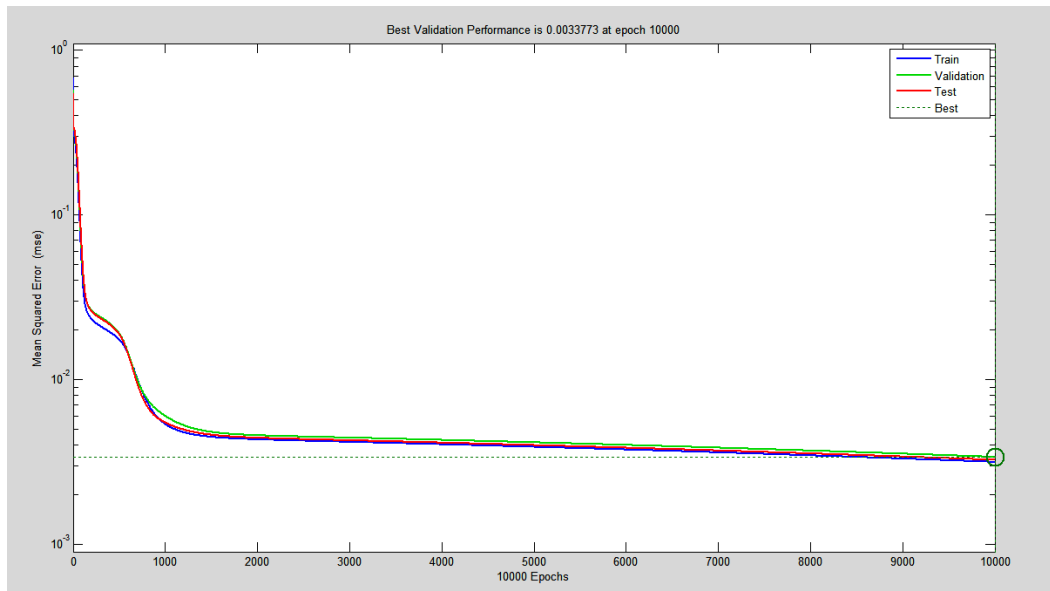


Figure 7.7 Mean square error for θ_2

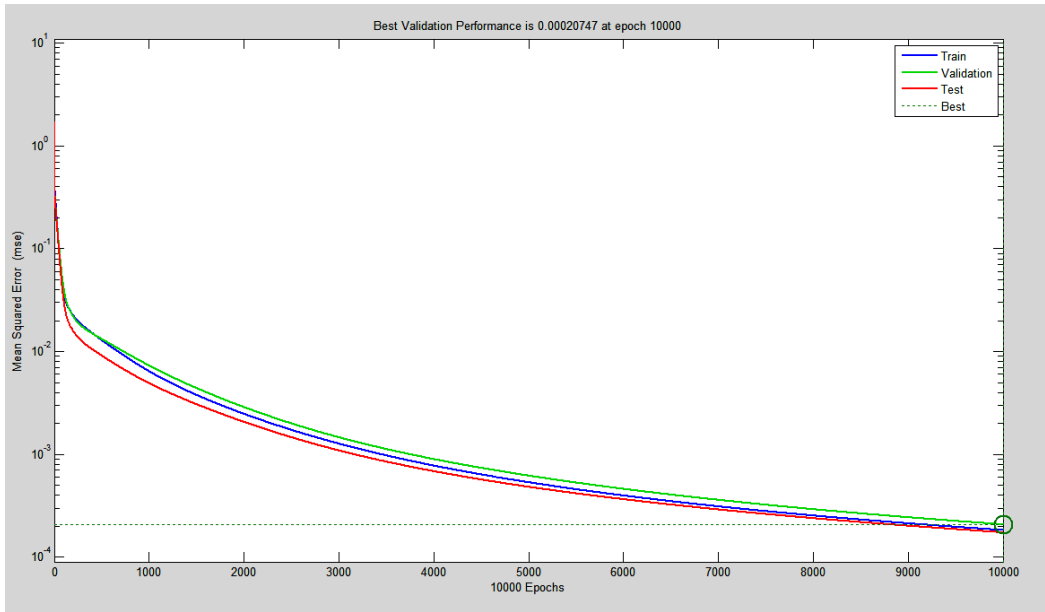


Figure 7.8 Mean square error for d_3

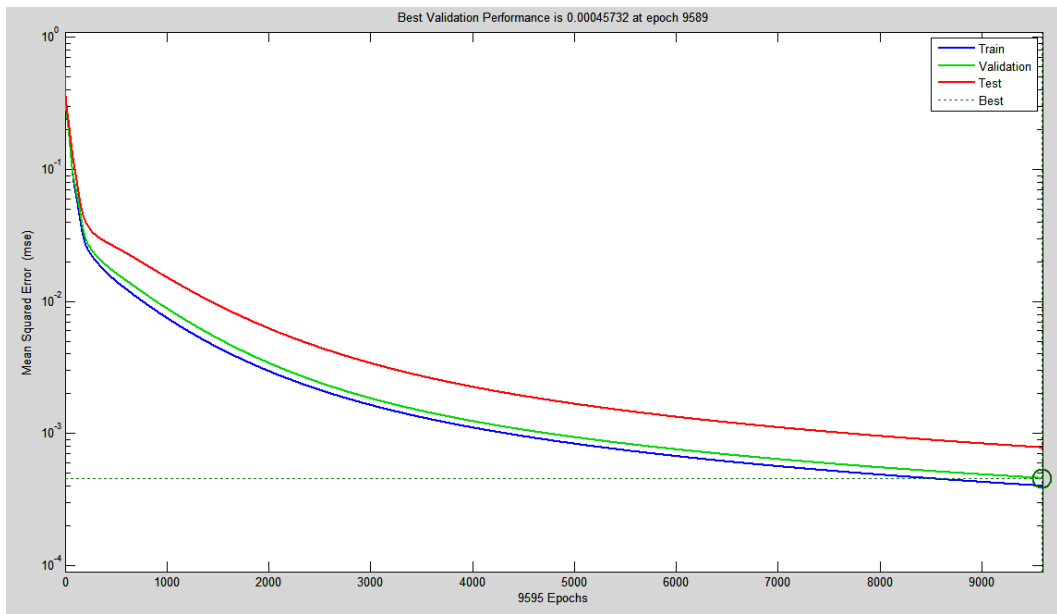


Figure 7.9 Mean square error for θ_4

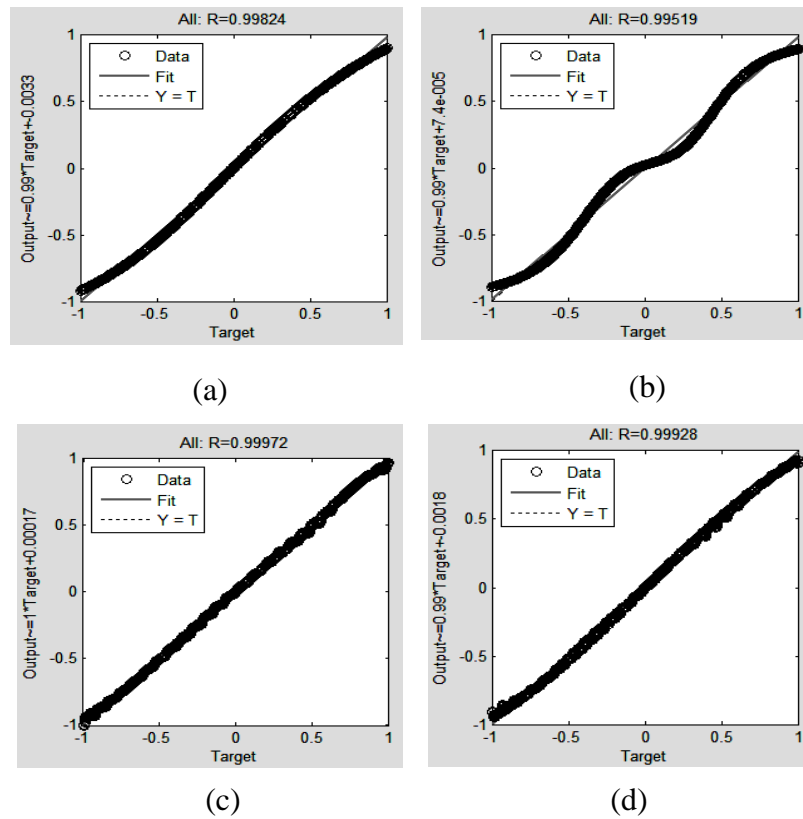


Figure 7.10 Graphical view of regression

Generalization tests were carried out with random target positions showing that the learned MLP generalize well over the whole space. From Table 7.3 it can understand that the mean square error for all joint variables is quite closer to zero. The regression coefficient analysis as per Table 7.4 that shows 99.9% matching for all joint variables which is acceptable for obtaining inverse kinematics of the SCARA manipulator. Resolutions of the AdeptOne SCARA robot given in Table 7.4 (obtained from AdeptOne robot with smart controller user's guide) are compared with the resolution obtained from the MLPNN model. Figure 7.10 represents the graphical view of regression analysis.

7.2.3 ANFIS results of 4-dof SCARA manipulator

The propose work is performed in MATLAB toolbox. The coordinates (X, Y and Z) and the angles (θ_1 , θ_2 , d_3 and θ_4) are used as training data to train ANFIS network with Gaussian membership function with hybrid learning algorithm. The training data for ANFIS model were taken from the Table 7.3 similar previous work. Table 7.5 shows configuration of ANFIS. Figure 7.11 through Figure 7.14 shows the validation curve for the problem of learning the inverse kinematics of the 4-DOF SCARA manipulator. Table 7.6 gives the average errors of joint variables using ANFIS and MLPNN. These

errors are small and the ANFIS algorithm is, therefore, acceptable for obtaining the inverse kinematics solution of the robotic manipulator.

Table 7.5 Configuration of ANFIS

Number of nodes	734
Number of linear parameters	343
Number of nonlinear parameters	63
Total number of parameters	406
Number of training data pairs	700
Number of fuzzy rules	343

Table 7.6 Comparison of results

Sl.	MSE of MLPNN	MSE of ANFIS
1	0.0076	0.00030124
2	0.00471	0.00002849
3	0.00031	0.00026932
4	0.00584	0.00039377

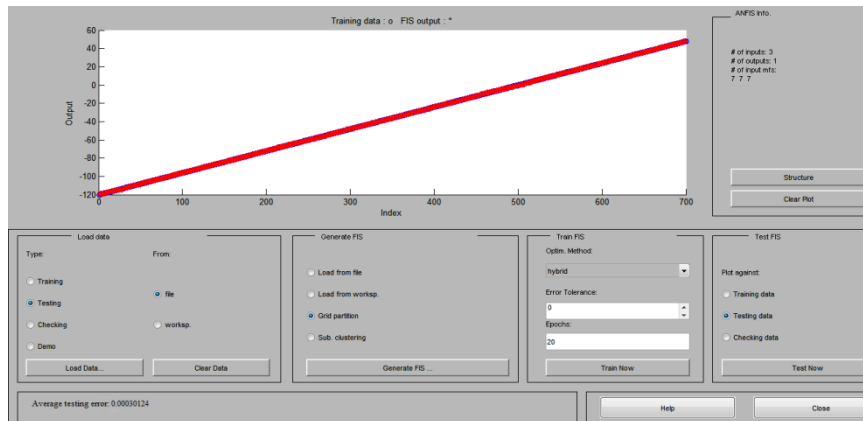


Figure 7.11 Mean square error for θ_1

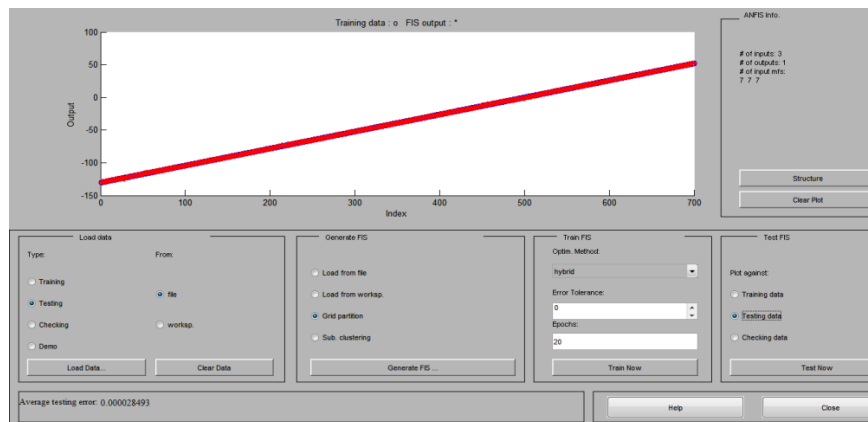


Figure 7.12 Mean square error for θ_2

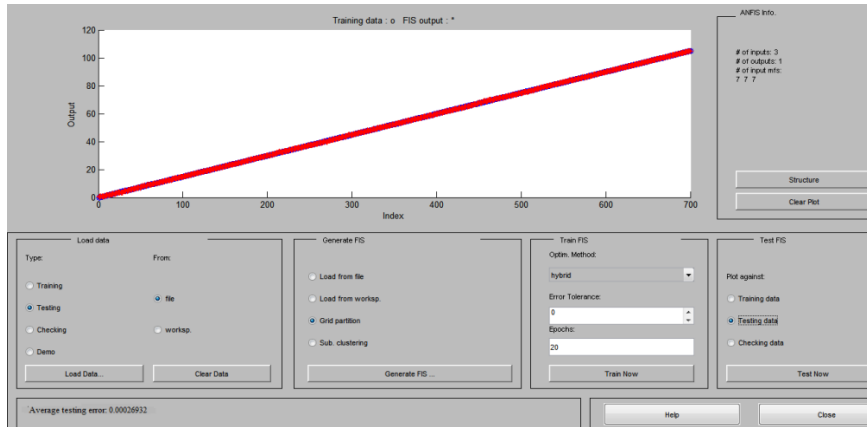


Figure 7.13 Mean square error for d_3

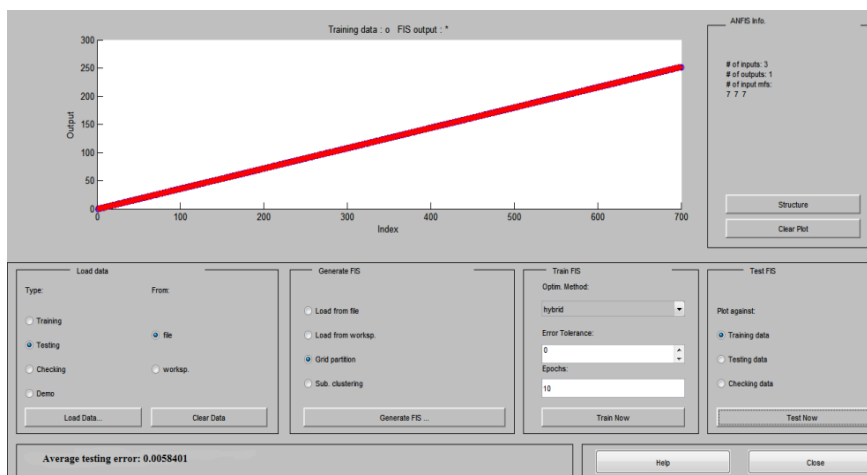


Figure 7.14 Mean square error for θ_4

7.2.4 Results of 5-dof revolute manipulator

Similar to SCARA manipulator proposed work is performed on the MATLAB Neural Networks Toolbox. 'Premnmx' function is used for preprocessing of input and output data. Then, the function 'newff' is used to create a feed forward network for inverse kinematics. Further, the same network is trained according to 'tansig' and 'logsig' transfer function. The training functions employed are 'trainoss' and 'trainlm', to validate the performance of MLPNN neural network for inverse kinematics problem. Then, the weights and biases are calculated for the network. To simulate the data corresponding to the task considered here, the new input data to the trained network are preprocessed with the 'traimnmx' function. Then, the outputs simulated by the trained network are post processed back using the 'postmnmx' function. The generated sample data sets for training adopted neural network models are given in Table 7.7.

Table 7.7 Desired joint variables determined through analytical solution

SN	Positions and joint variables determined through quaternion algebra							
	θ_1	θ_2	θ_3	θ_4	θ_5	X	Y	Z
1	112.5641	47.3165	8.2447	65.8373	39.8977	-186.6903	183.0670	-14.7039
2	153.1316	21.9812	126.9031	57.2629	30.4168	-92.6981	32.1423	157.3316
3	66.1779	143.2985	14.6124	73.6231	41.5228	-131.5420	-22.3866	-32.2155
4	57.6085	119.6396	104.5818	71.1946	33.8225	-10.7684	111.7435	77.4862
5	31.4308	2.9242	71.5757	63.0358	39.8749	64.7966	172.0372	151.5714
6	124.3702	116.7337	102.4999	53.1482	22.4807	-111.8590	-59.6708	60.8590
7	89.1765	13.1827	101.6747	80.3340	29.4704	-76.9533	96.2813	121.3505
8	5.6698	30.9685	57.2308	29.3079	29.6421	174.3873	107.6283	143.1839
9	131.5857	108.7086	92.8278	5.6664	36.4826	-104.6410	109.7511	40.5523
10	32.8579	102.3539	138.8770	26.4141	33.7466	146.4984	48.7416	54.4041
11	134.1878	70.4224	26.3511	82.2471	44.0566	-188.7864	15.4108	-53.9823

These data sets are used for training, evaluation and validating the neural network model for inverse kinematic solution. Out of the sets of 1000 data points, 900 were used as training data and 100 were used for testing for MLPNN as shown in Table 7.8. The following parameters were taken:

Table 7.8 Configuration of MLPNN

Sl. Parameters	Values taken
1 Learning rate	0.59
2 Momentum parameter	0.68
3 Number of epochs	10000
4 Number of hidden layers	2
5 Number of inputs	3
6 Number of output	5
7 Target datasets	1000
8 Testing datasets	900
9 Training datasets	100

Back-propagation algorithm was used for training the network and for updating the desired weights. The mean square curve shown in Figure 7.15 through Figure 7.19 in result, the used solution method gives the chance of selecting the output, which has the least error in the system. So, the solution can be obtained with less error. Figure 7.15 through Figure 7.19 shows the validation curve for the problem of learning the inverse kinematics of the 5-DOF manipulator. These errors are small and the MLPNN algorithm is, therefore, acceptable for obtaining the inverse kinematics solution of the robotic manipulator. Figure 7.20 shows the graphical view of regression with respect to number of epochs and it's almost gives 99.99%. In the next section, the comparison of MLPNN model with ANFIS has been presented.

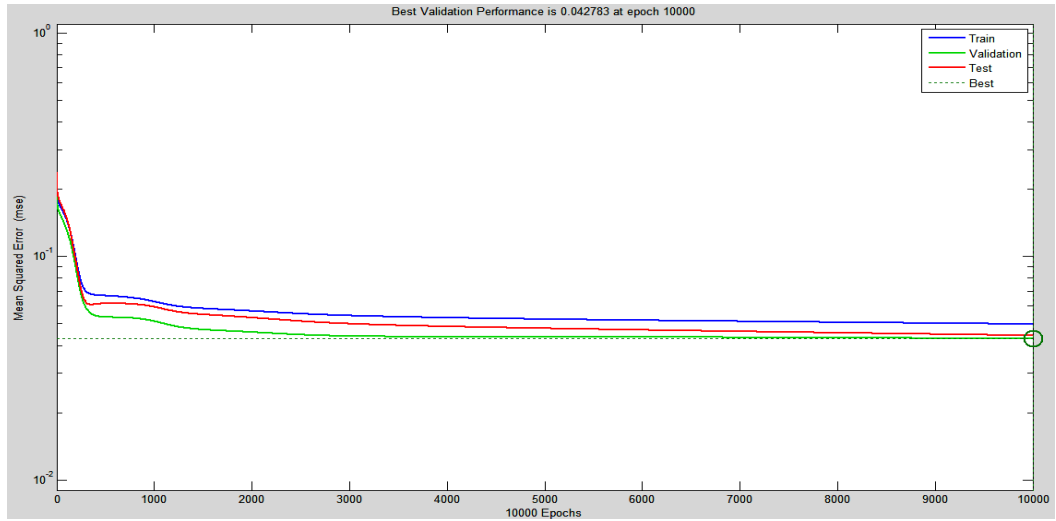


Figure 7.15 Mean square error for θ_1

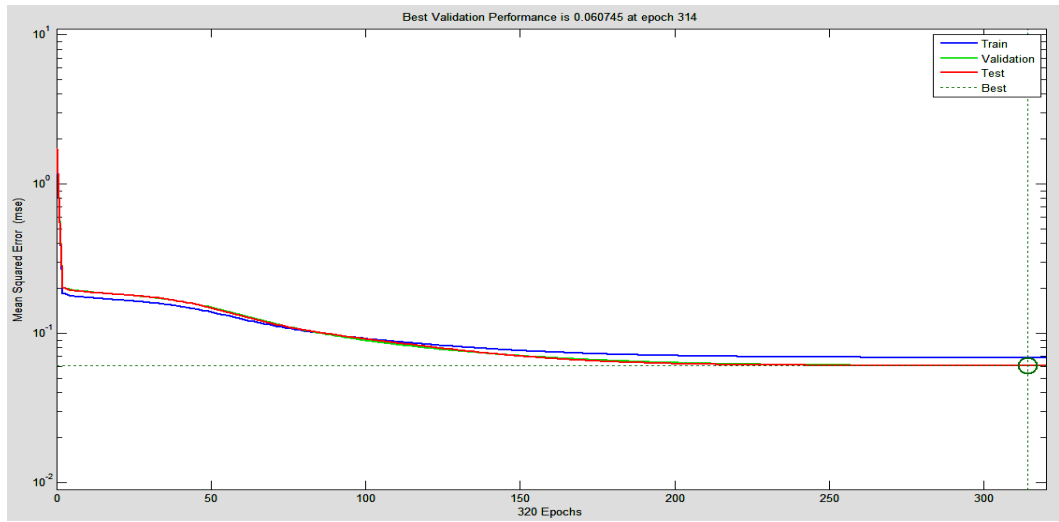


Figure 7.16 Mean square error for θ_2

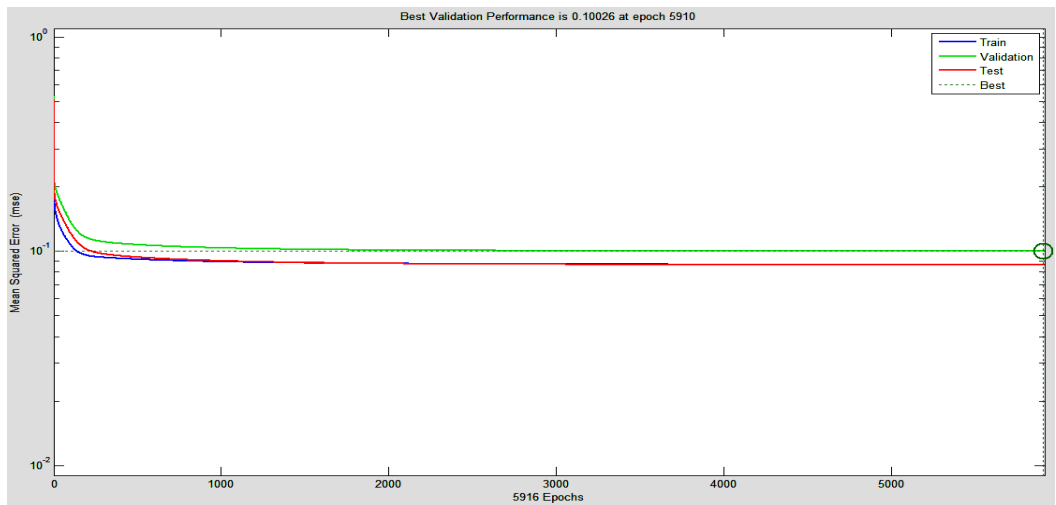


Figure 7.17 Mean square error for θ_3

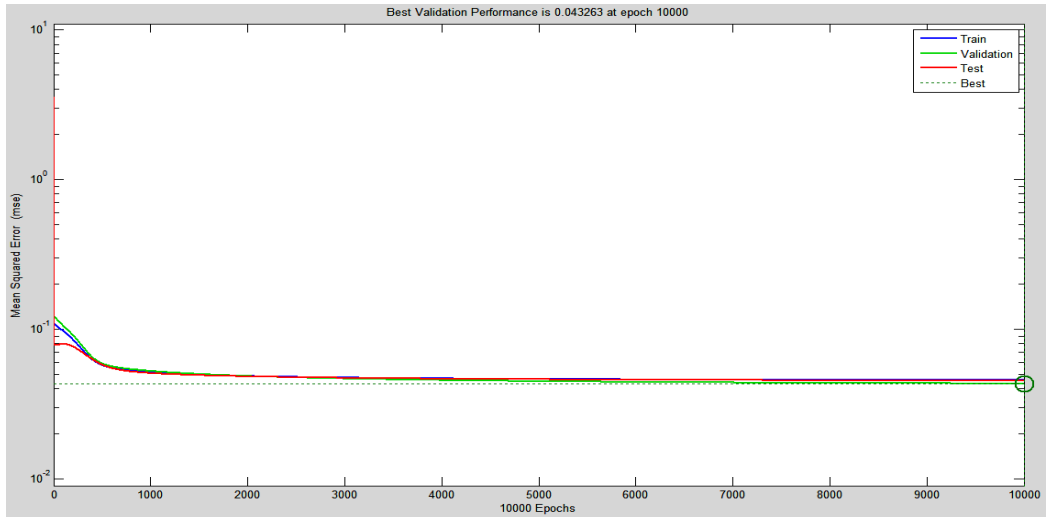


Figure 7.18 Mean square error for θ_4

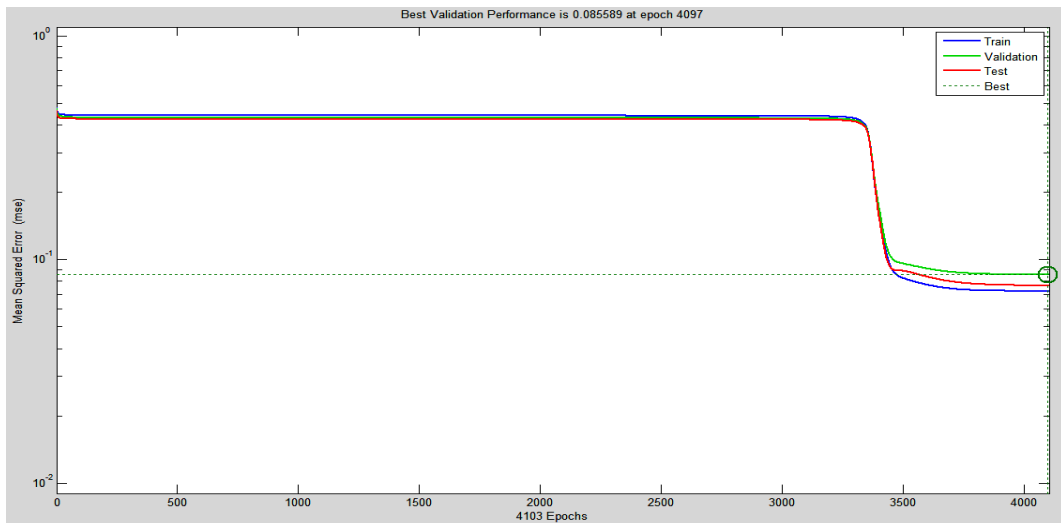


Figure 7.19 Mean square error for θ_5

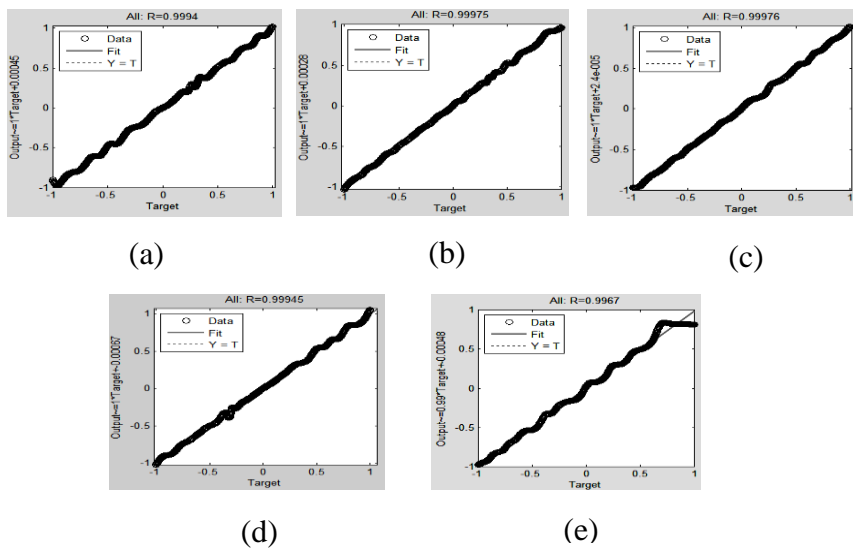


Figure 7.20 Regression coefficient plot for joint variables

7.2.5 ANFIS results of 5-dof revolute manipulator

In this section MLPNN result of 5-dof manipulator has been compared with the ANFIS results of inverse kinematic problem. Generated data sets for MLPNN training is used to training ANFIS network. Similar to previous work, the coordinates (X, Y and Z) and the angles (θ_1 , θ_2 , θ_3 , θ_4 , and θ_5) are used as training data to train ANFIS network with Gaussian membership function with hybrid learning algorithm. A set of 1000 data sets were first generated as per the formula for the input parameter X, Y and Z coordinates. Out of the sets of 1000 data points, 700 were used as training data and 300 were used for testing the performance of ANFIS. In the training phase, the membership functions and the weights will be adjusted such that the required minimum error is satisfied or if the number of epochs reached. At the end of training, the trained ANFIS network would have learned the input/output map and it is tested with the deduced inverse kinematics. Figure 7.21 through Figure 7.25 shows the difference in joint variables analytically and the data predicted with ANFIS.

Table 7.9 Configuration of ANFIS

Number of nodes	734
Number of linear parameters	343
Number of nonlinear parameters	63
Total number of parameters	406
Number of training data pairs	700
Number of checking data pairs	0
Number of fuzzy rules	343

Table 7.10 Comparison of results

Sl.	Average testing Error of MLPNN	Epoch Number MLPNN	Average testing Error of ANFIS	Epoch Number ANFIS
1	0.112475	10000	0.0035263	10
2	0.451253	10000	0.0029383	10
3	0.336321	10000	0.013536	10
4	0.258163	10000	0.0016652	10
5	0.321749	100000	0.00057395	10

Table 7.9 shows configuration of ANFIS. Figure 7.21 through Figure 7.25 shows the validation curve for the problem of learning the inverse kinematics of the 5-DOF manipulator. Table 7.10 shows comparison between the MLPNN with respect to ANFIS. Generalization tests were carried out with new random target positions showing that the learned MLPNN gives a deviation of 0.29599 of the error goal during the learning process and ANFIS gives 0.004448 average errors which is better than the

mean square error of MLPNN. The obtained errors are small and the ANFIS algorithm is, therefore, acceptable for obtaining the inverse kinematics solution of the robotic manipulator.

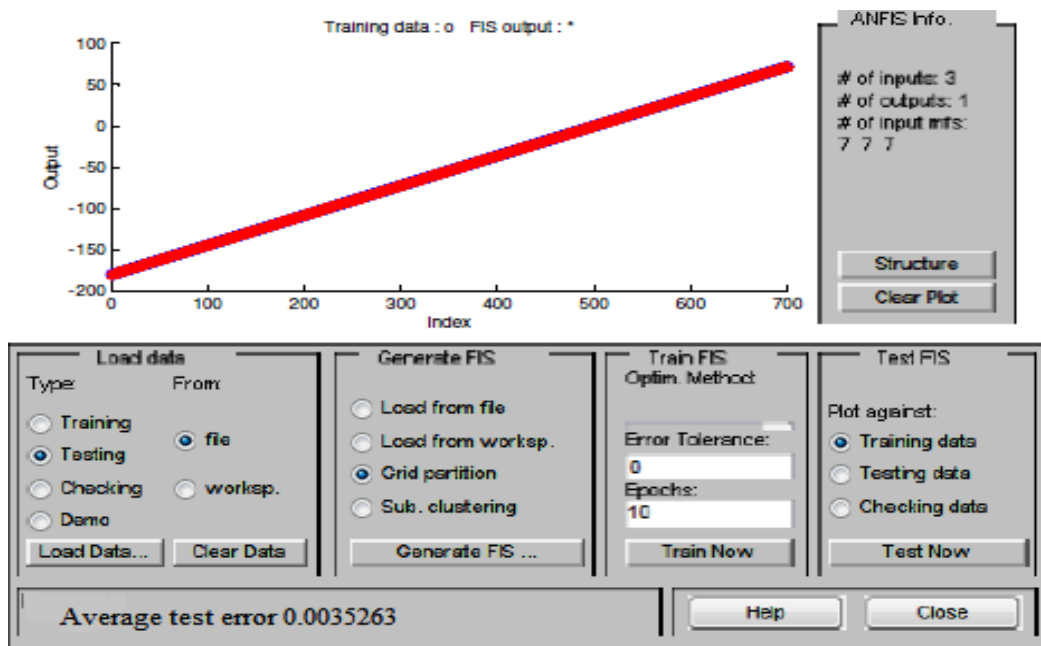


Figure 7.21 Mean square error for θ_1

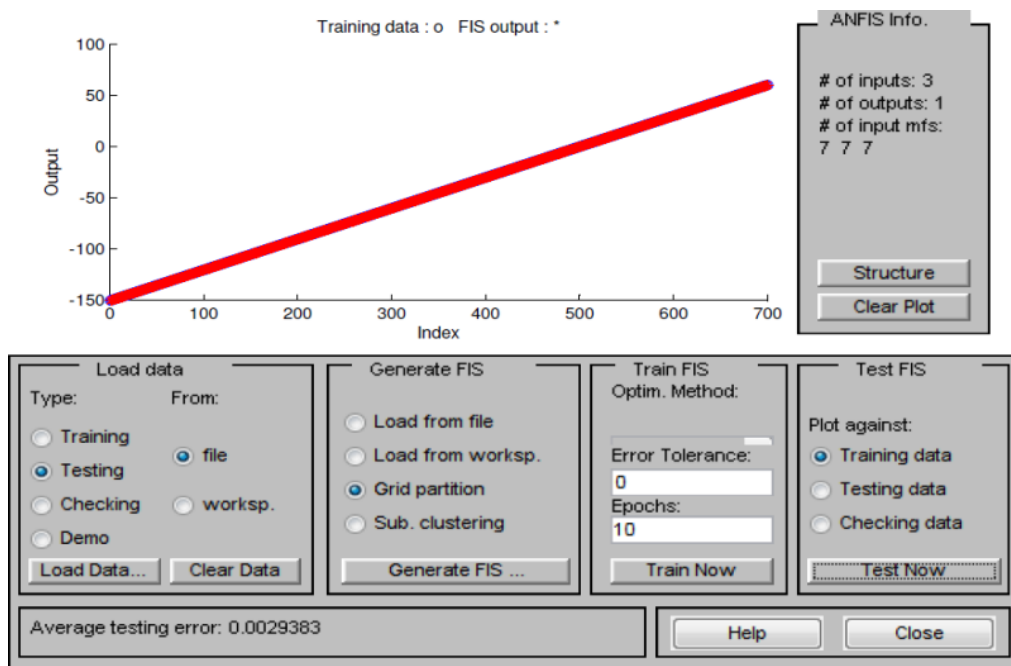


Figure 7.22 Mean square error for θ_2

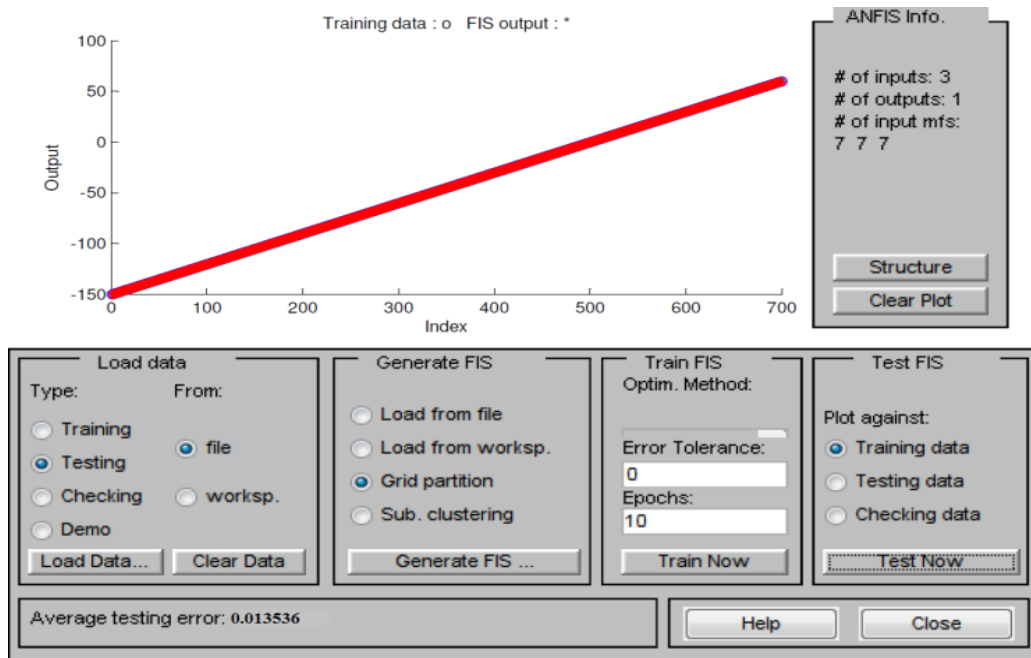


Figure 7.23 Mean square error for θ_3

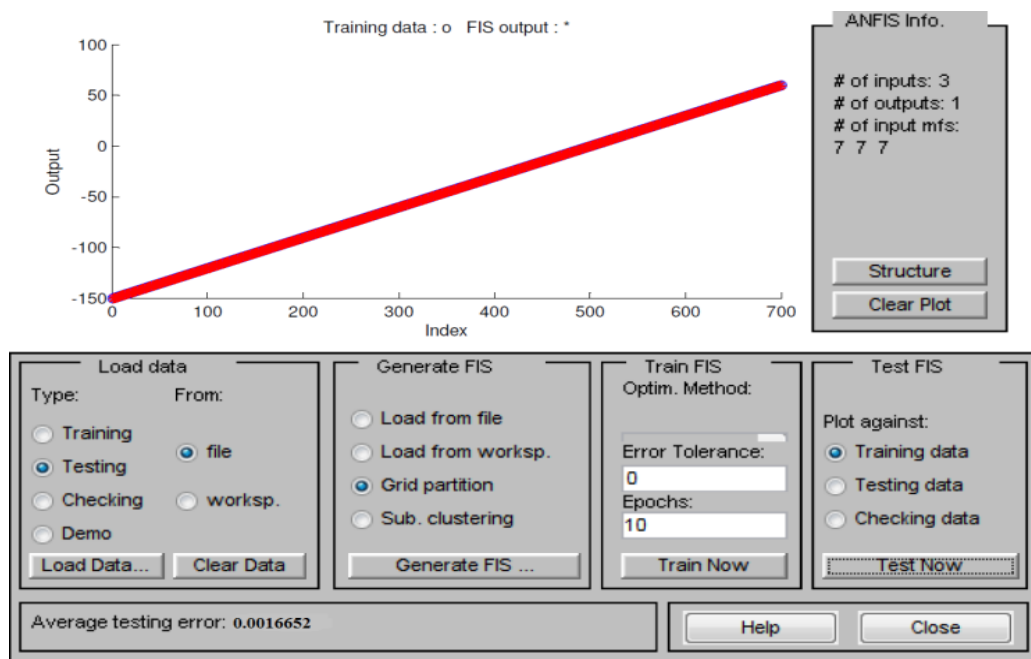


Figure 7.24 Mean square error for θ_4

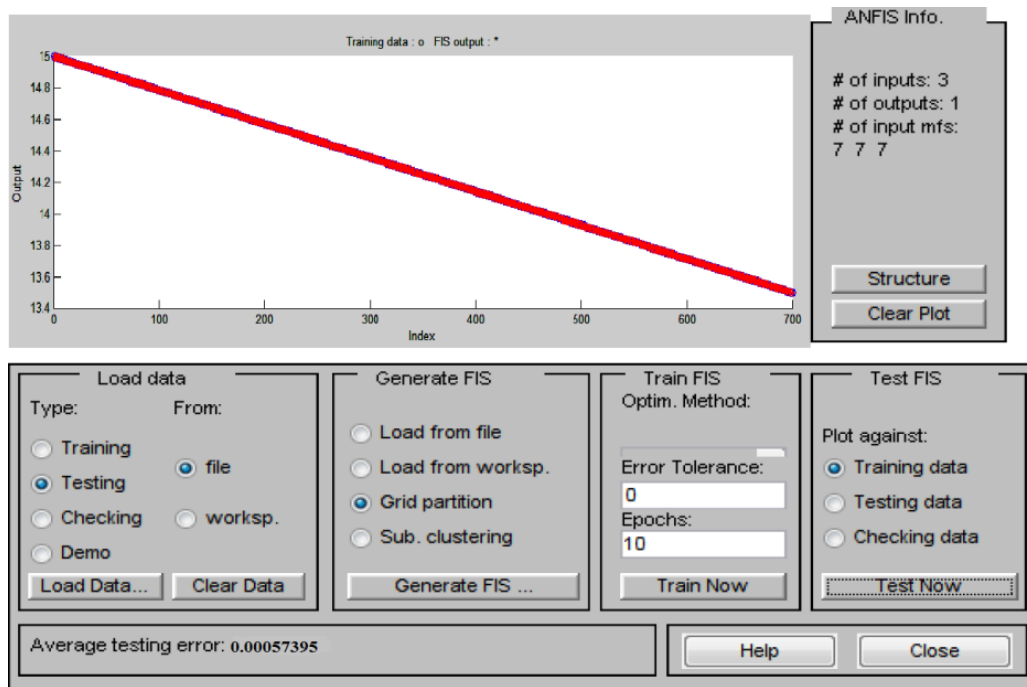


Figure 7.25 Mean square error for θ_5

7.3 Hybrid ANN approach for inverse kinematics solution

In this section metaheuristic algorithm like CIBO, PSO, GA, GWO, TLBO etc. are applied using weight and bias based optimization criteria and MLP neural network is used throughout this section. From the previous section neural network models are appropriate for solving inverse kinematic problem but the adopted models producing inappropriate mean square error for the different configurations of the robot manipulator. Therefore, hybridization of metaheuristic or population based algorithms method can be start with the introduction of the adopted algorithms with the specific model of neural network. The detailed discussions of inverse kinematic problem have been presented in chapter 4.

Hybrid ANN model are used for solving the inverse kinematic problem in the present work. Chapter 5 gives the detail discussion of hybridization scheme of MLP network with metaheuristic algorithms. The results obtained by solving the inverse kinematic problem for different configurations of 6-dof manipulator using hybrid ANN in MATLAB platform is presented in following sections.

7.3.1 Inverse kinematic solution of 4-dof SCARA manipulator

In this section, 4-dof SCARA manipulator is selected for the inverse kinematic analyses. SCARA robot manipulators are mostly used in light duty applications due to the high speed and precision of the manipulator. The detail kinematic modelling of this

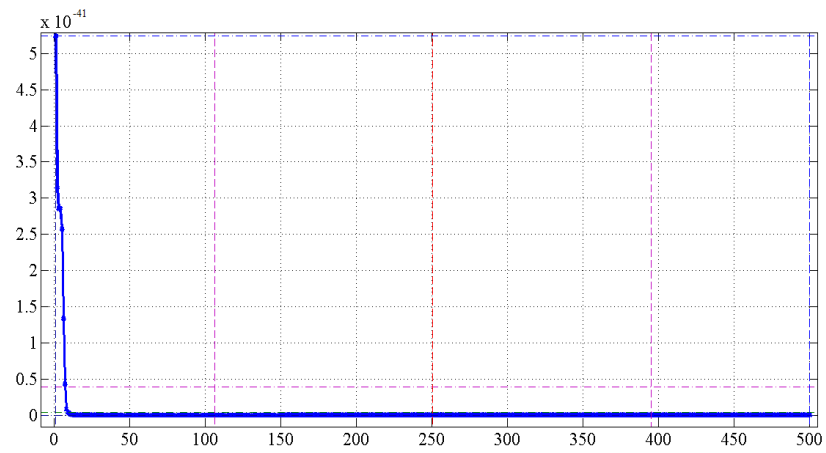
manipulator using conventional method is presented in chapter 4. Using forward kinematic equations from section 4.2.5 is used to create data sets for the training of MLPNN model. The results using back propagation algorithms with 2 hidden nodes for MLPNN is obtained in section 7.2.2. The present section gives the comparative analysis of inverse kinematic solution with the hybrid MLPNN technique. From section 7.2.2 the training of MLPNN model MATLAB neural network toolbox is used and later the obtained results are compared with the hybrid MLPNN method. A set of 1000 data sets are first generated as per the formula for the inputs X, Y and Z coordinate. The parameters for training MLP network is presented in section 7.4.2 (see Table 7.2).

