

**“INTELLIGENT CONTROL AND OPTIMIZATION OF NONLINEAR
SYSTEMS”**

A Thesis submitted to the
UPES

For the Award of
Doctor of Philosophy
In
Engineering

By
Devendra Rawat

Jan.2023

SUPERVISOR(s)

Dr. Mukul Kumar Gupta
Dr. Abhinav Sharma



**Department of Electrical & Electronics Engineering
School of Engineering (SOE)
UPES
Dehradun- 248007: Uttarakhand**

**“INTELLIGENT CONTROL AND OPTIMIZATION OF NONLINEAR
SYSTEMS”**

A Thesis submitted to the
UPES

For the Award of
Doctor of Philosophy
In
Engineering

By
Devendra Rawat
(SAP ID 500049754)

Jan.2023

SUPERVISOR(s)

Dr. Mukul Kumar Gupta
Associate Professor
Department of Electronics and Instrumentation
MJP Rohilkhand University Uttar Pradesh
Formally Assistant Professor (SG)
UPES

Dr. Abhinav Sharma
Assistant Professor(SG)
Department of Electrical & Electronics Engineering
UPES



**Department of Electrical & Electronics Engineering
School of Engineering (SOE)
UPES
Dehradun- 248007: Uttarakhand**

DECLARATION

I declare that the thesis entitled “Intelligent control and optimization of nonlinear systems” has been prepared by me under the guidance of Dr. Mukul Kumar Gupta Assistant Professor (SG), and Dr. Abhinav Sharma Assistant Professor (SG), Department of Electrical and Electronics Engineering, UPES, Dehradun. No part of this thesis has formed the basis for the award of any degree or fellowship previously.

A rectangular box containing a handwritten signature in black ink. The signature appears to be 'Dev.' with a horizontal line underneath.

Devendra Rawat

School of Engineering [SOE],

UPES

Dehradun-248007, Uttarakhand

CERTIFICATE




CERTIFICATE

I certify that Devendra Rawat has prepared his thesis entitled "Intelligent control and optimization of nonlinear systems", for the award of Ph.D. degree of the University of Petroleum & Energy Studies, under my guidance. He has carried out the work at School of Engineering, University of Petroleum & Energy Studies.


Dr. Mukul Kumar Gupta
School of Engineering,
University of Petroleum & Energy Studies
Dehradun -248007, Uttarakhand

Date:


Dr. Abhinav Sharma
School of Engineering,
University of Petroleum & Energy Studies
Dehradun -248007, Uttarakhand

Date:

ii

Energy Acres: Bicholi Via Prem Nagar, Dehradun-248 007 [Uttarakhand], India T: +91 1352770137, 2776053/54/91, 2776201, 9997799474 F: +91 135 2776090/95
Knowledge Acres: Kandoli Via Prem Nagar, Dehradun - 248 007 [Uttarakhand], India T: +91 8171979021/2/3, 7060111775

ENGINEERING | COMPUTER SCIENCE | DESIGN | BUSINESS | LAW | HEALTH SCIENCES AND TECHNOLOGY | MODERN MEDIA | LIBERAL STUDIES

NEW

ABSTRACT

Almost every physical system is nonlinear in nature. These systems are inherently uncertain, unstable, and complex. Nonlinear systems are always challenging to control because of the properties like No superposition and homogeneity, Chaotic behavior, Multiple equilibrium states, and Input sensitivity. Several methods are employed for analyzing and controlling such types of systems. Control of these systems is always a fascinating area for researchers. Robotic manipulators are an example of such a system that has highly nonlinear, uncertain, and unstable dynamics.

With technological advancement, robotic manipulators have gained much popularity in industrial as well as medical applications. These applications include pick and place, material handling, assembling, welding, teleoperations, haptic interfaces, prosthetic limbs, and many more. Taking into consideration the non-linear characteristics and complexity of robotic manipulators, effective and optimized control is essentially required. Because of these non-linear characteristics, it is difficult to approximate the systems aimed to enhance efficiency. Conventional approaches like optimal control, a nonlinear sliding mode control (SMC), robust and adaptive control, the three-term proportional-integral-derivative (PID), and fractional order PID (FOPID) are popular control methods but need the exact modeling of the system. Intelligent control methods imbibe artificial intelligence methods in conventional control methods to increase their performance and effectiveness.