Table 7.11 Mean square error for all training samples of hybrid MLPNN

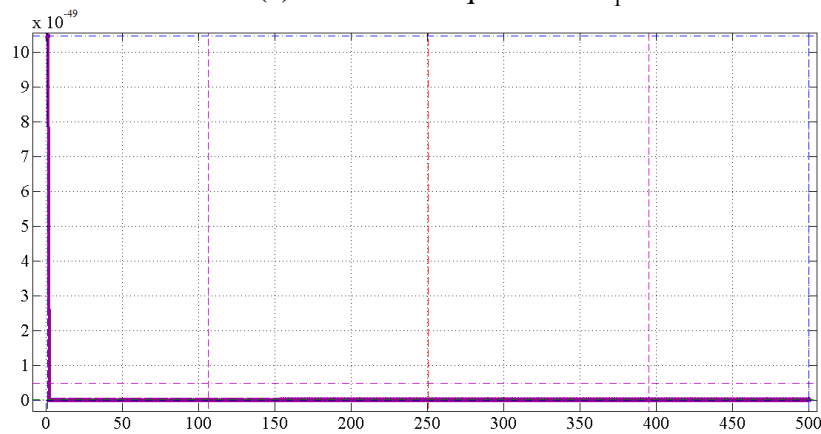
Output Algorithms	Number of Hidden Nodes										
	4	5	8	11	14	17	20	23	27	30	
θ_1	MLPBP	9.89e-3	2.52e-2	4.88e-1	9.67e-3	6.26e-1	1.69e-3	2.54e-1	0.51e-3	6.55e-2	8.29e-3
	MLPPSO	1.40e-6	2.08e-6	1.63e-8	0.54e-11	1.82e-16	7.90e-17	4.15e-15	5.70e-15	1.11e-19	4.16e-21
	MLPGA	1.05e-21	2.55e-23	9.09e-25	6.85e-21	5.21e-21	3.25e-29	7.89e-19	5.45e-18	6.58e-21	2.09e-21
	MLPTLBO	1.37e-07	1.00e-08	4.19e-11	6.58e-10	8.87e-09	3.96e-08	9.39e-11	6.56e-10	2.11e-13	5.61e-11
	MLPCIBO	1.65e-05	5.19e-07	1.85e-09	9.21e-11	2.22e-10	0.52e-09	8.47e-11	1.01e-13	7.67e-09	0.28e-11
	θ_2	MLPBP	1.23e-01	0.41e-02	4.96e-03	5.14e-01	6.85e-02	0.57e-03	4.26e-03	2.25e-02	6.94e-01
MLPPSO		0.09e-05	1.64e-09	2.73e-09	0.17e-11	0.34e-07	1.03e-12	2.24e-11	7.54e-12	8.16e-13	1.92e-11
MLPGA		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MLPTLBO		0.00	0.00	3.42e-39	6.44e-41	4.51e-33	5.48e-48	6.00e-39	7.88e-31	6.50e-32	0.89e-29
MLPCIBO		5.61e-11	2.32e-15	7.76e-11	8.28e-15	0.91e-16	4.86e-21	1.63e-14	4.73e-17	5.87e-16	9.39e-14
d_3		MLPBP	6.87e-01	2.53e-04	4.54e-02	7.62e-02	5.27e-01	3.42e-04	5.77e-03	2.57e-02	0.41e-03
	MLPPSO	7.74e-05	5.55e-07	3.26e-10	4.72e-11	0.23e-09	9.28e-11	7.03e-12	4.67e-13	8.52e-14	3.77e-16
	MLPGA	0.00	0.00	0.00	8.27e-49	9.15e-51	7.92e-49	7.00e-49	2.59e-39	3.71e-37	4.74e-29
	MLPTLBO	4.86e-11	8.47e-12	2.19e-13	8.67e-13	2.22e-11	9.99e-21	7.80e-19	7.86e-19	4.16e-15	7.66e-16
	MLPCIBO	1.52e-10	2.80e-09	7.14e-11	8.88e-11	0.46e-09	1.88e-13	2.22e-16	3.64e-11	5.96e-12	3.43e-11
	θ_4	MLPBP	8.93e-02	4.26e-01	2.45e-02	1.99e-04	7.61e-01	5.67e-02	0.76e-04	5.07e-03	4.37e-02
MLPPSO		2.27e-05	2.78e-07	4.11e-08	3.27e-09	9.07e-11	4.83e-16	6.49e-14	4.44e-11	6.43e-19	9.73e-11
MLPGA		6.74e-11	2.09e-29	3.77e-41	4.00e-31	0.00	0.00	0.00	0.00	0.00	0.00
MLPTLBO		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MLPCIBO		8.47e-12	4.75e-11	2.26e-14	6.03e-13	8.67e-21	7.58e-19	5.59e-15	6.66e-15	8.64e-26	0.04e-21

In this section, different numbers of hidden nodes are used and numbers of hybrid algorithms such as MLPPSO, MLPGA, and MLPTLBO MLPCIBO etc. have been compared with the MLPBP algorithm. Hybrid scheme has been presented in section 5.2, using the scheme of hybridization training of MLNN model is presented. The weight and bias of MLPNN model is optimized with the selected optimization algorithms. The parameters of the optimization algorithms have been chosen randomly. After the optimized training of the MLPNN model, the trained network is used to calculate the inverse kinematic of the SCARA manipulator. After calculation of the inverse kinematic solution the mean square error is obtained with the comparison of actual solution and desired solutions. Best mean square errors for all joint variables are

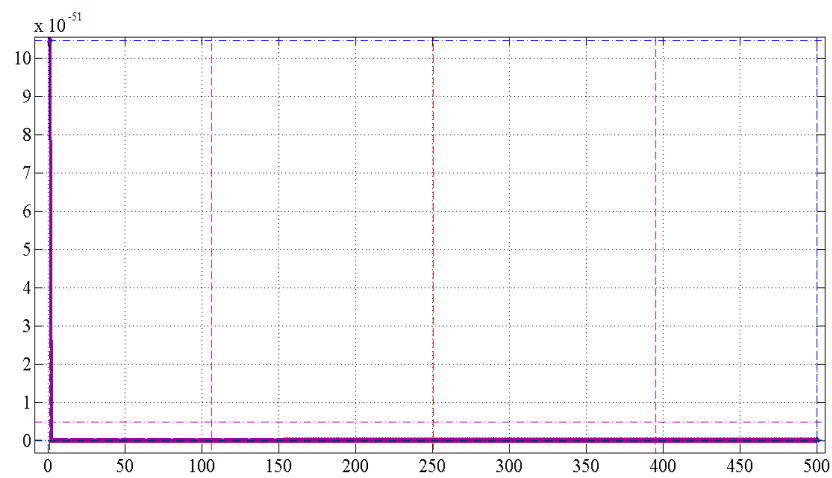
presented in Figure 7.26. Mean square error for all joint variables and comparison of different algorithms are presented in Table 7.11.



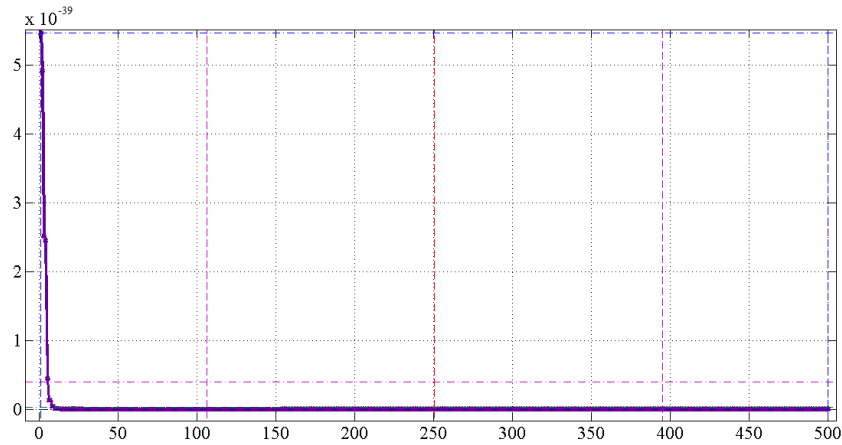
(a) Best mean square for θ_1



(b) Best mean square for θ_2



(c) Best mean square for d_3



(d) Best mean square for θ_4

Figure 7.26 (a), (b), (c), (d) and (e) are mean square error curve for all joint angles using MLPGA.

7.3.2 Inverse kinematic solution of 5-dof manipulator

In this section, 5-dof revolute manipulator is considered for the kinematic inversion. This manipulator is extensively used in industries as well as in research work. The inverse kinematic solution for the adopted manipulator is performed on the MATLAB R2013a. From the previous research work the adopted MLPNN models perform poor. Therefore, performance of hybrid ANN model and evaluations have been made. For the training of MLP network, MATLAB Neural Networks Toolbox is used (see section 7.2.4). ‘Premnmx’ function is used for preprocessing of input and output data. Then, the function ‘newff’ is used to create a feed forward network for inverse kinematics. Further, the same network is trained according to ‘tansig’ and ‘logsig’ transfer function. The training functions employed are ‘trainoss’ and ‘trainlm’, to validate the performance of MLP neural network for inverse kinematics problem. Then, the weights and biases are calculated for the network.

To simulate the data corresponding to the task considered here, the new input data to the trained network are preprocessed with the ‘traimnmx’ function. Then, the outputs simulated by the trained network are post processed back using the ‘postmnmx’ function. The training data sets were generated by using forward kinematic equation from chapter 4. A set of 1000 data sets are first generated as per the formula for the inputs X, Y and Z coordinate. The generated data sets are used to train the MLP network. The parameters for training MLP network is given in Table 7.8.

The abilities of several different hybrid algorithms such as MLPPSO, MLPGA, MLPTLBO and MLPCIBO have been compared. In the training phase, the weights and biases will be adjusted such that the required minimum error is satisfied or if the

number of iteration reached. At the end of training, the trained MLP network would have learned the input/output map, and it is tested with the deduced inverse kinematics. The results obtained through the MLPBP have been described in section 7.2.4. There is no specific tuning of the associated parameters are considered. The parameters for the PSO algorithm is discussed further and for the rest of the algorithm parameters are selected randomly without any tuning.

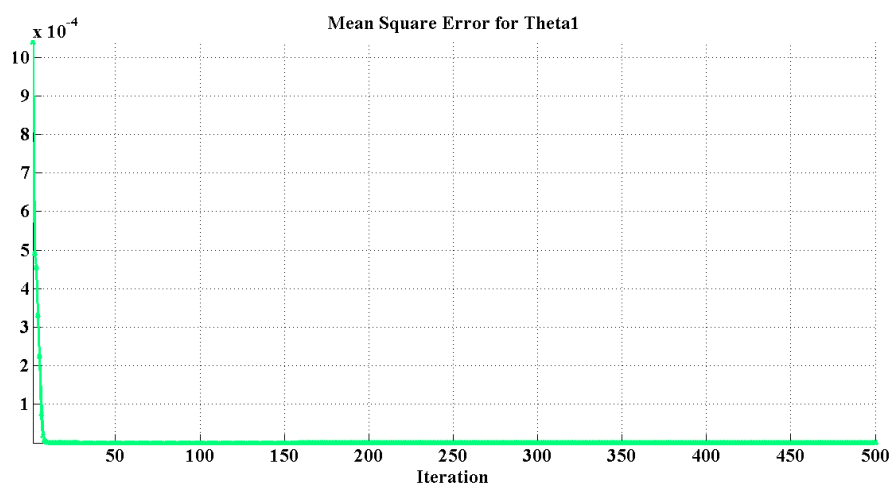
The initial approximations of every particle have been chosen randomly in the range of [0, 1]. For MLPPSO, maximum and minimum inertia weights are decreasing linearly from 0.9 to 0.4. C1 and C2 are set to 2; r1 and r2 are two random numbers in the interval of [0, 1] and the initial velocities of particles are randomly selected in the interval of [0, 1]. Finally the population sizes for each algorithm are 100. From section 7.2.4, Table 7.9 gives some of the desired data for the position of joints determined through analytical solution, which will further be used to calculate the MSE of MLPBP, MLPPSO, MLPGA, MLPTLBO and MLPCIBO.

Table 7.12 Mean square error for all training samples of hybrid MLPNN

Output Algorithms	Number of Hidden Nodes										
	4	5	8	11	14	17	20	23	27	30	
θ_1	MLPBP	1.25e-2	2.63e-3	9.10e-2	7.76e-2	3.62e-1	2.96e-1	4.45e-2	1.15e-1	7.15e-1	9.92e-1
	MLPPSO	2.04e-9	4.80e-9	2.36e-9	1.45e-8	2.28e-9	8.09e-12	5.51e-12	6.07e-14	2.07e-25	5.61e-25
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MLPTLBO	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MLPCIBO	2.56e-52	6.91e-51	0.00	0.00	3.33e-49	1.25e-53	0.00	0.00	0.00	1.82e-50
θ_2	MLPBP	2.32e-3	1.14e-2	5.69e-2	6.56e-3	7.58e-1	7.75e-3	5.62e-1	3.96e-1	7.89e-2	9.23e-1
	MLPPSO	1.90e-6	2.46e-12	3.37e-9	1.71e-20	1.43e-8	2.30e-13	3.42e-15	8.45e-15	9.09e-17	2.29e-12
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MLPTLBO	9.33e-19	2.64e-20	4.52e-18	1.66e-19	5.31e-18	6.84e-17	4.95e-19	8.61e-17	8.44e-19	1.09e-21
	MLPCIBO	4.16e-16	1.21e-17	6.65e-19	7.17e-18	9.89e-17	3.75e-25	2.52e-15	5.62e-19	4.14e-19	3.52e-24
θ_3	MLPBP	5.56e-2	3.35e-3	1.69e-3	8.26e-1	6.45e-2	4.24e-2	6.44e-1	3.85e-3	1.14e-1	9.10e-1
	MLPPSO	8.42e-7	1.68e-6	4.62e-12	5.27e-12	1.32e-9	1.68e-10	1.30e-11	1.41e-14	1.25e-15	2.87e-21
	MLPGA	6.12e-15	9.47e-16	7.24e-13	3.19e-13	5.51e-13	6.29e-13	6.11e-17	3.47e-19	3.17e-16	3.47e-19
	MLPTLBO	3.68e-16	9.84e-17	3.21e-33	9.76e-33	0.00	0.00	6.08e-30	1.86e-33	0.61e-16	8.88e-17
	MLPCIBO	2.63e-17	3.61e-16	8.25e-21	9.99e-19	1.11e-53	0.96e-49	1.01e-21	8.88e-21	6.96e-21	4.54e-25
θ_4	MLPBP	9.39e-03	5.62e-03	3.34e-03	2.45e-02	8.75e-02	6.78e-03	1.65e-01	4.96e-02	5.48e-01	2.23e-02
	MLPPSO	2.38e-07	1.15e-08	5.38e-09	2.16e-10	1.09e-08	3.72e-09	1.34e-11	5.67e-12	1.32e-20	1.37e-13
	MLPGA	3.62e-35	0.00	0.00	0.00	9.09e-47	0.07e-44	3.71e-36	0.00	0.00	3.36e-49
	MLPTLBO	0.00	0.00	0.00	0.00	9.85e-105	3.32e-111	6.45e-79	3.25e-78	1.12e-80	0.00
	MLPCIBO	0.00	0.00	1.62e-45	7.12e-49	9.58e-41	8.47e-25	6.48e-54	5.49e-59	9.78e-25	0.00
θ_5	MLPBP	1.85e-3	2.23e-3	4.45e-2	1.24e-1	6.34e-1	7.54e-2	4.21e-3	6.54e-2	4.26e-1	8.87e-1
	MLPPSO	2.42e-10	6.22e-11	5.84e-9	6.82e-16	4.76e-9	1.18e-9	2.34e-17	2.44e-14	9.10e-12	1.54e-11
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MLPTLBO	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MLPCIBO	6.89e-45	4.21e-71	3.85e-81	0.00	0.00	0.00	6.55e-79	4.66e-53	0.00	0.00

The MSE for MLPBP algorithm shown in Figure 7.15 through Figure 7.19 in result section 7.2.4, the used solution method provides the criteria for selection of the output if it produces less error. So, the solution can be obtained with less error. Table 7.12 gives the experimental results and comparison of all adopted algorithms for different hidden nodes. Best results for joint variables specified in bold letters and presented in Figure 7.27 through Figure 7.29. Figure 7.27 (a), (b), (c), (d) and (e) shows the selected best mean square curve of MLPGA for all joint variables. Similarly best chosen mean square curve of MLPTLBO from Table 7.12 depicted in Figure 7.28 (a), (b), (c), (d) and (e) for all joint variables.

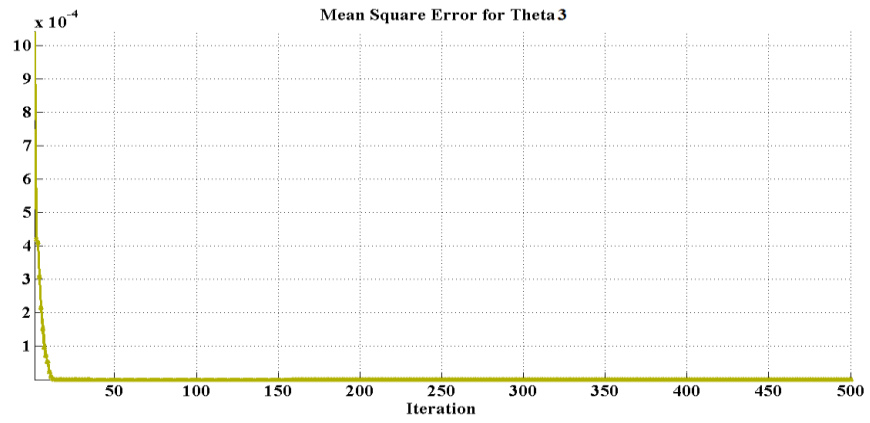
From Table 7.12, MLPBP does not give better results than the other adopted algorithms, due to trapping in local minima. Also it has been observed that MLPBP has been slow searching ability, and it consumes more CPU time. MLPGA gives the best results for all joint variables of the robot manipulator. MLPTLBO is a more stable algorithm as compared to MLPGA. Although results obtained through MLPTLBO is less as compared to MLPGA but due to better stability, MLPTLBO does not consume much CPU time as compared to MLPGA. It has been also observed that the MLPCIBO having slow searching process over MLPTLBO and MLPGA. However, CIBO has the strong exploration ability among all heuristic algorithms. For training MLP network it has been observed that both adopted heuristic algorithm yield good results of all joint variables. In other words, the proper utilization of an evolutionary algorithm with MLP networks gives outstanding performance for training the network. So these adopted hybrid techniques can be used for NP-hard problem, which generally suffers from trapping in local minima. Also these techniques guarantee faster convergence rate.



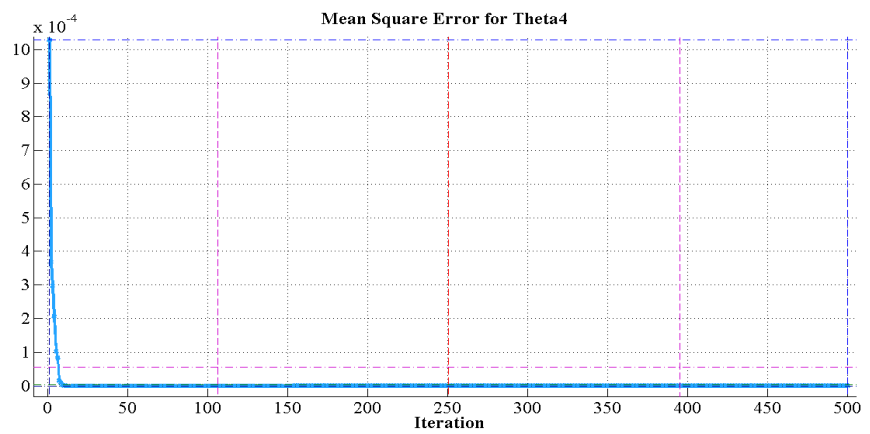
(a) Best mean square for θ_1



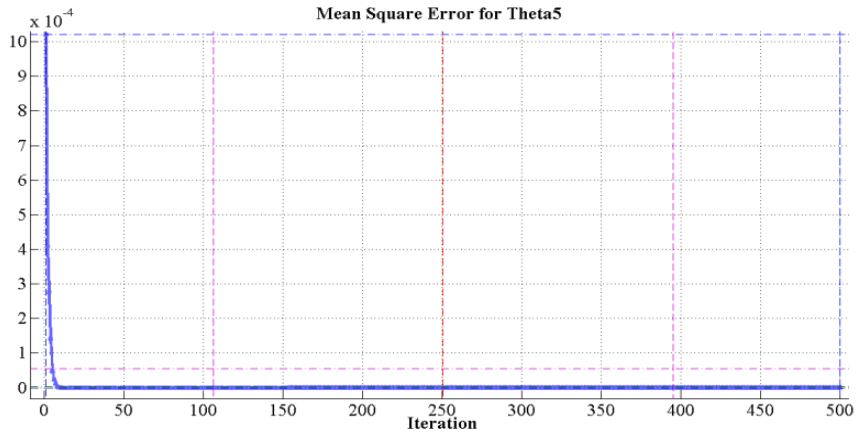
(b) Best mean square for θ_2



(c) Best mean square for θ_3

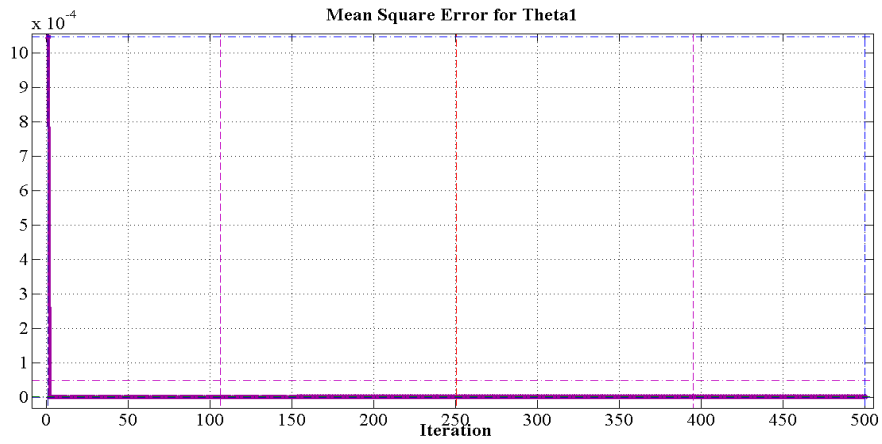


(d) Best mean square for θ_4

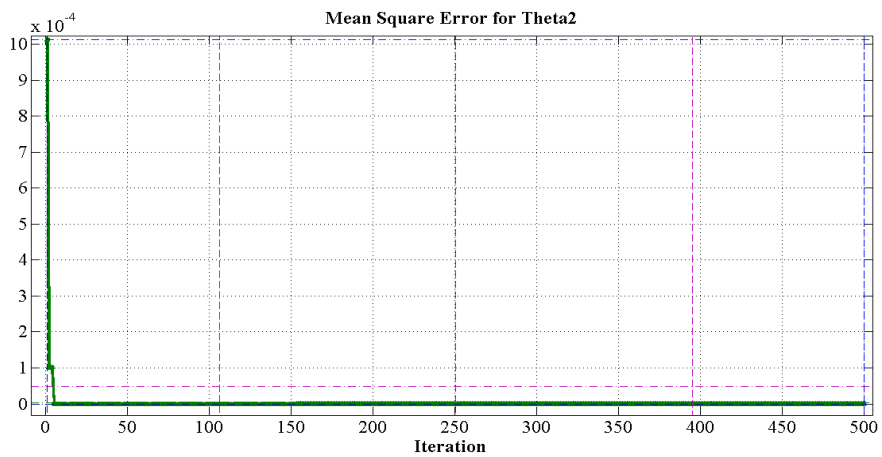


(e) Best mean square for θ_5

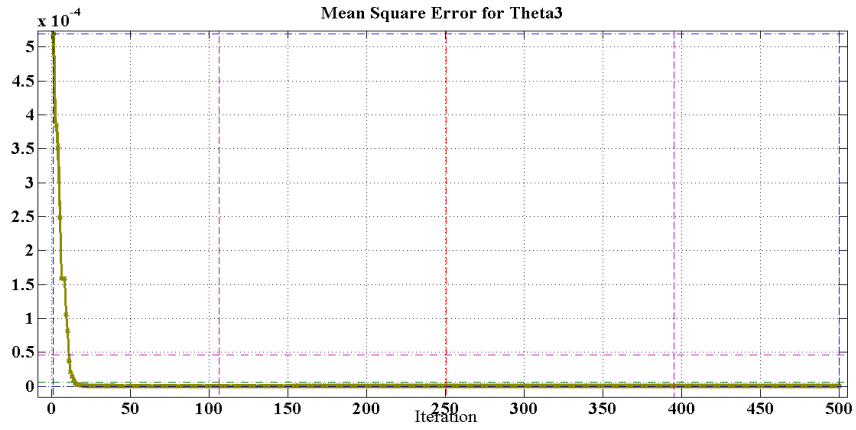
Figure 7.27 (a), (b), (c), (d) and (e) are mean square error curve for all joint angles using MLPGA.



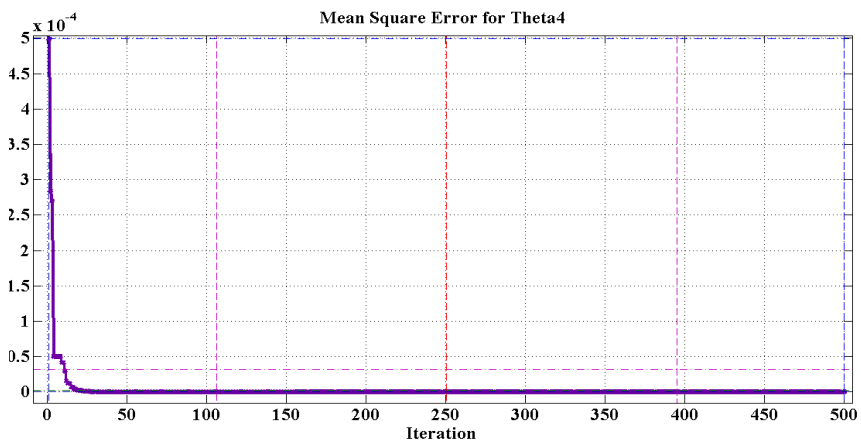
(a) Best mean square error for θ_1



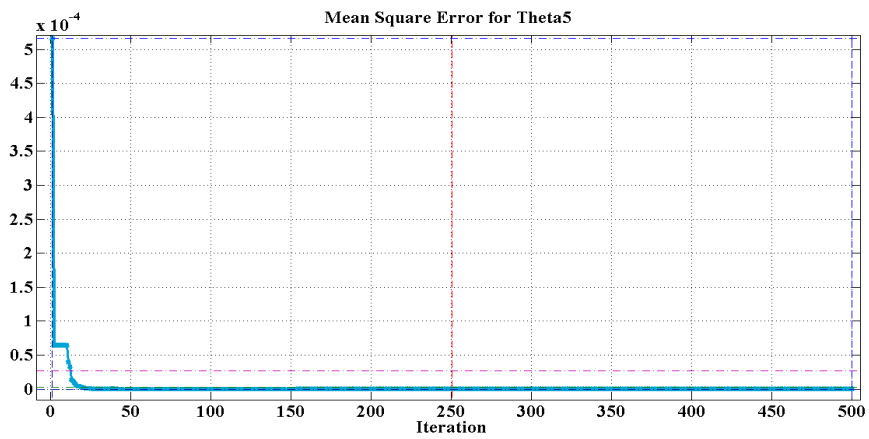
(b) Best mean square error for θ_2



(c) Best mean square error for θ_3

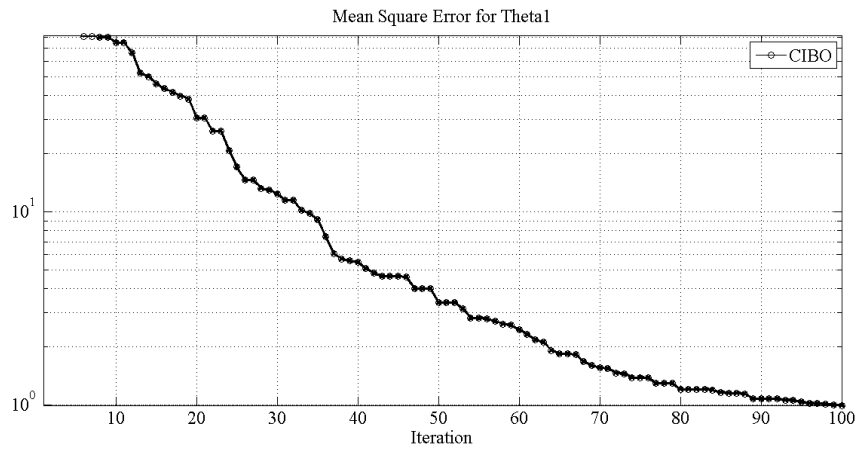


(d) Best mean square error for θ_4

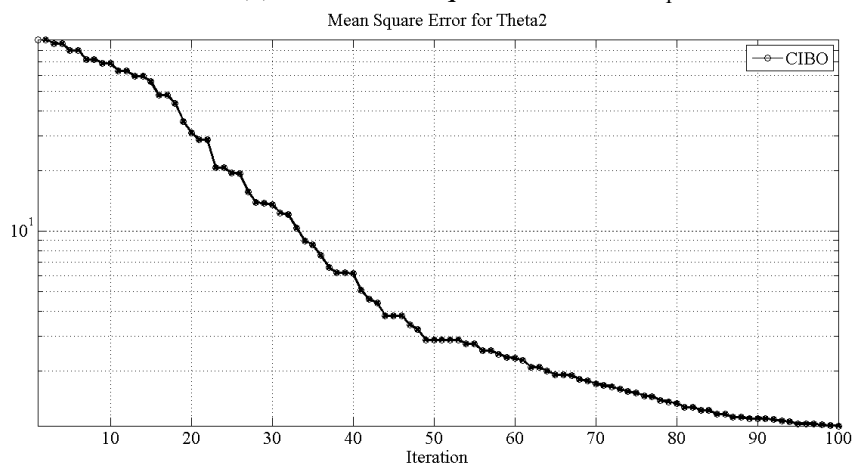


(e) Best mean square error for θ_5

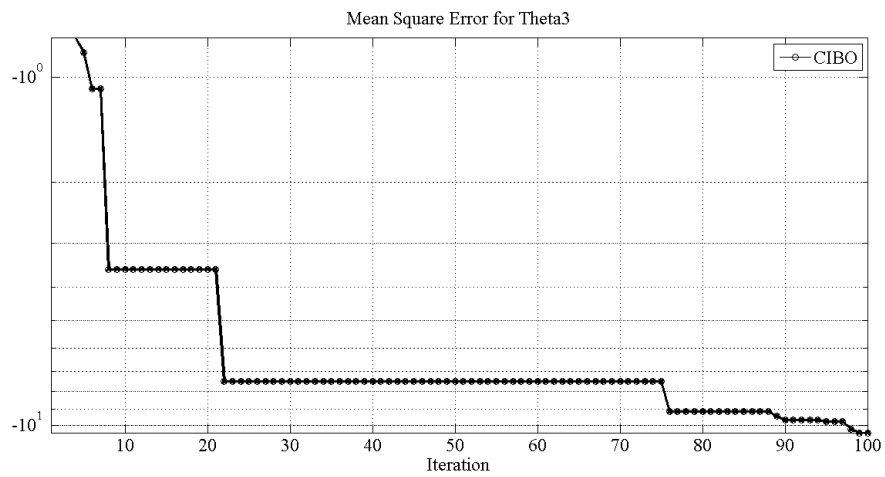
Figure 7.28 (a), (b), (c), (d) and (e) are mean square error curve for all joint angles using MLPTLBO.



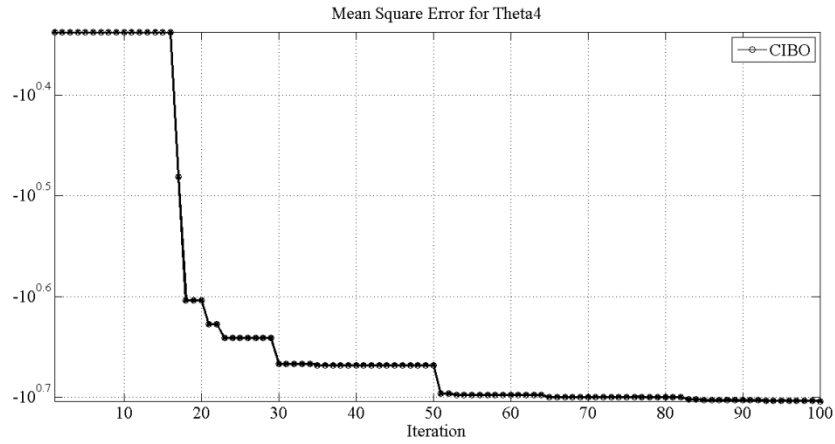
(a) Best mean square error for θ_1



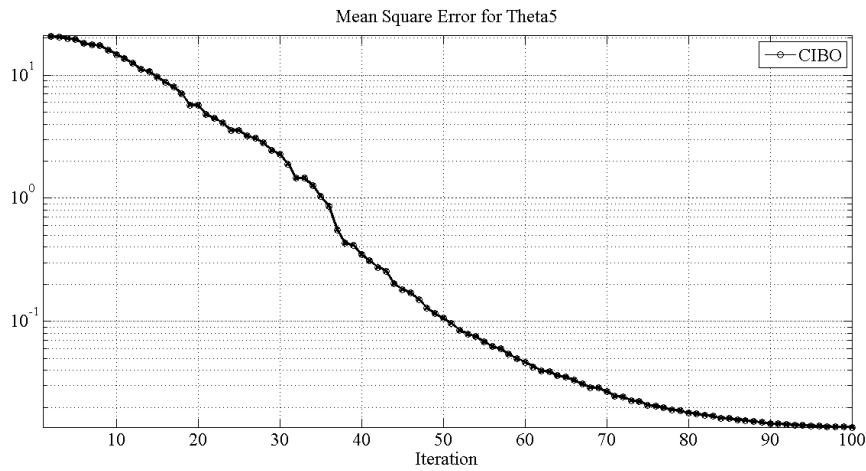
(b) Best mean square error for θ_2



(c) Best mean square error for θ_3



(d) Best mean square error for θ_4



(e) Best mean square error for θ_5

Figure 7.29 (a), (b), (c), (d) and (e) are mean square error curve for all joint angles using MLPCIBO.

7.3.3 Inverse kinematic solution of 6-dof PUMA manipulator

In this section, PUMA 560 robot manipulator is selected for the inverse kinematic inversion. This manipulator is one of the benchmark industrial manipulator which is widely used in industries and research work. The detail descriptions about this manipulator have been presented in chapter 3. The forward and inverse kinematic derivation using quaternion algebra is already discussed in chapter 4. Using the forward kinematic equations the data sets for the training of MLNN neural network is generated. MATLAB program is used to generate the data sets of the end effector coordinates and joint variables. These data sets were the basis for the training, evaluation and testing the MLP model.

Similar to previous work hybrid ANN method is used to resolve the problem of inverse kinematics of 6-dof PUMA manipulator. The comparison has been made with the several hybrid models like MLPPSO, MLPGA, MLPTLBO and MLPCIBO. Initial approximations for all adopted optimization algorithms are similar to the case of 5-dof manipulator. The desired joint variables are taken as input for the training of hybrid ANN model. The configuration and parameters is given in Table 7.13. Quaternion vector method is used to calculate the inverse kinematic solution for the PUMA manipulator and sample data sets are given in Table 7.14.

Table 7.13 Configuration of MLPNN

Sl. Parameters	Values taken
1 Learning rate	0.18
2 Momentum parameter	0.52
3 Number of hidden layers	1
4 Number of inputs	3
5 Number of output	6
6 Target datasets	1000
7 Testing datasets	300
8 Training datasets	700

Table 7.14 Desired joint variables determined through quaternion algebra

SN	Joints variables and positions determined through quaternion algebra								
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	X	Y	Z
1	26.912	131.557	79.577	138.870	6.78	63.56	-176.663	133.670	-104.739
2	13.295	32.899	102.499	26.351	91.75	21.52	-91.681	62.123	147.316
3	19.696	134.188	101.677	138.870	54.53	78.32	-121.520	-32.366	-132.255
4	12.942	109.606	57.238	26.351	95.24	69.58	-110.784	101.765	72.482
5	106.737	28.244	71.577	33.825	39.879	40.12	164.766	132.072	131.574
6	113.187	196.921	29.309	39.879	22.487	12.63	-11.850	-55.608	68.850
7	39.965	114.694	5.664	22.487	20.323	9.89	-176.933	97.283	111.305
8	61.912	93.065	26.411	33.825	6.35	45.45	134.373	167.623	133.139
9	121.557	96.636	82.241	39.879	62.56	54.78	-114.640	139.711	45.543
10	38.879	6.922	33.746	138.870	34.63	96.85	136.494	38.746	58.441
11	104.178	106.737	44.056	26.351	41.32	21.23	-158.764	45.408	-51.653

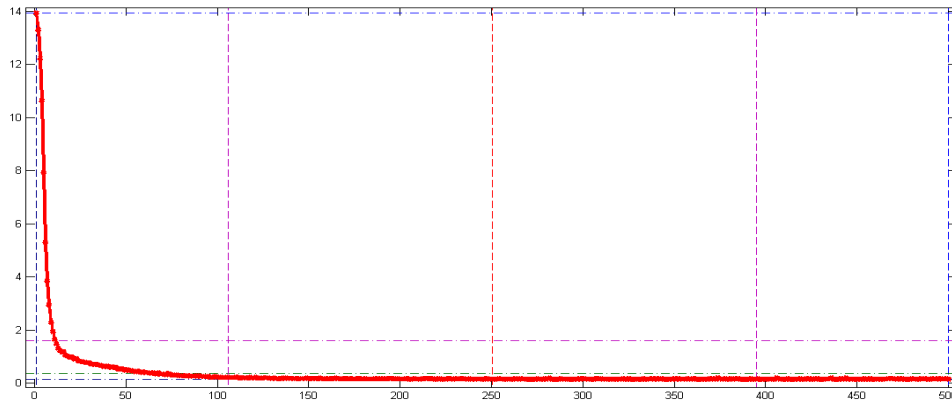
Similar to the results of previous research work MLPBP does not produce better results as compared to other hybrid algorithms. On the other hand, MLPGA gives best results among all other hybrid algorithms. But the convergence rate of MLPTLBO is more stable than GA. It has been observed that MLPGA is giving fast searching ability in case of 6-dof PUMA manipulator over other adopted algorithms. However, TLBO show strong exploration capability than others. Hybridization of metaheuristic

algorithms gives better performance as compared to back propagation algorithm. In other words, the proper utilization of an evolutionary algorithm with MLP networks gives outstanding performance for training the network. So these adopted hybrid techniques can be used for NP-hard problem, which generally suffers from trapping in local minima. Also these techniques guarantee faster convergence rate.

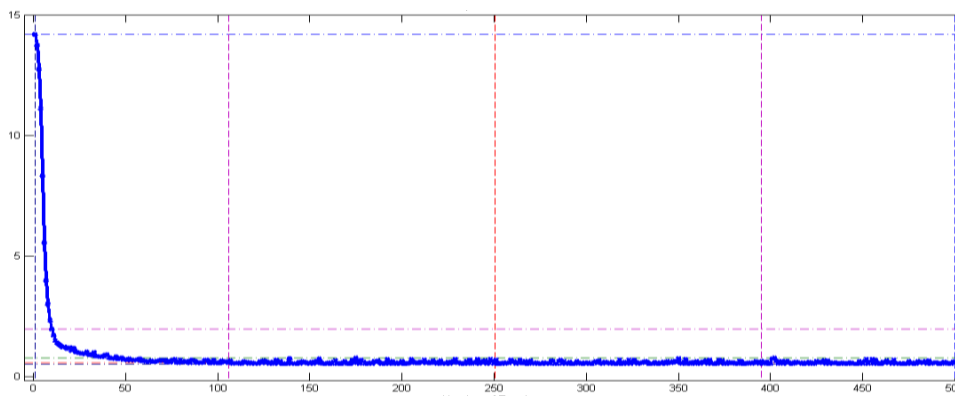
Table 7.15 Mean square error for all training samples of hybrid MLPNN

Output	Algorithms	Number of hidden neurons					
		4	11	17	20	27	30
θ_1	MLPBP	3.11e-1	6.16e-1	1.86e-2	3.41e-3	9.63e-3	7.72e-3
	ANFIS	4.12e-7	2.95e-9	7.00e-11	6.61e-11	1.98e-21	6.12e-21
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00
	MLPPSO	5.92e-40	6.85e-42	8.81e-96	5.65e-89	9.78e-56	0.44e-90
	MLPTLBO	0.00	0.00	0.00	0.00	0.00	0.00
	MLPCIBO	8.12e-56	5.89e-63	7.81e-56	8.68e-81	4.71e-65	1.34e-71
θ_2	MLPBP	6.32e-1	8.51e-2	2.32e-2	1.89e-3	5.92e-3	8.88e-1
	ANFIS	0.88e-9	8.45e-18	2.33e-19	5.43e-16	8.08e-18	3.23e-15
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00
	MLPPSO	6.55e-08	4.45e-19	6.88e-17	6.85e-07	6.32e-19	2.32e-11
	MLPTLBO	5.56e-20	2.36e-18	3.44e-18	4.12e-18	9.47e-18	6.87e-23
	MLPCIBO	1.11e-10	3.45e-13	2.19e-11	8.78e-11	6.97e-15	8.00e-24
θ_3	MLPBP	1.56e-1	9.21e-3	0.94e-3	1.84e-2	6.44e-3	7.11e-2
	ANFIS	5.42e-9	5.41e-11	8.78e-09	9.10e-13	9.47e-14	2.88e-19
	MLPGA	1.54e-21	0.00	3.54e-89	2.87e-91	6.15e-29	5.55e-31
	MLPPSO	9.14e-13	1.62e-09	3.74e-08	0.21e-07	0.99e-09	1.63e-08
	MLPTLBO	0.00	0.00	0.00	0.00	0.00	9.79e-90
	MLPCIBO	8.56e-13	4.38e-13	9.04e-14	5.54e-12	8.89e-31	6.66e-21
θ_4	MLPBP	1.99e-02	9.49e-03	4.69e-01	3.21e-02	7.47e-03	1.81e-01
	ANFIS	4.38e-08	3.34e-11	4.83e-11	2.45e-10	2.34e-21	2.48e-14
	MLPGA	0.00	0.00	1.18e-52	4.82e-44	0.00	0.00
	MLPPSO	2.22e-18	7.62e-17	7.34e-13	4.32e-14	7.74e-15	5.32e-19
	MLPTLBO	0.00	0.00	0.00	0.00	0.00	0.00
	MLPCIBO	5.49e-10	4.45e-17	5.94e-18	3.56e-18	3.45e-20	8.59e-19
θ_5	MLPBP	2.96e-2	2.35e-2	6.67e-01	3.32e-02	5.37e-03	9.98e-03
	ANFIS	3.53e-11	7.93e-17	0.09e-19	6.65e-16	0.09e-13	2.65e-12
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00
	MLPPSO	7.76e-19	7.25e-21	7.61e-19	6.21e-09	9.99e-21	7.32e-21
	MLPTLBO	0.00	0.00	0.00	0.00	0.00	0.00
	MLPCIBO	0.00	9.99e-79	5.68e-53	6.14e-51	0.00	0.00
θ_6	MLPBP	1.54e-03	8.54e-02	3.45e-01	4.65e-03	6.15e-01	3.33e-01
	ANFIS	1.78e-11	1.45e-78	3.56e-29	8.95e-31	0.00	0.00
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00
	MLPPSO	2.55e-09	4.65e-15	3.25e-19	1.44e-17	8.96e-09	4.54e-11
	MLPTLBO	0.00	0.00	0.00	0.00	0.00	0.00
	MLPCIBO	1.87e-09	1.48e-11	4.51e-09	2.02e-17	0.09e-21	9.11e-24

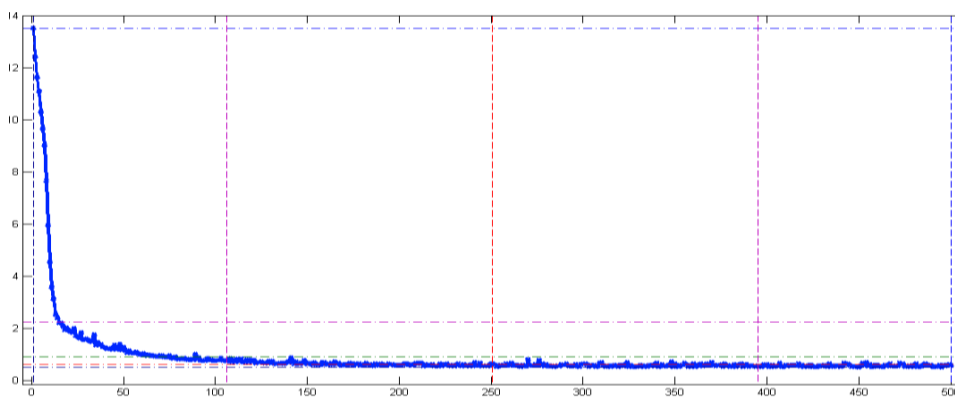
Table 7.15 gives the experimental results and comparison of all adopted algorithms for different hidden nodes. Best results for joint variables specified in bold letters and presented in Figure 7.30 through Figure 7.33. Figure 7.31 (a), (b), (c), (d) and (e) shows the selected best mean square curve of MLPGA for all joint variables. Similarly best chosen mean square curve of MLPTLBO from Table 7.16 depicted in Figure 7.32 (a), (b), (c), (d) and (e) for all joint variables.



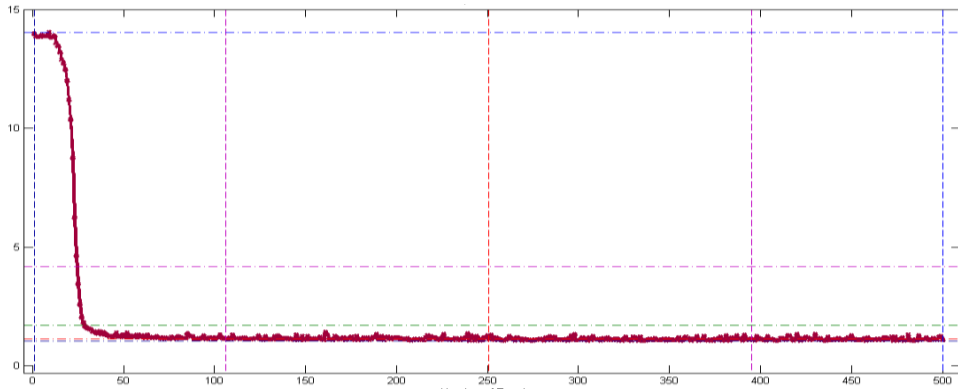
(a) Best mean square error for θ_1



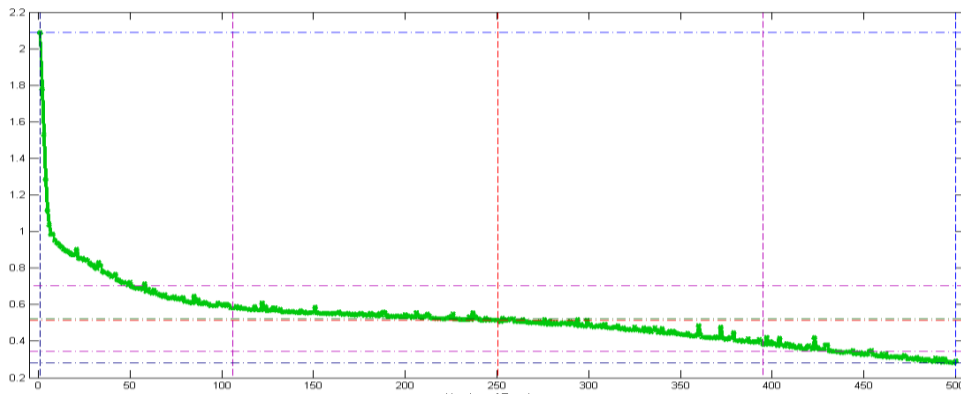
(b) Best mean square error for θ_2



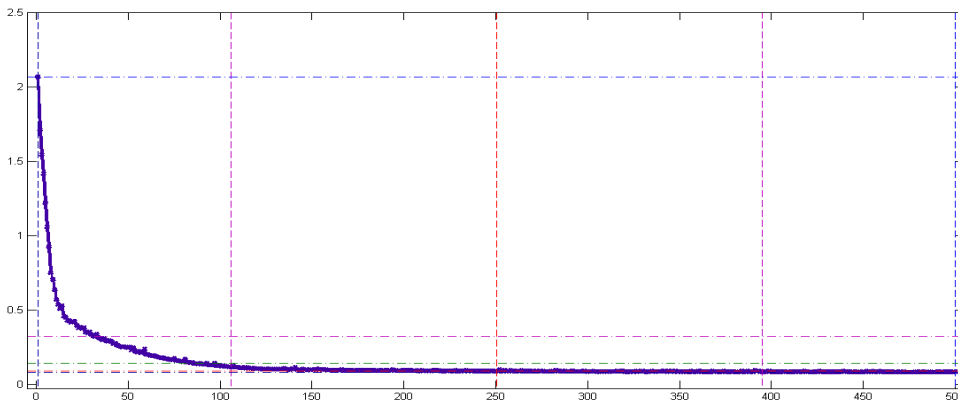
(c) Best mean square error for θ_3



(d) Best mean square error for θ_4

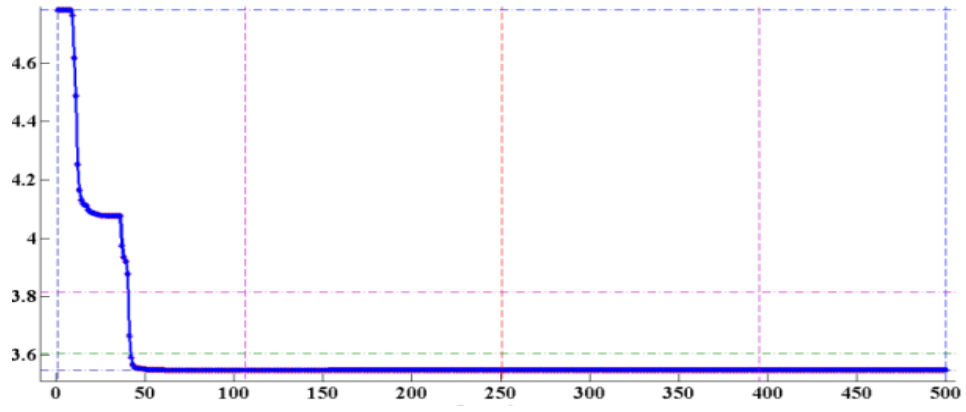


(e) Best mean square error for θ_5

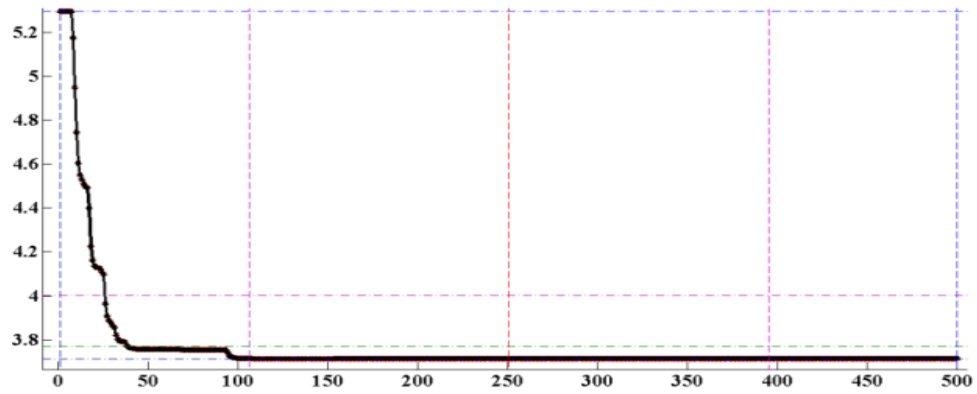


(f) Best mean square error for θ_6

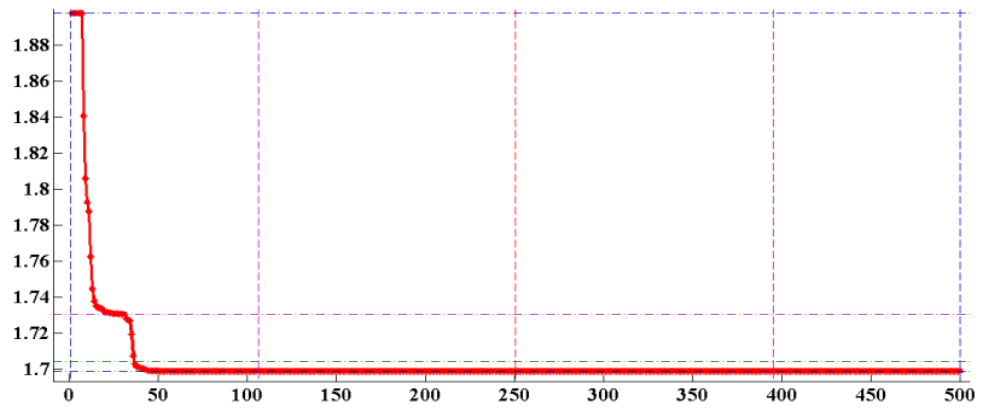
Figure 7.30 (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPBP for all joint angles.



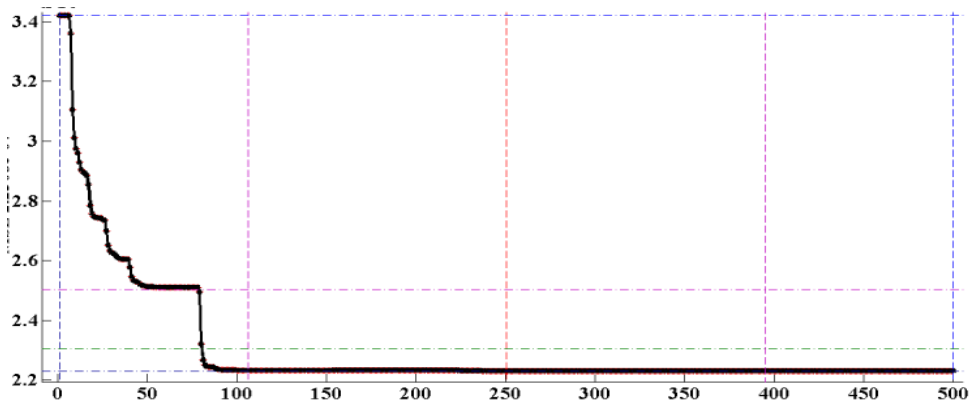
(a) Best mean square error for θ_1



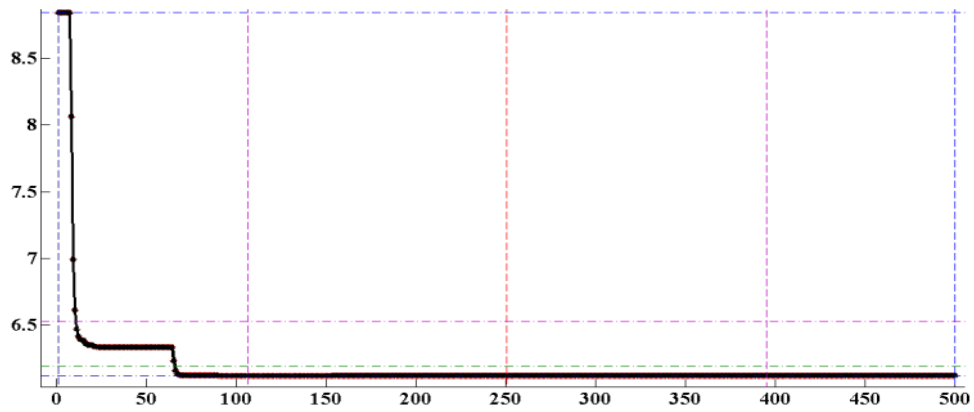
(b) Best mean square error for θ_2



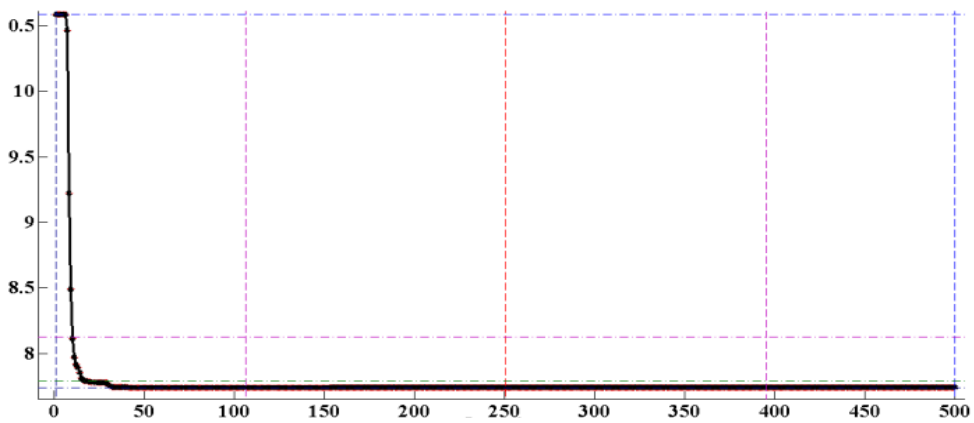
(c) Best mean square error for θ_3



(d) Best mean square error for θ_4

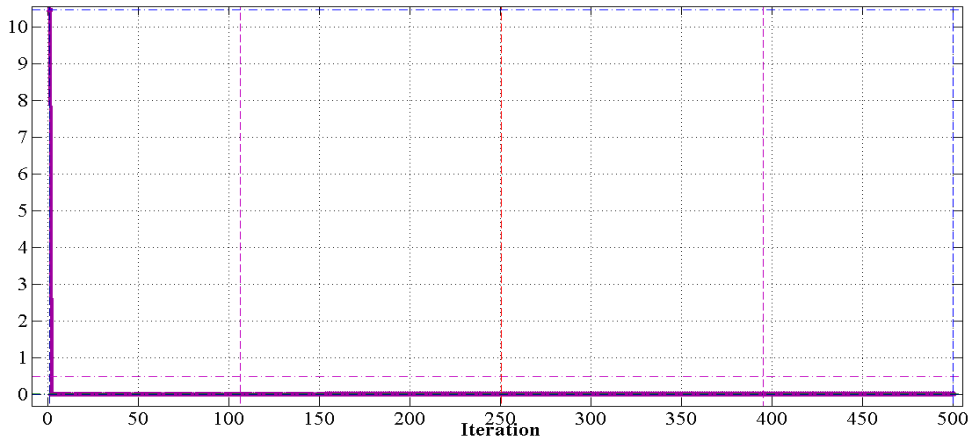


(e) Best mean square error for θ_5

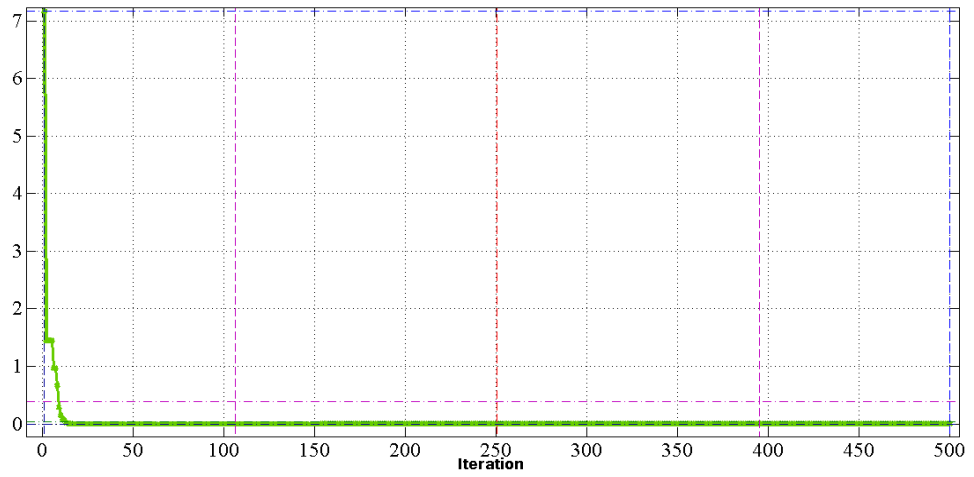


(f) Best mean square error for θ_6

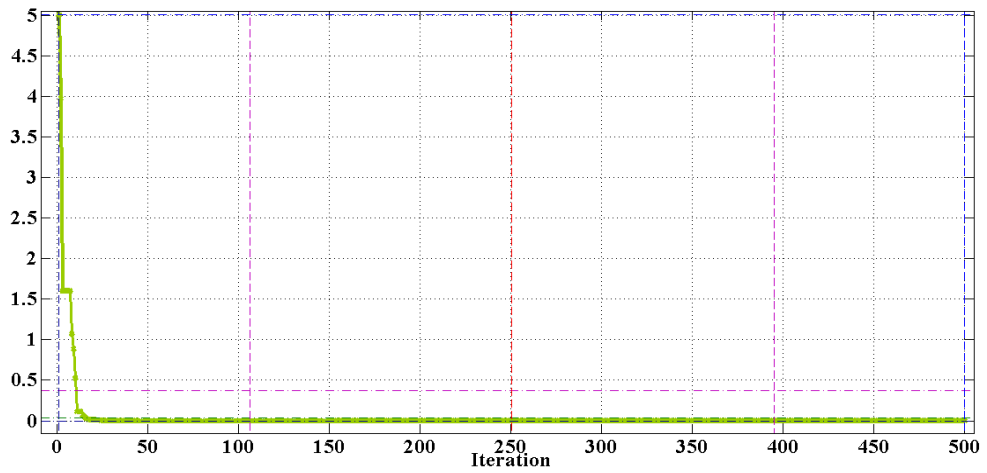
Figure 7.31 (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPGA for all joint angles.



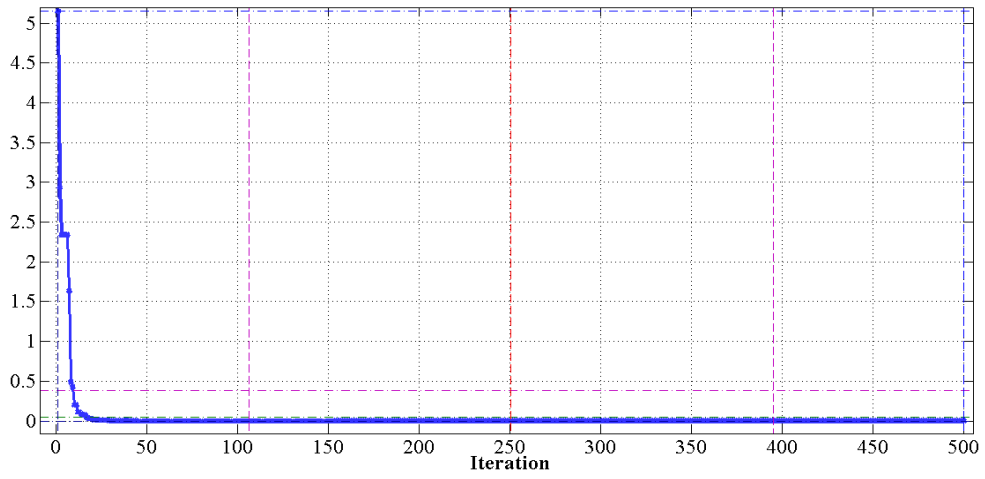
(a) Best mean square error for θ_1



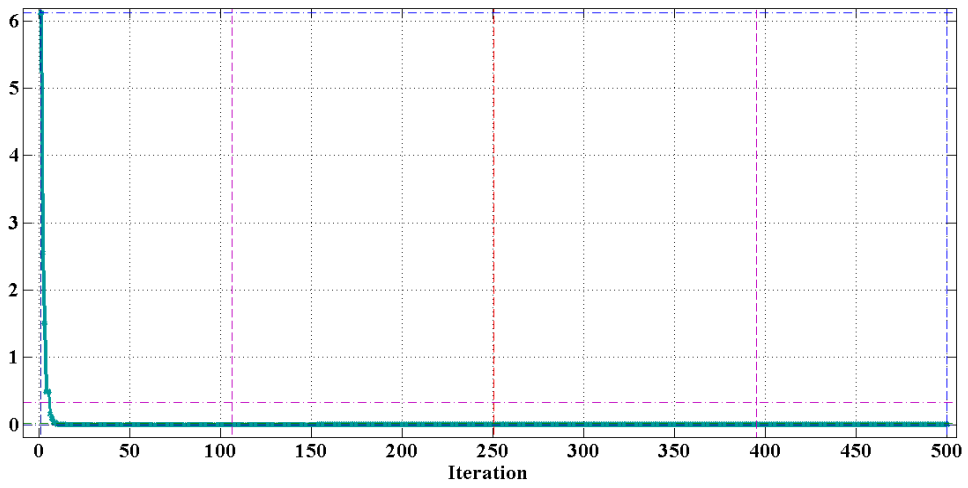
(b) Best mean square error for θ_2



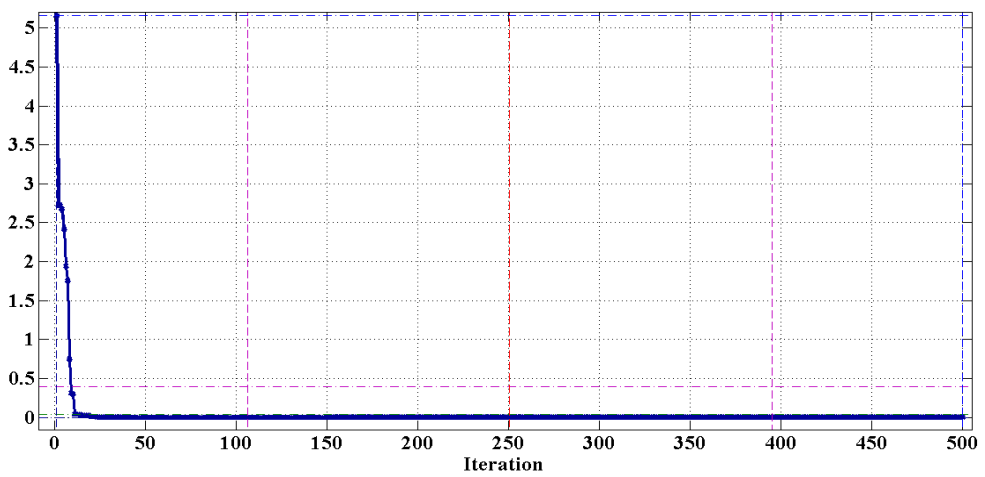
(c) Best mean square error for θ_3



(d) Best mean square error for θ_4

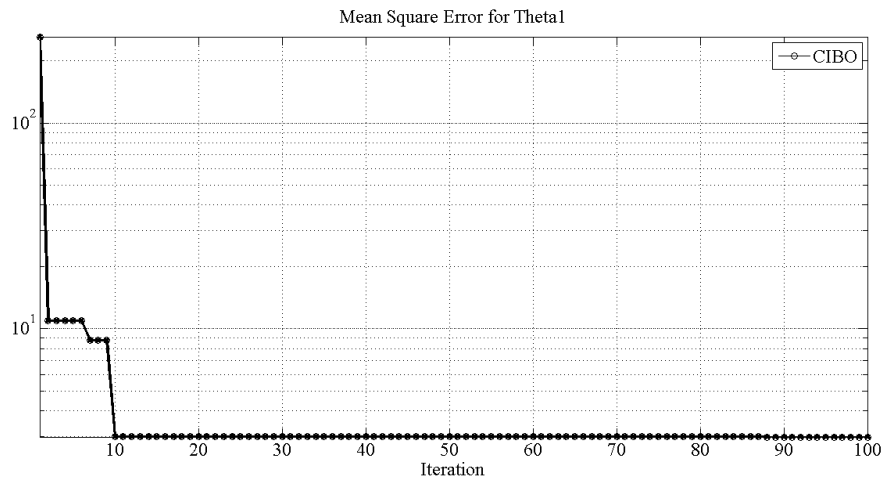


(e) Best mean square error for θ_5

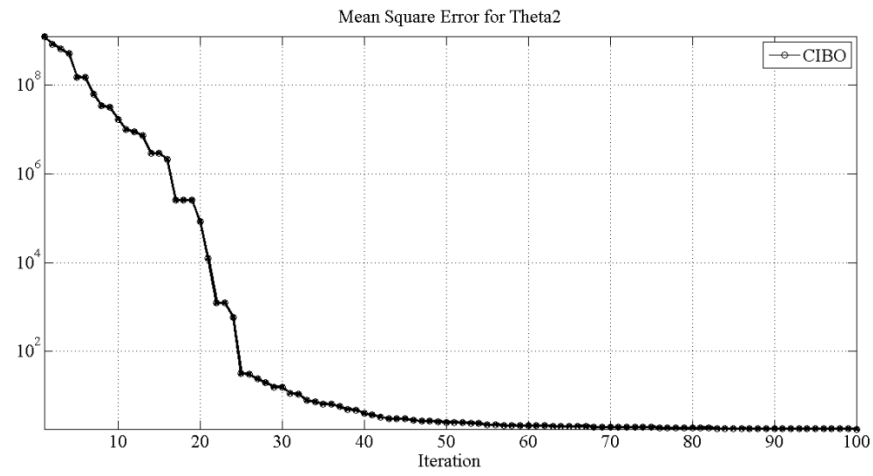


(f) Best mean square error for θ_6

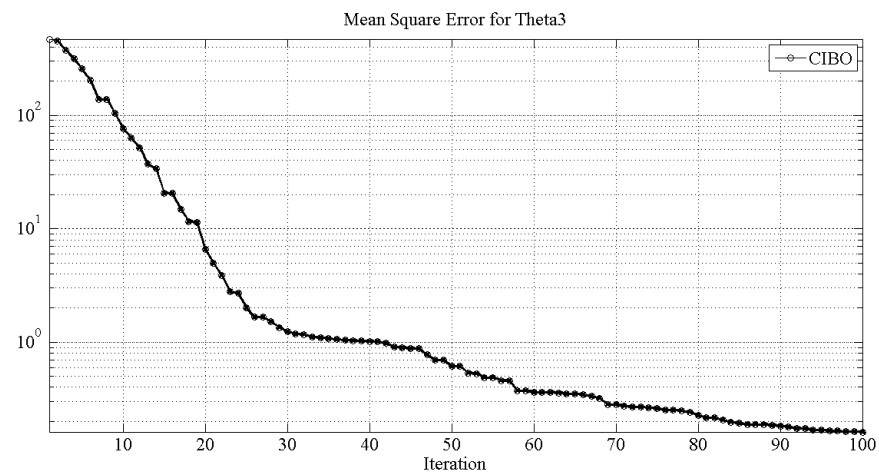
Figure 7.32 (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPTLBO for all joint angles.



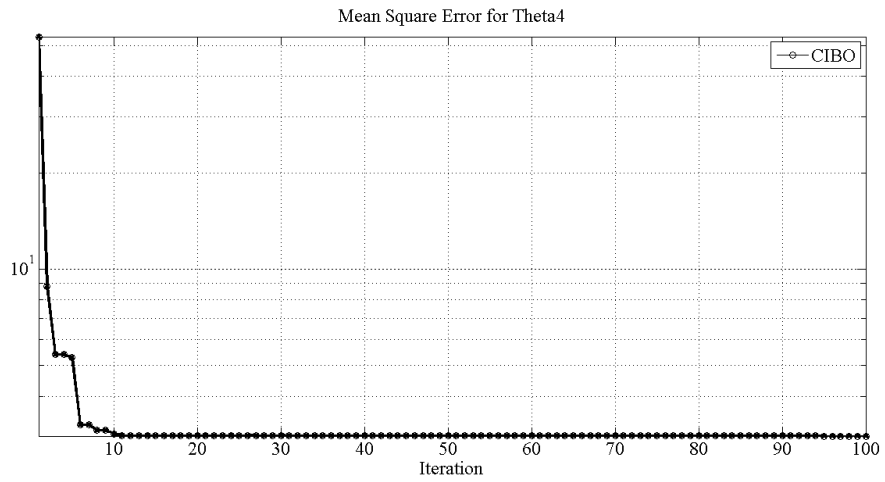
(a) Best mean square error for θ_1



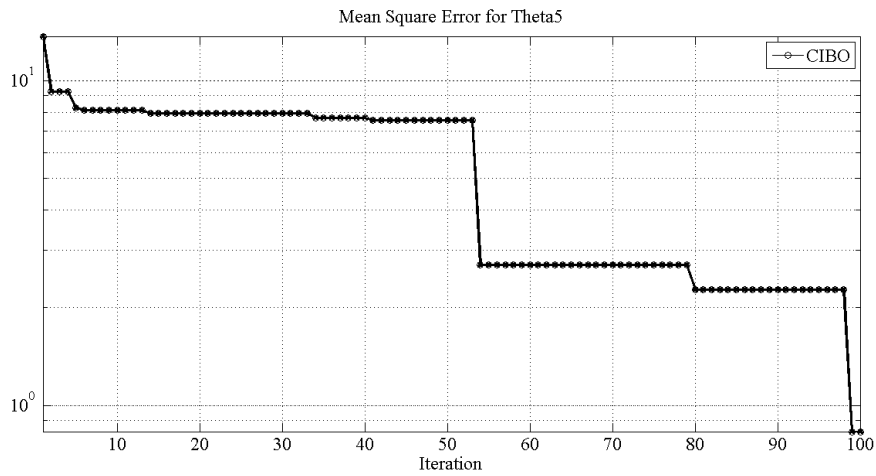
(b) Best mean square error for θ_2



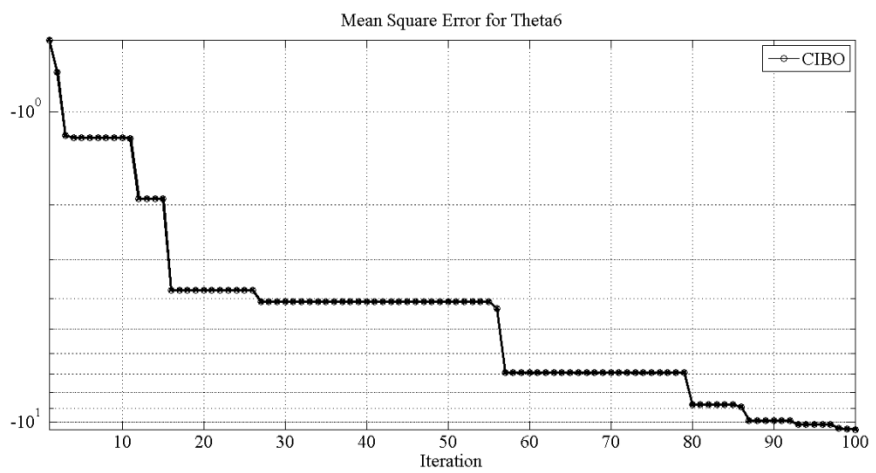
(c) Best mean square error for θ_3



(d) Best mean square error for θ_4



(e) Best mean square error for θ_5



(f) Best mean square error for θ_6

Figure 7.33 (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPCIBO for all joint angles.

7.3.4 Inverse kinematic solution of 6-dof ABB IRB-1400 manipulator

This section pertains, ABB IRB-1400 (Type A2) robot manipulator for the inverse kinematic solution. The detail description of the adopted robot manipulator model and kinematic parameters are presented in chapter 3. The quaternion vector based mathematical modelling of the adopted robot manipulator is given in chapter 4. Using kinematic equations from section 4.3.6, several joint variables and end effector coordinates are depicted in Table 7.17. MATLAB coding is used to generate the joint variables and end effector coordinates. The generated data sets are used for training and testing of the adopted intelligence based methods. Once the training is completed the actual output is compared with the desired output so as to get mean square error for all joint variables. The mean square errors for all joint variables are obtained using ANN and hybrid ANN models. The configuration of the MLPNN with different number of hidden neurons is presented in Table 7.16.

Table 7.16 Configuration of MLPNN

Sl. Parameters	Values taken
1 Learning rate	0.96
2 Momentum parameter	0.34
3 Number of epochs	1000
4 Number of inputs	3
5 Number of output	6
6 Target datasets	500
7 Testing datasets	100
8 Training datasets	700

Following the similar procedure of PUMA 560 manipulator, the inverse kinematic solution is presented using MLPNN and hybrid MLPNN methods. From the previous research work MLPNN produces poor results for the prediction of inverse kinematic solutions. Therefore, The ANN model is hybridized with the fuzzy logic and other optimization based algorithms. The comparison has been made with the several hybrid models like MLPPSO, MLPGA, MLPTLBO, ANFIS and MLPCIBO. Initial approximations for all adopted optimization algorithms are similar to the case of 5-dof and PUMA manipulators. Quaternion vector method is used to calculate the inverse kinematic solution for the PUMA manipulator and sample data sets are given in Table 7.17. The desired joint variables are taken as input for the training of hybrid ANN model.

Table 7.18 provides the experimental results and comparison of all adopted algorithms for different hidden nodes. Best performance of the algorithm is presented in bold letters with mean square error plots. The obtained mean square error values are very

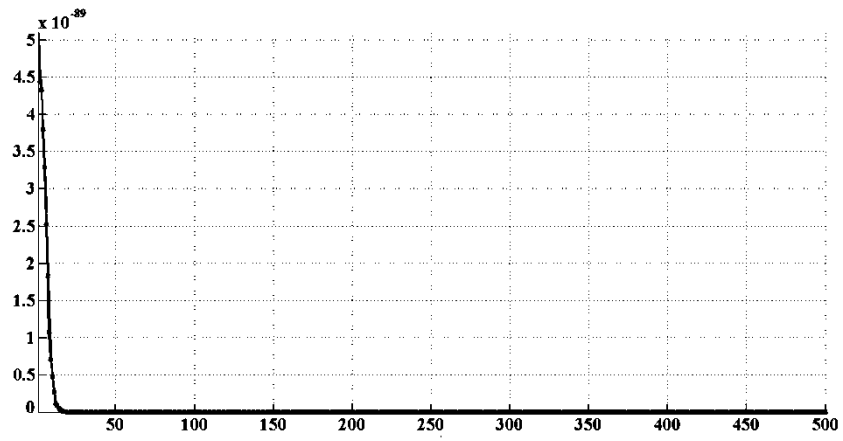
low as compared to MLPNN and ANFIS models. Therefore, fusion of MLPNN model with optimization algorithms proves strong convergence ability along with better prediction of inverse kinematic solution. On the other hand, ANFIS perform better than MLPBP see Table 7.18. Although, the results produces through the ANFIS is quite acceptable as compared to MLPBP but hybrid ANN's are more efficient and accurate. The convergence of MLPBP and ANFIS are poor due to its local searching ability. Therefore, hybrids ANN are more efficient and yields global searching ability. Figure 7.34 (a), (b), (c), (d) and (e) shows the selected best mean square curve of MLPGA for all joint variables.

Table 7.17 Desired joint variables determined through quaternion algebra

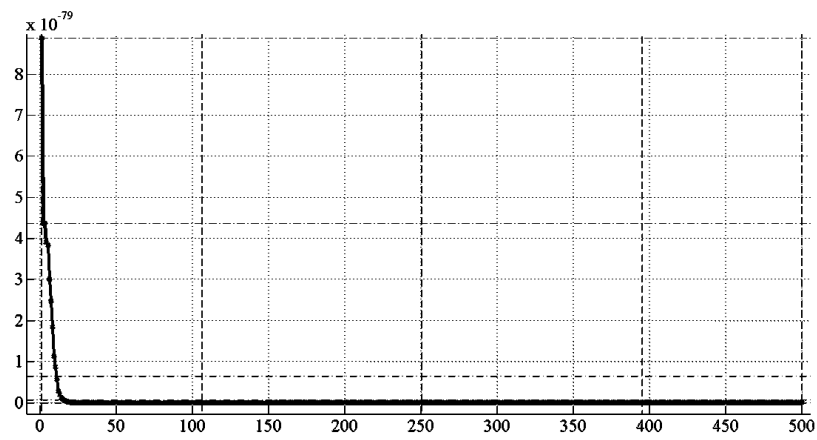
SN	Joints variables and positions determined through quaternion algebra								
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	X	Y	Z
1	58.3056	3.3921	39.6834	88.3088	72.2516	155.4015	12.4633	47.5939	104.7516
2	53.1530	20.0422	-56.6319	81.8867	14.0897	165.9524	-24.6403	-62.3966	91.8795
3	86.3071	45.3925	12.6747	50.4011	80.0651	41.8404	-21.6169	9.3034	118.0053
4	87.1289	38.6382	9.3534	44.1845	11.7174	264.8126	-12.1602	8.8344	126.5538
5	98.9140	49.3338	42.9245	85.5255	56.5194	36.9996	-50.3168	18.7116	103.2790
6	51.4737	66.4455	-13.9112	120.3588	47.1847	49.7577	2.3410	-23.3834	122.2380
7	40.9223	28.2572	7.1238	59.7338	103.5930	124.9550	-5.3869	15.7569	116.4455
8	121.7940	61.2329	63.2612	103.3004	59.3777	149.5440	-66.7556	-14.1633	83.3454
9	130.8836	4.7135	-48.6507	136.3077	77.2835	252.4995	35.9920	-48.1656	91.8021
10	134.5108	56.2707	-16.2299	17.0630	24.0029	283.9313	19.6269	-0.8582	125.0100
11	124.1020	23.6187	-45.5391	17.1552	61.1137	228.1513	38.6717	-31.1640	105.8820

Table 7.18 Mean square error for all training samples of hybrid MLPNN

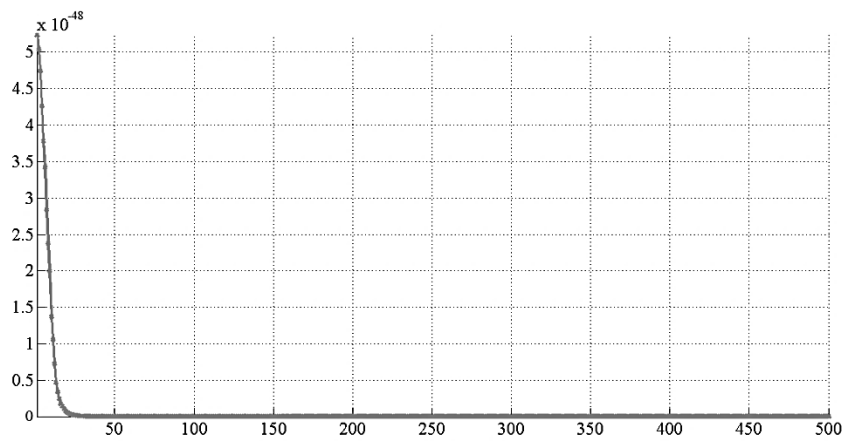
Output	Algorithms	Number of Hidden Nodes					
		4	11	17	20	27	30
θ_1	MLPBP	0.12e-01	3.86e-02	5.62e-03	1.85e-01	2.15e-03	4.56e-04
	ANFIS	9.04e-03	5.14e-01	7.24e-02	0.99e-04	8.73e-04	1.15e-03
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00
	MLPTLBO	0.00	0.00	0.00	1.51e-49	3.48e-51	8.45e-47
	MLPCIBO	4.52e-13	2.32e-13	6.96e-19	4.85e-17	6.99e-15	7.31e-24
	MLPPSO	9.41e-23	9.60e-37	2.63e-17	1.03e-31	2.87e-09	4.45e-08
θ_2	MLPBP	9.99e-03	1.81e-03	0.37e-02	4.73e-03	5.82e-02	0.99e-03
	ANFIS	1.06e-04	3.55e-03	7.34e-02	6.11e-04	2.04e-04	7.64e-02
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00
	MLPTLBO	3.01e-79	8.85e-80	7.25e-91	6.12e-91	5.47e-90	4.19e-90
	MLPCIBO	5.55e-98	7.36e-89	5.92e-40	9.49e-35	8.89e-40	6.85e-42
	MLPPSO	4.96e-89	8.81e-96	5.65e-89	6.87e-52	9.78e-56	0.44e-90
θ_3	MLPBP	1.60e-03	5.21e-02	3.48e-03	4.44e-02	7.34e-01	0.48e-02
	ANFIS	3.66e-04	4.72e-04	1.01e-01	8.24e-03	6.42e-03	2.89e-03
	MLPGA	0.00	0.00	6.89e-61	1.43e-63	1.11e-76	0.00
	MLPTLBO	5.56e-40	7.61e-30	1.89e-44	2.55e-41	3.12e-67	1.23e-36
	MLPCIBO	4.95e-80	3.31e-45	3.18e-51	9.04e-79	3.39e-16	6.11e-15
	MLPPSO	1.66e-14	3.91e-13	9.99e-19	2.29e-21	6.78e-21	7.78e-16
θ_4	MLPBP	0.07e-01	1.33e-03	9.09e-04	0.22e-01	6.48e-02	3.65e-03
	ANFIS	8.56e-05	0.04e-01	2.33e-03	0.86e-01	5.31e-03	1.91e-04
	MLPGA	1.33e-29	2.95e-31	0.18e-35	1.02e-39	0.00	0.00
	MLPTLBO	4.47e-16	3.33e-31	8.56e-11	3.85e-12	8.21e-19	6.45e-19
	MLPCIBO	6.55e-08	1.09e-09	5.21e-09	4.45e-19	3.11e-18	6.88e-17
	MLPPSO	6.87e-18	9.51e-19	8.76e-18	8.23e-15	3.54e-08	7.12e-19
θ_5	MLPBP	9.11e-03	1.86e-02	0.06e-01	4.67e-03	8.22e-02	6.49e-04
	ANFIS	1.08e-05	3.44e-02	4.86e-03	0.67e-02	3.55e-01	1.49e-04
	MLPGA	0.09e-28	8.64e-29	1.30e-33	6.54e-36	7.29e-29	4.86e-28
	MLPTLBO	6.85e-07	8.12e-03	6.32e-19	2.32e-11	7.67e-12	2.37e-12
	MLPCIBO	4.95e-15	4.35e-14	3.27e-16	1.18e-17	3.00e-09	0.74e-08
	MLPPSO	4.68e-11	2.21e-13	4.12e-17	2.87e-16	3.72e-15	2.11e-10
θ_6	MLPBP	0.02e-01	5.74e-04	9.23e-03	4.61e-02	3.00e-03	0.46e-01
	ANFIS	6.87e-04	3.48e-05	6.77e-03	8.15e-02	6.47e-01	0.18e-03
	MLPGA	5.94e-21	3.86e-26	0.89e-21	4.81e-19	2.32e-17	6.55e-16
	MLPTLBO	0.30e-19	1.34e-11	6.76e-15	7.15e-18	0.87e-12	3.51e-18
	MLPCIBO	9.14e-13	8.27e-11	6.25e-12	1.62e-09	8.45e-08	3.74e-08
	MLPPSO	0.21e-07	1.39e-04	0.99e-09	1.63e-08	8.44e-09	8.04e-08



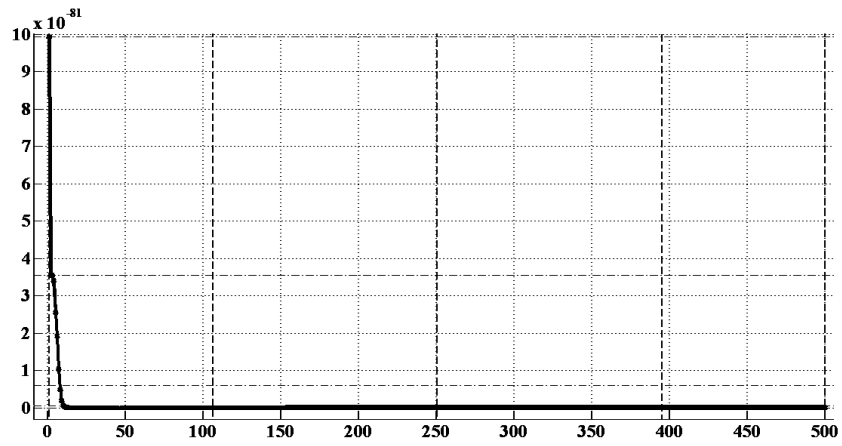
(a) Best mean square error for θ_1



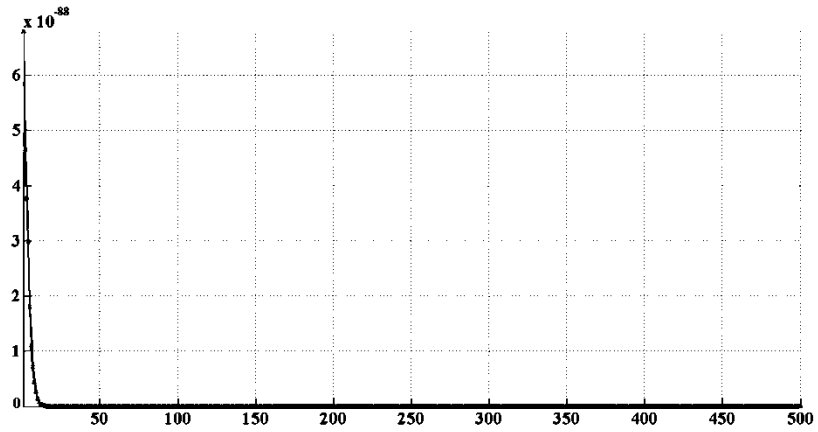
(b) Best mean square error for θ_2



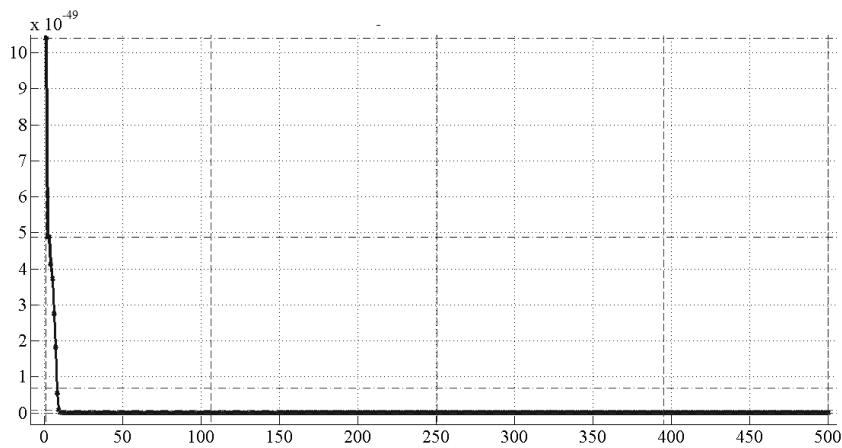
(c) Best mean square error for θ_3



(d) Best mean square error for θ_4



(e) Best mean square error for θ_5



(f) Best mean square error for θ_6

Figure 7.34 (a), (b), (c), (d), (e) and (f) are mean square error curve of MLPGA for all joint angles.

The optimization of ANN parameters such as weight and bias provides the better efficiency and prediction of results. MLPGA and MLPTLBO equally perform better than MLPSO and MLPCIBO. On the other hand, performances of MLPGA compared to all adopted algorithms are acceptable. It can also be observed that hybridization with GA shows the fast searching ability and with better exploitation of the solution. However, TLBO shows strong exploration ability than other algorithms. The results of the MLPTLBO are depicted in Table 7.18. Therefore, the hybridization of optimization algorithms with MLP models produces better results as compared to traditional back propagation algorithms. Also these techniques guarantee faster convergence rate.

7.3.5 Inverse kinematic solution of 5-dof ASEA IRb-6 manipulator

In this section, ASEA IRb-6 robot manipulator is considered for the inverse kinematic solution. This manipulator is widely used in industries as well as in research areas. The detail description of the adopted robot manipulator is presented in chapter 3. The mathematical modelling of the adopted robot manipulator is given in chapter 4. Following the kinematics equations and DH-parameters of the adopted robot, the joint variables and end effector coordinates are obtained using MATLAB coding. The prepared data sets are used as an input to train the adopted ANN models for the prediction of inverse kinematic of the manipulator. After training of the adopted configurations of the neural network models and hybrid neural networks, the inverse kinematic solutions are compared with the conventional method based solution. Further the mean square errors for all joint variables are calculated on the basis of actual data and desired data sets. The configuration of the MLPNN with different number of hidden neurons is presented in Table 7.19.

Table 7.19 Configuration of MLPNN

Sl. Parameters	Values taken
1 Learning rate	0.96
2 Momentum parameter	0.34
3 Number of epochs	1000
4 Number of inputs	3
5 Number of output	6
6 Target datasets	500
7 Testing datasets	100
8 Training datasets	700

Table 7.20 Desired joint variables determined through quaternion algebra

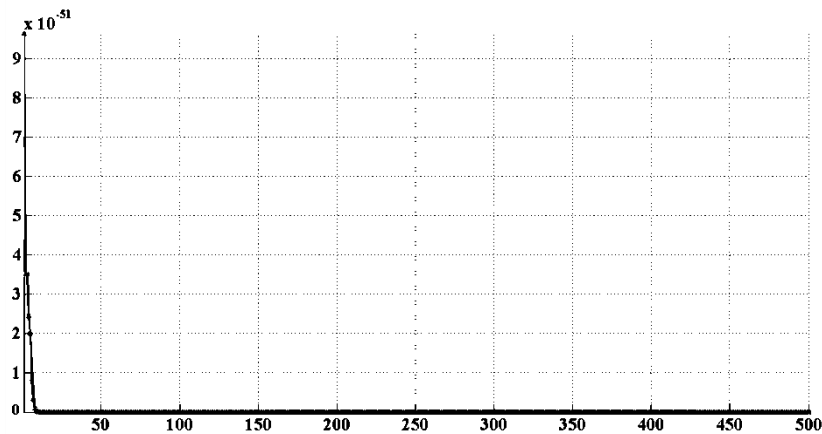
SN	Joints variables and positions determined through quaternion algebra							
	θ_1	θ_2	θ_3	θ_4	θ_5	X	Y	Z
1	99.6981	85.4732	-135.6945	-143.2309	109.1581	595.7279	-303.9048	-294.6464
2	310.6737	59.8666	140.0265	27.9008	39.4117	78.5678	-334.9327	71.8066
3	329.3916	81.7060	144.4626	-82.6927	333.3707	565.9467	-293.7725	43.1890
4	177.9217	95.9999	-142.7337	67.0121	283.7942	510.2091	-572.3394	-611.0379
5	157.1849	95.2089	143.8745	-219.1774	32.5522	109.9897	-242.1020	-191.0548
6	329.4058	74.2685	39.7596	91.7458	313.3460	458.8033	-373.0525	-200.7021
7	310.8852	69.1873	-146.0051	126.3939	189.3874	40.1121	-656.7506	-4.1938
8	195.4502	84.4757	48.2710	6.5640	138.6096	215.9464	-157.9838	-507.8093
9	251.0410	92.8778	-4.2599	-202.7799	178.0429	247.9422	-0.6056	-32.8324
10	106.2674	96.7097	40.1150	-34.5511	86.8883	21.1918	-390.7909	-179.9546
11	195.0605	100.0935	-138.8201	2.5618	207.9951	276.6552	-407.2539	-109.6221

Table 7.21 Mean square error for all adopted algorithms

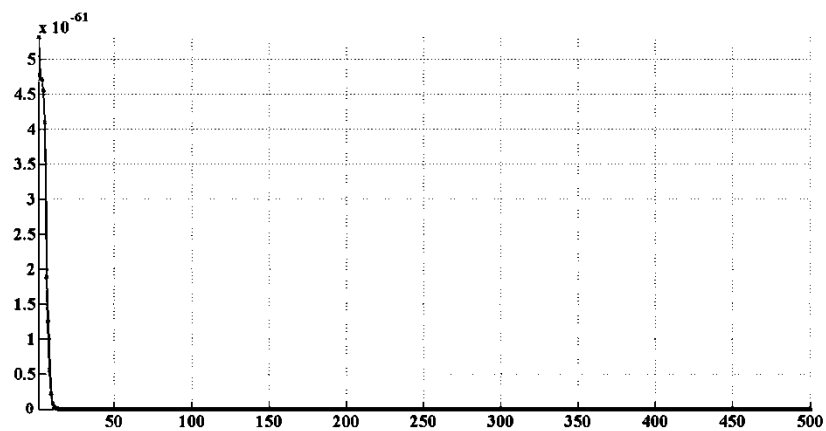
Output	Algorithms	Number of Hidden Nodes					
		4	11	17	20	27	30
θ_1	MLPBP	5.26e-01	4.16e-03	0.99e-01	3.98e-05	7.86e-04	1.74e-03
	ANFIS	9.47e-02	3.33e-01	5.00e-03	6.20e-03	4.51e-01	9.12e-02
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00
	MLPTLBO	4.65e-11	6.51e-21	3.08e-21	2.79e-18	2.13e-24	6.46e-21
	MLPCIBO	1.51e-11	0.51e-12	4.52e-08	6.18e-10	1.00e-11	0.54e-12
	MLPPSO	2.58e-06	5.70e-07	2.31e-09	3.15e-12	2.09e-10	3.50e-19
θ_2	MLPBP	7.65e-04	1.40e-02	0.43e-03	1.43e-03	3.25e-01	0.01e-03
	ANFIS	6.44e-02	3.77e-03	0.11e-05	3.52e-02	1.56e-02	6.42e-01
	MLPGA	0.00	1.63e-51	3.86e-48	5.74e-71	0.00	0.00
	MLPTLBO	2.22e-18	3.56e-14	7.78e-18	7.62e-17	3.35e-16	7.34e-13
	MLPCIBO	4.32e-14	4.26e-14	7.74e-15	5.32e-19	1.47e-05	4.60e-06
	MLPPSO	1.20e-08	2.04e-11	1.18e-15	7.40e-14	2.07e-20	3.42e-15
θ_3	MLPBP	5.44e-03	0.23e-04	4.58e-02	6.78e-01	7.21e-02	9.85e-03
	ANFIS	5.12e-03	0.56e-04	4.68e-03	4.89e-03	4.33e-02	7.29e-03
	MLPGA	6.64e-34	7.64e-31	2.22e-21	0.05e-30	0.00	0.00
	MLPTLBO	0.99e-21	5.35e-18	6.15e-10	5.21e-11	6.75e-10	6.16e-18
	MLPCIBO	7.76e-19	6.78e-18	1.00e-19	7.25e-21	0.85e-25	7.61e-19
	MLPPSO	2.26e-15	1.85e-21	9.11e-11	0.93e-21	9.42e-17	0.25e-18
θ_4	MLPBP	6.88e-03	0.99e-03	1.44e-03	4.00e-01	0.45e-01	7.42e-01
	ANFIS	6.48e-06	7.04e-01	6.47e-02	6.44e-03	8.11e-02	4.00e-03
	MLPGA	0.00	0.00	0.00	0.00	2.14e-69	3.15e-79
	MLPTLBO	6.21e-09	2.11e-19	9.99e-21	7.32e-21	5.45e-81	4.32e-61
	MLPCIBO	4.78e-19	5.55e-18	6.78e-17	6.54e-16	6.48e-15	7.15e-15
	MLPPSO	9.87e-11	6.58e-12	4.58e-19	7.47e-09	2.55e-09	6.45e-19
θ_5	MLPBP	7.44e-04	1.05e-01	7.53e-02	4.12e-01	6.41e-03	7.22e-03
	ANFIS	3.33e-01	1.67e-03	4.22e-04	8.04e-02	5.82e-02	7.77e-01
	MLPGA	0.00	0.00	0.00	0.00	0.00	0.00
	MLPTLBO	1.45e-21	4.65e-15	1.02e-11	3.25e-19	1.44e-17	1.11e-12
	MLPCIBO	8.96e-09	4.54e-11	8.54e-12	3.41e-11	2.07e-15	4.68e-21
	MLPPSO	0.85e-19	5.62e-11	3.13e-21	0.00	0.00	0.00

Following the similar procedure of PUMA 560 manipulator, the inverse kinematic solution is presented using MLPNN and hybrid MLPNN methods. The hybrid MLP models are as follows MLPSO, MLPGA, MLPTLBO, ANFIS and MLPCIBO.