The trajectory tracking of a robotic manipulator has been presented in this work using numerous methods. First, a robust adaptive sliding mode control then PID and FOPID-based controllers have been implemented to track the reference trajectory. Furthermore, four different metaheuristic algorithms namely grey wolf optimization algorithm (GWO), whale optimization algorithm (WOA), moth flame optimization (MFO), and multiverse optimization (MVO) have been implemented

for the trajectory control of a two-link linearized robotic manipulator. Afterward, the trajectory tracking has been achieved using other four recent metaheuristic algorithms namely the arithmetic optimization algorithm (AOA), atom search optimization (ASO), spotted hyena optimizer (SHO), and sooty tern optimization (STO). These optimization techniques have considered the weighted sum of IAE and ITAE as a performance index having the error between the reference and actual trajectory as the fitness value.

All these metaheuristic algorithms are stochastic in nature; therefore, a statistical analysis has been performed by running each algorithm 10 times. In order to assess their performance a parametric statistical Friedman's ANOVA test has been performed and a Friedman ranking has been assigned to each of the algorithms. For the linearized model, the MFO outperforms the others while for the nonlinear model, the STO outperforms the other algorithms. MFO and STO attain the first rank in this test for linear and nonlinear models respectively.

Further, a novel metaheuristics algorithm hybrid of the particle swarm optimization (PSO) and sooty tern optimization (STOPSO) has been proposed for the optimization of the controllers for trajectory tracking of the robotic manipulators. The inclusion of the PSO's exploitation capability with adjustable weight in sequential mode enhances the STO's exploitation capability greatly in the proposed STOPSO algorithm. As a result, the proposed algorithm is converging equally well to the true values with minimum error. The proposed STOPSO performs better in comparison to STO and other implemented algorithms. In addition to this, the performance of the proposed STOPSO algorithm is measured based on convergence analysis, robustness, reliability, and statistical analysis for trajectory control and compared with the previous algorithms existing in the literature. In trajectory control of robotic manipulators, some applications require tracking of the optimum point in the defined trajectory, for such applications an extremum-seeking control has been designed. The perturbation type extremum seeking control attains the optimum value of the trajectory with the help of designed control laws.

ACKNOWLEDGEMENT

Foremost, I would first like to thank my Ph.D. supervisors, Dr. Mukul Kumar Gupta, and Dr. Abhinav Sharma, from UPES Dehradun. Their patience, motivation, enthusiasm, and immense knowledge have constantly encouraged me during my Ph.D. study. They constantly provided me with guidance along the way about my paper writing and presentation abilities, which helped me gain confidence and become an accomplished researcher. Without their immense support, I could never consummate my Ph.D. study. The wisdom and accomplishments they imparted to me have shaped who I am today and will continue to do so in the future.

I am thankful to Dean SoE, Dean R&D, HoD Electrical Cluster, Research coordinator SoE, and all my supervisory team for providing consistent motivation to put their best efforts into this work.

The blessings of my parents and the love of my family are my biggest support during every stage of my life. My heartfelt gratitude to my wife Ujjwala and children Yug and Shatakshi without their motivation and kindness, I could never have finished my degree.

I am extremely grateful to all of my amazing colleagues for their constant encouragement and support. Finally, I want to express my gratitude to the almighty God for guiding me through all of the challenges I have experienced in my journey. I am thankful to the almighty for being my mentor day by day.

TABLE OF CONTENTS

Declaration	i
Certificate	ii
Abstract	iii
Acknowledgment	v
Contents	vi
List of Figures	ix
List of Tables	xi
List of Abbreviations	xii
List of Symbols	xiii
1. Introduction	1
1.1 Introduction to nonlinear systems.....	1
1.2 Control of Nonlinear systems.....	2
1.3 Robotic Manipulators.....	3
1.4 Motivation of the work.....	4
1.5 Objectives.....	4
1.6 Thesis Organization.....	5
1.6.1 Chapter 2: Literature Review.....	5
1.6.2 Chapter 3: Analysis of control strategies for robotic manipulator.....	5
1.6.3 Chapter 4: Optimization and statistical analysis of control techniques for linearized model.....	5
1.6.4 Chapter 5: Optimization and statistical analysis of control techniques for nonlinear model.....	5
1.6.5 Chapter 6: Conclusion and Future Work.....	6

2. Literature Review	7
2.1 Introduction.....	7
2.2 Intelligent Control Methods.....	8
2.2.1 Artificial Neural Networks.....	9
2.2.2 Fuzzy Logic Control.....	12
2.2.3 Expert Control Systems.....	16
2.2.4 Machine Learning Control.....	19
2.2.5 Optimization Algorithms.....	21
2.3 Comparison of intelligent control methods.....	26
3. Analysis of control strategies for robotic manipulators	29
3.1 Introduction.....	29
3.2 Dynamics of a two-link Robotic Manipulator.....	29
3.3 PID and FOPID Control.....	34
3.4 Robust Adaptive Sliding Mode Control.....	37
3.5 Stability Analysis.....	41
3.6 Extremum Seeking Control.....	43
3.6.1 Design of ESC for robotic manipulator.....	45
4. Optimization and statistical analysis of control techniques for linearized model	49
4.1 Introduction.....	49
4.2 Grey Wolf Optimization (GWO).....	50
4.3 Whale Optimization Algorithm (WOA).....	52
4.4 Moth Flame Optimization (MFO).....	55
4.5 Multi-Verse Optimization (MVO).....	58
4.6 Results and Discussions.....	60
5. Optimization and statistical analysis of control techniques for nonlinear model	70
5.1 Introduction.....	70
5.2 Arithmetic Optimization Algorithm (AOA).....	71

5.3 Atom Search optimization (ASO).....	74
5.4 Spotted Hyena Optimizer (SHO).....	76
5.5 Sooty Tern Optimization (STO).....	78
5.6 Novel Hybrid STOPSO Algorithm.....	81
5.6.1 Novelty of Work.....	81
5.6.2 Hybrid STOPSO Algorithm.....	81
5.7 Results and Discussion.....	84
6. Conclusion and Future Scope of the Work	94
6.1 Conclusion.....	94
6.2 Future Scope.....	96
 Bibliography	 97
Plagiarism Report	111
Curriculum Vitae	113

List of Figures

2.1 Intelligent Control Methods.....	8
2.2 Architecture of artificial neural networks.....	10
2.3 Architecture of ANN-based control methods.....	10
2.4 Fuzzy Logic-based control approach.....	13
2.5 Architecture of MLC.....	19
2.6. Generalized implementation of an optimization technique in control.....	21
3.1 A two-link robotic manipulator.....	30
3.2 Flow diagram of mathematical modeling.....	33
3.3 Simulink diagram of the mathematical model.....	33
3.4 Block diagram of PID controller design.....	35
3.5 Block diagram of FOPID and PID controller design.....	35
3.6 Block Diagram of the adaptive SMC controller.....	37
3.7 Error of link 1.....	40
3.8 Error of link 2.....	40
3.9 Tracking error of first joint using ASMC.....	41
3.10 Tracking error of second joint using ASMC.....	42
3.11 Block diagram of ESC.....	43
3.12 Classification of ESC Techniques.....	44
3.13 Polynomial Reference trajectory.....	45
3.14 Output of ESC.....	46
3.15 Optimum value tracking of the reference trajectory.....	46
3.16 Optimum value tracking of the reference trajectory.....	47
3.17 SIMULINK model of ESC for trajectory control of robotic manipulator.....	47
4.1 Description of GWO algorithm.....	51
4.2 Flow diagram of GWO algorithm.....	52
4.3 Description of WOA.....	54
4.4 Flow diagram of WOA.....	54
4.5 Description of MFO algorithm.....	55