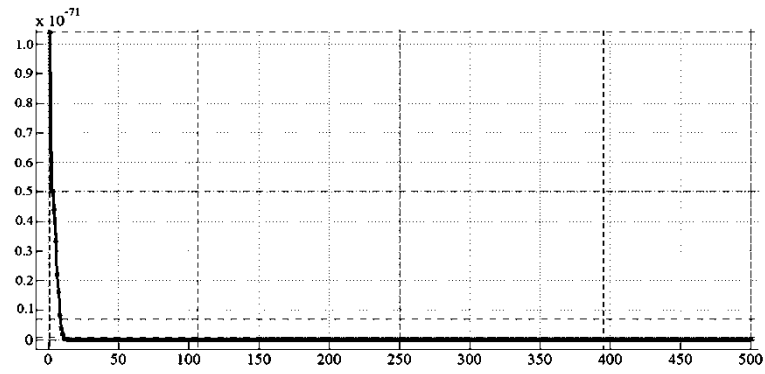
Initialization of algorithms and certain approximations are selected randomly without any specific tuning similar to the previous research work. After initialization of the algorithm the joint variables for the selected manipulator is obtained and compared with the quaternion vector method based solution. The sample data set generated by the conventional tool is presented in Table 7.20. From the generated data 100 sets of joint variables are considered for the training of the network. The comparisons of all adopted algorithms are presented in Table 7.21 with different number of hidden nodes. The predicted inverse kinematic solutions from the adopted models are compared with the actual solution and later the mean square error is calculated. The best mean square error is presented in bold letters (see Table 7.21). Figure 7.35 (a), (b), (c), (d) and (e) shows the selected best mean square curve of MLPGA for all joint variables. Similarly best chosen mean square error of other adopted algorithm is presented in Table 7.21.



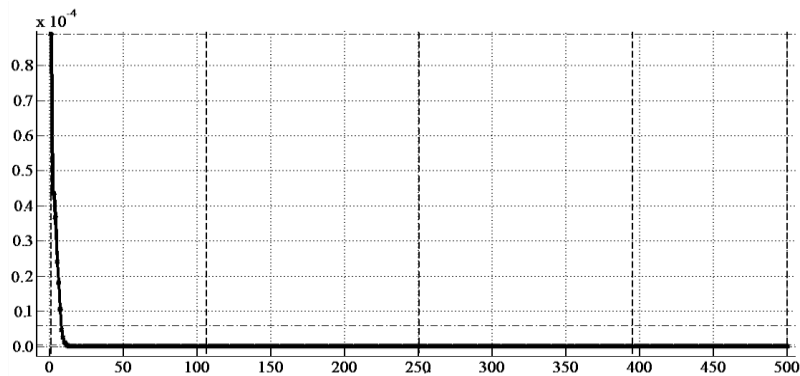
(a) Best mean square error for θ_1



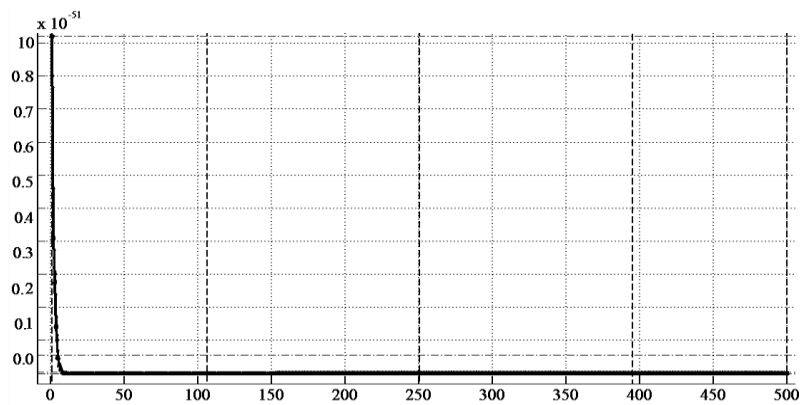
(b) Best mean square error for θ_2



(c) Best mean square error for θ_3



(d) Best mean square error for θ_4



(e) Best mean square error for θ_5

Figure 7.35 (a), (b), (c), (d) and (e) are mean square error curve of MLPGA for all joint angles

7.3.6 Inverse kinematic solution of 6-dof STAUBLI RX160 L manipulator

In this section, 6-dof STAUBLI RX160 L robot manipulator is adopted for the inverse kinematic solution. The detail explanation of the adopted robot manipulator is presented in chapter 3. The mathematical modelling using conventional tool of the adopted robot manipulator is given in chapter 4. Using the kinematic equations of the adopted

manipulator the training data sets are prepared for the prediction of invers kinematic solution. The work is perform in the MATLAB 2013 a. All joint variables and end effector positions are calculated using the kinematic equations. The prepared data sets are presented in Table 7.23 which is later used as an input for the training of the neural network models. Once the training is completed, the desired output is compared with the actual data sets. Hence the mean square error from the desired value and actual value has been calculated. The configuration of the MLPNN with different number of hidden neurons is presented in Table 7.22.

Table 7.22 Configuration of MLPNN

Sl. Parameters	Values taken
1 Learning rate	0.06
2 Momentum parameter	0.31
3 Number of inputs	3
4 Number of output	6
5 Target datasets	500
6 Testing datasets	200
7 Training datasets	300

The configuration of the neural network model from Table 7.22 gives the different parameters for the training the neural network models. Based on the above configurations the prediction of inverse kinematic solution is done. Later the other parameters such as weight and bias are updated using the back propagation algorithm. The experimental results and comparison with other hybrid models are presented in Table 7.24. The adopted algorithms for the prediction of inverse kinematic solution are similar to the previous work. The comparison has been made on the basis of mean square error of the solution. There is no specific tuning of the associated parameters for the training of ANN models. The desired joint variables and end effector positions are presented in Table 7.23.

Table 7.23 Desired joint variables determined through quaternion algebra

SN	Joints variables and positions determined through quaternion algebra								
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	X	Y	Z
1	109.9995	58.2048	30.9958	164.4046	105.3653	190.8809	5.9422	-6.4931	34.2919
2	57.1830	7.1038	141.7618	219.3840	115.7053	160.3292	-16.6496	-9.1300	17.8256
3	109.3382	41.0370	109.5175	182.9499	112.0530	109.3752	-1.5343	-3.7763	28.4963
4	126.0248	32.7009	67.1041	172.8914	113.6765	219.8415	-0.1470	-4.4910	33.5340
5	136.5495	47.7934	94.1527	14.0607	110.3387	216.8484	4.3838	-64.9257	37.5117
6	103.3780	123.7474	59.1928	130.5230	112.9944	120.9618	19.0998	-1.2647	50.2394
7	129.4171	34.1353	106.0208	64.6734	118.2626	53.1042	-21.0814	-44.0966	56.9321
8	77.5296	64.7463	101.2878	157.0967	116.5815	140.5798	-5.8106	-3.6520	41.8311
9	40.9348	87.3007	111.7118	51.4675	105.8061	207.2419	-48.2125	-26.5668	52.5295
10	64.7363	119.7839	89.1456	12.7334	117.3080	103.4015	5.2430	-65.0266	36.8321

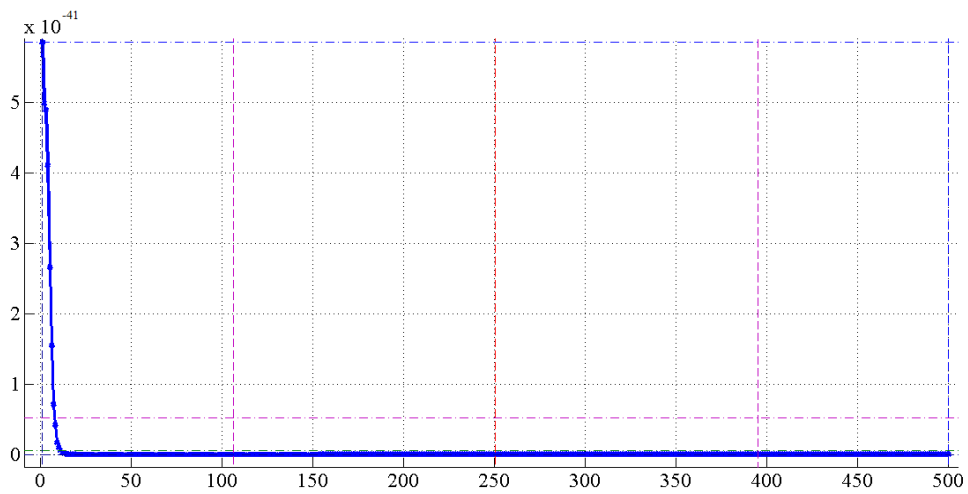
Table 7.24 Mean square error for all training samples of hybrid MLPNN

Output	Algorithms	Number of Hidden Nodes					
		4	11	17	20	27	30
θ_1	MLPBP	0.01e-02	3.09e-01	7.63e-02	7.16e-03	0.11e-03	6.11e-01
	ANFIS	5.44e-04	6.78e-02	0.18e-03	5.07e-03	4.99e-04	7.19e-03
	MLPGA	1.23e-78	8.54e-71	3.85e-89	1.84e-75	0.00	0.00
	MLPTLBO	8.34e-24	8.70e-27	3.88e-27	6.03e-30	6.78e-80	1.25e-75
	MLPCIBO	2.59e-12	8.56e-16	2.66e-15	5.94e-14	2.18e-18	1.99e-19
	MLPPSO	5.70e-14	3.12e-14	7.25e-18	2.51e-16	7.99e-16	6.35e-22
θ_2	MLPBP	7.24e-03	2.11e-01	9.78e-03	7.22e-02	0.22e-03	1.70e-01
	ANFIS	9.26e-03	4.99e-04	6.78e-01	7.24e-01	6.00e-03	1.08e-03
	MLPGA	3.55e-99	3.96e-67	0.00	0.00	0.00	0.00
	MLPTLBO	1.01e-69	6.67e-81	3.25e-80	8.32e-90	4.47e-91	5.89e-80
	MLPCIBO	6.12e-55	4.47e-41	3.21e-50	1.09e-49	0.00	0.00
	MLPPSO	6.32e-41	8.59e-40	3.39e-45	5.14e-49	7.43e-51	9.81e-45
θ_3	MLPBP	2.14e-04	3.77e-03	4.99e-01	6.17e-05	0.16e-01	7.33e-02
	ANFIS	8.33e-01	6.13e-03	7.49e-04	9.12e-02	6.07e-03	5.42e-02
	MLPGA	1.21e-19	9.82e-22	8.31e-18	1.14e-20	8.87e-41	7.89e-79
	MLPTLBO	3.15e-09	6.12e-12	8.11e-07	1.09e-08	1.21e-09	3.85e-09
	MLPCIBO	3.21e-09	5.45e-09	3.89e-08	9.51e-09	4.16e-10	8.47e-11
	MLPPSO	1.71e-25	2.05e-17	1.33e-14	6.03e-30	2.56e-10	1.24e-11
θ_4	MLPBP	0.023	5.11e-03	4.00e-03	6.89e-02	3.71e-02	0.17e-03
	ANFIS	7.89e-02	9.88e-05	4.37e-03	6.66e-01	5.41e-03	1.03e-01
	MLPGA	1.94e-13	1.88e-26	0.00	0.00	9.94e-31	7.23e-30
	MLPTLBO	7.97e-18	1.75e-22	3.66e-24	4.47e-19	5.71e-15	8.25e-14
	MLPCIBO	2.01e-08	8.87e-07	9.91e-09	7.12e-10	1.32e-09	2.22e-10
	MLPPSO	5.72e-13	1.38e-17	6.52e-14	3.18e-13	1.29e-14	5.19e-20
θ_5	MLPBP	7.56e-04	6.89e-01	4.66e-02	6.99e-02	1.00e-03	0.01e-06
	ANFIS	8.99e-06	1.06e-03	8.49e-03	3.88e-01	7.86e-06	5.45e-03
	MLPGA	7.36e-14	2.71e-24	0.00	0.00	0.00	0.00
	MLPTLBO	3.72e-19	9.05e-29	2.97e-24	1.68e-19	1.02e-25	5.35e-24
	MLPCIBO	3.72e-03	3.27e-03	6.07e-02	1.17e-03	6.06e-03	8.94e-03
	MLPPSO	2.26e-02	2.34e-02	6.72e-03	9.25e-04	1.11e-17	2.45e-15
θ_6	MLPBP	7.66e-03	1.04e-05	6.44e-03	1.33e-02	4.99e-05	4.04e-03
	ANFIS	4.71e-03	9.08e-03	7.91e-03	1.05e-02	6.03e-01	8.44e-03
	MLPGA	6.04e-13	4.23e-15	7.75e-11	7.06e-19	0.00	0.00
	MLPTLBO	6.45e-17	6.51e-18	2.24e-15	6.23e-14	3.21e-15	3.15e-17
	MLPCIBO	1.47e-05	4.69e-06	1.20e-08	2.04e-10	1.98e-11	2.49e-18
	MLPPSO	6.63e-18	4.21e-21	6.04e-13	4.23e-15	7.75e-11	7.06e-19

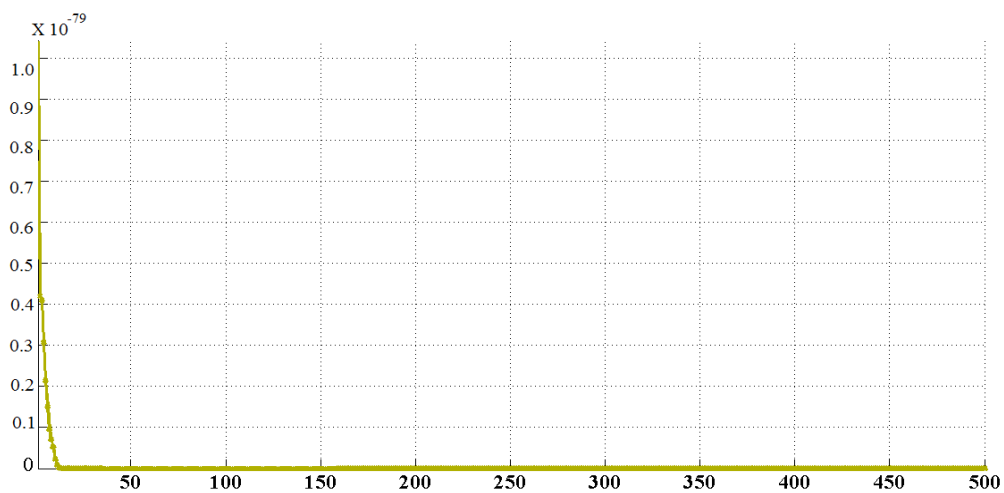
The experimental results and comparison of all adopted algorithms with several different numbers of hidden nodes are presented in Table 7.24. The perform work is similar to previous sections. The use of fuzzy sets with the neural network models is producing better results as compared to MLPNN model. The results for MLPBP and ANFIS are given in Table 7.24. The updating of the weight and bias using back propagation algorithm is slow and stagnate at local optimum point. Moreover, back

propagation algorithm consumes more computational time as compared to ANFIS and other hybrid algorithms. Therefore, to increase the exploration and exploitation ability, it is required to fuse optimization algorithm with MLPNN model.

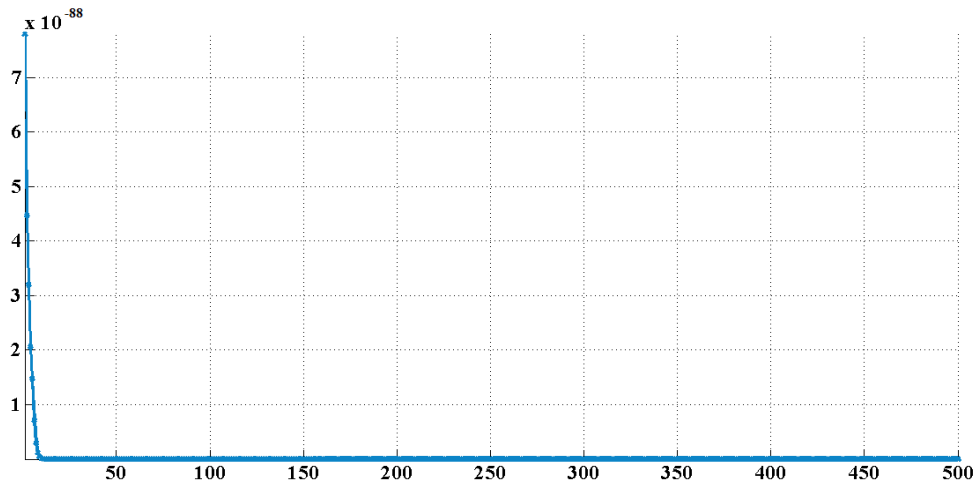
The results of the hybrid ANN models are presented in Table 7.24. The best mean square error is specified in bold letter for all adopted algorithms. Figure 7.36 gives the overall best performance of the hybrid model using genetic algorithm. Figure 7.36 (a), (b), (c), (d) and (e) shows the selected best mean square curve of MLPGA for all joint variables. Similarly best chosen mean square error for all other algorithms can be obtained from Table 7.24.



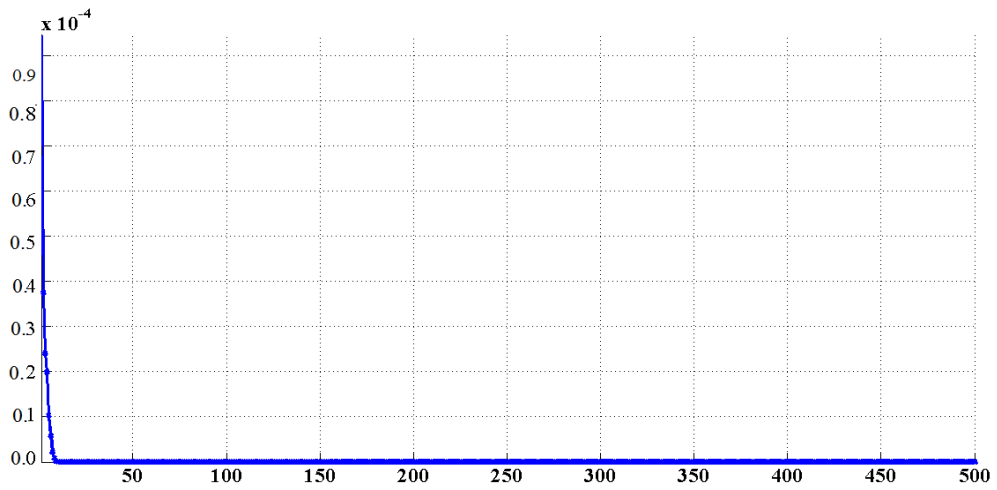
(a) Best mean square error for θ_1



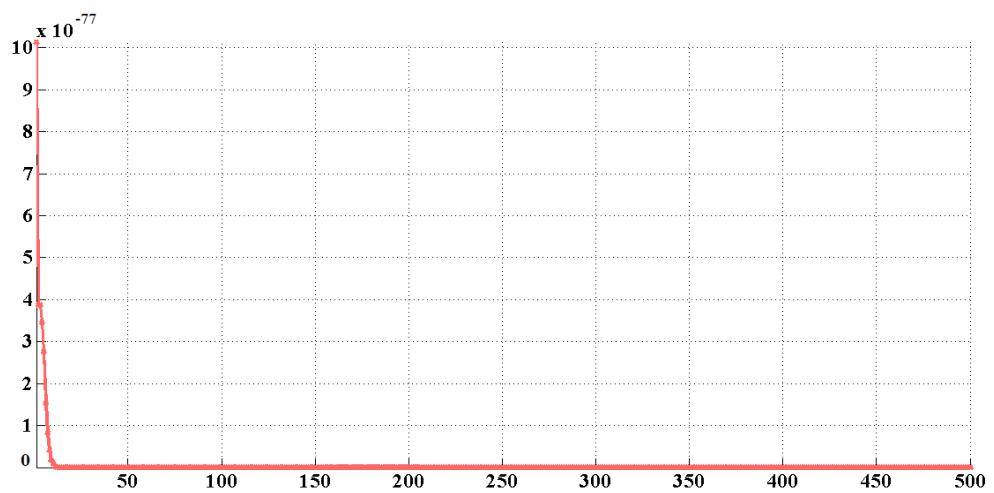
(b) Best mean square error for θ_2



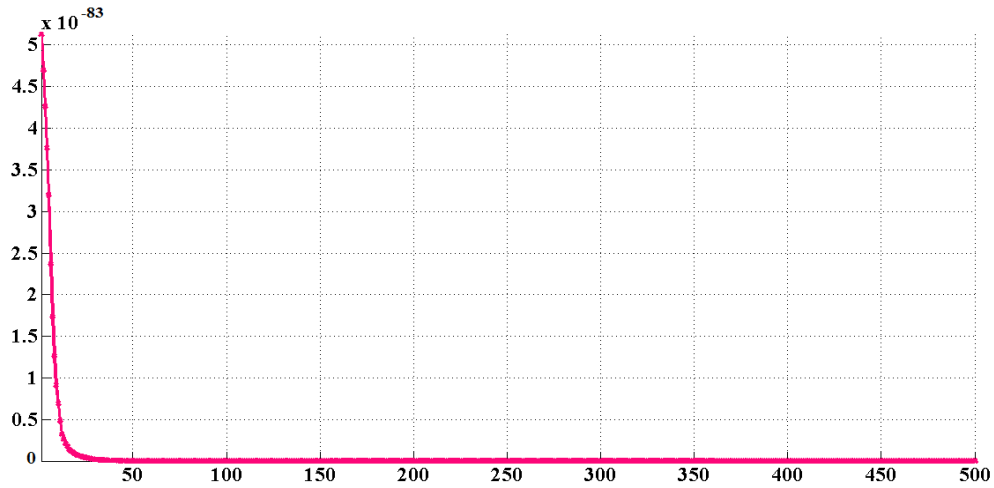
(c) Best mean square error for θ_3



(d) Best mean square error for θ_4



(e) Best mean square error for θ_5



(f) Best mean square error for θ_6

Figure 7.36 (a), (b), (c), (d) and (e) are mean square error curve of MLPGA

7.4 Metaheuristic approach for inverse kinematics solution

Inverse kinematics solution of robot manipulators has been considered and developed different solution scheme in last recent year because of their multiple, nonlinear and uncertain solutions. Optimization methods can be applied to solve inverse kinematics of manipulators and or general spatial mechanism. Basic numerical approaches like Newton-Raphson method can solve nonlinear kinematic formulae or another approach is predictor corrector type methods to assimilate differential kinematics formulae. But the major issues with the numerical method are that, when Jacobian matrix is ill conditioned or possess singularity then it does not yield a solution. Moreover, when the initial approximation is not accurate then the method becomes unbalanced even though initial approximation is good enough might not converge to optimum solution. Therefore optimization based algorithms are quite fruitful to solve inverse kinematic problem. Generally these approaches are more stable and often converge to global optimum point due to minimization problem. The key factor for optimization algorithms is to design objective function which might be complex in nature. On the other hand, metaheuristic algorithms generally based on the direct search method which generally do not need any gradient based information. In case of heuristic based algorithms local convergence rate is slow therefore some global optimization algorithms like GA, TLBO, PSO etc. can be gainfully used.

Therefore, the key purpose of this work is focused on minimizing the Euclidian distance of end effector position based resolution of inverse kinematics problem with comparison of adopted optimization algorithms obtained solution for 5-dof and 6-dof revolute robot manipulators. The objective function (fitness function) mathematical

modelling is given in previous chapter. The result of each algorithm is weighed by using inverse kinematics equations to obtain statistics about their error. In other words, Cartesian coordinates have been used as an input to calculate each joint angle. Finally 4th order spline is used to generate trajectory and corresponding joint angles of manipulator using optimization algorithms and quaternion for 5-dof manipulator. The mathematical modelling of the adopted configuration of robot manipulators and detail derivation of forward and inverse kinematics of 5-dof manipulator using quaternion algebra is given in chapter 3. The experimental results as obtained from simulations are discussed elaborately in later section.

7.4.1 Inverse kinematic solution for 4-dof SCARA manipulator

In this section, inverse kinematic solution of 4-dof SCARA robot manipulator is presented. The mathematical formulations and background of the research topic has been presented in chapter 6. The formulation of the objective function (fitness function) is obtained using position and orientation based error method. The detail description of the formulation of the objective function is presented in section 6.4. Using the position and orientation error subjected to the joint variables constrained is solved using several optimization algorithms. Moreover, the obtained inverse kinematic solutions are compared with the conventional solution of the inverse kinematic problem. Simulations and MATLAB programs are used to check the performance and effectiveness of the adopted optimization algorithms for the inverse kinematic solutions. Five different positions of the end effector have been considered for the comparative evaluation of the invers kinematic solutions. Five different positions of the end effector are presented in Table 7.25 using conventional tool.

Table 7.25 Five different positions and joint variables

Positions	Joint angles			
	θ_1	θ_2	d_3	θ_4
P1(102.86, 302.11, -233.33)	13.3333	14.44	83.33	200.00
P2(-16.91, 193.65, -250.00)	40.00	43.33	100.00	240.00
P3(-256.40, -295.35, -266.67)	66.66	72.22	116.66	280.00
P4(63.17, -77.59, -283.33)	93.33	101.11	133.33	320.00
P5(351.91, 167.19, -300.00)	120.0	130.13	150	360

Table 7.26 represents the comparative results of the all adopted algorithms for the evaluation of the inverse kinematic problem of 4-dof SCARA manipulator. The experiment of the adopted optimization algorithms doesn't follow any specialized tuning for the associated parameters. From Table 7.26, it can be observed that the objective function value for genetic algorithm is better than all other algorithms. On the other hand, TLBO and GWO are performing equally on the basis of function

evaluation. The optimum value for the Euclidean distance norm is 0; the obtained results for adopted algorithms are acceptable if it varies within the limit of 0.001. Therefore, it can be observed that the optimization based inverse kinematic solutions are acceptable. The comparisons on the basis of computational time for all adopted algorithms are discussed in chapter 8.

Table 7.26 Five different positions and joint variables through adopted algorithm

Positions	PSO Joint angles				Function Value
	θ_1	θ_2	d_3	θ_4	
P1	10.25	99.41	88.02	159.26	12.90
P2	40.11	-45.36	75.96	200.14	0
P3	55.89	70.25	100.94	270.87	-0.46
P4	22.56	18.96	80.23	-126.36	-290.99
P5	71.54	30.14	66.74	265.11	0
Positions	GWO Joint angles				Function Value
	θ_1	θ_2	d_3	θ_4	
P1	11.02	100.93	83.10	270.36	0
P2	14.43	152.91	77.04	265.15	-320.56
P3	42.86	16.48	55.47	124.68	0
P4	12.45	75.86	37.95	276.42	0.0094
P5	10.63	77.52	86.34	310.24	0
Positions	TLBO Joint angles				Function Value
	θ_1	θ_2	d_3	θ_4	
P1	10.22	35.68	71.24	105.26	-222.56
P2	66.89	15.73	83.14	265.66	-147.8
P3	71.59	9.31	149.22	191.36	0
P4	11.33	57.16	150.36	200.18	110.6
P5	96.85	55.48	120.30	222.59	0
Positions	GA Joint angles				Function Value
	θ_1	θ_2	d_3	θ_4	
P1	11.36	14.25	71.25	270.23	0
P2	41.25	44.11	91.63	230.55	0
P3	60.88	72.99	115.94	270.71	0
P4	-95.24	100.08	121.27	280.45	0
P5	120.29	121.56	149.56	310.29	0
Positions	CIBO Joint angles				Function Value
	θ_1	θ_2	d_3	θ_4	
P1	10.25	55.26	46.58	180.25	-120.63
P2	11.69	45.89	115.85	310.79	-1.03
P3	44.89	51.26	149.66	280.44	0.0094
P4	73.94	67.61	111.09	198.46	0.0768
P5	96.15	15.63	101.76	175.48	0

The comparison of the obtained solutions using optimization algorithms are presented in Figure 7.37 through Figure 7.41. The solution obtained through the optimization methods are compared with the quaternion based solution. All joint variables for five

positions of end effector are indicated in Figure 7.37 through Figure 7.41. Figure 7.37 gives comparative results for the position one (P1) using all adopted algorithms. For position P1 the joint variables using PSO and GWO are closed to the solution obtained through quaternion method. Similarly for positions P2 to P4, the solutions obtained through the GA are better as compared to other optimization algorithms. Therefore, the functional value and joint variables is acceptable using GA algorithm. Moreover, TLBO and GWO equally perform for the inverse kinematic solution.

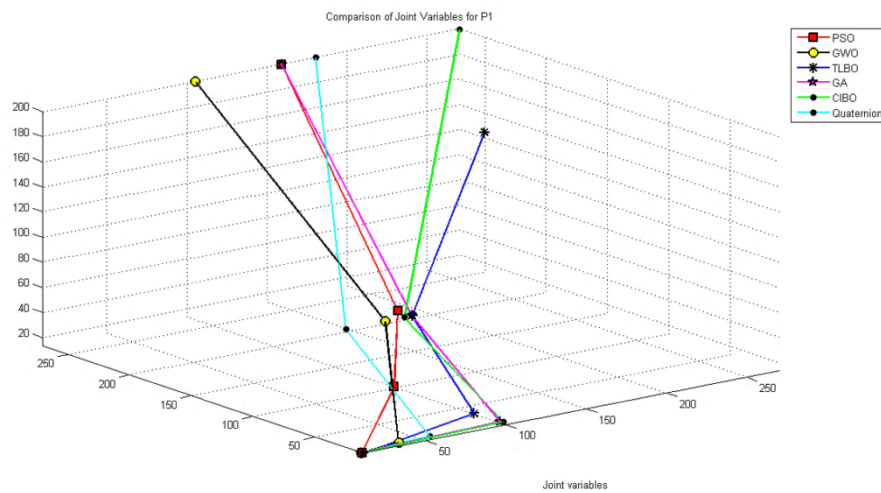


Figure 7.37 Comparison of joint variables for position 1

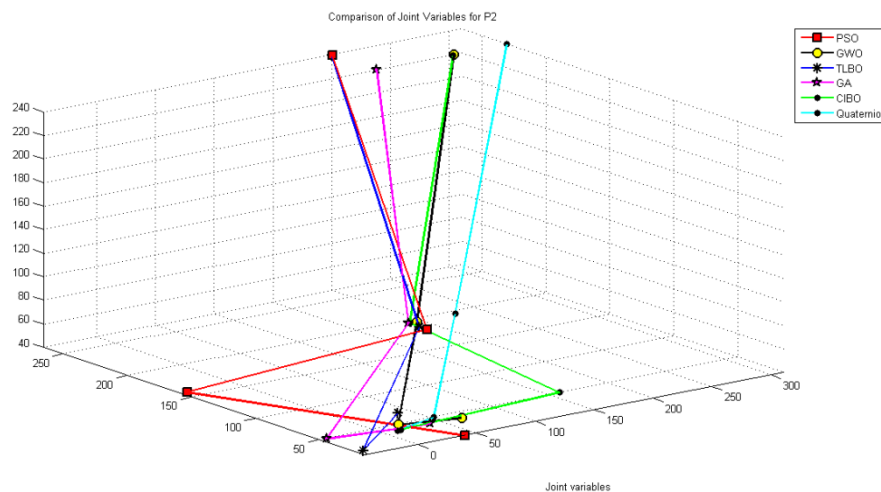


Figure 7.38 Comparison of joint variables for position 2

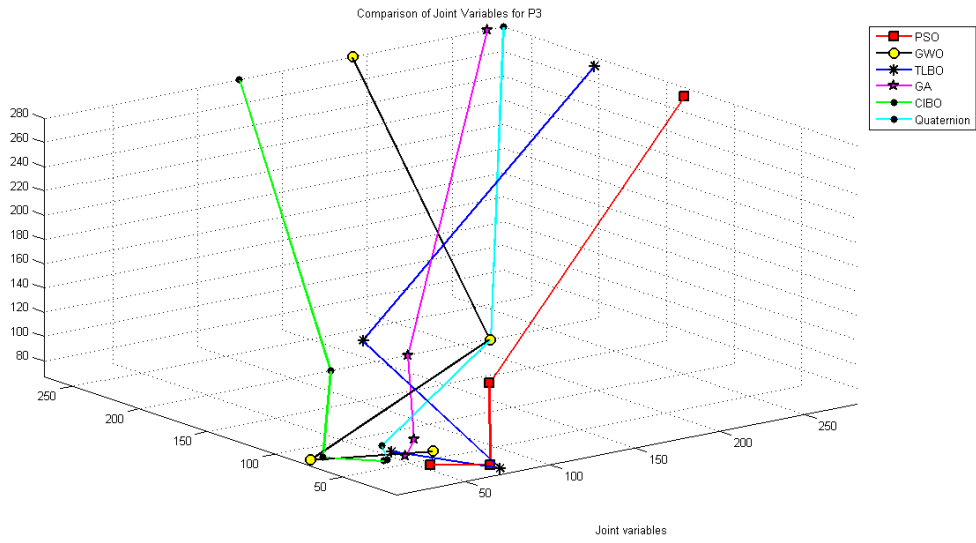


Figure 7.39 Comparison of joint variables for position 3

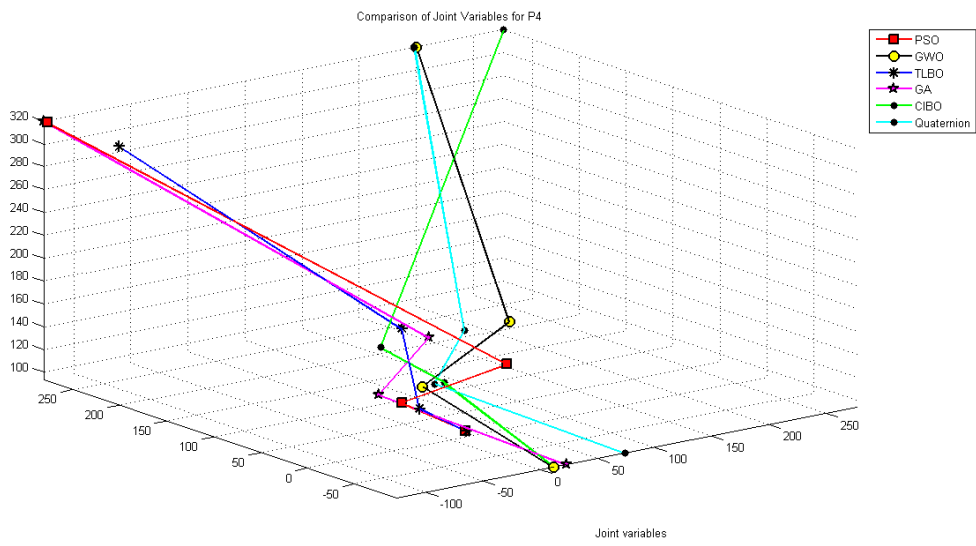


Figure 7.40 Comparison of joint variables for position 4

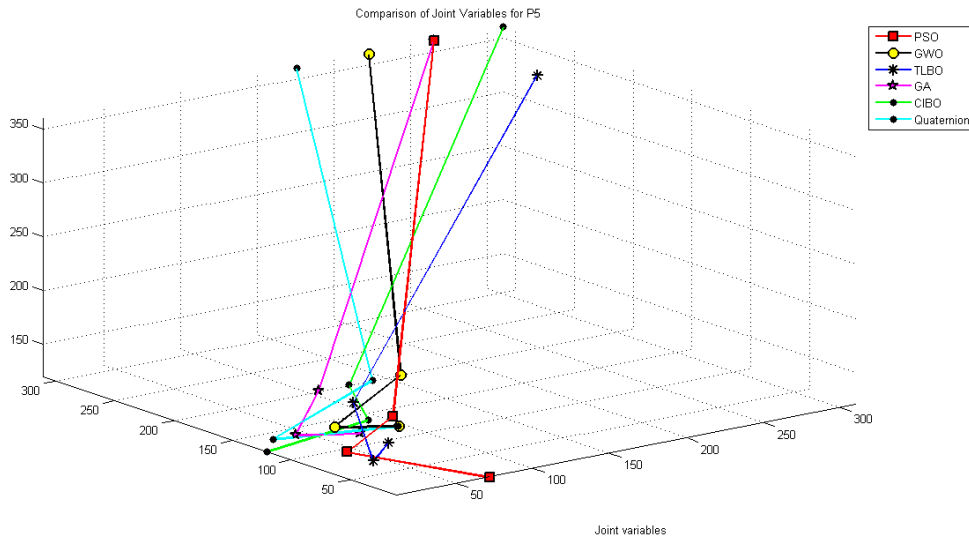


Figure 7.41 Comparison of joint variables for position 5

7.4.2 Inverse kinematic solution for 5-dof manipulator

Simulations have been made to check the performance and effectiveness of adopted optimization algorithms and comparison to solve inverse kinematic problem of 5-dof manipulator. Five different positions of end effector have been considered for inverse kinematics evaluation as presented in Table 7.27. The proposed work is performed on the MATLAB R2013a. In this work, comparison data sets were generated by using quaternion vector based method from chapter 4.

Table 7.27 Five different positions and joint variables

Positions	Joint angles				
	θ_1	θ_2	θ_3	θ_4	θ_5
P1(-76.09, 54.36, -61.94)	84.559	77.518	101.74	30.616	38.697
P2(89.69, 192.55, 90.87,)	84.791	97.25	130.44	50.771	36.428
P3(-4.24, 94.08, 97.55)	18.384	78.688	35.234	77.708	34.889
P4(29.10,154.02, -31.52)	104.43	115.47	124.11	7.3372	33.774
P5(-184.33, -43.21, 8.27)	39.177	107.13	97.052	65.672	15.374

Table 7.7 gives some of the desired data for the position of joints determined through analytical solution (Quaternion), which will further be used to calculate the difference between the adopted algorithms and analytical solution of inverse kinematic. Table 7.28 represents the comparative results of optimization algorithms for the evaluation of inverse kinematic fitness function and joint variables. Conducted experiment doesn't follow any special tuning of associated parameters of all algorithms. It is observed from Table 7.28 TLBO performing well as compared to GA on the basis of fitness evaluations for position 4. In case of the Euclidean distance norm the minimum value of the considered fitness functions is 0, the obtained result is accepted if it varies from

the optimum value by less than 0.01 and all algorithms are near to the stated value. Hence it can be understood from the obtained results that the proposed solution scheme performing quite well for metaheuristic algorithms.

Table 7.28 Five different positions and joint variables through adopted algorithm

Positions	PSO Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	21.12	71.20	54.94	2.55	90.65	0.037
P2	57.37	9.69	119.60	23.77	95.81	0
P3	3.62	93.48	67.93	28.40	103.78	-120.23
P4	65.33	95.23	70.63	33.66	104.04	-119.99
P5	74.64	14.38	29.40	15.68	104.61	0

Positions	GWO Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	47.55	35.60	52.18	55.73	92.99	0.019
P2	6.81	41.17	0.65	40.87	90.88	-314.89
P3	50.84	39.36	79.09	36.34	97.01	10.36
P4	5.03	66.25	115.93	64.46	101.52	0.88
P5	13.83	39.68	86.01	26.95	90.64	0.0003

Positions	TLBO Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	86.59	72.16	72.16	40.89	30.45	0
P2	83.87	69.89	69.89	39.60	30.76	0
P3	84.51	70.42	70.42	39.90	30.68	0
P4	85.56	71.30	71.30	40.40	30.53	0.0137
P5	87.81	73.18	73.18	41.46	30.36	0

Positions	GA Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	60.61	49.50	58.38	62.28	27.90	0
P2	88.29	34.09	14.43	15.24	51.73	0
P3	55.00	49.27	63.94	47.84	33.63	0
P4	72.59	22.68	68.29	85.88	27.04	0
P5	25.66	70.58	31.34	66.80	52.88	0

Positions	CIBO Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	60.15	98.96	65.99	-14.23	8.06	10.78
P2	35.42	97.27	56.32	-1.28	9.76	-41.03
P3	48.63	96.01	66.61	22.52	4.39	0.0012
P4	29.81	92.70	1.97	-10.17	1.15	0.931
P5	71.30	94.78	63.84	13.72	20.86	0

Table 7.29 Computational time for inverse kinematic evaluations

SN	Method	Computational time
1	TLBO	15.671s
2	GA	7.932s
3	GWO	3.45s
4	PSO	16.88s
5	CIBO	29.41s

Figure 7.42 through 7.51 represents the best function value and corresponding joint variables for all positions. These figures give the performance of the adopted algorithms for the solution of inverse kinematic problem of 5-dof manipulator. The convergence of the fitness function goes to zero error for both adopted algorithm but in case of TLBO it gives 0.013 errors for position 4. Therefore it can be say that the TLBO is performing less accurate as compared to GA. Figure 7.47 through 7.51 gives the performance of the GA for the adopted model and histograms gives the value in radians which is converted into degree and presented in Table 7.28. Using GA MATLAB toolbox the program was testes and the results converge to zero displacement error and corresponding joint variable for single run is shown in Figure 7.45 through 7.49. In this figure the adopted algorithm is producing multiple solutions for the single position but as per give termination criteria and among those generated results minimum value of joint angle has taken for the comparison. It has been observe that the convergence of the solution for GA is taking less computation time as compared to algorithms. Computational times for all adopted algorithms are given in Table 7.29. Overall computation time for the calculation of inverse kinematic solution is 15.671seconds for TLBO which is more than other algorithms, while the GA is taking only 7.932 second. Therefore it can also be compared on the basis of computational cost that quaternion algebra is taking slowest time among other adopted method.

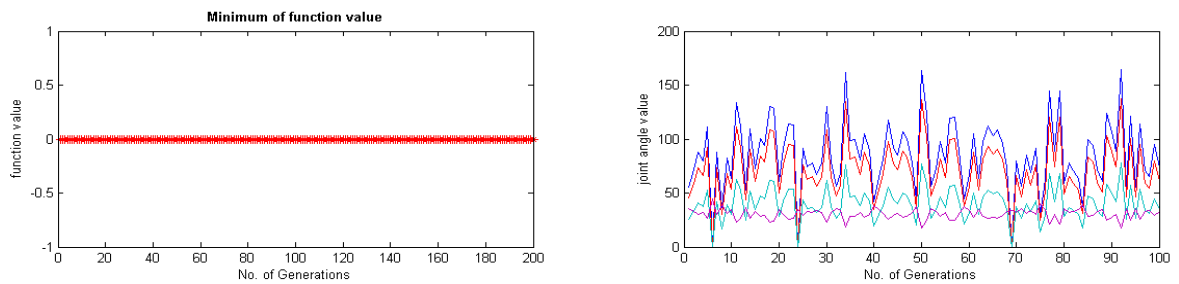


Figure 7.42 Function value and joint variables for P1

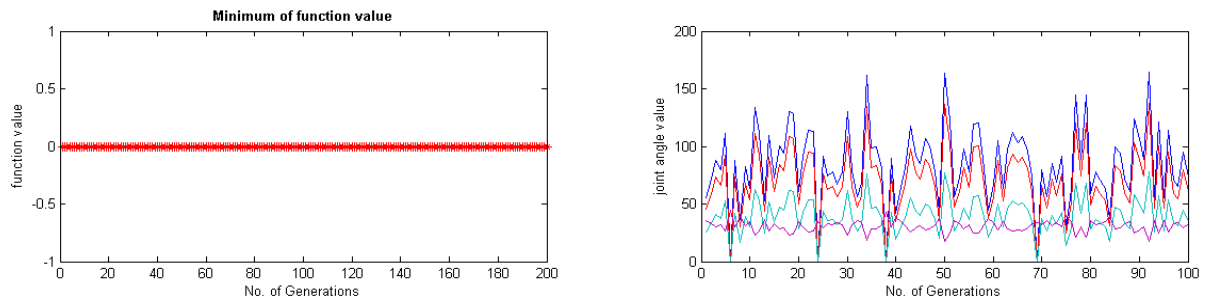


Figure 7.43 Function value and joint variables for P2

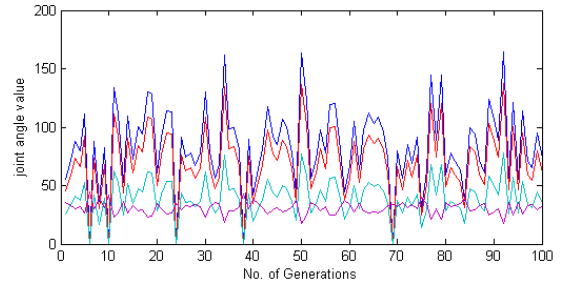
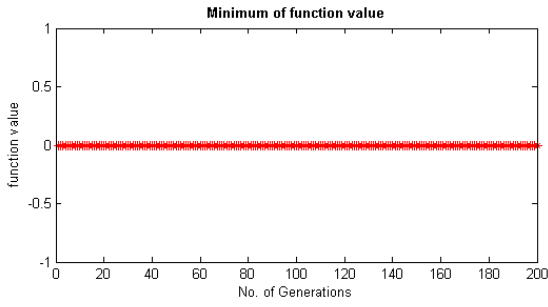


Figure 7.44 Function value and joint variables for P3

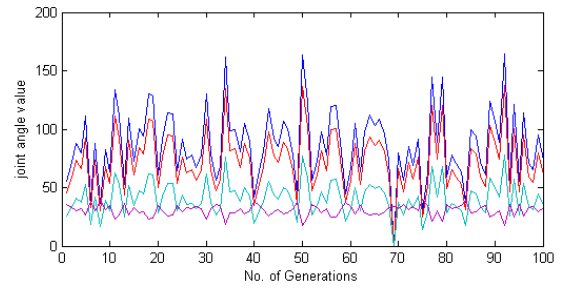
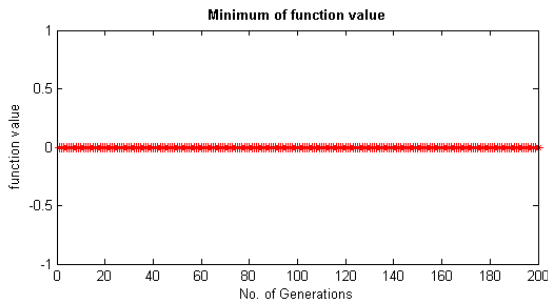


Figure 7.45 Function value and joint variables for P4

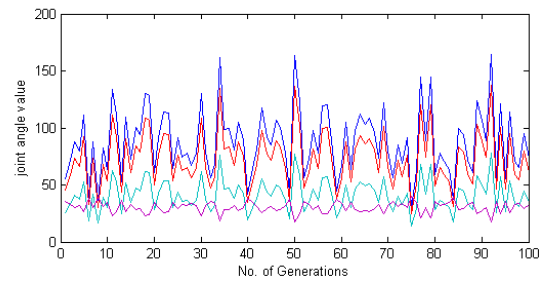
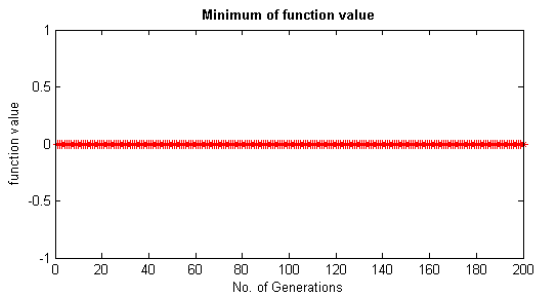


Figure 7.46 Function value and joint variables for P5

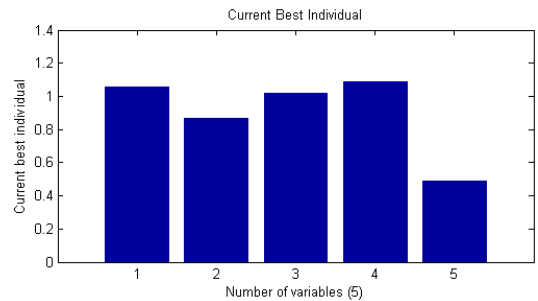
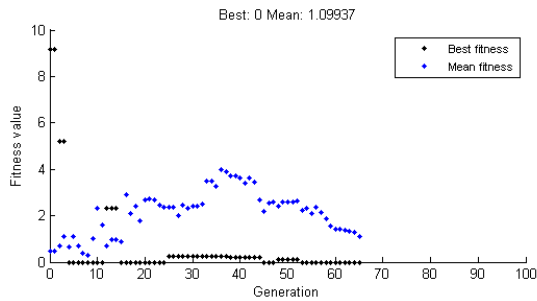


Figure 7.47 Function value and joint variables for P1

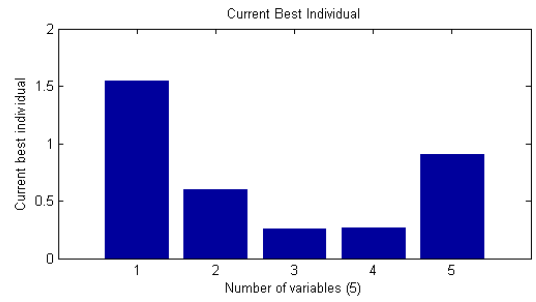
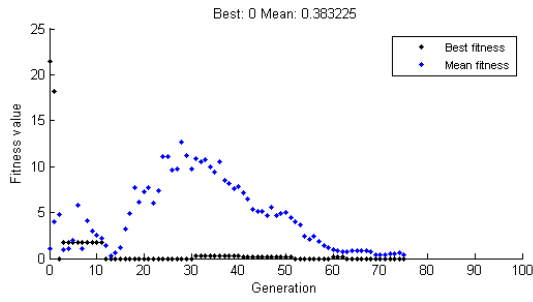


Figure 7.48 Function value and joint variables for P2

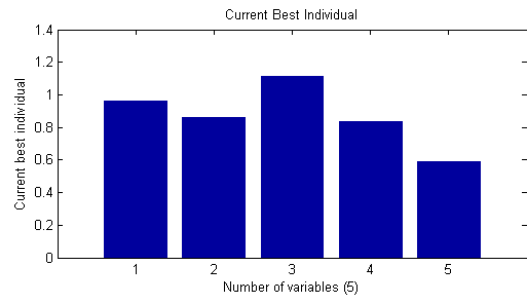
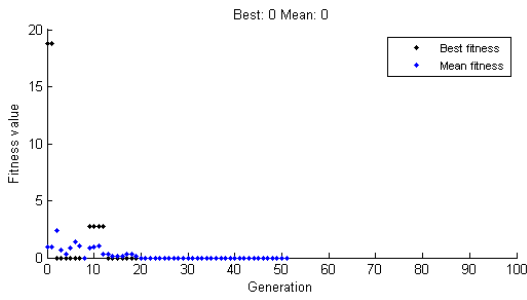


Figure 7.49 Function value and joint variables for P3

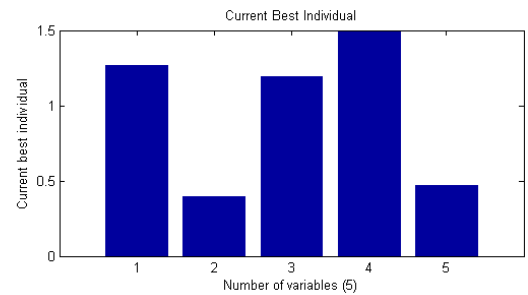
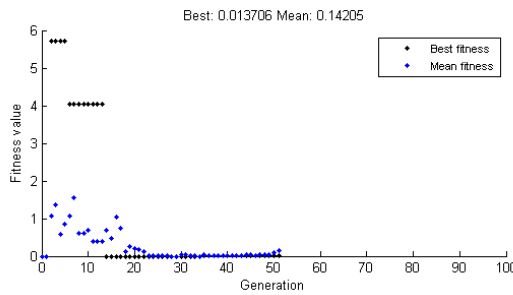


Figure 7.50 Function value and joint variables for P4

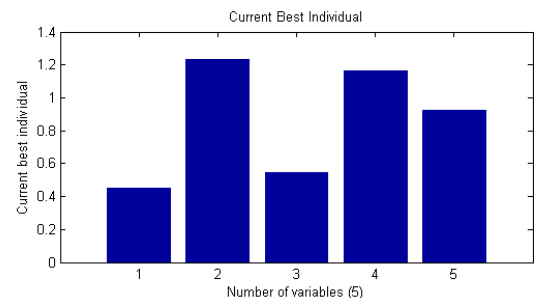
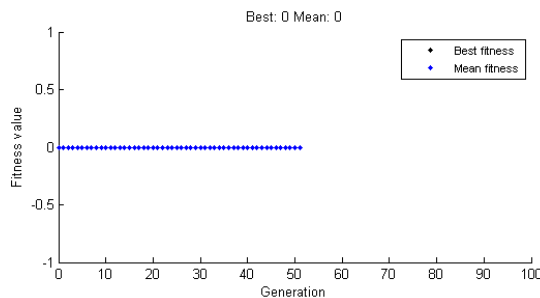


Figure 7.51 Function value and joint variables for P5

7.4.3 Inverse kinematic solution of 6-dof PUMA manipulator

In this section, type C PUMA 560 manipulator is used for the inverse kinematic analysis. Simulations studies are carried out to check the performance and efficiency of proposed GWO, PSO, CIBO, TLBO algorithm and comparison with the GA to solve inverse kinematic problem of PUMA manipulator. Five different positions of end effector have been considered for inverse kinematics evaluation as indicated in Table 7.30. The proposed work is implemented on the MATLAB R2013a. Table 7.14 provides sample of the target data for the position of end effector determined through analytical solution (Quaternion), which is used to determine the difference between the adopted algorithms and analytical solution of inverse kinematic.

Table 7.31 denotes the comparative results of GWO, PSO, TLBO, CIBO and GA algorithms for the evaluation of inverse kinematic fitness function and joint variables. The parameters of all adopted algorithms are used without any specialized tunings similar to previous work. The objective function formulations are presented in chapter 6 which is based on the position and orientation based error. The minimum functional value of the distant based norm is 0 while in this thesis the minimum functional value of objective function is allowed to the limit of 0.01 for all algorithms. It has been observed from the comparison Table 7.31; all adopted algorithms are performing precisely to achieve minimum functional value. The overall performance to get minimum function value for the objective function using GA is yielding better results as compared to other adopted algorithm.

It is also observed from Table 7.31 GWO performing well as compared to other adopted algorithms on the basis of fitness evaluations for positions P1, P2, P3 and P5 while TLBO is performing better in case of P4. Hence it can be understood from the obtained results that the proposed solution scheme performing quite well for metaheuristic algorithms.

Table 7.30 Five different positions and joint variables through quaternion

Positions (X, Y, Z)	Joint angles					
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
P1(12.46, 47.59, 104.75)	-43.90	-211.96	205.28	-91.55	67.26	-20.30
P2(-24.64, -62.39, 91.87)	23.41	-161.7	31.48	101.09	10.56	201.94
P3(-21.61, 9.30, 118.00)	72.06	-183.33	-22.34	15.56	-49.21	180.11
P4(-12.16, 8.83, 126.55)	-16.32	-206.70	90.38	-49.86	72.13	44.86
P5(-50.31, 18.71, 103.27)	103.88	-129.00	26.92	99.86	61.08	11.86

Table 7.31 Comparative results for joint variable and function value

Positions (X, Y, Z)	Joint angles by PSO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	14.34	-0.67	88.01	100.15	91.68	214.14	-1.52
P2	-82.25	-12.77	109.37	21.89	-11.47	-102.69	-10.82
P3	-151.10	-179.99	87.30	-99.23	74.24	230.05	-11.8639
P4	-51.41	2.10	87.30	81.52	73.22	87.21	-2.77
P5	32.51	-0.01	-69.32	1.03	29.46	77.48	155.2629
Positions	Joint angles by GWO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	-34.40	-10.04	97.10	109.09	-56.42	182.34	-217.09
P2	-21.60	-3.034	90.8145	66.48	-90.34	156.14	-244.5808
P3	-64.53	0.004	-0.97	14.36	22.50	-81.33	-257.8689
P4	-57.03	-179.43	86.124	71.54	9.82	40.15	-96.025
P5	-95.16	-2.595	90.05	61.59	43.10	67.71	-219.9465
Positions	Joint angles by TLBO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	-107.65	-180.83	-0.83	30.59	-51.72	81.45	-0.66
P2	-14.03	-101.84	78.160	-135.75	94.58	234.16	-186.667
P3	-36.48	-120.78	59.21	110.00	-45.21	20.89	-228.959
P4	-12.64	-100.66	79.33	74.55	83.19	7.06	-159.472
P5	-108.06	-181.18	-1.179	101.09	-44.56	241.16	-2.22257
Positions	Joint angles by GA						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	15.93	25.50	37.59	-61.29	44.87	-230.24	-1.43
P2	-21.81	13.98	11.91	99.89	71.56	55.48	-158.33
P3	-26.98	15.82	16.61	39.67	86.42	62.84	-1.9668
P4	16.33	19.05	22.16	46.82	16.76	105.68	-68.7128
P5	-83.19	2.70	85.54	57.19	37.24	-109.58	-2.3067
Positions	Joint angles by CIBO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	21.89	6.95	91.69	30.54	-84.50	133.49	-0.52
P2	-65.36	-111.21	19.85	37.19	99.15	222.56	-7.12
P3	-131.63	-169.99	79.89	150.36	102.35	64.84	-9.32
P4	-35.56	10.12	63.78	46.87	1.43	-125.14	-2.03
P5	83.36	15.091	-71.56	94.25	21.54	46.79	10.15

Figure 7.52 through 7.76 represents the best function value and corresponding joint variables for all considered positions. Visualization of the fitness function using different domains of the variables can be found in the left side (first column) of the Figure 7.52 through 7.76. Surf plot function is used in an area of given range of variables, where focus or impression is given on the XY plane depicting the global optimum range from [-50, 50]. For all surf plot in the left side of the figure represents the different properties of the considered fitness function. When looking at the inner surface area, the fitness function looks different, wherein many small valleys and peaks are visible. These peaks and valleys increases when the considered problem is higher dimensional. Moreover zooming of the surf plot can yield the desired location of the

optimum point within the small search area. In order to analyse the convergence behaviors of the adopted algorithms search history and corresponding joint angles of the manipulator is presented in Figures 7.52 through 7.76 in second columns. In this work three search agents for GWO has been considered to find out the optimum value of fitness function, similarly three sets of learner for TLBO, three particles for PSO and three genes considered for GA. It has been observed that all considered search agents or individuals for adopted algorithm having ability of exploration and exploitation of best fitness evaluations. From Figure 7.65, TLBO is giving minimum of fitness function for the position P4 as compared to other adopted algorithms while for rest of the considered positions GWO is performing better.

The convergence of the fitness function goes less than zero error for all adopted algorithm but in case of PSO it gives 155.2629 errors for position P5. Therefore it can be said that the PSO is performing less accurate as compared to other algorithms. Figure 7.67 through 7.71 gives the performance of the GA for the adopted model and histograms give the value in radians which is converted into degree and presented in Table 7.31. Using GA MATLAB toolbox the program was tested and the results converge to zero displacement error and corresponding joint variable for single run is shown in Figure 7.67 through 7.71. In this figure the adopted algorithm is producing multiple solutions for the single position but as per given termination criteria and among those generated results minimum value of joint angle has taken for the comparison.

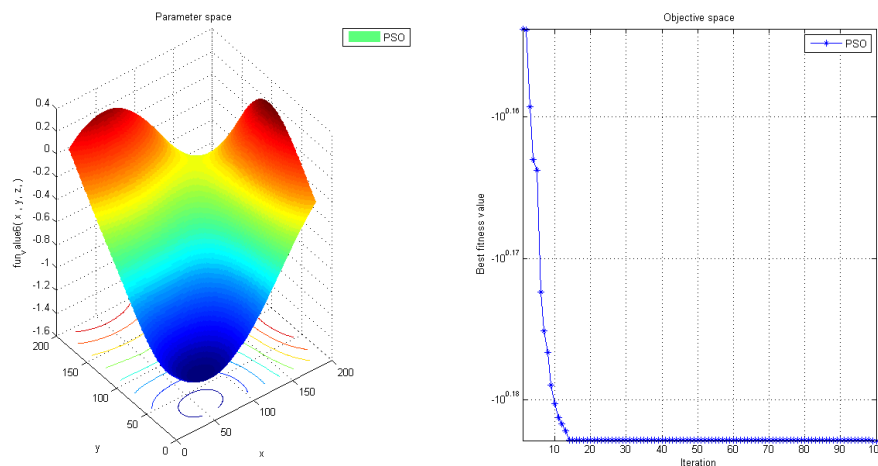


Figure 7.52 PSO Function value and joint variables for P1

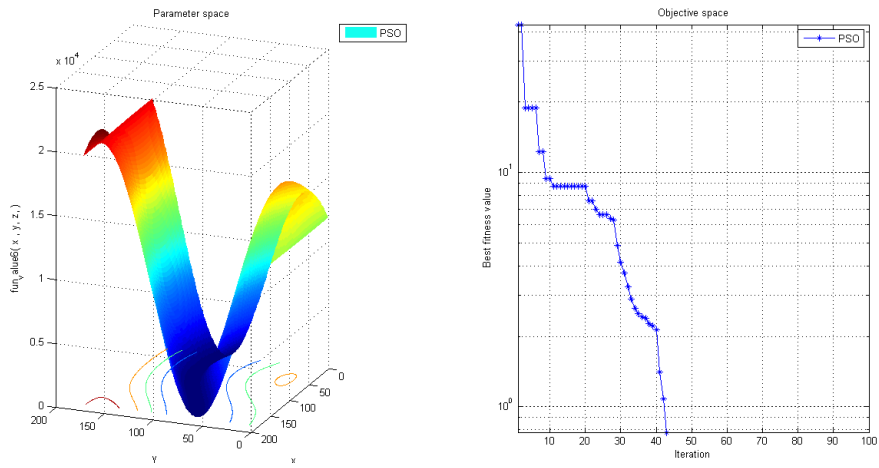


Figure 7.53 PSO Function value and joint variables for P2

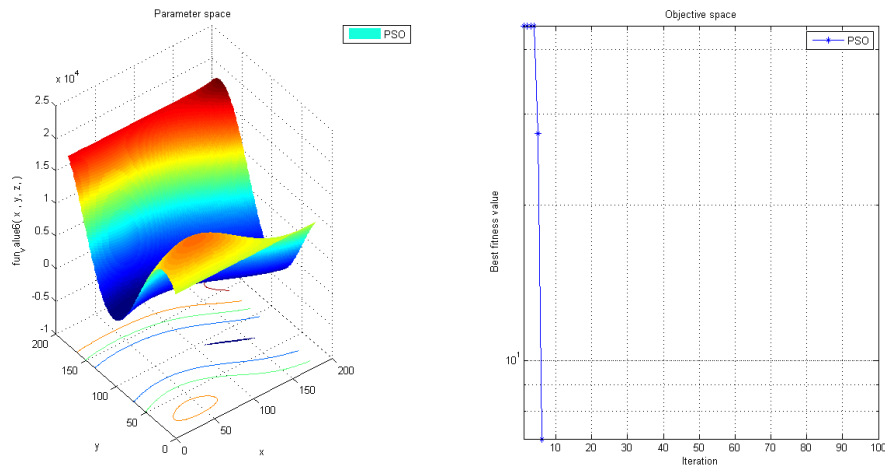


Figure 7.54 PSO Function value and joint variables for P3

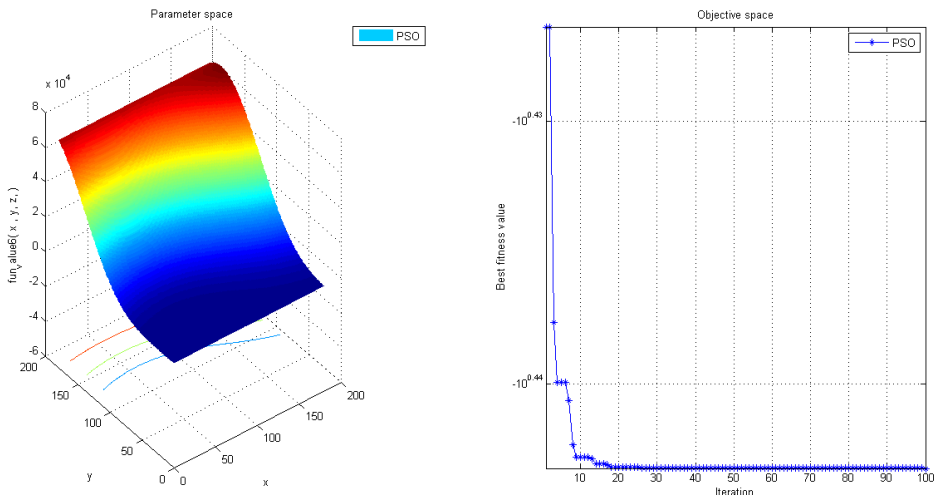


Figure 7.55 PSO Function value and joint variables for P4

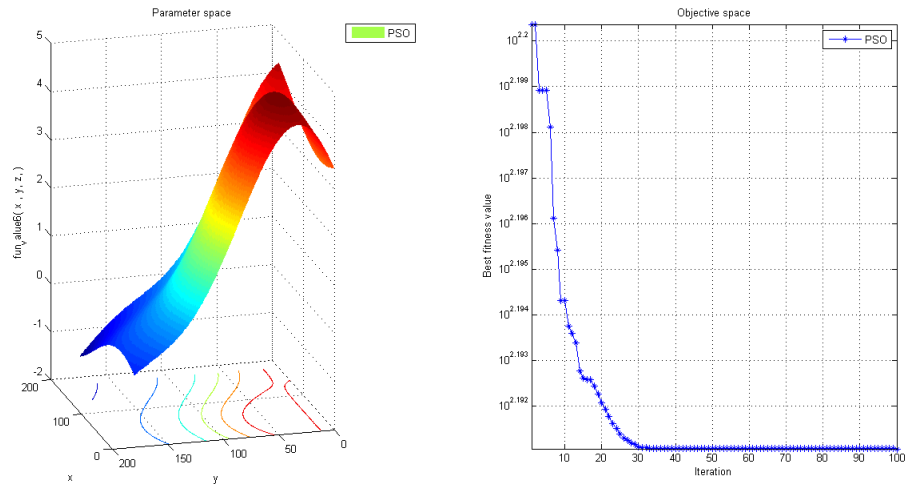


Figure 7.56 PSO Function value and joint variables for P5

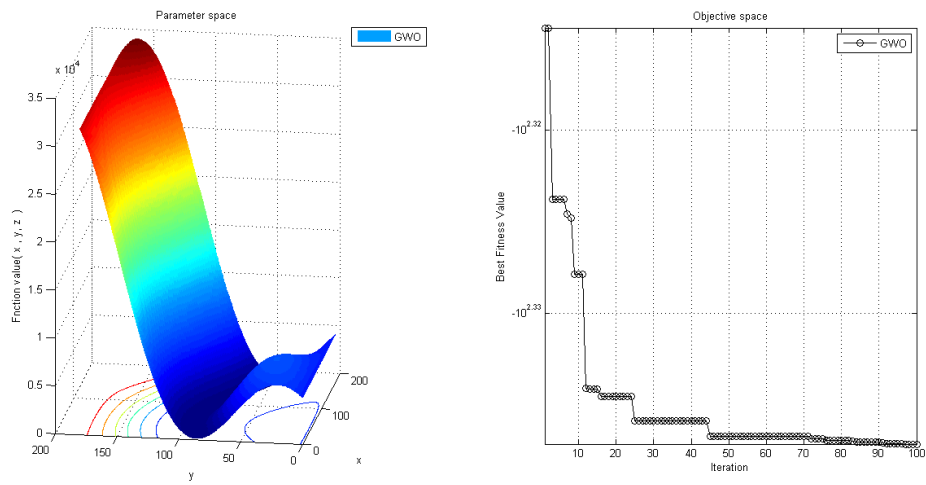


Figure 7.57 GWO Function value and joint variables for P1

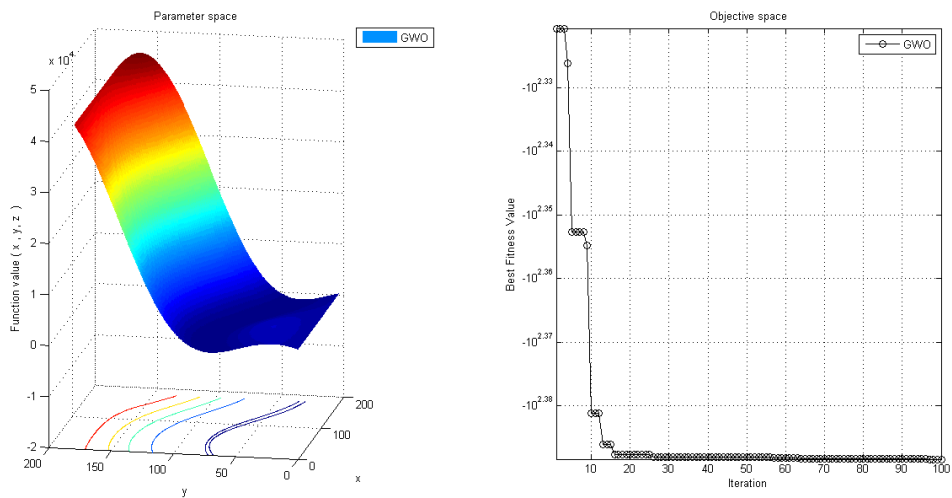


Figure 7.58 GWO Function value and joint variables for P2

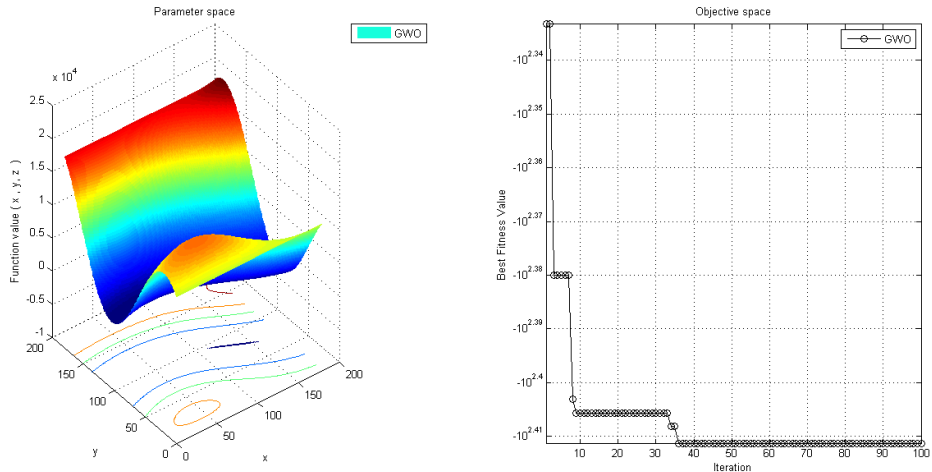


Figure 7.59 GWO Function value and joint variables for P3

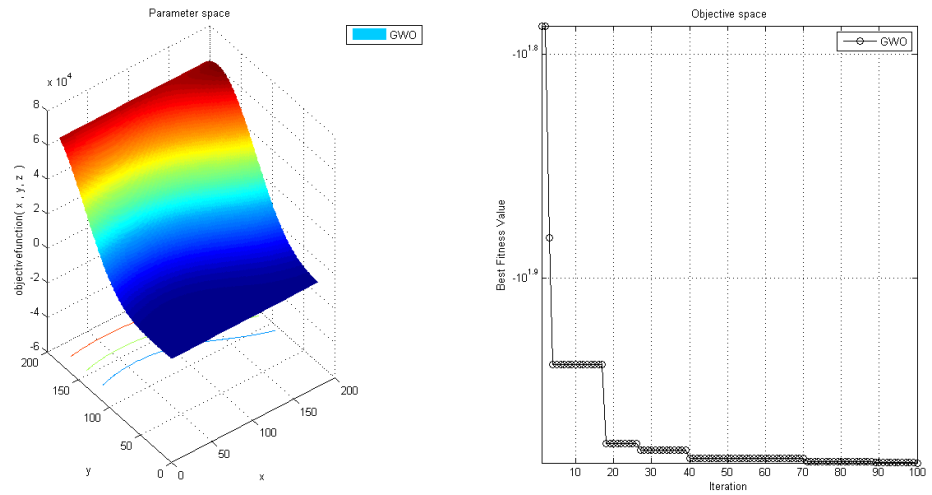


Figure 7.60 GWO Function value and joint variables for P4

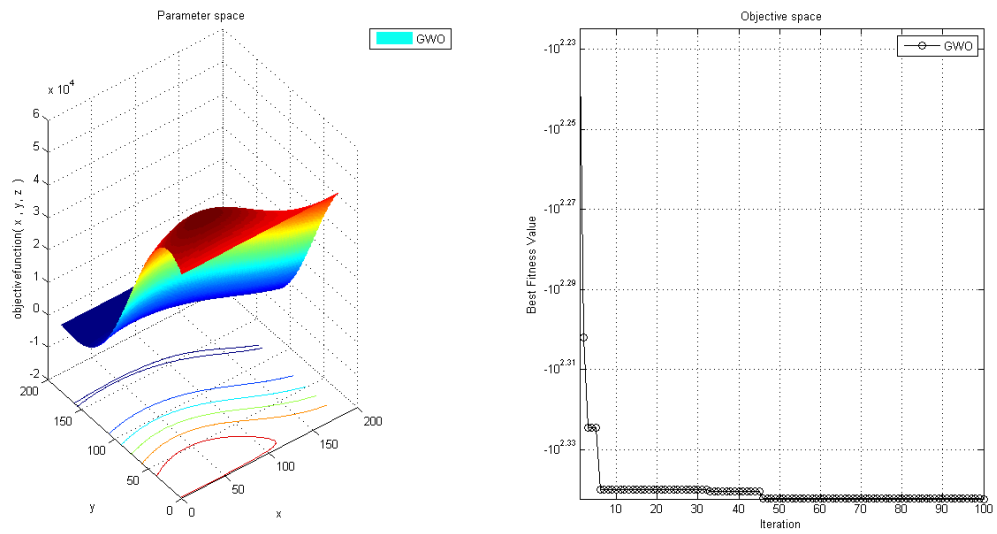


Figure 7.61 GWO Function value and joint variables for P5

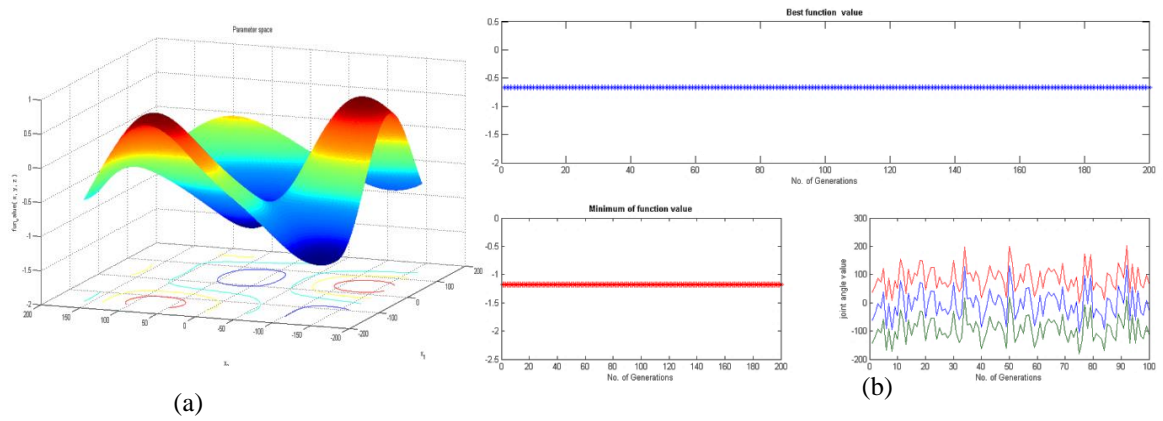


Figure 7.62 TLBO Function value and joint variables for P1

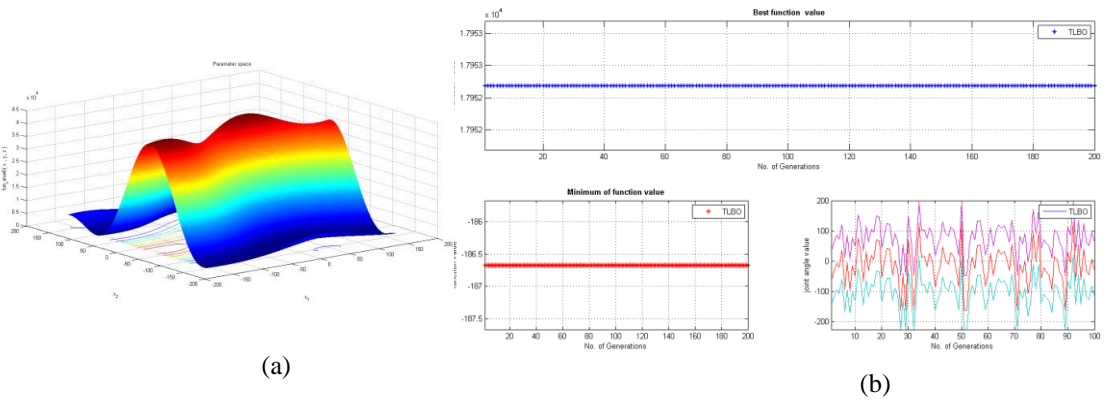


Figure 7.63 TLBO Function value and joint variables for P2

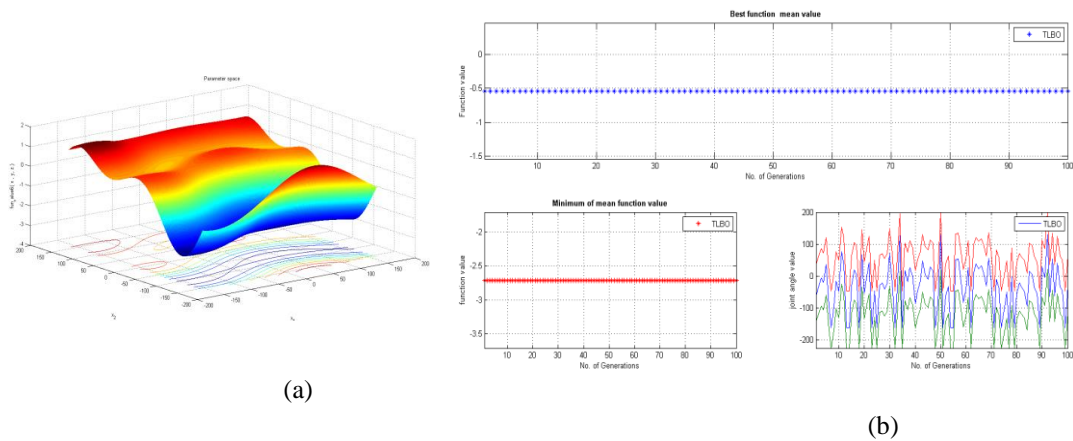


Figure 7.64 TLBO Function value and joint variables for P3

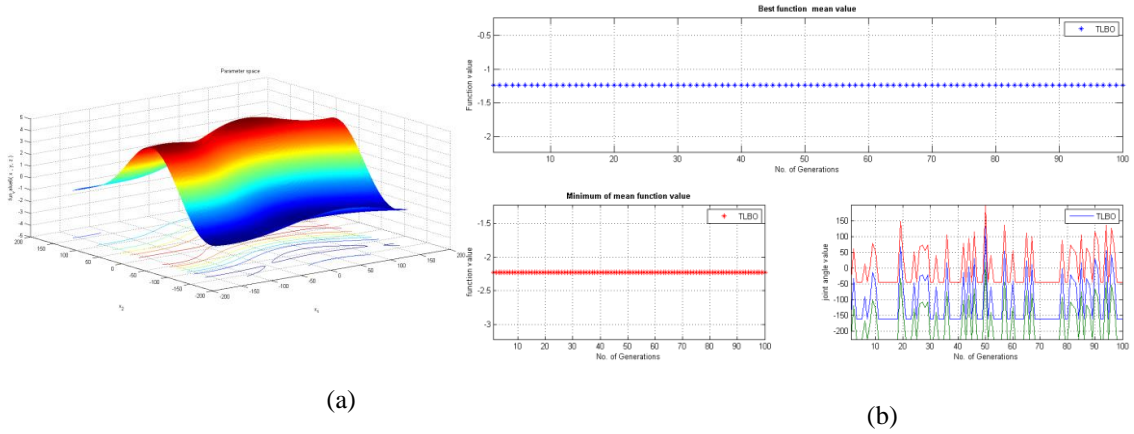


Figure 7.65 TLBO Function value and joint variables for P4

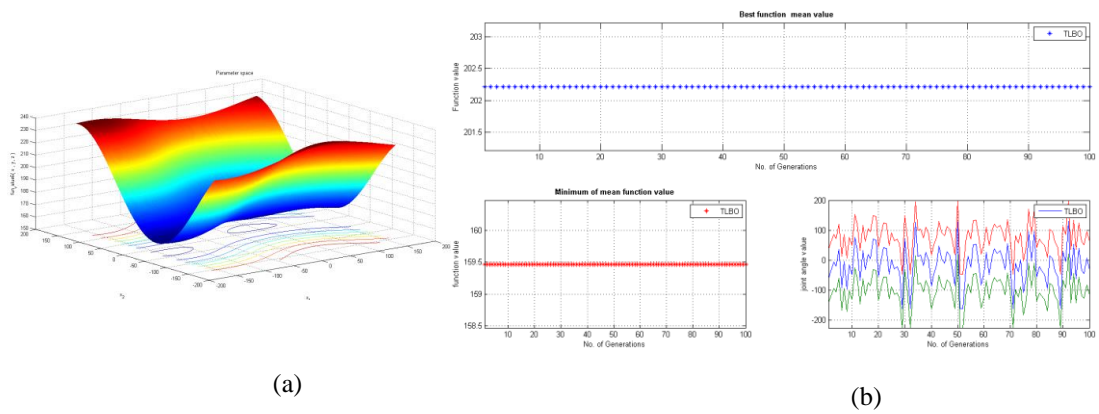


Figure 7.66 TLBO Function value and joint variables for P5

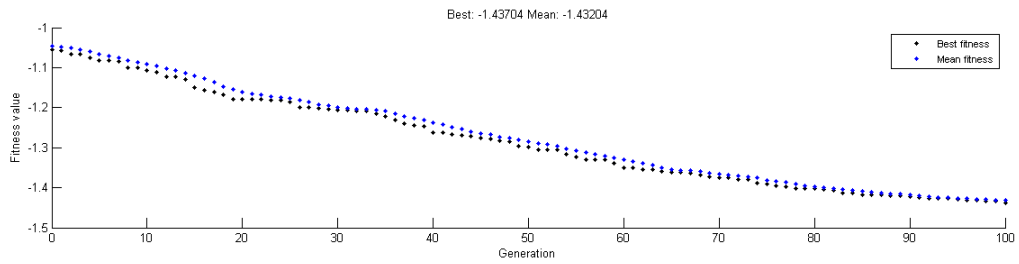


Figure 7.67 GA Function value and joint variables for P1

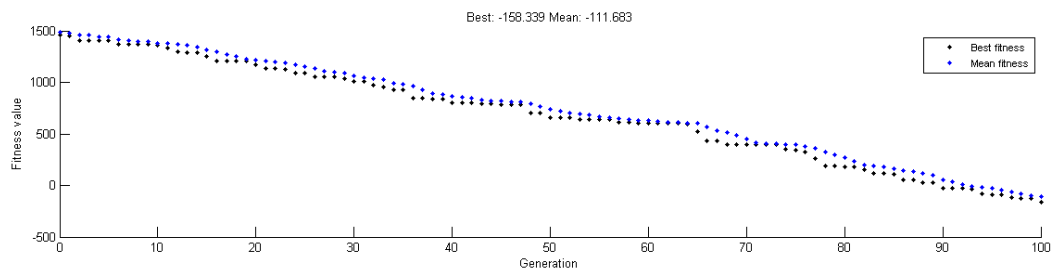


Figure 7.68 GA Function value and joint variables for P2

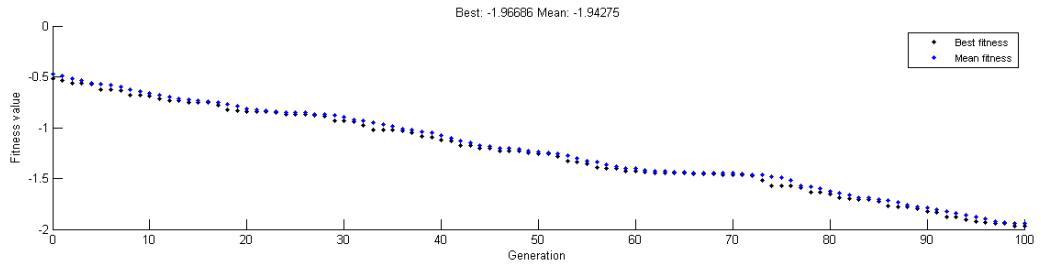


Figure 7.69 GAFunction value and joint variables for P3

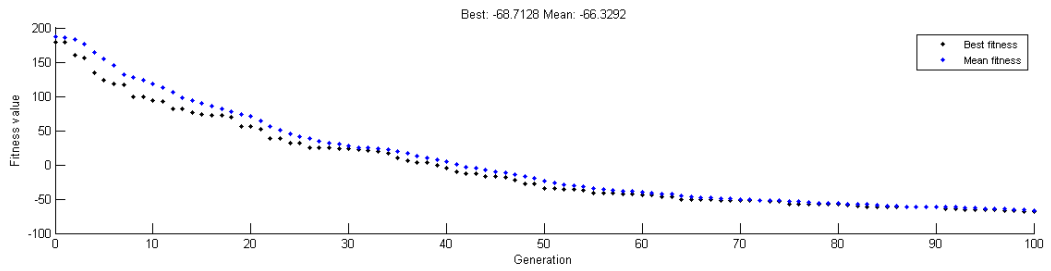


Figure 7.70 GAFunction value and joint variables for P4

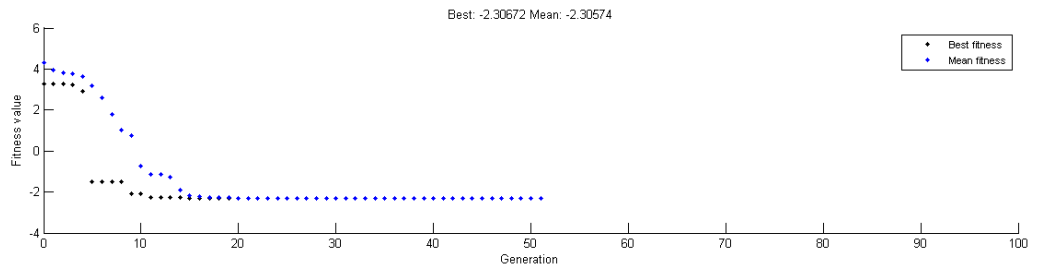


Figure 7.71 GAFunction value and joint variables for P5

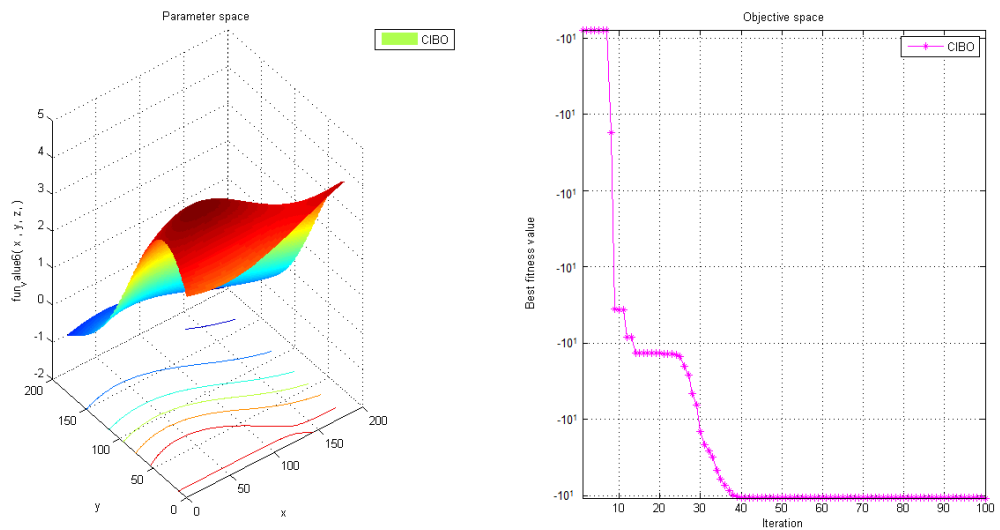


Figure 7.72 CIBO Function value and joint variables for P1

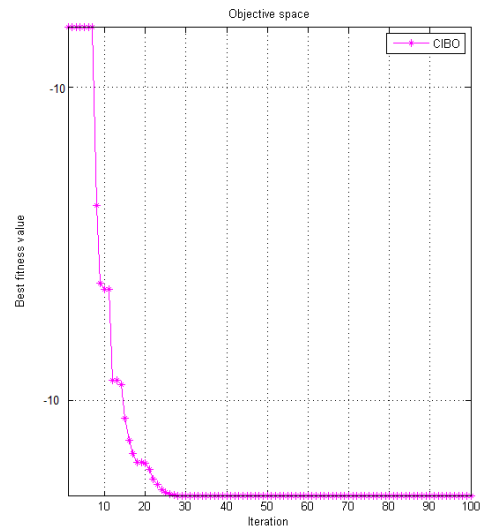
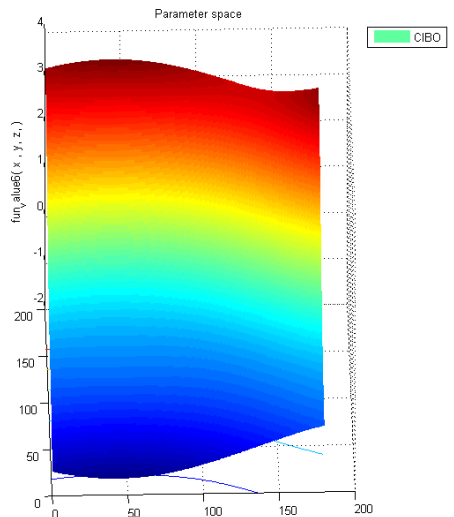


Figure 7.73 CIBO Function value and joint variables for P2

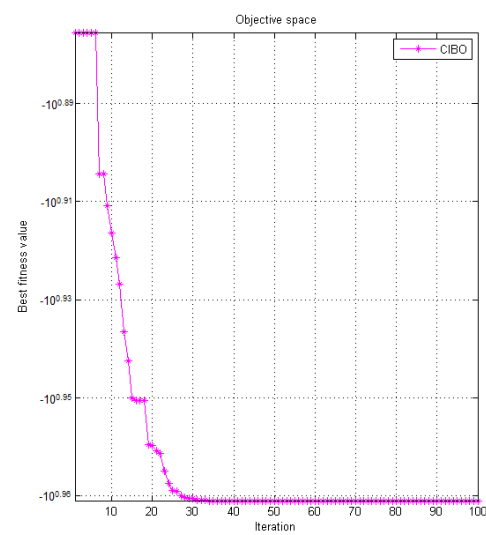
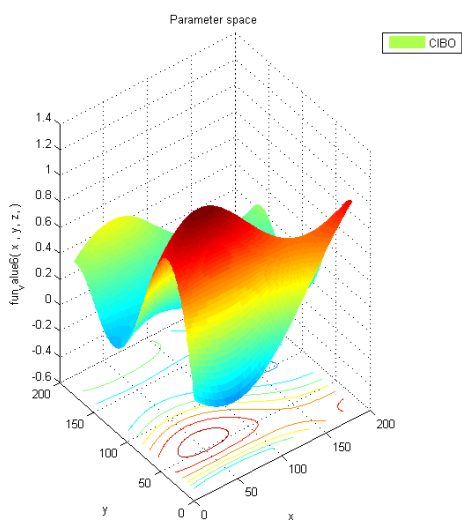


Figure 7.74 CIBO Function value and joint variables for P3

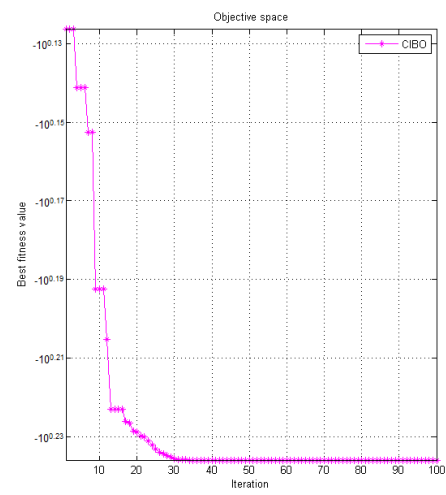
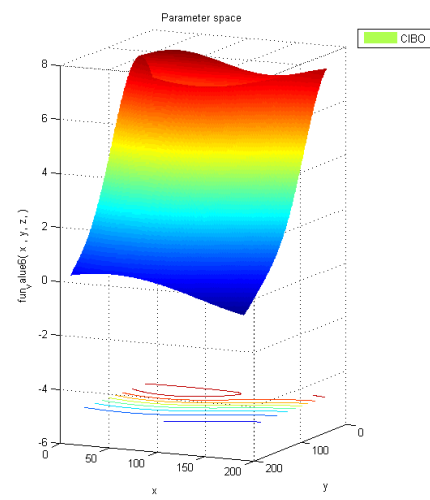


Figure 7.75 CIBO Function value and joint variables for P4

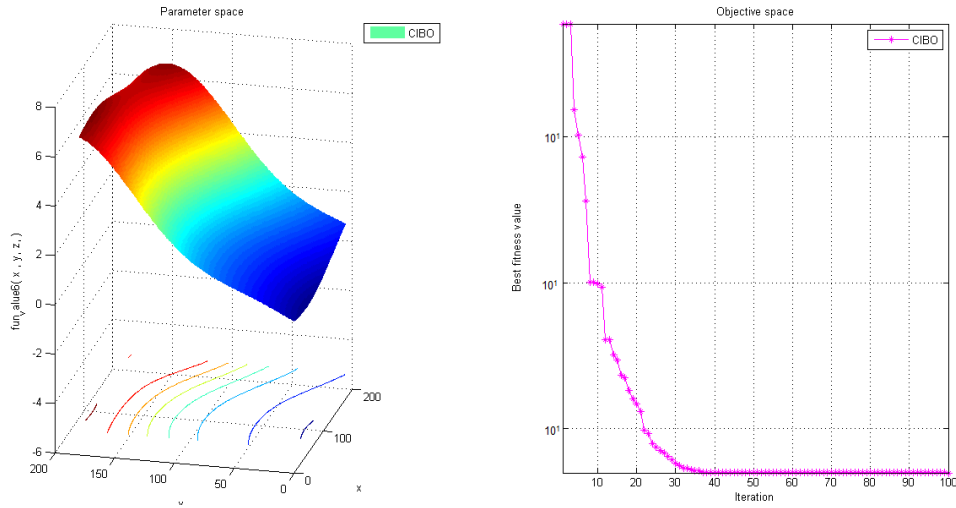


Figure 7.76 CIBO Function value and joint variables for P5

Finally it has been observe that the convergence of the solution for GA is taking less computation time as compared to GWO, PSO, and TLBO as given in Table 7.32. Overall computation time for the calculation of inverse kinematic solution is 31.864 seconds for PSO which is more than other algorithms, while the GA is taking only 5.896 seconds. Therefore it can also be compared on the basis of computational cost.

Table 7.32 Computational time for inverse kinematic evaluations

SN	Method	Computational time
1	GWO	25.821s
2	PSO	31.864s
3	TLBO	29.547s
4	GA	5.896s
5	CIBO	30.568s

In order to obtain desired joint angles for adopted manipulator the actual solution using quaternion algebra is presented in Figure 7.77 through 7.81. The comparison of all algorithms has been made on the basis of best joint angle found by the adopted algorithms. In case of position P1, P2 and P4 genetic algorithm is closer to the standard solution which is highlighted in pink line in Figure 7.77-7.78 and Figure 7.80, While in case of P5, GA, PSO and TLBO is performing similar for all joint variables.

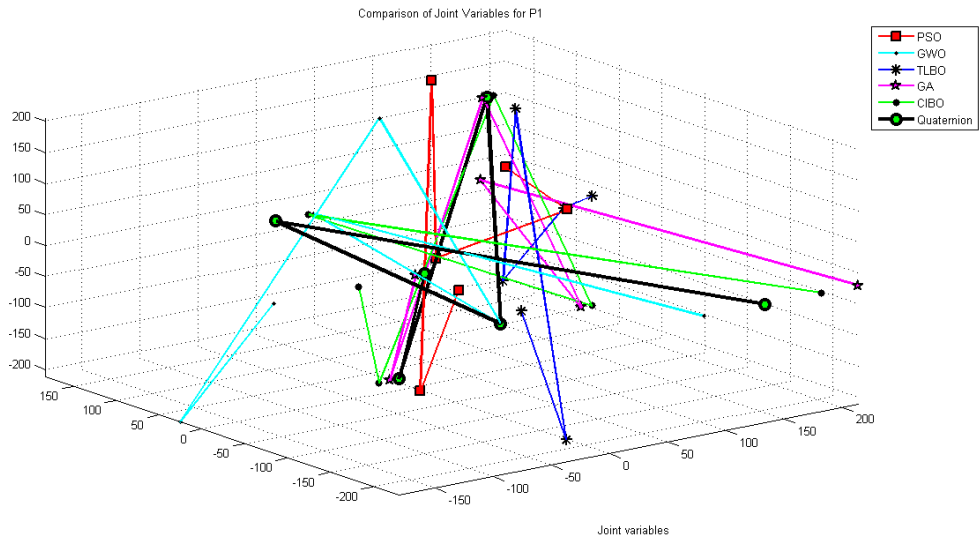


Figure 7.77 Comparison of joint variables for position 1

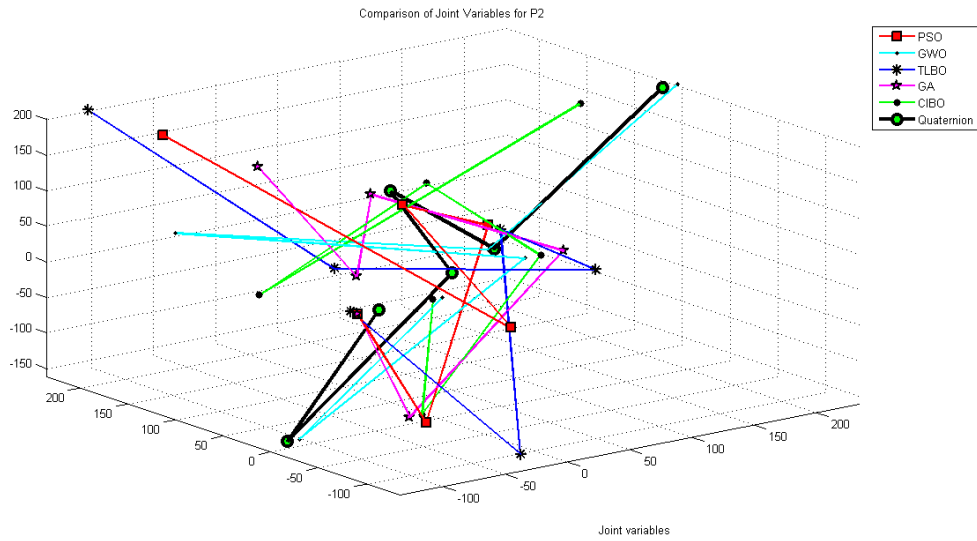


Figure 7.78 Comparison of joint variables for position 2

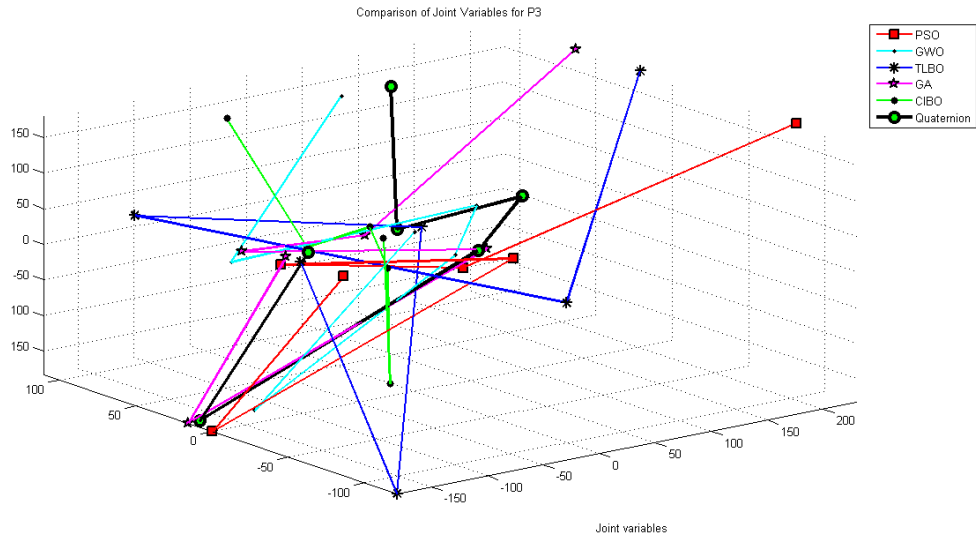


Figure 7.79 Comparison of joint variables for position 3

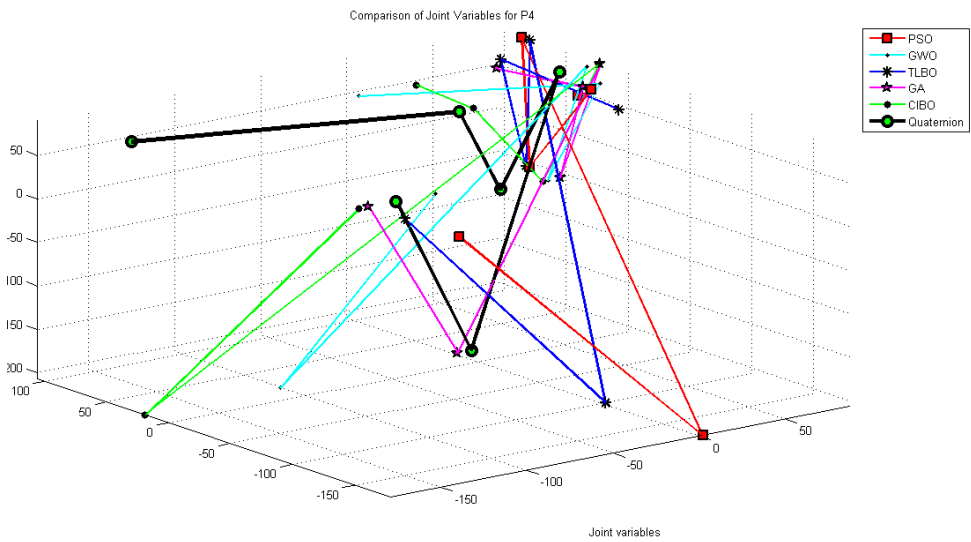


Figure 7.80 Comparison of joint variables for position 4

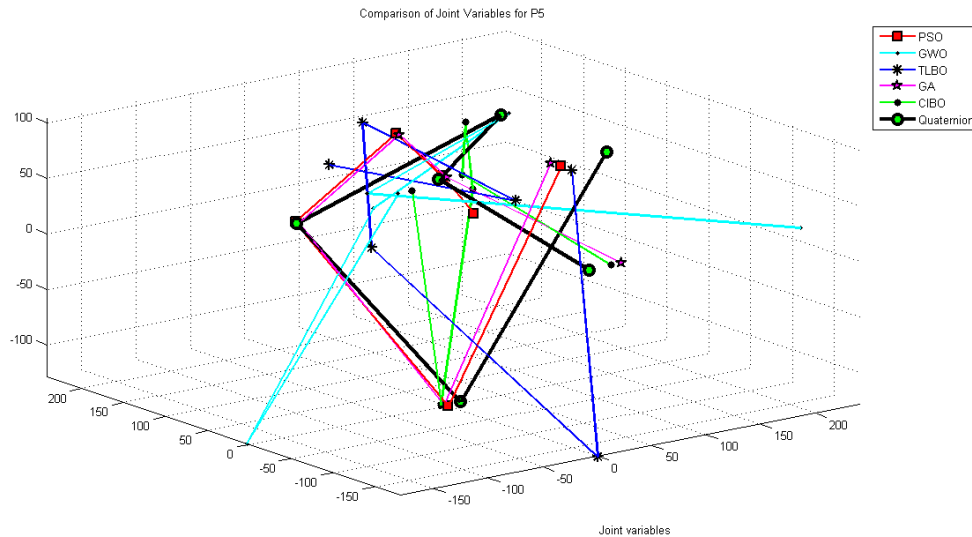


Figure 7.81 Comparison of joint variables for position 5

It has been also described that the adopted algorithm is much appropriate for constrained problems. In order to evaluate the effectiveness of the adopted techniques used, the obtained results are compared with standard quaternion solution. In the results section Table 7.31 shows comparison with the results obtained through different algorithms. In this approach forward and inverse kinematic model of the PUMA manipulator is used for generating the objective function for GWO, PSO, TLBO and GA. All adopted algorithms gives faster convergence rate and improves the problem of trapping in local minima. Future research will be on the hybridization of GWO, PSO, TLBO with ANN, tuning parameter, epoch numbers can be used to refine optimum solution.

7.4.4 Inverse kinematic solution of 6-dof ABB IRB-1400 manipulator

In this section, type A2 ABB IRB-1400 manipulator is adopted for the inverse kinematic analysis. The material description is provided in chapter 3 with kinematic parameters and joint variables. The mathematical modelling of forward and inverse kinematic is given in chapter 4. Using the mathematical equations of kinematics five different positions and joint variables are calculated for the comparative experiments. Five different positions and joint variables are presented in Table 7.33. Simulations of the proposed model and their kinematic relationship are performed to check the quality and efficiency of the solution using all adopted algorithms. The detail discussions of the inverse kinematic solution scheme and application of the optimization algorithms are presented in chapter 6. Based on the application of optimization algorithm and objective function formulations the comparison has been made with the quaternion algebra kinematics. The different optimization algorithms for the comparison are considered as

follows, (a) GWO, (b) PSO, (c) TLBO, (d) CIBO and (e) GA. The proposed work and adopted algorithms are performed in MATLAB.

Table 7.17 gives the sample of the target data for the position of end effector determined through analytical solution (Quaternion), which is further used to determine the difference between the adopted algorithms and analytical solution of inverse kinematic.

Table 7.33 Five different positions and joint variables through quaternion

Positions (X, Y, Z)	Joint angles					
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
P1(12.46, 47.59, 104.75)	58.30	3.39	39.68	88.30	72.25	155.40
P2(-24.64, -62.39, 91.87)	53.15	20.04	-56.63	81.88	14.08	165.95
P3-21.61, 9.30, 118.00)	86.30	45.39	12.67	50.40	80.06	41.84
P4(-12.16, 8.83, 126.55)	87.12	38.63	9.35	44.18	11.71	264.81
P5(-50.31, 18.71, 103.27)	98.91	49.33	42.92	85.52	56.51	36.99

Table 7.34 presents the comparative results of all adopted algorithms for the evaluation of inverse kinematic using objective function and constraints. Similar to the previous work, parameters for all adopted algorithms have chosen randomly. The development of the objective function is based on the Euclidean distant norm which is having minimum of function value 0. Moreover, in this work the minimum functional value is considered as 0.01 for all algorithms. The number of the dimension depends on the joint variables. Hence six search agents or wolves have been considered for the GWO based solution. Similarly six dimensions for PSO, six sets of learners for TLBO and six genes for GA are considered as a candidate solution. It has been observed that the all considered search agents or individuals for adopted algorithm having ability of exploration and exploitation of best fitness evaluations.

Table 7.34 represents the comparative results of all algorithms. The results obtained through the GA are better than all other algorithms while TLBO and GWO performs equally up to certain limit. Performances of the PSO and CIBO are similar in case of function evaluation. Joint variables for position P1 using PSO is near to the conventional based solution. On the other hand, for positions P2, P3 and P4 genetic algorithm is giving better solution. Figure 7.82 through Figure 7.86 indicates the overall comparison of all adopted algorithms with quaternion algebra based solutions. From Figure 7.82 through Figure 7.86 it can be understood that the implementation of the optimization algorithms are fruitful. Therefore, to avoid the mathematical complexities

of inverse kinematic solution optimization algorithms can be applied. In this work the adopted algorithm is producing multiple solutions for the single position but as per give termination criteria and among those generated results minimum value of joint angle has taken for the comparison.

Table 7.34 Comparative results for joint variable and function value

Positions (X, Y, Z)	Joint angles by PSO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	49.96	-4.12	40.22	90.25	72.66	149.78	-11.96
P2	50.23	18.54	-56.55	80.65	-104.35	102.85	0.0152
P3	86.85	-90.21	-12.99	49.47	90.25	40.25	-294.35
P4	87.36	-40.15	-10.51	-45.67	10.99	243.82	0.0017
P5	100.35	50.89	-65.68	-26.85	125.86	10.61	0.1950
Positions	Joint angles by GWO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	48.65	35.96	14.95	-58.99	-1.96	102.65	0
P2	98.45	44.62	-91.56	-110.54	201.68	-14.63	0
P3	88.65	-45.65	65.85	1.29	-111.48	89.56	-316.856
P4	102.65	-90.36	49.58	61.25	91.39	88.24	0
P5	73.52	64.25	-34.95	-72.52	94.68	66.45	0
Positions	Joint angles by TLBO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	91.68	55.36	-102.51	31.84	59.51	15.52	0
P2	-14.57	165.85	-97.34	22.86	19.47	-28.36	0
P3	94.45	-53.35	-23.84	201.84	-90.47	1.95	-296.47
P4	104.35	-52.86	20.35	65.32	14.69	44.58	0.0025
P5	99.21	149.57	-45.67	16.57	19.06	0.25	0
Positions	Joint angles by GA						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	59.21	-4.69	40.25	102.65	65.85	100.35	0
P2	50.36	20.99	-45.56	80.68	102.63	30.51	0
P3	99.58	-14.65	-23.68	71.16	-41.69	32.85	0
P4	55.68	24.36	16.52	-63.78	0.10	36.52	0
P5	99.51	50.36	41.35	102.96	-61.52	-40.63	0
Positions	Joint angles by CIBO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	35.56	14.23	40.41	22.96	90.30	-150.26	0.0016
P2	50.26	66.81	-20.14	37.61	55.14	100.52	-102.36
P3	96.58	50.24	64.81	33.55	-45.09	-12.99	0.0036
P4	69.58	-10.25	-155.62	23.57	61.00	29.85	0.856
P5	57.21	-21.86	44.36	-34.51	85.26	9.53	-102.96

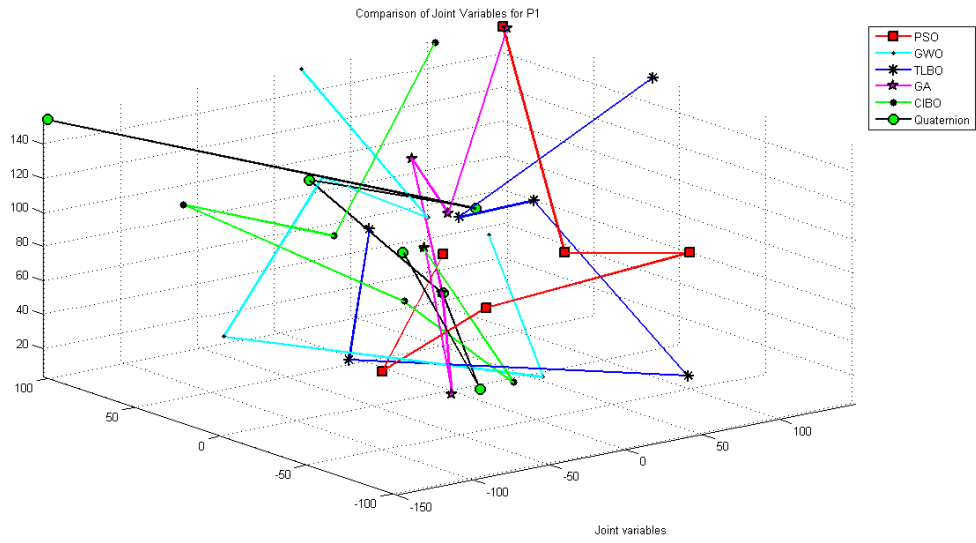


Figure 7.82 Comparison of joint variables for position 1

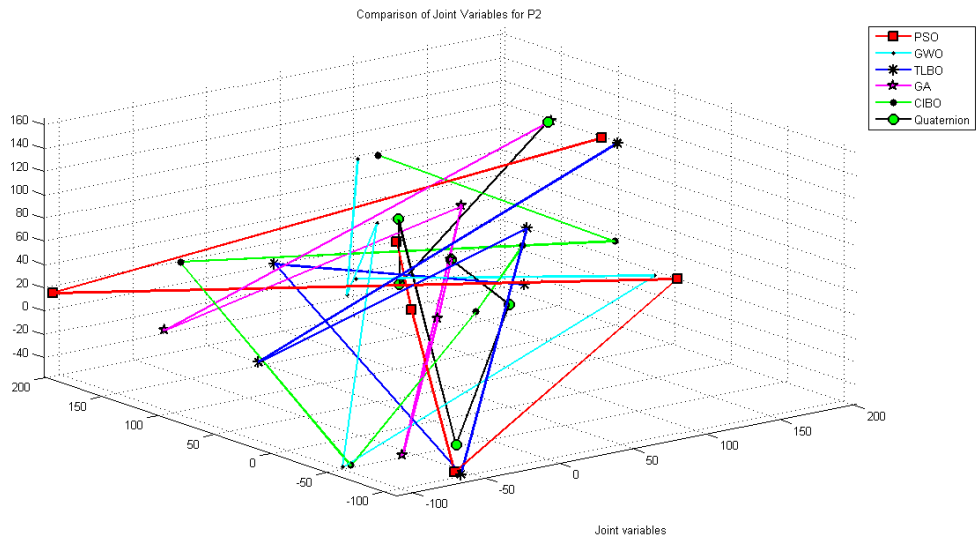


Figure 7.83 Comparison of joint variables for position 2

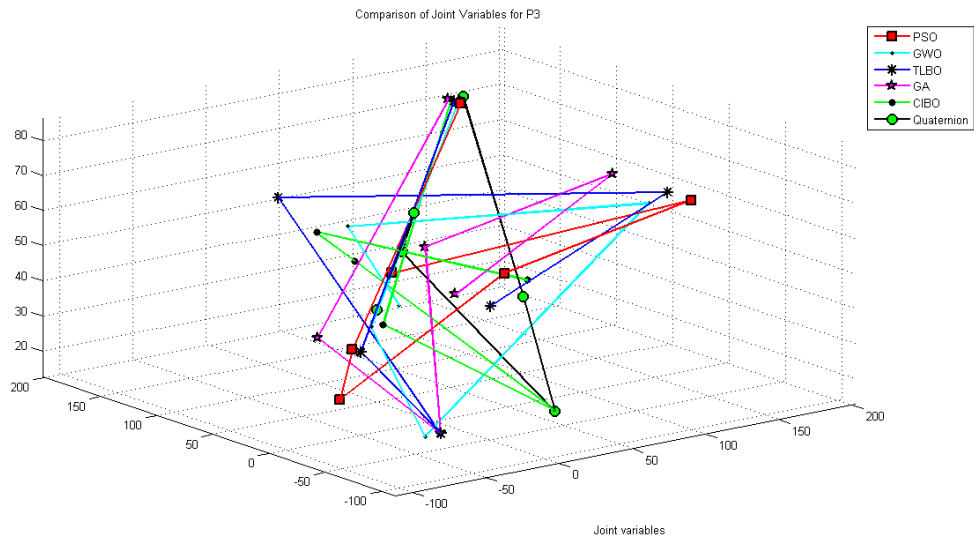


Figure 7.84 Comparison of joint variables for position 3

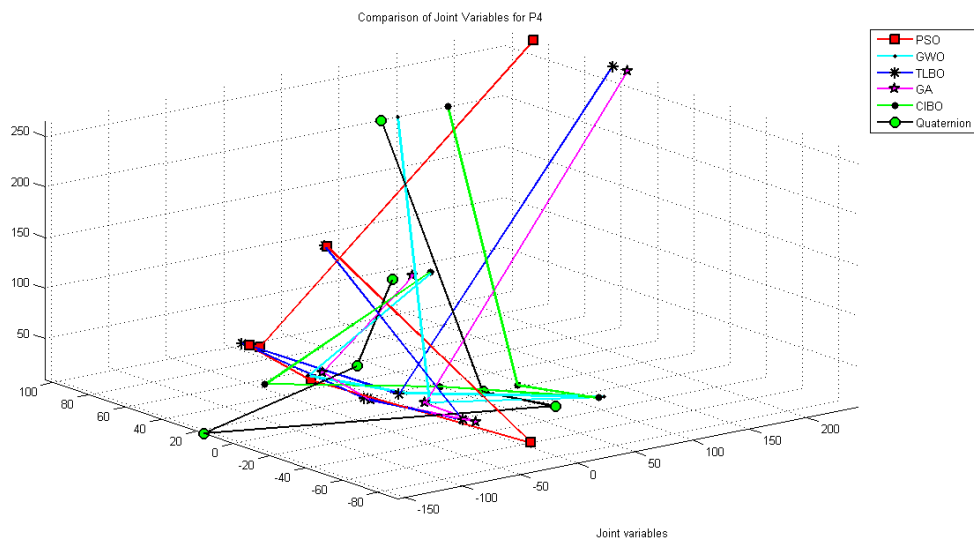


Figure 7.85 Comparison of joint variables for position 4

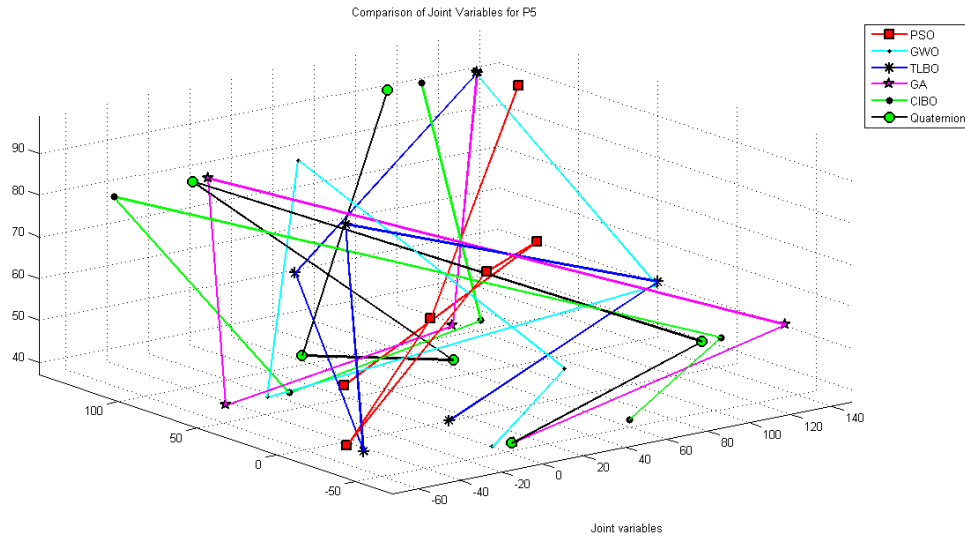


Figure 7.86 Comparison of joint variables for position 5

7.4.5 Inverse kinematic solution of 5-dof ASEA IRb-6 manipulator

In this section, type A1 ASEA IRb-6 manipulator is used for the inverse kinematic analysis. The joint variables and kinematic parameters for the adopted robot manipulator are described in chapter 3. In chapter 4 thorough description and mathematical modelling of forward and inverse kinematics of the selected manipulator is presented. Using the kinematic formulations five different positions of the end effector and their joint variables are calculated for the comparative evaluations of the adopted algorithms. The joint variables and end effector coordinates for the selected manipulator is presented in Table 7.35.

Table 7.35 Five different positions and joint variables

Positions	Joint angles				
	θ_1	θ_2	θ_3	θ_4	θ_5
P1(595.72, -303.90, -294.64)	99.6981	85.4732	-135.6945	-143.2309	109.1581
P2(78.56, -334.93, 71.80)	310.6737	59.8666	140.0265	27.9008	39.4117
P3(565.94, -293.77, 43.18)	329.3916	81.7060	144.4626	-82.6927	333.3707
P4(510.20, -572.33, -611.03)	177.9217	95.9999	-142.7337	67.0121	283.7942
P5(109.98, -242.10, -191.05)	157.1849	95.2089	143.8745	-219.1774	32.5522

To check the quality and efficiency of the adopted algorithms simulations and comparisons have been made. The thorough description of the inverse kinematic solution scheme and application of the adopted algorithms are discussed in chapter 6. The formulations for the objective function for the inverse kinematic evaluations are

based on the position and orientation error of the manipulator. The proposed work and adopted algorithms are performed in MATLAB. Some samples of the joint variables and end effector coordinates using MATLAB program is presented in Table 7.20. The considered optimization algorithms are GWO, TLBO, CIBO, GA and PSO which is further compared with the quaternion algebra based inverse kinematic solutions. The comparative results for function evaluations and joint variables are given in Table 7.36 for all adopted algorithms.

Table 7.36 Comparative results for joint variable and function value

Positions	PSO Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	100.23	-26.85	-130.25	-14.26	91.26	0.0023
P2	256.35	-60.53	114.52	30.25	24.63	-10.25
P3	211.36	56.24	-45.85	-0.425	74.35	24.63
P4	52.96	112.54	46.98	111.41	-81.21	0.096
P5	34.68	12.48	6.69	-90.73	30.54	0.0006
Positions	GWO Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	100.36	26.95	45.95	10.43	109.66	0.0024
P2	270.15	60.03	-114.68	-11.58	40.97	-100.67
P3	-96.24	55.18	88.65	67.16	-251.63	0.0048
P4	6.15	67.24	-36.74	-51.86	280.57	0
P5	66.54	100.29	11.96	49.89	73.54	0
Positions	TLBO Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	52.67	69.89	108.16	9.82	-47.65	0
P2	31.93	14.51	107.19	5.30	57.69	0
P3	29.56	7.72	108.35	18.33	57.63	0
P4	42.82	24.48	106.28	3.22	45.44	-248.36
P5	36.57	49.80	107.41	15.26	-53.72	0.0025
Positions	GA Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	57.20	121.10	68.84	-12.10	128.83	0
P2	143.86	150.6	91.36	-30.89	14.39	0
P3	174.22	90.48	128.31	31.85	185.97	0
P4	170.85	35.33	54.50	-43.94	87.04	0
P5	23.35	88.18	69.67	-25.62	16.12	0
Positions	CIBO Joint angles					Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	
P1	35.76	62.85	108.84	3.55	-16.48	0.0074
P2	53.25	50.55	105.46	6.73	20.31	10.63
P3	39.13	45.61	106.45	5.52	-56.41	0.0042
P4	32.25	37.70	108.66	18.49	-62.37	0
P5	39.79	47.63	106.6	6.84	-7.52	0

The parameters for tuning of the optimization algorithms are chosen randomly which is similar to the previous work. The objective function is based on the distance norm having minimum function value 0. Moreover, in this work the minimum functional value is considered as 0.01 for all algorithms. The number of particle for swarm optimization or solution dimension is depends on the number of variables of the objective function. Therefore, in this work number of dimension is five for all algorithms. The ability of exploration and exploitation for all algorithms is better as compared to conventional optimization techniques. The metaheuristic or nature based optimization algorithms having ability to avoid local optimum points. Therefore, the adopted algorithms having strong exploration and exploitation ability for the searching of global optimum point.

The evaluations of the function value and joint variables are given in Table 7.36. The functional values of GA for all positons are 0, which is better than all other algorithms. The computational cost for the GA is less as compared to other optimization based algorithms. From Figure 7.87 through 7.91, the joint variables obtained through GWO for position P2, P4 and P5 are better than all other algorithm. On the hand, for position P1 and P3, TLBO is yielding better result as compared to GWO as shown in Figure 7.87 and Figure 7.89. The overall performance of TLBO and GA is better than other adopted algorithm. Therefore, the mathematical complexities of higher order polynomial equations can be avoided using the optimization based solutions. Moreover, all adopted algorithms having ability to solve inverse kinematic problem. Although the adopted algorithm produces number of solution for the problem but eventually the termination of algorithm yields optimum result of inverse kinematic.

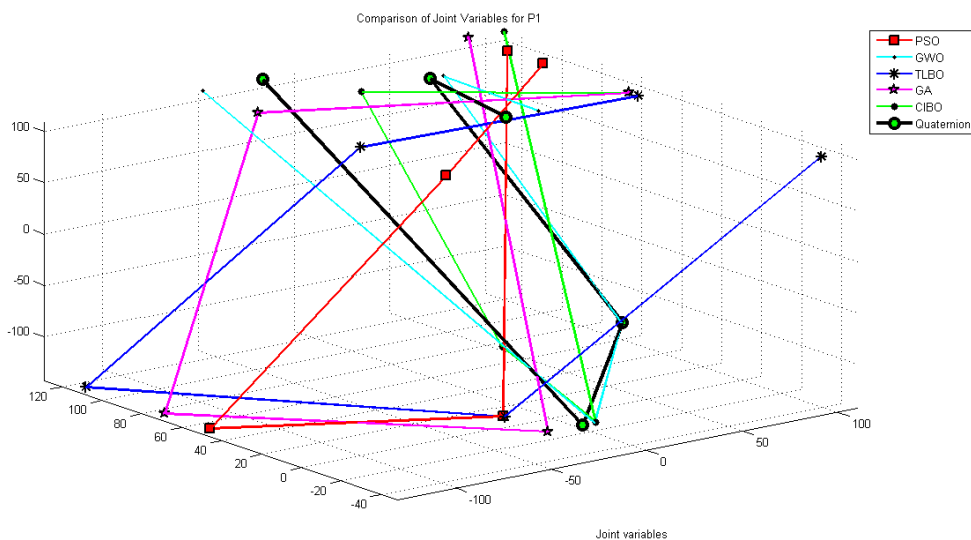


Figure 7.87 Comparison of joint variables for position 1

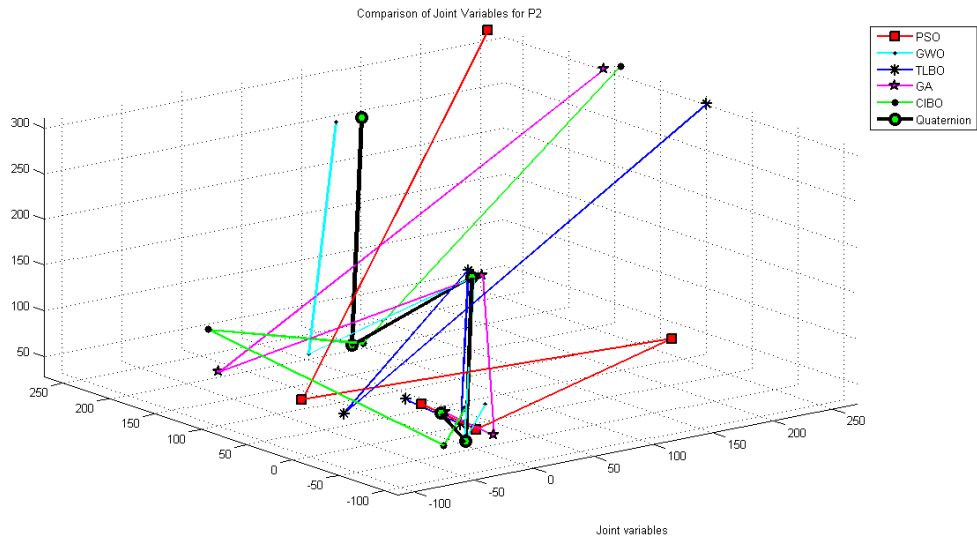


Figure 7.88 Comparison of joint variables for position 2

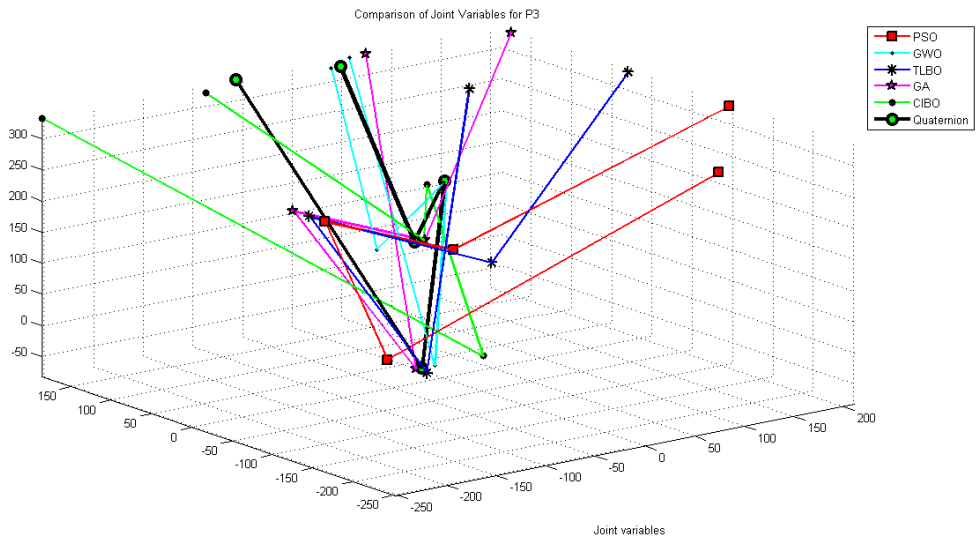


Figure 7.89 Comparison of joint variables for position 3

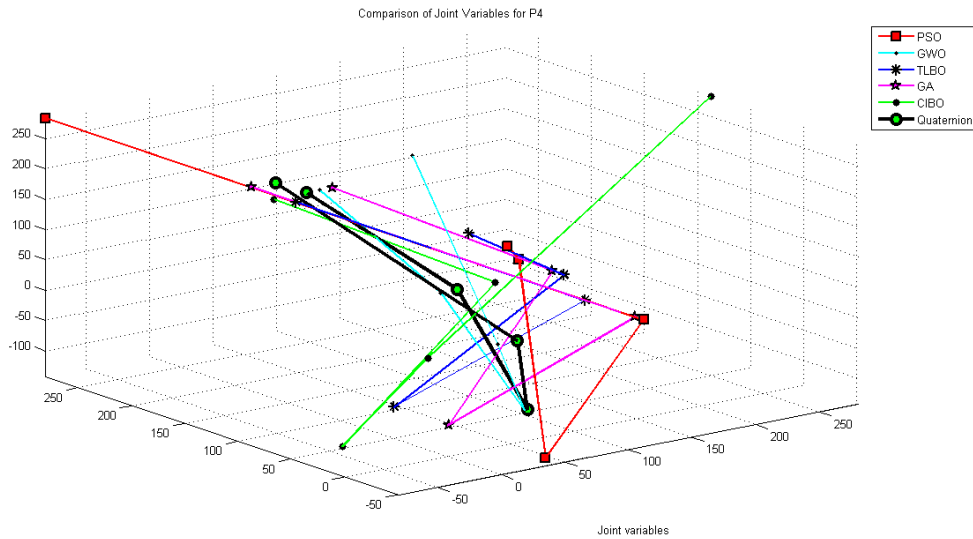


Figure 7.90 Comparison of joint variables for position 4

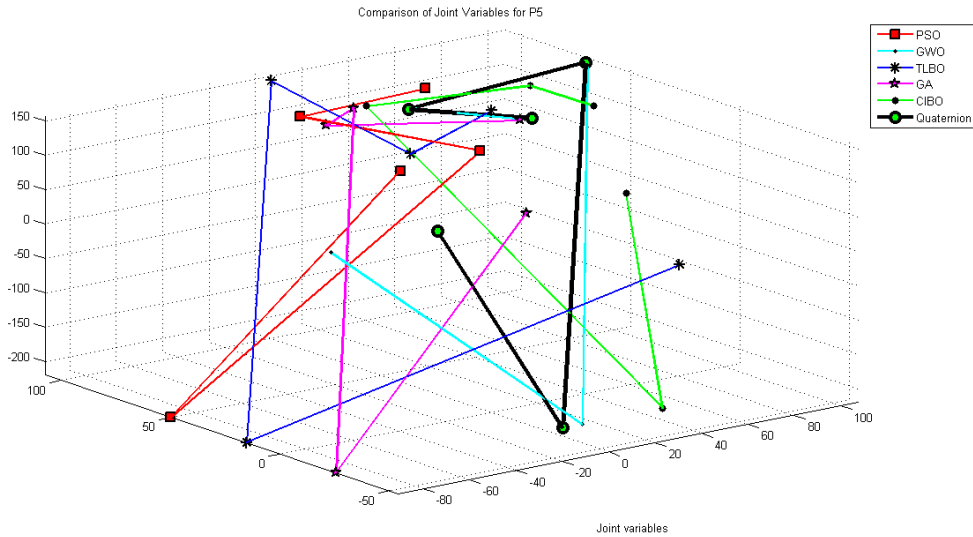


Figure 7.91 Comparison of joint variables for position 5

7.4.6 Inverse kinematic solution of 6-dof STAUBLI RX160 L manipulator

In this section, type A2 6-dof STAUBLI RX160 L manipulator is used for the inverse kinematic analysis. The model description and associated kinematic parameters are discussed in chapter 3. On the other hand, a derivation of the inverse kinematic problem is presented in chapter 4. The forward and inverse kinematic formulations are used to generate the sample data for the comparison and evaluation of the adopted algorithm as explained earlier. Using the kinematic formulations from chapter 4, five different positions are considered for the inverse kinematic solution using optimization algorithm. The generated data samples for five different positions are presented in Table 7.37.

Table 7.37 Five different positions and joint variables through quaternion

Positions (X, Y, Z)	Joint angles					
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
P1(5.94, -6.49, 34.29)	109.99	58.2030.99	164.40	105.36	190.88	
P2(-16.64, -9.13, 17.82)	57.18	7.10	141.76	219.30	115.70	160.32
P3(-1.53, -3.77, 28.49)	109.33	41.03109.51	182.94	112.05	109.37	
P4(-0.14, -4.49, 33.53)	126.02	32.7067.10	172.89	113.67	219.84	
P5(4.38, -64.92, 37.51)	136.54	47.7994.15	14.06	110.33	216.84	

This generated data is the basis for the simulation and comparison of the adopted methodologies. Moreover, the application of the optimization algorithm for the solution of inverse kinematic problem is discussed in chapter 6. The preparation of the fitness function is also described in chapter 6 along with the mathematical formulations. The fitness function (objective function) is based on the distance norm with the imposed joint variables as constraints to solve the inverse kinematic problem. Overall work is performed in the MATLAB environment. Flexibility of the objective function provides better opportunity to adopt various optimization algorithms to solve the inverse kinematic problem. In this work, five different optimization algorithms are considered namely GWO, PSO, TLBO, GA and CIBO. The adopted algorithm is later compared with the conventional based solution of the inverse kinematic problem. The comparative results are produced in Table 7.38.

The tuning parameters for all adopted optimization algorithms are taken randomly, there no specific tuning of the parameters. For example in PSO it is require to set acceleration constant, inertia weight and constriction factor. Similarly for all other algorithms possess some parameter which directly affects the performance. The number of joint variable is six in this case therefore the selected dimension for the optimization algorithm is six. The adopted algorithms have the ability to avoid the local optimum or near optimal solutions. Moreover, the exploration and exploitation of algorithms play crucial role to get the global optimal solution. The overall performance and quality of the results are presented in chapter 8. The comparative results and functional values are given in Table 7.38. The objective function values are zero for GA and TLBO algorithms for all considered positions. Therefore the convergence of the GA and TLBO is better than other adopted algorithms.

Table 7.38 Comparative results for joint variable and function value

Positions (X, Y, Z)	Joint angles by PSO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	50.50	136.59	101.21	29.5	31.67	111.06	0.0078
P2	156.97	19.59	106.62	64.77	178.37	108.57	10.36
P3	146.38	71.12	26.13	97.56	183.49	105.96	-215.63
P4	139.06	14.52	44.78	23.30	10.29	115.65	0
P5	29.09	55.46	13.16	126.70	145.79	108.57	0
Positions	Joint angles by GWO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	1.12	88.27	14.97	169.35	114.04	155.89	0
P2	126.5	13.83	85.62	51.54	119.15	36.96	0
P3	124.0	52.19	63.36	64.60	105.17	53.69	0
P4	20.35	43.77	87.72	261.65	115.42	52.57	0
P5	45.39	130.82	20.15	226.51	116.36	66.97	0
Positions	Joint angles by TLBO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	98.34	55.16	134.24	118.62	119.71	98.34	0.0058
P2	35.32	5.11	196.6	106.8	112.32	35.32	0.00034
P3	96.79	11.28	23.95	107.40	131.63	96.79	-211.63
P4	61.1	75.20	129.40	106.80	136.74	61.19	0
P5	17.74	71.39	110.98	112.37	112.70	17.74	0
Positions	Joint angles by GA						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	60.03	6.08	31.83	108.07	242.29	60.03	0
P2	75.42	11.79	205.14	105.14	259.89	75.42	0
P3	94.35	35.76	225.22	118.75	66.43	94.35	0
P4	111.2	70.00	134.37	112.21	227.22	111.2	0
P5	87.71	107.92	179.51	114.49	58.58	87.71	0
Positions	Joint angles by CIBO						Function Value
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
P1	32.82	148.12	231.26	107.95	9.62	32.82	-20.36
P2	75.79	78.73	143.81	118.75	250.04	75.79	0.0086
P3	82.27	106.23	16.71	118.01	242.53	82.27	0
P4	104.25	1.41	140.99	117.23	185.74	104.25	0
P5	35.36	145.29	178.50	112.06	52.86	35.36	-315.36

From Figure 7.92 through Figure 7.96 gives the overall comparison for all positions with the quaternion based solution. For position P1 GA is yielding better results as compared to other algorithms. On the other hand, GWO, TLBO and GA are performing better than PSO and CIBO for position P2. For position P3 the performance of CIBO is similar to GA and TLBO. For positions P4 and P5, GA and TLBO perform equally. All adopted algorithms are giving minimum function values with optimized joint variables. Therefore, optimization algorithms are strong tool to solve inverse kinematic problem.

All adopted algorithms produces multiple solutions for the inverse kinematic problem but when the algorithm reaches to the maximum iteration the joint variable are optimized with functional value.

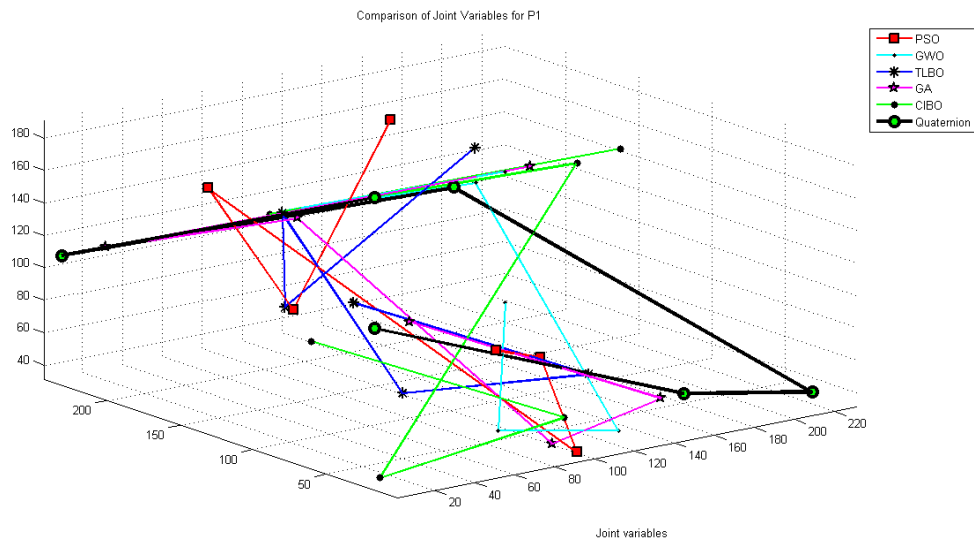


Figure 7.92 Comparison of joint variables for position 1

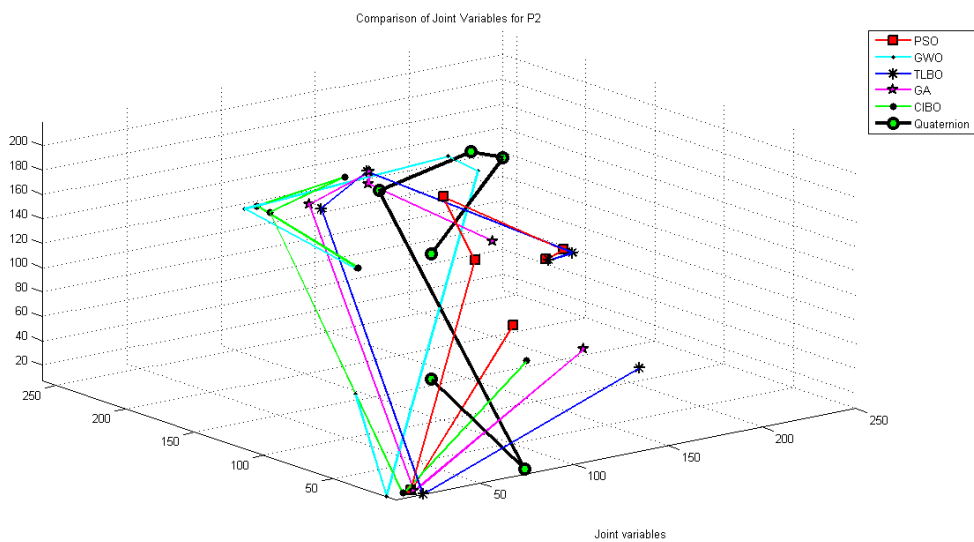


Figure 7.93 Comparison of joint variables for position 2

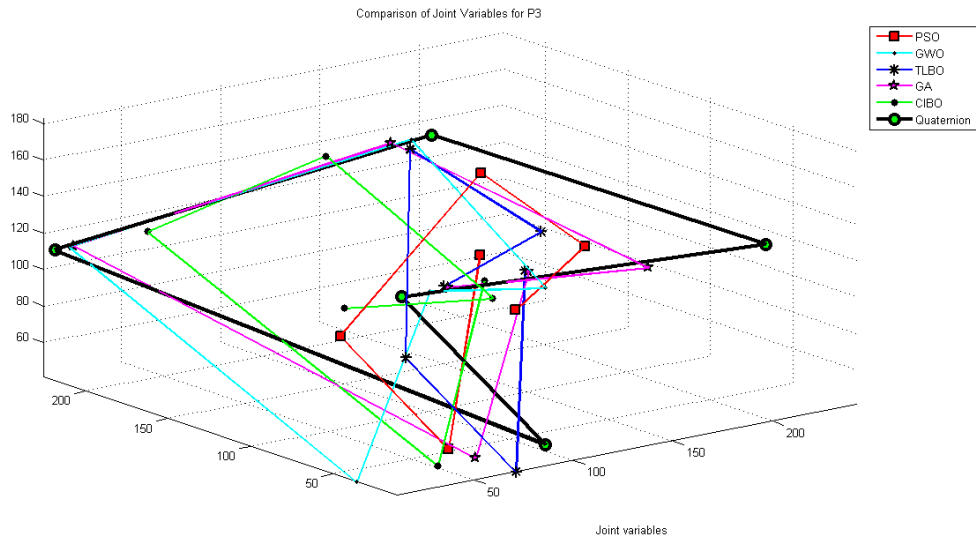


Figure 7.94 Comparison of joint variables for position 3

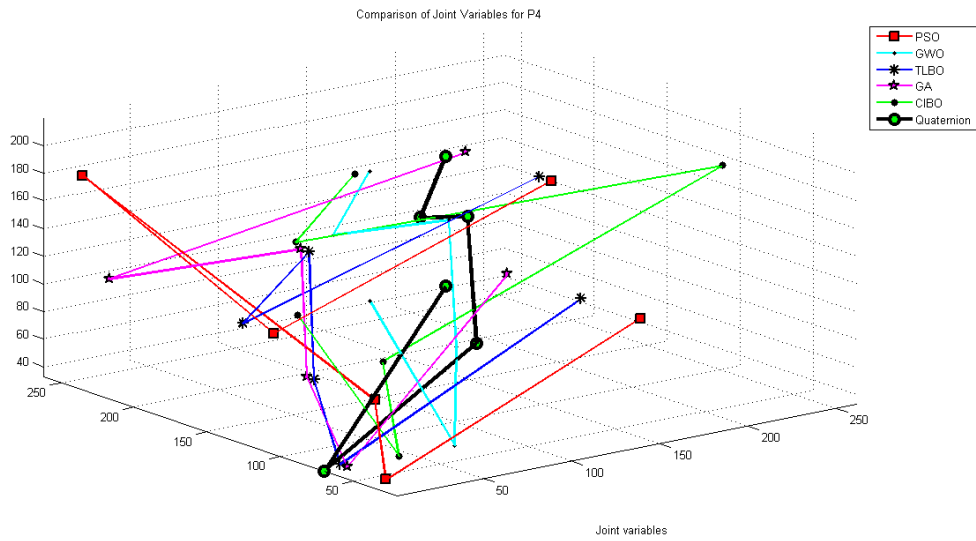


Figure 7.95 Comparison of joint variables for position 4

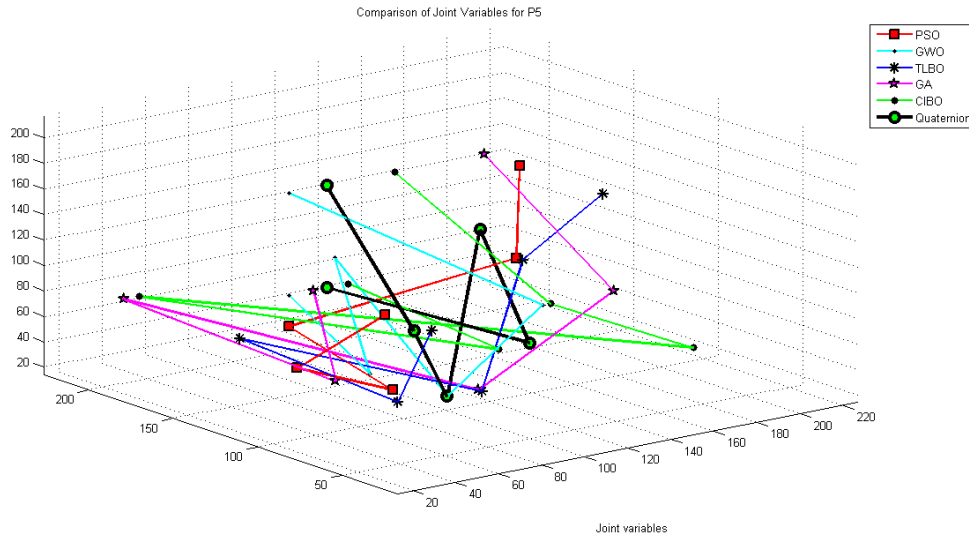


Figure 7.96 Comparison of joint variables for position 5

7.5 Discussions

Inverse kinematic solutions of different configurations of industrial robot manipulators are presented in this chapter. The detail plan of the materials and methods are presented in Table 3.10. Conventional tools such as quaternion algebra as well as homogeneous transformation methods are used to determine the inverse kinematic solution of the adopted manipulators. In homogenous transformation method, it is required to store all orientation vector or transformation matrices of each coordinate system with respect to previous one from the beginning. Whereas quaternions of each coordinate system are calculated from the unit line vector. The total space required for the quaternion algebra is eight while homogeneous matrix method takes 12 memory locations. The space requirement affects the overall computational cost due to the cost of attracting an operand from the memory surpasses the cost of execution a basic mathematical operations.

For the calculation of computational cost the system, Intel Core i5 with 4 GB RAM and the 3.10 GHz processor was used. On the other hand, comparison has been made on the basis of the number of solutions for all adopted manipulators. The number of solution for all adopted manipulator is presented in Table 7.39. An overall result of quaternion algebra and homogeneous transformation based approach towards the solution of inverse kinematic problem is presented in Table 7.39.

Table 7.39 Comparative analysis of conventional tools

Methods	Adopted On			Outcomes	Remarks			
	Robots	Structures	Types		No. of Solutions	MO		CT (Seconds)
						+	*	
Conventional approaches 1. HT 2. QA	3-dof revolute	Rigid (R-R-R)	Planar	θ_1, θ_2 and θ_3	HT: 2 QA: 2	HT:72 QA:69	HT:114 QA:105	HT:0.36 QA:0.11
	SCARA (4-dof)	Flexible (R-R-P-R)	SCARA	θ_1, θ_2, d_3 and θ_4	Multiple solutions	HT:108 QA:91	HT:168 QA:144	HT:0.37 QA:0.19
	Pioneer arm2 (5-dof)	Rigid (R-R-R-R-R)	Spatial	$\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5	Multiple solutions	HT:144 QA:113	HT:222 QA:183	HT:0.40 QA:0.29
	PUMA 560 (6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-C	$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ and θ_6	Multiple solutions	HT:180 QA:135	HT:276 QA:222	HT:0.49 QA:0.33
	ABB IRb-1400(6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-A1	$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ and θ_6	Multiple solutions	HT:180 QA:135	HT:276 QA:222	HT:0.40 QA:0.31
	ASEA IRb6 (5-dof)	Rigid (R-R-R-R-R)	Spatial Type-A2	$\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5	Multiple solutions	HT:144 QA:113	HT:222 QA:183	HT:0.39 QA:0.27
	STÄUBLI RX160 L (6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-B2	$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ and θ_6	Multiple solutions	HT:180 QA:135	HT:276 QA:222	HT:0.446 QA:0.318

The intelligence based approaches are quite convenient tool for the solution of inverse kinematic problem due to its flexibility to adapt non-linear functions. The detail description of the adopted neural network models and their application towards the solution of inverse kinematic problem has been discussed in chapter 5. The obtained results using the intelligence based approaches are presented in Table 7.40. The comparison has been made on the basis of number of solutions as well as computational time. An intelligent based approach gives multiple solutions as compared to optimization based approach.

The overall computation time for the determination of inverse kinematic solution is more than the conventional tool. An intelligent based method requires the higher level of programming which makes maximum use of memory locations. A concise result of the intelligence based approaches ae presented in Table 7.40.

Table 7.40 Comparative analysis of intelligent approaches

Methods		Adopted On			Outcomes	Remarks	
		Robots	Structures	Types	Joint variables	No. of Solutions	CT (Seconds)
Intelligence approaches 1. MLPBP 2. ANFIS 3. MLPPSO 4. MLPGWO 5. PMLTLB 6. MLPGA 7. MLPCIBO		3-dof revolute	Rigid (R-R-R)	Planar	θ_1, θ_2 and θ_3	Multiple solutions	1. MLP:4.5 2. PPN:3.9 3. Pi-Sigma:3.1
		SCARA (4-dof)	Flexible (R-R-P-R)	SCARA	θ_1, θ_2, d_3 and θ_4	Multiple solutions	1. MLPBP:29.3 2. ANFIS:23.2 3. MLPPSO:21.8 4. MLPGWO:15.3 5. MLPTLBO:17.8 6. MLPGA:6.13 7. MLPCIBO:18.21
		Pioneer arm2 (5-dof)	Rigid (R-R-R-R-R)	Spatial	$\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5	Multiple solutions	1. MLPBP:29.22 2. ANFIS:24.36 3. MLPPSO:18.45 4. MLPGWO:14.36 5. MLPTLBO:16.5 6. MLPGA:6.99 7. MLPCIBO:15.26
		PUMA 560 (6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-C	$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ and θ_6	Multiple solutions	1. MLPBP:31.01 2. ANFIS:29.85 3. MLPPSO:19.96 4. MLPGWO:14.22 5. PMLTLBO:14.6 6. MLPGA:7.59 7. MLPCIBO:15.79
		ABB IRb-1400(6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-A1	$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ and θ_6	Multiple solutions	1. MLPBP:30.88 2. ANFIS:29.64 3. MLPPSO:19.22 4. MLPGWO:14.60 5. MLPTLBO:13.1 6. MLPGA:7.06 7. MLPCIBO:15.49
		ASEA IRb6 (5-dof)	Rigid (R-R-R-R-R)	Spatial Type-A2	$\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5	Multiple solutions	1. MLPBP:28.50 2. ANFIS:25.41 3. MLPPSO:16.04 4. MLPGWO:13.71 5. MLPTLBO:13.9 6. MLPGA:5.23 7. MLPCIBO:15.55
		STÄUBLI RX160 L (6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-B2	$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ and θ_6	Multiple solutions	1. MLPBP:31.33 2. ANFIS:30.26 3. MLPPSO:21.84 4. MLPGWO:19.76 5. MLPTLBO:18.2 6. MLPGA:8.90 7. MLPCIBO:19.06

Apart from conventional tools as well as neural network based approaches for solving inverse kinematic problem, optimization based approaches are also considered. The detail description and mathematical modelling of the objective function is presented in chapter 6. As can be realized from the conventional and intelligence based solution, it is lengthy and time consuming method. Therefore, to overcome the problem of mathematical complexities as well as higher level of programming, the position and orientation error based optimization function is used. Although the computational time is nearly similar to intelligent based methods but the efficiency and quality of the solution is more reliable. The comparison has been made on the basis of computational time as well as number of solutions. The optimization based approaches produces multiple solution during the iteration but once the algorithm reached to the maximum iteration or termination point it provides optimized solution for the inverse kinematic problem. Therefore, summarized results obtained through the different optimization based approaches and their comparison has been presented in Table 7.41.

Table 7.41 Comparative analysis of optimization algorithms

Methods		Adopted On			Outcomes		Remarks	
		Robots	Structures	Types	Joint variables	No. of Solutions	CT (Seconds)	
Optimization approaches 1. PSO 2. GWO 3. TLBO 4. GA 5. CIBO	SCARA (4-dof)	Flexible (R-R-P-R)	SCARA	θ_1, θ_2, d_3 and θ_4	Multiple solutions	1. PSO: 17.22 2. GWO: 11.68 3. TLBO: 14.23 4. GA: 4.96 5. CIBO: 14.08		
	Pioneer arm2 (5-dof)	Rigid (R-R-R-R-R)	Spatial	$\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5	Multiple solutions	1. PSO: 16.88 2. GWO: 3.45 3. TLBO: 15.67 4. GA: 7.92 5. CIBO: 29.41		
	PUMA 560 (6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-C	$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ and θ_6	Multiple solutions	1. PSO: 31.86 2. GWO: 25.82 3. TLBO: 29.54 4. GA: 5.89 5. CIBO: 30.56		
	ABB IRb-1400 (6-dof)	Rigid (R-R-R-R-R-R)	Spatial Type-A1	$\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ and θ_6	Multiple solutions	1. PSO: 30.14 2. GWO: 12.89 3. TLBO: 12.05 4. GA: 5.03 5. CIBO: 14.30		
	ASEA IRb6 (5-dof)	Rigid (R-R-R-R-R)	Spatial Type-A2	$\theta_1, \theta_2, \theta_3, \theta_4$ and θ_5	Multiple solutions	1. PSO: 29.11 2. GWO: 15.36 3. TLBO: 15.58 4. GA: 5.04 5. CIBO: 15.77		

	STÄUBLI RX160 L (6-dof)	Rigid (R-R-R-R- R-R)	Spatial Type-B2	$\theta_1, \theta_2 \theta_3 \theta_4 \theta_5$ and θ_6	Multiple solutions	1. PSO:21.74 2. GWO:14.28 3. TLBO:16.47 4. GA:8.99 5. CIBO:21.54

7.6 Summary

A detail analysis of inverse kinematic problem of selected benchmark manipulators and their simulation studies are carried out with some adopted conventional and reactive approaches. In this chapter few models of ANN is used to find out the inverse kinematic solution of selected benchmark manipulators as presented in Table 3.1. On the other hand, optimization algorithms are adopted and comparison has been made with the hybrid ANN algorithms. The obtained results are presented in the form of graphs and tables. MATLAB programs are used to resolve the problem of inverse and forward kinematics of selected manipulators. Results obtained through adopted methods are compared with the results of conventional methods and are presented in tables as well as in figures. Application of optimization algorithms and their comparison are presented in figures and tables. The quality and efficiency of the proposed algorithms have been presented in this chapter. On the basis of performed kinematic analysis on selected configuration of robot manipulators can be summarized as:

- A conventional approach provides closed form solution of inverse kinematic problem but it requires complex mathematics. Hence quaternion vector method has been used to calculate the inverse kinematic of selected manipulators. On the other hand, homogeneous transformation method is also used to resolve the inverse kinematic problem for selected configurations. After calculation of inverse kinematic of robot manipulators these data sets were used to train ANN models.
- ANN based approach are quite flexible and easy to resolve kinematic problems. Adopted ANN models are MLP, PPN and Pi-sigma are performing well for inverse kinematic problem. In comparison with the adopted ANN models MLPNN with back propagation algorithm giving better results than other models. Hence MLPNN has been applied for all configurations of robot manipulator.

- The hybrid MLPNN method provides less error as compared to normal MLPNN method. Hybridisation of optimization algorithms gives fast exploration and exploitation ability to the network. The results obtained using hybrid ANN and ANFIS are compared and verified with the conventional solution of inverse kinematics of robot manipulator. Obtained results are reasonable and accurate as compared to normal ANN models, therefore it can be accepted.
- The metaheuristic algorithm produces flexible structure for the resolution of inverse kinematic problem. The bio-inspired population based algorithms reveals satisfactory performance for the considered problem. The result exhibits constant convergence behaviour of the adopted algorithms for different configurations of manipulator. Genetic algorithm is performing best for all considered manipulator configuration.

Chapter 8

CONCLUSIONS AND FUTURE WORK

8.1 Overview

Inverse kinematics of any robot manipulator can generally be defined as finding out the joint angles for specified Cartesian position as well as orientation of an end effector and opposite of this, determining position and orientation of an end effector for given joint variables is known as forward kinematics. Forward kinematic having unique solution but in case of inverse kinematics it does not provide any closed form or unique solution thus it is require to have some suitable technique to resolve the problem for any configuration of robot manipulator. Hence, inverse kinematics solution is very much problematic and computationally expensive. For real time control of any configuration manipulator will be expensive and generally it takes long time. Most of the robotic applications are dependent on the joint variables of manipulator due to fact that the requirement of the desired position of the end effector. For the computing the analogous joint angles at high speed requires inverse kinematic transformation of each joint. Therefore, the current research work proposes inverse kinematic solution for various configurations of robot manipulator. The basic kinematics and mathematical modelling of the configurations are discussed thoroughly and subsequently kinematic analyses of selected configurations have been done. The concept and application of neural network models for inverse kinematic resolutions are discussed in length. To overcome the drawbacks of ANN model hybridization with optimization algorithms and their strategies are also made. In chapter 6, numerical solutions of the inverse kinematic problem of selected manipulator based on metaheuristic algorithms have been made. Optimization approaches are used to transform the kinematic mapping problem of the manipulators into constrained non-linear metaheuristic models. This approach gives the freedom to direct search of feasible configuration space of the robot end effector to

yield the joint variables of the manipulator with the minimization of position and orientation of end effector. The present work is summarized with the concluding remarks in the next section stating that contributions of the present research work. The scope of future work to extend or to modify or to add some other new concept to the work is suggested in the present chapter.

8.2 Conclusions

Inverse kinematic analysis of any configuration of robot manipulator is playing major role for robotic system. From the viewpoint of different configurations to simulation and real time control kinematic relationship of the robot plays crucial role for completion of given task. Mathematical complexities of inverse kinematic formulations using conventional approaches are expensive and time consuming but apart from the mathematical expenses it provides the closed form solution. To overcome the problem of mathematical operations of inverse kinematic of robot manipulator some techniques from the neural network models are required. ANN based approaches are quite fruitful for the inverse kinematic inversion. The architecture and working principle of the ANN provides the complex and non-linear functional organisation of the input output data. The data sets used for training can be generated from the forward kinematic equations of manipulator. Moreover, the generated data sets should be large so as to minimize the learning error of the network. The learning from the forward kinematic data sets is expensive and time consuming. The major drawback of ANN models are, it requires the optimization mechanism for the training of the structure and mostly stuck at local optimum point. Conventional methods like gradient descent learning algorithms gives effective and stable results to inverse kinematic problem. However, this classical algorithm provides stable solution but it converges into local optimum point and carries out constraint on the fitness function. Hence, the classical algorithms can be fruitfully used for the constrained robot manipulators. Therefore, to overcome the problem for higher dof, different population based optimization algorithms are hybridized with the ANN model.

On the other hand, for higher dof and complex task of robot manipulator, population based optimization algorithms can be used with the generic formulation of objective function. The optimization algorithms should be able to handle the problem of nonlinear, NP-hard and multimodal search problems. The major advantages of the optimization based inverse kinematic solution are; (1) it can cope with any configuration of robot manipulator (2) forward kinematic formulation are the only requirement for the generation of the objective function, (3) no. of optimization algorithms can be used for the single objective function (4) solution can be easily

obtained in the joint coordinates (5) it can handle constraint into the search space. Therefore, application of optimization algorithms and their theories are discussed thoroughly. Moreover, application on some selected manipulators and their performance are discussed in the previous chapter. This particular piece of research work aimed at achieving a precise and faster solution to inverse kinematic problems of industrial robots by using appropriate techniques. The major highlights of this research work are presented in the following lines.

- ✚ In the present research work DH-algorithm, homogeneous transformation and quaternion vector based methods and their significance for the kinematic analysis have been studied. Mathematical modelling of the forward and inverse kinematic problem of open chain revolute as well as SCARA robot with 3 to 6 joint axis is done. In this work quaternion vector based kinematic formulations have been done for selected configurations of the robot manipulator. The conventional kinematic equations of the open chain manipulator are transformed into consecutive quaternion transformations matrices and then articulated using quaternion. From the comparative results of homogeneous transformation methods with the quaternion based approach, mathematical operations are more in case of homogeneous transformation method. To maintain the accuracy of the obtained solution and reduce mathematical operations, quaternion based approach are much better. From chapter 4, it can be clearly understood that the quaternion vector based method delivers a very effective and efficient tool as compared to other conventional approach. Further, the adopted method is cost effective due to its less mathematical operations. Comparing with the homogeneous transformation methods, it can be observed that quaternion method produces same results with less time consumption. Therefore, this method can be applied to any configuration of robot manipulator. This can be used as general tool for the kinematic solution of n-dof robot manipulator. Finally the data sets of the selected manipulator can be prepared either by homogeneous or quaternion method for training of ANN models.
- ✚ Since inverse kinematic solution yields number of alternate solutions, an appropriate iterative or intelligence based technique can be used. Forward kinematic solution of any configuration is producing exact solution. Therefore, the generated data sets can be easily used to train the ANN models. Further trained network predicts the inverse kinematic solution of the selected configuration of the robot manipulator. In the present work MLP, PPN and Pi-sigma neural network is use to solve inverse kinematic problem. Later ANN based solutions are used to compare with the ANSFIS and hybrid ANNs.

Similar to ANN models, ANFIS was trained from the generated data sets within the limit of workspace. The ANFIS tool box from MATLAB is used to calculate the joint variables of the some selected configuration of robot manipulator. This method works on the principle of multiple inputs with single output (MISO) system. For all selected configuration of robot manipulator FIS (Fuzzy inference system) structure are obtained and applied for prediction of the individual joint angles. The process of training and testing of ANFIS structure for the particular problem is quite lengthy process. However, once the training of the structure is completed then it can be saved and used for number of inputs with minimal developed error.

- Despite the advantages of the neural network and ANFIS approach for inverse kinematic resolution, a chief concern that often comes is about the convergence and stability of the solution. These networks training generally converged into the local optimum point as discussed in chapter 5. Therefore, neural network models can be hybridize with population based optimization algorithm to update the weight and bias of the network. The hybridization scheme has already been discussed in chapter 5. Moreover, it is also enclosed the combination of optimization algorithms with MLP neural network as well as comparison of gradient descent learning algorithms and appropriate scheme. After the application of the metaheuristic algorithms and trained neural network, is applied to find out the inverse kinematic solution of the robot manipulators. Different types of configuration of the robot manipulators have been taken for the kinematic analysis.
- Although there are many advantages of ANN and hybrid ANN that can be easily implemented for the inverse kinematic solution but important concern is computational cost and convergence speed of the algorithm. ANN models with back propagation learning gives poor performance for the higher dof robot manipulators. The nonlinear functional relationship for higher dof problem become unstable and produces unacceptable error at the end of learning process. Therefore population based optimization algorithms can be gainfully used to find out the inverse kinematic solution. The only requirement for the application of optimization algorithms is to develop the objective function for the concern manipulator. In chapter 6, objective function formulations are discussed in detail which can be further applied with minor modifications to any configuration of manipulator. Moreover, the objective function produces the candidate solution of each individual joint variable and that can be defined by the configuration vector of manipulator with number of point within the workspace limit. This method requires only the formulation of the forward kinematic equations of the

robot manipulator and associated constant or parameters. This method provides flexibility to complete many task related to robot manipulator like design, kinematic analysis, synthesis of kinematic structures etc.

- On the other hand, many optimization algorithms require the number of control parameters setting and this increases the complexity of the adopted algorithm. The parameter associated with the algorithms can make the differences in the results like accuracy, convergence speed, efficiency, global optimum point and computational cost. Therefore, to avoid many parameter setting, novel effectual nature-inspired metaheuristic optimization technique grounded on crab behaviour is proposed (see chapter 6). The proposed Crab Intelligence Based Optimization (CIBO) technique is a population centered iterative metaheuristic algorithm for D-dimensional and NP-hard problems. Besides using Jacobian matrix for the mapping of task space to the joint variable space, forward kinematic equations are used. Kinematic singularity is avoided using these formulations as compared to other conventional Jacobian matrix based methods. Different types of configuration of the robot manipulators have been taken for the kinematic analysis. In general, proposed crab based algorithm gives generic solution of the inverse kinematic problem for some selected benchmark manipulators. But the proposed CIBO algorithm having some limitations like, it cannot apply for real time control and application for higher dof manipulator; it takes time to converge in single optimum point, etc.

8.3 Contributions

The major contributions of the current work towards the inverse kinematic solutions are:

- i. The developed mathematical modelling of various configurations of robot manipulator provides the generic solution to the specific problem. Quaternion vector pair based methods provides efficient tool for the inverse kinematic resolution. This method yield similar result as compared to other conventional methods therefore it can be generalized for the kinematic inversion. The developed derivation for the selected manipulators can be used to find the inverse kinematic solution.
- ii. ANN models generally doesn't not guarantee exact solution of inverse kinematic problem, therefore hybrid scheme has been proposed to solve the above problem. The major contribution of the work is to present novel hybrid ANN algorithm and their application on the kinematic problem.

Different proposed hybrid ANN models are discussed in chapter 5 and their consequently results are tabulated in chapter 7.

- iii. To avoid the problem of singularity or Jacobian matrix based numerical solution of the inverse kinematic; a population based optimization algorithms have been proposed for the kinematic inversion. The major challenge with the numerical solution is stability of the solution and it increase with the number of dof of manipulator.
- iv. A novel CIBO algorithm is proposed to solve the inverse kinematic solution of the robot manipulator. The proposed algorithm is based on the some specific behaviour of the Crabs such as social behaviour, recognition behaviour and crossing behaviour as discussed in chapter 6.

8.4 Future research

Inverse kinematic problem is one of the major concerns for many researchers. From past few decades many researchers have been trying to produce general solution method for different configuration and also for n-dof manipulators. The result obtained through the previous research is the foundation for the development of general solution of the problem. The adopted methods like intelligence based approach, conventional approach and an optimization algorithm provides a basic tool for the inverse kinematic solution. Therefore in the present work first stone of the foundation has been laid and hopefully it might motivate other researcher to develop novel methods for the kinematic inversion. Further research can be focused on modifying the ANN models to get less training error. Particularly setting of some tuning parameters like learning rate, momentum coefficient etc. could be useful to avoid local optimum points. Hybridisation of the ANN models can be done with the hybrid optimization algorithm with other ANN models like, RBFN, Elman's neural network etc.

Control parameter of inverse kinematic objective function formulation can be modified for the comparative analysis of the optimization algorithms. Therefore, alternative representation of the pose error formulations and application of other metaheuristic algorithms could be verified and comparison has to be done. The developed optimization algorithm can be used to calculate the inverse kinematic of constrained robotic systems.

The performance on the basis of accuracy and computational cost for the alternate objective function can be examined.

Conventional methods like dual quaternion, exponential matrix algebra, Lie algebra or Grobner bases can be used to solve inverse kinematic problem.

REFERENCES

- [1] J. Lenarčič, T. Bajd and M. Stanišić, *Robot Mechanisms*. Dordrecht: Springer, 2013.
- [2] J. Funda and R. Paul, 'A computational analysis of screw transformations in robotics', *IEEE Trans. Robot. Automat*, vol. 6, no. 3, pp. 348-356, 1990.
- [3] J. Funda, R. Taylor and R. Paul, 'On homogeneous transform, quaternions, and computational efficiency', *IEEE Trans. Robot. Automat*, vol. 6, no. 3, pp. 382-388, 1990.
- [4] S. Mitsi, K. Bouzakis and G. Mansour, 'Optimization of robot links motion in inverse kinematics solution considering collision avoidance and joint limits', *Mechanism and Machine Theory*, vol. 30, no. 5, pp. 653-663, 1995.
- [5] A. Nearchou, 'Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm', *Mechanism and Machine Theory*, vol. 33, no. 3, pp. 273-292, 1998.
- [6] M. Özgören, 'Kinematic analysis of a manipulator with its position and velocity related singular configurations', *Mechanism and Machine Theory*, vol. 34, no. 7, pp. 1075-1101, 1999.
- [7] B. Karlik and S. Aydin, 'An improved approach to the solution of inverse kinematics problems for robot manipulators', *Engineering Applications of Artificial Intelligence*, vol. 13, no. 2, pp. 159-164, 2000.
- [8] C. Chen, M. Her, Y. Hung and M. Karkoub, 'Approximating a robot inverse kinematics solution using fuzzy logic tuned by genetic algorithms', *The International Journal of Advanced Manufacturing Technology*, vol. 20, no. 5, pp. 375-380, 2002.
- [9] M. Perez Rueda, A. Lara Feria, J. Fraile Marinero, J. Delgado Urrecho and J. Gonzalez Sanchez, 'Manipulator Kinematic Error Model In a Calibration Process Through Quaternion-Vector Pairs', in *International Conference on Robotics & Automation*, Washington D. C., 2002, p. 135-140.
- [10] R. Köker, C. Öz, T. Çakar and H. Ekiz, 'A study of neural network based inverse kinematics solution for a three-joint robot', *Robotics and Autonomous Systems*, vol. 49, no. 3-4, pp. 227-234, 2004.
- [11] Z. Bingul, H.M. Ertunc and C. Oysu, 'Comparison of Inverse Kinematics Solutions Using Neural Network for 6R Robot Manipulator with Offset', *ICSC Congress on Computational Intelligence Methods and Applications*, 2005.

- [12] D. Xu, C. Acosta Calderon, J. Gan, H. Hu and M. Tan, 'An analysis of the inverse kinematics for a 5-DOF manipulator', *International Journal of Automation and Computing*, vol. 2, no. 2, pp. 114-124, 2005.
- [13] R. Köker, 'Reliability-based approach to the inverse kinematics solution of robots using Elman's networks', *Engineering Applications of Artificial Intelligence*, vol. 18 pp. 685–693, 2005.
- [14] R. Mayorga and P. Sanongboon, 'Inverse kinematics and geometrically bounded singularities prevention of redundant manipulators: An Artificial Neural Network approach', *Robotics and Autonomous Systems*, vol. 53, no. 3-4, pp. 164-176, 2005.
- [15] Y. Aydin and S. Kucuk, 'Quaternion based inverse kinematics for industrial robot manipulators with Euler wrist', *ICM, IEEE 3rd International Conference on Mechatronics*, 2006.
- [16] A. Hasan, A. Hamouda, N. Ismail and H. Al-Assadi, 'An adaptive-learning algorithm to solve the inverse kinematics problem of a 6 D.O.F serial robot manipulator', *Advances in Engineering Software*, vol. 37, no. 7, pp. 432-438, 2006.
- [17] S. Tabandeh, W. Melek and C. Clark, 'An adaptive niching genetic algorithm approach for generating multiple solutions of serial manipulator inverse kinematics with applications to modular robots', *Robotica*, vol. 28, no. 04, pp. 493-507, 2009.
- [18] J. Xie and W. Qiang, 'Inverse Kinematics Problem for 6-DOF Space Manipulator Based On the Theory of Screws', *IEEE Conference on Robotics and Biomimetic*, pp. 1659–1663, 2008.
- [19] M. L. Husty, M. Pfurner, and H. Schro, 'A new and efficient algorithm for the inverse kinematics of a general serial 6 R manipulator', *Mechanism and Machine Theory*, vol. 42, pp. 66–81, 2007.
- [20] F. C. Park, 'Computational aspects of the product-of-exponentials formula for robot kinematics', *IEEE Transactions on Automatic Control*, vol. 39, no. 3, pp. 643–647, 1994.
- [21] D. T. Pham, M. Castellani, and A. A. Fahmy, 'Learning the Inverse Kinematics of a Robot Manipulator using the Bees Algorithm', *IEEE Conference on Industrial Informatics*, 2008.
- [22] S. Alavandar and M. J. Nigam, 'Inverse Kinematics Solution of 3DOF Planar Robot using ANFIS', *International Journal of Computers, Communications and Control*, vol. III, pp. 150–155, 2008.

- [23] F. Y. C. Albert, S. P. Koh, S. K. Tiong, C. P. Chen, and F. W. Yap, 'Inverse Kinematic Solution in Handling 3R Manipulator via Real-Time Genetic Algorithm', *International Symposium on Information Technology*, 2008.
- [24] M. S. Dutra, I. L. Salcedo, L. Margarita, and P. Diaz, 'New technique for inverse kinematics problem using Simulated Annealing', *International Conference on Engineering Optimization*, pp. 1–5, 2008.
- [25] E. Sariyildiz, E. Cakiray and H. Temeltas, 'A Comparative Study of Three Inverse Kinematic Methods of Serial Industrial Robot Manipulators in the Screw Theory Framework', *Int J Adv Robotic Sy*, p. 1, 2011.
- [26] C. Ayiz and S. Kucuk, 'The Kinematics of Industrial Robot Manipulators Based on the Exponential Rotational Matrices', *IEEE International Symposium on Industrial Electronics (ISIE)*, Seoul Olympic Parktel, Seoul, Korea July 5-8, 2009.
- [27] J. H. Martin, J. de Lope and M. Santos, 'A method to learn the inverse kinematics of multi-link robots by evolving neuro-controllers', *Neurocomputing*, vol. 72, no. 13-15, pp. 2806-2814, 2009.
- [28] L. Wenjun, L. Yufeng, Y. Tingli, S. Zhixing and F. Meitao, 'Numerical study on inverse kinematic analysis of 5R serial robot', *International Forum on Information Technology and Applications (IFITA)*, 2010.
- [29] S. S. Chiddarwar and N. R. Babu, 'Comparison of RBF and MLP neural networks to solve inverse kinematic problem for 6R serial robot by a fusion approach', *Engineering Applications of Artificial Intelligence*, vol. 23, no. 7, pp. 1083–1092, 2010.
- [30] A. Hasan, N. Ismail, A. Hamouda, I. Aris, M. Marhaban and H. Al-Assadi, 'Artificial neural network-based kinematics Jacobian solution for serial manipulator passing through singular configurations', *Advances in Engineering Software*, vol. 41, no. 2, pp. 359-367, 2010.
- [31] Y. Cui, 'Kinematics simulation of an aided fruit-harvesting manipulator based on ADAMS', *International Conference on Computer Design and Applications (ICCCA)*, 2010.
- [32] A. Olaru, S. Olaru, D. Paune and O. Aurel, 'Assisted Research and Optimization of the Proper Neural Network Solving the Inverse Kinematics Problem', *AMR*, vol. 463-464, pp. 827-832, 2012.
- [33] J. A. Ramírez and A. F. Rubiano, 'Optimization of Inverse Kinematics of a 3R Robotic Manipulator using Genetic Algorithms', *World Academy of Science, Engineering and Technology*, vol. 5, pp. 1425–1430, 2011.

- [34] P. Zhang, 'A psgo-based method for inverse kinematics analysis of serial dangerous articles disposal manipulator', *International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (ICQR2MSE)*, 2011.
- [35] R. Köker, 'A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization', *Information Sciences*, vol. 222, pp. 528-543, 2013.
- [36] T. Morishita and O. Tojo, 'Integer inverse kinematics method using Fuzzy logic', *Intelligent Service Robotics*, vol. 6, no. 2, pp. 101-108, 2012.
- [37] A. Perez and J. McCarthy, 'Dual Quaternion Synthesis of Constrained Robotic Systems', *J. Mech. Des.*, vol. 126, no. 3, p. 425, 2004.
- [38] L. G. Herrera-Bendezu, E. Mu and J. T. Cain, 'Symbolic computation of robot manipulator kinematics', *IEEE International Conference on Robotics and Automation*, 1988.
- [39] D. R. Smith and H. Lipkin, 'Analysis of fourth order manipulator kinematics using conic sections', *IEEE International Conference on Robotics and Automation*, 1990.
- [40] P. L. Chebyshev, 'Théorie des mécanismes connus sous le nom de parallélogrammes', *Mém. Acad. Sci. Pétersb* 7, pp. 539-568, 1854.
- [41] J. J. Sylvester, 'On recent discoveries in mechanical conversion of motion', *Proc. Roy. Inst. Great Britain*, Vol. 7/5, pp. 179-198.
- [42] M. Grübler, "Allgemeine Eigenschaften der swanglaufigen ebeden kinematischen ketten", *part I Zivilingenieur* 29, pp. 179-223, 1885.
- [43] P. O. Somov and J. P. Chen, 'On the DoF of kinematic chains', pp. 443-447, 1985.
- [44] K. I. Gokhman, 'Equation for mobility definition and solution of mechanism classification', Odessa, 1889.
- [45] G. Koenigs, 'Introduction a une théorie nouvelle des mécanismes', *Librairie Scientifique A. Hermann, Paris*, pp. 27-28, 1905.
- [46] L. V. Assur, 'Investigation of plane linkage mechanisms with lower pairs from point of view of their structure and classification', *unpublished dissertation*, p.529, 1916.
- [47] R. Mueller, 'Die zwanglanfigkeit kinematische ketten', *unpublished dissertation (as quoted in Federhofer, K., 1932. "Graphische kinematic and kinetostatic" (Springer, Berlin))*, 1920.
- [48] A. P. Malushev, 'Analysis and synthesis of mechanisms from point of view of their structure', Tomsk, p. 78, 1929.
- [49] K. Kutzbach, 'Mechanische leitungsverzweigung, ihre geetze und anwendungen, Maschinenbau', *Betrieb*, Vol. 8, pp. 710-716, 1929.

- [50] N. I. Kolchin, 'Experience structure of widening structural classification of mechanisms and base to its structural table of mechanisms', *Analysis and Synthesis of Mechanisms, Moscow*, pp. 85-97, 1960.
- [51] I. I. Artobolevskii, 'To structure of spatial mechanisms', Vol. 10, pp. 110-152, and 'Experience of structural analysis, Structure and classification of mechanisms', Moscow, pp. 49-66, 1935-1939.
- [52] V. V. Dobrovolskii, 'Main principles of rational classification', *AS USSR*, pp. 5-48, 1939.
- [53] U. F. Moroshkin, 'Geometry problems of complex kinematic chains', *AS USSR*, Vol. 119, pp. 38-41, 1958.
- [54] R. Voinea, and M. Atanasiu, 'Contribution a l'etude de la structure des chaines cinematiques', *Bulrtinul Institutului Pol., Bucharesti*, 21-1, 1960.
- [55] B. Paul, 'A unified criterion for the degree of constraint of plane kinematic chains', *J. Appl. Mechanics*, Vol. 27, pp. 196-200, 1960.
- [56] W. Rössner, 'Zur strukturellen ordnung der getriebe', *Wissenschaft. Tech. Univ., Dresden*, Vol. 10, pp. 1101-1115, 1960.
- [57] H. Boden, 'Zum zwanglauf genuscht räumlichebener getriebe', *Maschinenbautechnik* Vol. 11, pp. 612-615, 1962.
- [58] O. G. Ozol, 'Expansion of structural theory and classification plane mechanisms with lower pairs', *Latv. Acrycult. Acad*, Vol. 13, pp. 71-91, 1963.
- [59] K. J. Waldron, 'The constraint analysis of mechanisms', *J. Mech.*, Vol. 1, pp. 101-114, 1966.
- [60] N. I. Manolescu, 'A method based on Baranov trusses and using graph theory to find the set of planar jointed kinematic chains and mechanisms', *Elsevier, IFToMM J. Mech. Mach. Theory*, Vol. 1, pp. 3-22, 1973.
- [61] C. Bagci, 'Degrees of freedom of motion in mechanisms', *ASME J. Eng. Industry*, Vol. 93, B pp. 140-148, 1971.
- [62] P. Antonescu, 'Extending of structural formula of Dobrovolski to the complex mechanisms with apparent family', *Proceedings of the SYROM*, Bucharest, 1973.
- [63] F. Freudenstein and R. I. Alizade, 'On the degree of freedom of mechanisms with variable general constraint', *IV World IFToMM Congress*, England, pp. 51-56, 1975.
- [64] K. H. Hunt, 'Kinematic geometry of mechanisms', *Oxford Univ. Press, Oxford*, 1978.
- [65] J. M. Herve, 'Analyse structurelle des mécanismes par groupe des déplacements', *Mech. Mach. Theory*, Vol. 13, pp. 437-450, 1978.

- [66] A. Gronowicz, 'Identifizierungs-Methode der Zwanglaufbedingungen von kinematischen ketten', *Elsevier, IFToMM J. Mech. Mach. Theory*, Vol. 16, pp. 127-135, 1981.
- [67] T. H. Davies and F. R. E. Crossley, 'Structural analysis of plane linkages by Franke's condensed notation', *J. Mech.*, Vol. 1, pp. 171-183, 1966.
- [68] V. P. Agrawal and J. S. Rao, 'Fractionated freedom kinematic chains and mechanisms', *Elsevier, IFToMM J. Mech. Mach. Theory*, Vol. 22, pp. 125-130, 1987.
- [69] F. Dudita and D. Diaconescu, 'Optimizarea structurala a mecanismelor', *Technica, Bucuresti*, pp.36-45, pp. 229-254, 1987.
- [70] J. Angeles and C. Gosselin, 'Determination du degre de liberte des chaines cinematiques', *Trans. CSME*, Vol.12/4, pp. 219-226, 1988.
- [71] R. I. Alizade, 'Investigation of linkage mechanisms with lower pairs from point of view of its structure and classification', *Azerb. Poly. Inst.*, Baku, pp. 111-127, 1988.
- [72] J. M. McCarthy, 'Geometric design of linkages', *Springer-Verlag, N.Y.*, p.320, 2000.
- [73] Z. Huang and Q. C. Li, 'Type synthesis of symmetrical lower-mobility parallel mechanisms using the constraint-synthesis method', *J. Robotics Res.*, Vol. 22, pp. 59-79, 2003.
- [74] R. I. Alizade and C. Bayram, 'Structural synthesis of parallel manipulators', *Elsevier, IFToMM J. Mech. Mach. Theory*, Vol. 39, pp. 857-870, 2004.
- [75] G. Gogu, 'Mobility of mechanisms: a critical review', *Elsevier, IFToMM J. Mech. Mach. Theory*, Vol. 40, pp. 1068-1097, 2005.
- [76] R. Alizade, C. Bayram, and E. Gezgin, 'Structural synthesis of Serial Platform Manipulators', Elsevier, *IFToMM J. Mech. Mach. Theory*, MMT 40-129, 2006.
- [77] Y. J. Kanayama and G. W. Krahn, 'Theory of Two-Dimensional Transformations', vol. 14, no. 5, pp. 827-834, 1998.
- [78] R. P. Paul and H. Zhang, 'Computationally Efficient Kinematics for Manipulators with Spherical Wrists Based on the Homogeneous Transformation Representation', *the International Journal of Robotics Research*, vol. 5 no. 2, 32-44, 1986.
- [79] N. Aspragathos and J. Dimitros, 'A comparative study of three methods for robot kinematics', *IEEE Trans. Syst., Man, Cybern. B*, vol. 28, no. 2, pp. 135-145, 1998.
- [80] D. Xu, C. Acosta Calderon, J. Gan, H. Hu and M. Tan, 'An analysis of the inverse kinematics for a 5-DOF manipulator', *Int J Automat Comput*, vol. 2, no. 2, pp. 114-124, 2005.

- [81] D. Manocha and J. Canny, 'Efficient inverse kinematics for general 6R manipulators', *IEEE Trans. Robot. Automat.* vol. 10, no. 5, pp. 648-657, 1994.
- [82] J. Lai and C. Menq, 'Motion Control of Manipulators with Closed-Form Inverse Solutions Near Wrist Singularities', *Journal of Engineering for Industry*, vol. 111, no. 1, p. 87, 1989.
- [83] G. Pennock and M. Raghavan, 'Spatial Mechanisms and Robot Manipulators', *Journal of Mechanical Design*, vol. 128, no. 1, p. 149, 2006.
- [84] S. Kucuk and Z. Bingul, 'The inverse kinematics solutions of industrial robot manipulators', *IEEE International Conference on ICM, Mechatronics*, 2004.
- [85] M. Walker, 'Manipulator kinematics and the epsilon algebra', *IEEE J. Robot. Automat.*, vol. 4, no. 2, pp. 186-192, 1988.
- [86] T. Balkan, M. Kemal Özgören, M. Sahir Arıkan and H. Murat Baykurt, 'A kinematic structure-based classification and compact kinematic equations for six-dof industrial robotic manipulators', *Mechanism and Machine Theory*, vol. 36, no. 7, pp. 817-832, 2001.
- [87] D. R. Smith and H. Lipkin, 'Analysis of fourth order manipulator kinematics using conic sections', *IEEE International Conference on Robotics and Automation*, 1990.
- [88] M. Ceccarelli, 'Editorial: Kinematic Design of Manipulators', *The Open Mechanical Engineering Journal*, vol. 4, no. 1, pp. 36-36, 2010.
- [89] K. Low and R. Dubey, 'A Comparative Study of Generalized Coordinates for Solving the Inverse- Kinematics Problem of a 6R Robot Manipulator', *The International Journal of Robotics Research*, vol. 5, no. 4, pp. 69-88, 1986.
- [90] A. Perez-Gracia, 'Synthesis of Spatial RPRP Closed Linkages for a Given Screw System', *J. Mechanisms Robotics*, vol. 3, no. 2, p. 021009, 2011.
- [91] G. K. Singh and J. Claassens, 'An Analytical Solution for the Inverse Kinematics of a Redundant 7DoF Manipulator with Link Offsets', *International Conference on Intelligent Robots and Systems*, pp. 2976–2982, 2010.
- [92] J. Nielsen and B. Roth, 'Formulation and Solution for the Direct and Inverse Kinematics Problems for Mechanisms and Mechatronics Systems', *Computational Methods in Mechanical Systems NATO ASI Series*, Vol. 161, pp 33-52, 1998.
- [93] S. Xin, L. Feng, H. Bing and Y. Li, 'A Simple Method for Inverse Kinematic Analysis of the General 6R Serial Robot', *J. Mech. Des.*, vol. 129, no. 8, p. 793, 2007.
- [94] C. Mavroidis, E. Lee and M. Alam, 'A New Polynomial Solution to the Geometric Design Problem of Spatial R-R Robot Manipulators Using the

- Denavit and Hartenberg Parameters', *Journal of Mechanical Design*, vol. 123, no. 1, p. 58, 2001.
- [95] I. Chen, G. Yang and I. Kang, 'Numerical inverse kinematics for modular reconfigurable robots', *Journal of Robotic Systems*, vol. 16, no. 4, pp. 213-225, 1999.
- [96] J. Rico, J. Gallardo and B. Ravani, 'Lie Algebra and the Mobility of Kinematic Chains', *Journal of Robotic Systems*, vol. 20, no. 8, pp. 477-499, 2003.
- [97] A. Perez and J. McCarthy, 'Dual Quaternion Synthesis of Constrained Robotic Systems', *J. Mech. Des.*, vol. 126, no. 3, p. 425, 2004.
- [98] A. Perez and J. McCarthy, 'Clifford Algebra Exponentials and Planar Linkage Synthesis Equations', *J. Mech. Des.*, vol. 127, no. 5, p. 931, 2005.
- [99] L. Radavelli, R. Simoni, E. D. E. Pieri, and D. Martins, 'A Comparative Study of the Kinematics of Robots Manipulators by Denavit-Hartenberg and Dual', *Mecánica Computacional, Multi-Body Systems*, vol. 31 (15), 2833-2848, 2012.
- [100] E. S. Serra and A. Perez-gracia, "Kinematic synthesis using tree topologies," *MAMT*, vol. 72, pp. 94–113, 2014.
- [101] V. Krovi, G. Ananthasuresh and V. Kumar, 'Kinematic and Kinetostatic Synthesis of Planar Coupled Serial Chain Mechanisms', *Journal of Mechanical Design*, vol. 124, no. 2, p. 301, 2002.
- [102] E. Lee, C. Mavroidis and J. Merlet, 'Five Precision Point Synthesis of Spatial RRR Manipulators Using Interval Analysis', *J. Mech. Des.*, vol. 126, no. 5, p. 842, 2004.
- [103] A. Perez and J. McCarthy, 'Clifford Algebra Exponentials and Planar Linkage Synthesis Equations', *J. Mech. Des.*, vol. 127, no. 5, p. 931, 2005.
- [104] G. Hegedüs, J. Schicho and H. Schröcker, 'Four-Pose Synthesis of Angle-Symmetric 6R Linkages', *J. Mechanisms Robotics*, vol. 7, no. 4, p. 041006, 2015.
- [105] X. Zhang and C. Nelson, 'Multiple-Criteria Kinematic Optimization for the Design of Spherical Serial Mechanisms Using Genetic Algorithms', *J. Mech. Des.*, vol. 133, no. 1, p. 011005, 2011.
- [106] A. Müller, 'On the Manifold Property of the Set of Singularities of Kinematic Mappings: Modeling, Classification, and Genericity', *J. Mechanisms Robotics*, vol. 3, no. 1, p. 011006, 2011.
- [107] C. Mavroidis, "Method to determine uncertain configurations of 6R manipulators," 1989.
- [108] T. Balkan, M. Kemal Özgören, M. Sahir Arıkan and H. Murat Baykurt, 'Structure Based Classification and Kinematic Analysis of Six-Joint Industrial

- Robotic Manipulators', *Industrial Robotics: Theory, Modelling and Control*, Sam Cubero (Ed.), pp. 964, ISBN: 3-86611-285-8, 2006.
- [109] M. K. Özgören, 'Kinematic Analysis of Spatial Mechanical Systems Using Exponential Rotation Matrices', *Transactions of the ASME*, 1144 / Vol. 129, 2007.
- [110] E. Pennestri and P. P. Valentini, 'Linear Dual Algebra Algorithms and their Application to Kinematics', *Computational Methods and Applications*, vol. 12, pp. 207–229, 2009.
- [111] E. Lee and C. Mavroidis, 'Geometric Design of 3R Robot Manipulators for Reaching Four End-Effector Spatial Poses', *IJRR*, vol. 23, no. 3, pp. 247-254, 2004.
- [112] Z. Liang, S. Meng and D. Changkun, 'Accuracy Analysis of SCARA Industrial Robot Based on Screw Theory', *IEEE*, 2011.
- [113] H. Zhuang, Z. Roth and R. Sudhakar, 'Simultaneous robot/world and tool/flange calibration by solving homogeneous transformation equations of the form $AX=YB$ ', *IEEE Trans. Robot. Automat.* vol. 10, no. 4, pp. 549-554, 1994.
- [114] S. Yahya, M. Moghavvemi, and H. A. F. Mohamed, 'Simulation Modelling Practice and Theory Geometrical approach of planar hyper-redundant manipulators : Inverse kinematics, path planning and workspace', *Simul. Model. Pract. Theory*, vol. 19, no. 1, pp. 406–422, 2011.
- [115] Y. Cui, J. Hua and P. Shi, 'Kinematics Simulation of an Aided Fruit-harvesting Manipulator Based on ADAMS', *International Conference on Computer Design and Applications*, (ICCD) 2010.
- [116] S. Ahmed and A. Pechev, 'Performance Analysis of FIK and DLS Inverse Kinematics Using Six Degree of Freedom Manipulator', *IEEE International Conference on Robotics and Biomimetics*, 2009.
- [117] Y. Wei, S. Jian, S. He and Z. Wang, 'General approach for inverse kinematics of nR robots', *Mechanism and Machine Theory*, vol. 75, pp. 97-106, 2014.
- [118] M. Palacios, 'The unified orthogonal architecture of industrial serial manipulators', *Robotics and Computer-Integrated Manufacturing*, vol. 29, pp. 257–271, 2013.
- [119] R. Muszyński, 'A solution to the singular inverse kinematic problem for a manipulation robot mounted on a track', *Control Engineering Practice*, vol. 10, no. 1, pp. 35-43, 2002.
- [120] Z. Bhatti, A. Shah, F. Shahidi and M. Karbasi, 'Forward and Inverse Kinematics Seamless Matching Using Jacobian', *Sindh Univ. Res. Jour. (Sci. Ser.)*, Vol. 45 (2) pp. 387-392, 2013.

- [121] R. Herrera, S. Alcántara, and A. Velázquez, 'Kinematic and Dynamic Modelling of Serial Robotic Manipulators Using Dual Number Algebra', *Kinematics, Dynamics, Control and Optimization*, ISBN: 978-953-51-0437-72007, 2012.
- [122] X. Luo and W. Wei, 'A New Immune Genetic Algorithm and Its Application in Redundant Manipulator Path Planning', *J. Robotic Syst.*, vol. 21, no. 3, pp. 141-151, 2004.
- [123] J. Karpińska, K. Tchoń and M. Janiak, 'Approximation of Jacobian Inverse Kinematics Algorithms: Differential Geometric vs. Variational Approach', *Journal of Intelligent & Robotic Systems*, vol. 68, no. 3-4, pp. 211-224, 2012.
- [124] M. Brandstotter, A. Angerer, and M. Hofbauer, 'An Analytical Solution of the Inverse Kinematics Problem of Industrial', *Proceedings of the Austrian Robotics Workshop, 22-23 May, 2014, Linz, Austria, 2014*.
- [125] N. Kofinas, E. Orfanoudakis and M. Lagoudakis, 'Complete Analytical Forward and Inverse Kinematics for the NAO Humanoid Robot', *Journal of Intelligent & Robotic Systems*, vol. 77, no. 2, pp. 251-264, 2014.
- [126] T. Szkodny, 'Forward and inverse kinematics of IRb-6 manipulator', *Mechanism and Machine Theory*, vol. 30, no. 7, pp. 1039-1056, 1995.
- [127] X. Wang, D. Han, C. Yu and Z. Zheng, 'The geometric structure of unit dual quaternion with application in kinematic control', *Journal of Mathematical Analysis and Applications*, vol. 389, no. 2, pp. 1352-1364, 2012.
- [128] X. Feng and W. Wan, 'Dual Quaternion Blending Algorithm and Its Application in Character Animation', *TELKOMNIKA Indonesian Journal of Electrical Engineering*, vol. 11, no. 10, 2013.
- [129] Y. Gu and J. Luh, 'Dual-number transformation and its applications to robotics', *IEEE J. Robot. Automat.* vol. 3, no. 6, pp. 615-623, 1987.
- [130] M. Wenz, and H. Worn, 'Solving the Inverse Kinematics Problem Symbolically by Means of Knowledge-Based and Linear Algebra-Based Methods', *IEEE*, pp. 1346-1353, 2007.
- [131] S. Neppalli, M. Csencsits, B. Jones and I. Walker, 'Closed-Form Inverse Kinematics for Continuum Manipulators', *Advanced Robotics*, vol. 23, no. 15, pp. 2077-2091, 2009.
- [132] V. Olunloyo, O. Ibidapo-obe, D. Olowookere, and M. Ayomoh, "Inverse Kinematics Analysis of a Five Jointed Revolute Arm Mechanism," *Journal of Control Science and Engineering*, vol. 2, pp. 7-15, 2014.
- [133] N. Yildirim and M. Bayram, 'Derivation of conservation relationships for metabolic networks using MAPLE', *Applied Mathematics and Computation*, vol. 112, no. 2-3, pp. 255-263, 2000.

- [134] K. Der, R. Sumner and J. Popović, 'Inverse kinematics for reduced deformable models', *ACM Trans. Graph.*, vol. 25, no. 3, p. 1174, 2006.
- [135] N. Zoric, M. Lazarevic, and A. Simonovic, 'Multi-Body Kinematics and Dynamics in Terms of Quaternions : Langrange Formulation in Covariant Form – Rodriguez Approach', *FME Transactions*, no. 10, pp. 19–28, 2010.
- [136] C. A. A. Calderon, E. M. R. P. Alfaro, J. Q. Gan, and H. Hu, 'Trajectory Generation and Tracking of a 5-DOF robotic Arm', *CONTROL*, 2004.
- [137] S. Ahmmad, R. Khan, M. Rahman and M. Billah, 'Position Control of a Four Link Hyper Redundant Robotic Manipulator', *Asian J. of Scientific Research*, vol. 6, no. 1, pp. 67-77, 2013.
- [138] V. Fedák and F. Ďurovský, 'Analysis of Robotic System Motion in SimMechanics and MATLAB GUI Environment', *Numerical Analysis and Scientific Computing*, 2014.
- [139] M. Gouasmi, M. Ouali, B. Fernini and M. Meghatria, 'Kinematic Modelling and Simulation of a 2-R Robot Using SolidWorks and Verification by MATLAB/Simulink', *Int J Adv Robotic Sy*, p. 1, 2012.
- [140] A. B. Rehiara, 'Kinematics of Adept Three Robot Arm, Robot Arms', *InTech Robotics and Automation*, 2011.
- [141] M. Dahari and J. Tan, 'Forward and Inverse Kinematics Model for Robotic Welding Process Using KR-16KS KUKA Robot', *International Conference on Modelling, Simulation and Applied Optimization (ICMSAO)*, pp. 2–7, 2011.
- [142] L. Soares and V. Alcalde, 'A Modeling and Simulation Platform for Robot Kinematics Aiming Visual Servo Control', *Visual Servoing, Rong-Fong Fung* (Ed.), ISBN: 978-953-307-095-7, 2010.
- [143] Wang, L. Hang and T. Yang, 'Inverse Kinematics Analysis of General 6R Serial Robot Mechanism Based on Groebner Base', *Frontiers of Mechanical Engineering in China*, vol. 1, no. 1, pp. 115-124, 2006.
- [144] D. Gan, Q. Liao, S. Wei, J. Dai and S. Qiao, 'Dual quaternion-based inverse kinematics of the general spatial 7R mechanism', *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 222, no. 8, pp. 1593-1598, 2008.
- [145] Y. Chelnokov, 'Biquaternion solution of the kinematic control problem for the motion of a rigid body and its application to the solution of inverse problems of robot-manipulator kinematics', *Mech. Solids*, vol. 48, no. 1, pp. 31-46, 2013.
- [146] R. Pérez-rodríguez, A. Marcano-cedeño, Ú. Costa, J. Solana, C. Cáceres, E. Opiiso, J. M. Tormos, J. Medina, and E. J. Gómez, 'Applications Inverse kinematics of a 6 DoF human upper limb using ANFIS and ANN for

- anticipatory actuation in ADL-based physical Neurorehabilitation', *Expert systems*, vol. 39, pp. 9612–9622, 2012.
- [147] S. Chiddarwar and N. Ramesh Babu, 'Comparison of RBF and MLP neural networks to solve inverse kinematic problem for 6R serial robot by a fusion approach', *Engineering Applications of Artificial Intelligence*, vol. 23, no. 7, pp. 1083-1092, 2010.
- [148] R. Köker, 'A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization', *Information Sciences*, vol. 222, pp. 528-543, 2013.
- [149] B. Karlik and S. Aydin, 'An improved approach to the solution of inverse kinematics problems for robot manipulators', *Engineering Applications of Artificial Intelligence*, vol. 13, pp. 159–164, 2000.
- [150] A. Hasan, A. Hamouda, N. Ismail and H. Al-Assadi, 'An adaptive-learning algorithm to solve the inverse kinematics problem of a 6 D.O.F serial robot manipulator', *Advances in Engineering Software*, vol. 37, no. 7, pp. 432-438, 2006.
- [151] B. Bocsi, D. Nguyen-Tuong, L. Csato, and S. Bernhard, 'Learning Inverse Kinematics with Structured Prediction', *International Conference on Intelligent Robots and Systems (IROS)*, 2011.
- [152] A. Hasan, N. Ismail, A. Hamouda, I. Aris, M. Marhaban and H. Al-Assadi, 'Artificial neural network-based kinematics Jacobian solution for serial manipulator passing through singular configurations', *Advances in Engineering Software*, vol. 41, no. 2, pp. 359-367, 2010.
- [153] A. Olaru, S. Olaru, D. Paune and O. Aurel, 'Assisted Research and Optimization of the Proper Neural Network Solving the Inverse Kinematics Problem', *AMR*, vol. 463-464, pp. 827-832, 2012.
- [154] R. Mayorga and P. Sanongboon, 'Inverse kinematics and geometrically bounded singularities prevention of redundant manipulators: An Artificial Neural Network approach', *Robotics and Autonomous Systems*, vol. 53, no. 3-4, pp. 164-176, 2005.
- [155] P. Kalra, 'A Neuro-genetic Algorithm Approach for Solving the Inverse Kinematics of Robotic Manipulators', *IEEE International Conference on Systems, Man and Cybernetics*, pp. 1979–1984, 2003.
- [156] T. Bhattacharjee and A. Bhattacharjee, 'A Study of Neural Network Based Inverse Kinematics Solution for a Planar', *Assam University Journal of Science & Technology: Physical Sciences and Technology*, pp. 1–7, 2010.

- [157] P. Martin and M. R. Emami, 'A neuro-fuzzy approach to real-time trajectory generation for robotic rehabilitation', *Rob. Auton. Syst.*, vol. 62, no. 4, pp. 568–578, 2014.
- [158] Y. Feng, W. Yao-nan and Y. Yi-min, 'Inverse Kinematics Solution for Robot Manipulator based on Neural Network under Joint Subspace', *International Journal of Computers Communications & Control*, vol. 7, no. 3, p. 459, 2014.
- [159] Z. Bingul, H. M. Ertunc, C. Oysu, 'Applying Neural Network to Inverse Kinematic Problem for 6R Robot Manipulator with Offset Wrist', *Proceedings of the International Conference in Coimbra*, Portugal, 2005.
- [160] A. Hasan, N. Ismail, A. Hamouda, I. Aris, M. Marhaban and H. Al-Assadi, 'Artificial neural network-based kinematics Jacobian solution for serial manipulator passing through singular configurations', *Advances in Engineering Software*, vol. 41, no. 2, pp. 359-367, 2010.
- [161] P. J. Alsina, 'Robot inverse kinematics: a modular neural network approach', *Proceedings of the 38th Midwest Symposium on Circuits and Systems*, 1995.
- [162] K. Onozato and Y. Maeda, 'Learning of Inverse-dynamics and Inverse-kinematics for Two-link SCARA Robot Using Neural Networks', *SICE, Annual Conference*, pp. 1031–1034, 2007.
- [163] M. Al-khedher and M. Alshamasin, 'SCARA Robot Control using Neural Networks', *International Conference on Intelligent and Advanced Systems (ICIAS)*, pp. 126–130, 2012.
- [164] R. Mayorga and P. Sanongboon, 'Inverse kinematics and geometrically bounded singularities prevention of redundant manipulators: An Artificial Neural Network approach', *Robotics and Autonomous Systems*, vol. 53, no. 3-4, pp. 164-176, 2005.
- [165] B. Daachi and A. Benallegue, 'A Neural Network Adaptive Controller for End-effector Tracking of Redundant Robot Manipulators', *J Intell Robot Syst*, vol. 46, no. 3, pp. 245-262, 2006.
- [166] D. Howard and A. Zilouchian, 'Application of Fuzzy Logic for the Solution of Inverse Kinematics and Hierarchical Controls of Robotic Manipulators', *Journal of Intelligent and Robotic Systems*, pp. 217–247, 1998.
- [167] S. Kumar and K. Irshad, 'Implementation of Artificial Neural Network applied for the solution of inverse kinematics of 2-link serial chain', *International Journal of Engineering Science and Technology (IJEST)*, vol. 4, no. 09, pp. 4012–4024, 2012.
- [168] E. Oyama and N. Y. Chong, 'Inverse Kinematics Learning by Modular Architecture Neural Networks with Performance Prediction Networks',

Proceedings of the 2001 IEEE International Conference on Robotics & Automation, 2001.

- [169] S. Tejomurtula and S. Kak, 'Inverse kinematics in robotics using neural networks', *Information Sciences*, vol. 116, pp. 147–164, 1999.
- [170] S. Kim and J. Lee, 'Inverse kinematics solution based on fuzzy logic for redundant manipulators', *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 904–910, 1993.
- [171] S. Alavandar and M. J. Nigam, 'Inverse Kinematics Solution of 3DOF Planar Robot using ANFIS', *Int. J. of Computers, Communications & Control*, vol. III, pp. 150–155, 2008.
- [172] C. Kozalziewicz, T. Ogiso, and N. Miyake, 'Partitioned Neural Network architecture for Inverse Kinematic calculation of a 6 dof robot manipulator', *IEEE International Joint Conference on Neural Networks*, no. 2, 2001.
- [173] Y. Kuroe, Y. Nakai and T. Mori, 'A New Neural Network Learning of Inverse Kinematics of Robot Manipulator', *IEEE*, 1994.
- [174] H. Jack, D. Lee, R. Buchal and W. H. Elmaraghy, 'Neural networks and the inverse kinematics problem', *Journal of Intelligent Manufacturing* vol. 4, 43-66, 1993.
- [175] A. Aristidou and J. Lasenby, 'FABRIK : A fast, iterative solver for the Inverse Kinematics problem', *Graph. Models*, vol. 73, no. 5, pp. 243–260, 2011.
- [176] M. Engell-Nørregard and K. Erleben, 'A projected back-tracking line-search for constrained interactive inverse kinematics', *Computers & Graphics*, vol. 35, 288–298, 2011.
- [177] Y. Zhang, Z. Tan, K. Chen, Z. Yang, and X. Lv, 'Repetitive motion of redundant robots planned by three kinds of recurrent neural networks and illustrated with a four-link planar manipulator ' s straight-line example', *Rob. Auton. Syst.*, vol. 57, no. 6–7, pp. 645–651, 2009.
- [178] M. Duguleana, F. Grigore, A. Teirelbar, and G. Mogan, 'Robotics and Computer-Integrated Manufacturing Obstacle avoidance of redundant manipulators using neural networks based reinforcement learning', *Robot. Comput. Integr. Manuf.*, vol. 28, no. 2, pp. 132–146, 2012.
- [179] B. Daya, S. Khawandi, and M. Akoum, 'Applying Neural Network Architecture for Inverse Kinematics Problem in Robotics', *J. Software Engineering & Applications*, vol. 2010, no. March, pp. 230–239, 2010.
- [180] W. Xiulan and S. Danghong, 'Manipulator inverse kinematics control based on particle swarm optimization neural network', *International Symposium on Instrumentation and Control Technology*, vol. 7129, pp. 1–6, 2008.

- [181] M. Aghajarian and K. Kiani, 'Inverse Kinematics Solution of PUMA 560 Robot Arm Using ANFIS', *International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pp. 574–578, 2011.
- [182] W. Shen, J. Gu, W. Shen, J. Gu, and E. E. Milios, 'Self-Configuration Fuzzy System for Inverse Kinematics of Robot Manipulators', *IEEE*, 2006.
- [183] K. Kinoshita, H. Matsushita, M. Izumida, and K. Murakami, 'Estimation of Inverse Kinematics Model by Forward-Propagation Rule with a High-Order Term', *IEEE, ICARCV*, pp. 0–5, 2006.
- [184] A. Borboni, 'Solution of the inverse kinematic problem of a serial manipulator by a fuzzy algorithm', *IEEE International Fuzzy Systems Conference*, no. 1, pp. 336–339, 2001.
- [185] H. Meshref, H. Vanlandingham, and C. Engineering, 'Immune Network Simulation', *IEEE Mountain Workshop on Soft Computing in Industrial Applications*, pp. 81–85, 2001.
- [186] Y. Al-Mashhadany, 'Inverse Kinematics Problem (IKP) of 6-DOF Manipulator by Locally Recurrent Neural Networks (LRNNs)', *IEEE*, 2010.
- [187] G. Asuni, G. Teti, C. Laschi, E. Guglielmelli, and P. Dario, 'A Bio-Inspired Sensory-Motor Neural Model for a Neuro-Robotic Manipulation Platform', *IEEE International Conference on Advanced Robotics*, pp. 607–612, 2005.
- [188] S. Yildirim and I. Eski, 'A QP Neural Network Inverse Kinematic Solution for Accurate Robot Path Control', *Journal of Mechanical Science and Technology*, vol. 20, No. 7, pp. 917-928, 2006.
- [189] P. Y. Zhang, T. S. Lu and L. B. Song, 'RBF networks-based inverse kinematics of 6R manipulator', *Int J Adv Manuf Technol*, vol. 26, pp. 144–147, 2005.
- [190] R. Köker, 'A neuro-simulated annealing approach to the inverse kinematics solution of redundant robotic manipulators', *Engineering with Computers*, vol. 29, no. 4, pp. 507-515, 2012.
- [191] M.-G. Her, C.-Y. Chen, Y.-C. Hung and M. Karkoub, 'Approximating a Robot Inverse Kinematics Solution Using Fuzzy Logic Tuned by Genetic Algorithms Serial Manipulators with an Ortho-parallel Basis and a Spherical Wrist', *Int J Adv Manuf Technol*, vol. 20, pp. 375–380, 2002.
- [192] C. Hua, W. Chen and T. Xie, 'Wavelet network solution for the inverse kinematics problem in robotic manipulator', *J. Zhejiang Univ. - Sci. A*, vol. 7, no. 4, pp. 525-529, 2006.
- [193] V. Agarwal, 'Trajectory planning of redundant manipulator using fuzzy clustering method', *Int. J. Adv. Manuf. Technology*, pp. 727–744, 2012.

- [194] L. Qi and Y. Li, 'Inverse Kinematics Solution of Manipulator for the Steel Plate Bending Forming by Line Heating Based on SVM and GA', *International Conference On Mechatronics Technology*, pp. 415–418, 2012.
- [195] H. Liu and D. J. Brown, 'An Extension to Fuzzy Qualitative Trigonometry and Its Application to Robot Kinematics', *IEEE International Conference on Fuzzy Systems*, pp. 1111–1118, 2006.
- [196] P. Martin and M. R. Emami, 'A neuro-fuzzy approach to real-time trajectory generation for robotic rehabilitation', *Rob. Auton. Syst.*, vol. 62, no. 4, pp. 568–578, 2014.
- [197] M. C. Netto, A. Esukoffand and M. Dutra, 'Fuzzy Systems to Solve Inverse Kinematics Problem in Robots Control: Application to a Hexapod Robots Leg', *IEEE*, 2000.
- [198] D. H. Song and S. Jung, 'Geometrical Analysis of Inverse Kinematics Solutions and Fuzzy Control of Humanoid Robot Arm under Kinematics Constraints', *IEEE International Conference on Mechatronics and Automation*, pp. 1178–1183, 2007.
- [199] M. Crengani, R. Breaz, G. Racz, and O. Bologna, 'The Inverse Kinematics Solutions of a 7 DOF Robotic Arm Using Fuzzy Logic', *IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pp. 518–523, 2013.
- [200] T. Morishita and O. Tojo, 'Integer inverse kinematics method using Fuzzy logic', *Intel Serv Robotics*, pp. 101–108, 2013.
- [201] K. Neumann, M. Rolf, J. Steil, and M. Gienger, 'Learning Inverse Kinematics for Pose-Constraint Bi-manual Movements', *Springer-Verlag Berlin Heidelberg, LNAI*, pp. 478–488, 2010.
- [202] B. Hashim, N. Ismail, and H. M. Ahmad, 'Application of Soft Computing to Serial Manipulator Kinematic Problems', *Pertanika J. Sci. & Techno. Supplement*. vol. 12, no. 2, pp. 191–200, 2004.
- [203] A. Nearchou, 'Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm', *Mechanism and Machine Theory*, vol. 33, pp. 273-292, 1998.
- [204] L. Wang and C. Chen, 'A combined optimization method for solving the inverse kinematics problems of mechanical manipulators', *IEEE Trans. Robot. Automat.* vol. 7, no. 4, pp. 489-499, 1991.
- [205] J. Parker, A. Khoogar, and D. Goldberg, 'Inverse Kinematics of Redundant Robots using Genetic Algorithms', *IEEE International Conference on Robotics and Automation*, 1989.
- [206] S. Kim and J. Kim, 'Optimal Trajectory Planning of a Redundant Manipulator using Evolutionary Programming', *IEEE*, no. 1, 1996.

- [207] A. Piazzil and A. Visioli, 'A Global Optimization Approach to Trajectory Planning for Industrial Robots', *IEEE*, 1997.
- [208] J. Ahuactzin, 'Completeness Results for a Point-to-point Inverse Kinematics Algorithm', *IEEE International Conference on Robotics & Automation*, pp. 1526–1531, 1999.
- [209] F. Chapelle and P. Bidaud, 'A Closed Form for Inverse Kinematics Approximation of General 6R Manipulators using Genetic Programming', *IEEE International Conference on Robotics & Automation*, pp. 3364–3369, 2001.
- [210] S. Khatami and F. Sassani, 'Isotropic Robotic Design Optimization of Manipulators Using a Genetic Algorithm Method', *Proceedings of the 2002 IEEE International Symposium on Intelligent Control*, Vancouver, Canada, October 27-30.2002.
- [211] P. Kalra, 'On the Solution of Multimodal Robot Inverse Kinematic Functions using Real-coded Genetic Algorithms', *IEEE*, 2003.
- [212] J. Korein and N. Badler, 'Techniques for Generating the Goal -Directed Motion of Articulated Structures', *IEEE CG&A*, November, 1982.
- [213] S. Tabandeh, C. Clark, and W. Melek, 'A Genetic Algorithm Approach to solve for Multiple Solutions of Inverse Kinematics using Adaptive Niching and Clustering', *IEEE Congress on Evolutionary Computation*, pp. 1815–1822, 2006.
- [214] G. He, H. Gao, G. Zhang and L. Wu, 'Using Adaptive Genetic Algorithm to the Placement of Serial Robot Manipulator', *IEEE*, 2006.
- [215] A. Rajpar, Q. Huang, W. Zhang, D. Jia, and K. Li, 'Object Manipulation of Humanoid Robot Based on Combined Optimization Approach', *IEEE International Conference on Mechatronics and Automation*, pp. 1148–1153, 2007.
- [216] S. Liu and S. Zhu, 'An Optimized Real Time Algorithm for the Inverse Kinematics of General 6R Robots', *IEEE International Conference on Control and Automation*, Guangzhou, CHINA - May 30 to June 1, 2007
- [217] M. H. Jaryani, 'An Effective Manipulator Trajectory Planning With Obstacles Using Virtual Potential Field Method', *IEEE*, 2007.
- [218] D. T. Pham, M. Castellani, and A. A. Fahmy, 'Learning the Inverse Kinematics of a Robot Manipulator using the Bees Algorithm', *IEEE International Conference on Industrial Informatics (INDIN)*, 2008.
- [219] F. Albert, S. Koh, S. Tiong, C. Chen, and F. Yap, 'Inverse Kinematic Solution in Handling 3R Manipulator via Real-Time Genetic Algorithm', *IEEE*, 2008.
- [220] H. Huang, S. Xu, and H. Hsu, 'Hybrid Taguchi DNA Swarm Intelligence for Optimal Inverse Kinematics Redundancy Resolution of Six-DOF Humanoid

- Robot', *Mathematical Problems in Engineering*, Vol. 2014, Article ID 358269, 9 pages, 2014.
- [221] W. Bailón, E. Cardiel, I. Campos and A. Paz, 'Mechanical Energy Optimization in Trajectory Planning for Six DOF Robot Manipulators Based on Eighth-Degree Polynomial Functions and a Genetic Algorithm', *International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE 2010)*, Tuxtla Gutiérrez, Chiapas, México. September 8-10, 2010.
- [222] A. P. Paramani, 'Optimum Inverse Kinematic Method for a 12 DOF Manipulator', *Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation*, August 7 - 10, Beijing, China.
- [223] X. Lei-ping, C. Zi-li and S. Shao-jie, 'Obstacle Avoiding Research on the Manipulator based on Genetic Algorithm', *International Conference on Instrumentation, Measurement, Computer, Communication and Control*, 2011.
- [224] F. Rubio, F. Abu-Dakka, F. Valero and V. Mata, 'Comparing the efficiency of five algorithms applied to path planning for industrial robots', *Industrial Robot: An International Journal* vol. 39/6, 580–591, 2012.
- [225] M. Galicki, 'Control-Based Solution to Inverse Kinematics for Mobile Manipulators Using Penalty Functions', *Journal of Intelligent and Robotic Systems*, vol. 42, pp. 213–238, 2005.
- [226] T. Cavdar and M. Milani, 'A New Heuristic Approach for Inverse Kinematics of Robot Arms', *American Scientific Publishers*, 2012.
- [227] W. Lalo, T. Brandt, D. Schramm and M. Hiller, 'Linear Optimization Approach to Inverse Kinematics of Redundant Robots with Respect to Manipulability', *International symposium on automation and robotics in construction*, June 26-29, 2008.
- [228] A. Bernal, 'Meta-Heuristic Optimization Techniques and Its Applications in Robotics', *Meta-Heuristic Optimization Techniques and Its Applications in Robotics*, <http://dx.doi.org/10.5772/54460>, 2013.
- [229] S. N. Cubero, 'Blind Search Inverse Kinematics for Controlling All Types of Serial-link Robot Arms', *University of Southern Queensland*.
- [230] R. Konietschke and G. Hirzinger, 'Inverse Kinematics with Closed Form Solutions for Highly Redundant Robotic Systems', *IEEE International Conference on Robotics and Automation*, Kobe International Conference Center Kobe, Japan, May 12-17, 2009.
- [231] Y. Zhang, L.-M. Xie, H. Shen and L. Jin, 'Simulation on Whole Inverse Kinematics of A 5R Robot Based on Hybrid Genetic Algorithm', *Proceedings of the Eighth International Conference on Machine Learning and Cybernetics*, Baoding, 12-15 July 2009.

- [232] W. Huang, S. Tan and X. Li, 'Inverse Kinematics of Compliant Manipulator Based on the Immune Genetic Algorithm', *Fourth International Conference on Natural Computation*, 2008.
- [233] S. Kumar, N. Sukavanam, and R. Balasubramanian, 'An Optimization Approach To Solve The Inverse Kinematics Of Redundant Manipulator', *International journal of information and systems sciences*, Volume 6, Number 4, Pages 414–423, 2010.
- [234] J. Xu, W. Wang and Y. Sun, 'Two optimization algorithms for solving robotics inverse kinematics with redundancy', *J. Control Theory Appl*, vol. 8 (2) pp. 166–175, 2010.
- [235] S. Mazhari and S. Kumar, 'PUMA 560 Optimal Trajectory Control using Genetic Algorithm, Simulated Annealing and Generalized Pattern Search Techniques', *World Academy of Science, Engineering and Technology*, pp. 800–809, 2008.
- [236] J. Ramírez and A. Rubiano, 'Optimization of Inverse Kinematics of a 3R Robotic Manipulator using Genetic Algorithms', *World Academy of Science, Engineering and Technology*, pp. 1425–1430, 2011.
- [237] Y. Feng and W. Yao-nan, 'Inverse Kinematic Solution for Robot Manipulator Based on Electromagnetism-like and Modified DFP Algorithms', *ACTA AUTOMATICA SINICA*, vol. 37, no. 1, 2011.
- [238] E. Henten, E. Schenk, L. Willigenburg, J. Meuleman, and P. Barreiro, 'Collision-free inverse kinematics of the redundant seven-link manipulator used in a cucumber picking robot', *Biosyst. Eng.*, vol. 106, no. 2, pp. 112–124, 2010.
- [239] N. Rokbani and A. Alimi, 'Inverse Kinematics Using Particle Swarm Optimization, A Statistical Analysis', *Procedia Eng.*, vol. 64, pp. 1602–1611, 2013.
- [240] M. Dutra, I. Salcedo, L. Margarita, and P. Diaz, 'New technique for inverse kinematics problem using Simulated Annealing', *International Conference on Engineering Optimization*, June, pp. 1–5, 2008.
- [241] X. Zhang and C. Nelson, 'Multiple-Criteria Kinematic Optimization for the Design of Spherical Serial Mechanisms Using Genetic Algorithms', *J. Mech. Des.*, vol. 133, no. 1, p. 011005, 2011.
- [242] N. Rokbani and A. M. Alimi, 'IK-PSO, PSO Inverse Kinematics Solver with Application to Biped Gait Generation', *International Conference on Design and Manufacturing*, vol. 58, no. 22, pp. 33–39, 2012.
- [243] X. Luo and W. Wei, 'New Immune Genetic Algorithm and Its Application in Redundant Manipulator Path Planning', *Journal of Robotic Systems*, vol.21 (3), pp. 141–151, 2004.

- [244] H. Al-Dois, A. Jha and R. Mishra, 'Task-based design optimization of serial robot manipulators', *Engineering Optimization*, vol. 45, no. 6, pp. 647-658, 2013.
- [245] S. Števo, I. Sekaj and M. Dekan, 'Optimization of Robotic Arm Trajectory Using Genetic Algorithm', *The International Federation of Automatic Control*, Cape Town, South Africa. August 24-29, 2014.
- [246] J. Denavit and R. Hartenberg, 'A kinematic notation for lower-pair mechanisms based on matrices', *Trans ASME J. Appl. Mech.*, vol. 23, pp. 215–221, 1955.
- [247] S. Haykin and S. Haykin, *Neural networks and learning machines*. New York: Prentice Hall/Pearson, 2009.
- [248] J. Kennedy, R. Eberhart and Y. Shi, *Swarm intelligence*. San Francisco: Morgan Kaufmann Publishers, 2001.
- [249] R. Rao, V. Savsani and D. Vakharia, 'Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems', *Computer-Aided Design*, vol. 43, no. 3, pp. 303-315, 2011.
- [250] S. Mirjalili, S. Mohd Hashim and H. Moradian Sardroudi, 'Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm', *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 11125-11137, 2012.
- [251] J. Zhang, J. Zhang, T. Lok and M. Lyu, 'A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training', *Applied Mathematics and Computation*, vol. 185, no. 2, pp. 1026-1037, 2007.
- [252] H. Zang, S. Zhang and K. Hapeshi, 'A Review of Nature-Inspired Algorithms', , *Journal of Bionic Engineering* vol. 7, pp. S232-S237, 2010.
- [253] A. Gandomi and A. Alavi, 'Krill herd: A new bio-inspired optimization algorithm', *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831-4845, 2012.
- [254] B. Akay and D. Karaboga, 'A modified Artificial Bee Colony algorithm for real-parameter optimization', *Information Sciences*, vol. 192, pp. 120-142, 2012.
- [255] Z. Michalewicz, *Genetic algorithms + data structures =*. Berlin: Springer-Verlag, 1996.
- [256] A. Eiben and J. Smith, *Introduction to evolutionary computing*. New York: Springer, 2003.
- [257] E. Mezura-Montes and C. Coello, 'A Simple Multimembered Evolution Strategy to Solve Constrained Optimization Problems', *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 1, pp. 1-17, 2005.

- [258] P. Stroud, 'Kalman-extended genetic algorithm for search in nonstationary environments with noisy fitness evaluations', *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 66-77, 2001.
- [259] D. Fogel, 'An introduction to simulated evolutionary optimization', *IEEE Trans. Neural Netw.*, vol. 5, no. 1, pp. 3-14, 1994.
- [260] K. Passino, 'Biomimicry of bacterial foraging for distributed optimization and control', *IEEE Control Syst. Mag.*, vol. 22, no. 3, pp. 52-67, 2002.
- [261] S. Muller, J. Marchetto, S. Airaghi and P. Kournoutsakos, 'Optimization based on bacterial chemotaxis', *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 16-29, 2002.
- [262] A. Ahrari and A. Atai, 'Grenade Explosion Method—A novel tool for optimization of multimodal functions', *Applied Soft Computing*, vol. 10, no. 4, pp. 1132-1140, 2010.
- [263] S. Mirjalili, S. Mirjalili and A. Lewis, 'Grey Wolf Optimizer', *Advances in Engineering Software*, vol. 69, pp. 46-61, 2014.
- [264] D. Wolpert and W. Macready, 'No free lunch theorems for optimization', *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67-82, 1997.
- [265] R. Carmichael, D. Rutecki, B. Annett, E. Gaines and I. Valiela, 'Position of horseshoe crabs in estuarine food webs: N and C stable isotopic study of foraging ranges and diet composition', *Journal of Experimental Marine Biology and Ecology*, vol. 299, no. 2, pp. 231-253, 2004..
- [266] B. Hazlett and W. Bossert, 'A statistical analysis of the aggressive communications systems of some hermit crabs', *Animal Behaviour*, vol. 13, no. 2-3, pp. 357-373, 1965.
- [267] L. Waldrop, 'Ontogenetic Scaling of the Olfactory Antennae and Flicking Behavior of the Shore Crab, *Hemigrapsus oregonensis*', *Chemical Senses*, vol. 38, no. 6, pp. 541-550, 2013.
- [268] M. Small and R. Thacker, 'Land hermit crabs use odors of dead conspecifics to locate shells', *Journal of Experimental Marine Biology and Ecology*, vol. 182, no. 2, pp. 169-182, 1994.
- [269] R. Carmichael, D. Rutecki, B. Annett, E. Gaines and I. Valiela, 'Position of horseshoe crabs in estuarine food webs: N and C stable isotopic study of foraging ranges and diet composition', *Journal of Experimental Marine Biology and Ecology*, vol. 299, no. 2, pp. 231-253, 2004.
- [270] R. Rotjan, J. Chabot and S. Lewis, 'Social context of shell acquisition in *Coenobita clypeatus* hermit crabs', *Behavioral Ecology*, vol. 21, no. 3, pp. 639-646, 2010.

- [271] B. E. Gravel, 'The use of artificial shells for exploring shell preference in the marine hermit crab *Pagurus longicarpus* (Say)', *Ann. Zool. Fennici*, vol. 41, pp. 477-485, 2004.
- [272] H. Murakami, T. Tomaru, Y. Nishiyama, T. Moriyama, T. Niizato and Y. Gunji, 'Emergent Runaway into an Avoidance Area in a Swarm of Soldier Crabs', *PLoS ONE*, vol. 9, no. 5, p. e97870, 2014.

Publications

JOURNALS:

- 1) **Panchanand Jha** and B B Biswal, "A Neural Network Approach for Inverse Kinematic of a SCARA Manipulator", International Journal of Robotics and Automation (IJRA) Vol. 3, No. 1, March 2014, pp. 31~40, ISSN: 2089-4856.
- 2) **Panchanand Jha**, Bibhuti B. Biswal, and Om Prakash Sahu, "Inverse Kinematic Solution of Robot Manipulator Using Hybrid Neural Network," International Journal of Materials Science and Engineering, Vol. 3, No. 1, pp. 31-38, March 2015.
- 3) Om Prakash Sahu, Bibhuti Bhusan Biswal, Saptarshi Mukharjee, and **Panchanand Jha**, "Development of Robotic End-Effector Using Sensors for Part Recognition and Grasping," International Journal of Materials Science and Engineering, Vol. 3, No. 1, pp. 39-43, March 2015.
- 4) **Panchanand Jha** et. al, "Development of an Intelligent System for Prediction of Inverse Kinematics of Robot Manipulator Arm ", IJNNA, International Science Press 5(2) July-December 2012, pp. 81-88.
- 5) **Panchanand Jha** and B B Biswal, "Inverse Kinematic Solution of 5R Manipulator using ANN and ANFIS", International Journal of Robotics and Automation (IJRA) Vol. 4, No. 2, June 2015, ISSN: 2089-4856.
- 6) **Panchanand Jha** and B B Biswal, "New Optimization Approach for Inverse Kinematic Resolution", Intelligent Service Robot, Springer. (Communicated)
- 7) **Panchanand Jha** and B B Biswal, "Metaheuristic Approach for Comparative Evaluation of Inverse Kinematic Problem", Swarm and Evolutionary Computation, Elsevier. (Communicated)

CONFERENCES:

- 1) **Panchanand Jha** and B B Biswal, "Inverse Kinematic Solution of Redundant Robot Manipulator using Artificial Neural Network ", Proceedings of International Conference on Advances in Mechanical Engineering May 29-31, 2013, COEP, Pune.
- 2) **Panchanand Jha** and B B Biswal, "Development of an Intelligent System for Prediction of Inverse Kinematics of Robot Manipulator", 4th International & 25th All India Manufacturing Technology, Design and Research Conference

(AIMTDR 2012) December 14th-16th, 2012, Jadavpur University, Kolkata, India.

- 3) Bunil Kumar Balabantaray, Panchanand Jha and B B Biswal, "Application of Edge Detection Algorithm for Vision Guided Robotics Assembly System", Proceeding of SCIEI, The 6th International Conference on Machine Vision (ICMV 2013) London, The United Kingdom November 16-17, 2013.
- 4) B B Biswal et, al., "Generation of Optimized Robotic Assembly Sequence using Immune Based Technique", Proceedings of the ASME 2012 International Mechanical Engineering Congress & Exposition IMECE2012 November 9-15, 2012, Houston, Texas, USA.
- 5) **Panchanand Jha** and B B Biswal, "Hybrid training of neural network for the computation of inverse kinematics of revolute manipulator", AIMTDR, IIT Guwahati.

CHAPTERS:

- 1) **Panchanand Jha** and B B Biswal, "Intelligent Computation and Kinematics of 4-DOF SCARA Manipulator using ANN and ANFIS", Proceedings of Springer-Verlag Berlin Heidelberg, SEMCCO 2013, Part II, LNCS 8298.
- 2) **Panchanand Jha**, Bibhuti B. Biswal, and Om Prakash Sahu, "Intelligent Computation of Inverse Kinematics of a 5-dof Manipulator Using MLPNN," Proceedings of Springer-Verlag Berlin Heidelberg, TAROS 2014, LNAI 8717, pp. 243–250, 2014.

CURRICULUM VITAE

Panchanand Jha

PhD. Research Scholar
Department of Industrial Design
National Institute of Technology
Rourkela, Odisha
Mob. No. - **+91-9437777934**
E-mail: jha_ip007@hotmail.com

Professional Objective:

To pursue a challenging career and be part of a progressive organization that gives scope to enhance my knowledge, skills and to reach the pinnacle in the computing and research field with sheer determination, dedication and hard work.

Academic Profile: -

Qualification: -

- Research Scholar in the Department of Industrial Design from **NIT Rourkela**.

Topic: Inverse Kinematics Analysis of Robot Manipulator.

- **Master of Technology** in Mechanical Engineering from **NIT Rourkela** with aggregate CGPI of **8.61/10** in year **2009**.

Thesis Title: Novel Artificial Neural Network Application for Prediction of Inverse Kinematics of Manipulator.

- **Bachelor of Engineering** in Industrial and Production Engineering from Institute of Technology, Central **University, Bilaspur (C.G.)** with aggregate CPI of **7.02/10** in year **2007**.

Schooling

- HSSC from Government school, Machadoli (Korba), with 61% in year 2003.
- HSC from Government school, Machadoli (Korba), with 76% in year 2001.

Research Area

- Industrial Robotics, Artificial Intelligence, Engineering Optimization, Design and Manufacturing.

International Exposure

- Received Slovakia Government scholarship for study stay program for five months from 01/09/2014 to 31/01/2015.

- Worked as a research scholar in the department of robotics and cybernetics, Slovak Technical University in Bratislava from **25/09/2014 to 23/12/2014**.

Work Experience

- Worked as Assistant Professor from **2/08/2009 to 04/02/2011** (06Months) in the Department of Mechanical Engineering in JKIE, Gatora, Bilaspur.
- Worked as Assistant Professor from **08/02/2011 to 30/06/2011** (06Months) in the Department of Mechanical Engineering in RCET, Bhilai.

Vocational training

- Completed 1 month of vocational training from Hasdev Bango Hydral Power Plant(C.G)
- Completed 1 month of vocational training from Precision Machine Tool Ltd. Haryana

Skills

- MATLAB R2008a
- PRO-E
- CATIA
- MINITAB 14
- AUTO CAD
- SOLID WORKS
- Hypermesh
- Operating System : Windows98, Windows8, Windows7, Windows XP.
- Office Tools : MS Office, 2007, 2010(MS Word, Excel, Power Point).

Hobbies

- Listening Music, Reading, and sports.

Personal Profile

- Date of Birth : 06th Sep.,1985
- Sex : Male
- Nationality : Indian
- Father's Name : Shri S N Jha
- Mother's Name : Smt.Vimla Jha

- Marital status : Single
- Languages known : English, Hindi.

I hereby accept that the information given is true to the best of my knowledge and belief.

Date:

Place:
JHA)

(PANCHANAND