4.6 Flow diagram of MFO algorithm.....	58
4.7 Flow chart of MVO algorithm.....	60
4.8 Friedman’s Ranking for PID controller.....	64
4.9 Friedman’s Ranking for FOPID Controller.....	65
4.10 Convergence curve of the algorithms for PID controller.....	66
4.11 Convergence curve of the algorithms for FOPID controller.....	66
4.12 GWO tuned PID and FOPID controller response.....	67
4.13 WOA tuned PID and FOPID controller response.....	68
4.14 MFO tuned PID and FOPID controller response.....	69
4.15 MVO tuned PID and FOPID controller response.....	69
5.1 Flow chart of AOA.....	71
5.2. Flow chart of ASO.....	74
5.3. Flow chart of SHO.....	77
5.4. Flow chart of STO.....	79
5.5 Flow chart of proposed hybrid STOPSO algorithm.....	84
5.6. Polynomial Reference trajectory.....	85
5.7 Reference trajectory generation in SIMULINK.....	85
5.8 Friedman’s ranking of the metaheuristic algorithms on PID controller.....	89
5.9 Trajectory tracking using ASO tuned PID.....	89
5.10 Trajectory tracking using AOA tuned PID.....	90
5.11 Trajectory tracking using SHO tuned PID.....	90
5.12 Trajectory tracking using STO tuned PID.....	91
5.13 Trajectory tracking using hybrid STOPSO tuned PID.....	92
5.14 Convergence curve of all the metaheuristic algorithms.....	92

List of Tables

2.1 Development model of an expert control system.....	17
2.2 Applications of metaheuristic algorithms in robotic manipulators.....	22
2.3 Key features of intelligent control techniques.....	27
4.1 Parameters for metaheuristic algorithms.....	61
4.2. PID Controller gains, error values, and objective function values for various algorithms.....	62
4.3 Statistical parameters for PID Controller.....	62
4.4 FOPID Controller gains, error values, and objective function values for various algorithms.....	63
4.5 Statistical parameters for FOPID Controller.....	63
4.6 Ranking of the metaheuristic algorithms on PID and FOPID controller designed according to the Friedman test.....	64
5.1 Values of the Parameters considered for simulation.....	86
5.2 Controller gains and objective function values for metaheuristic algorithms...86	86
5.3 Statistical Analysis of the fitness function in 10 runs.....	87
5.4 Ranking of the metaheuristic algorithms on PID controller designed according to the Friedman's Test.....	88
5.5 Comparative study of the proposed algorithm.....	93

List of Abbreviations

A

ABC	Artificial bee colony	ACO	Ant colony optimization
AOA	Arithmetic optimization Algorithm	ASO	Atom search optimization
ASMC	Adaptive sliding mode control	AI	Artificial Intelligence
ANN	Artificial neural networks		

C

CS	Crow search algorithm	CSA	Cuckoo search algorithm
CSO	Chicken swarm optimization		

D

DoF Degree of freedom

E

ESC Extremum seeking control

F

FOPID Fractional order proportional integral derivative

FLC Fuzzy logic controller FFA Firefly algorithm

G

GA Genetic algorithm GWO Grey wolf optimization

I

ILC Iterative learning control IAE Integral absolute error

ITAE Integral time absolute error IACCO Integral absolute change in
controller output ITSE Integral time SQUARE error

L

LSA Least square algorithm

M

MFO Moth flame optimization

MVO Metaverse optimization

MIMO Multi input multi output

MLC Machine learning control

N

ND Negative definite

NSD Negative semidefinite

P

PD Proportional derivative

PID Proportional integral derivative

PSO Particle swarm optimization

PSD Positive semidefinite

R

RNN Recurrent neural networks

RBFNN Radial Basis function neural networks

S

SHO Spotted hyena optimization

STO Sooty tern optimization

SMC Sliding mode control

SVM Support vector machine

T

TDR Travelling distance rate

W

WOA Whale optimization algorithm

WEP Wormhole existence probability

List of Symbols

θ_1	Angular position of link 1	θ_2	Angular position of link 2
L	Lagrangian function	K	Kinetic energy
U	Potential energy	X	Displacement
F	Force	T	Torque
M_1	Mass of link 1	M_2	Mass of link 2
l_1	Length of link 1	l_2	Length of link 2
M	Mass matrix	C	Coriolis term
G	Gravitational term	C_1	Cosine of Angular position of link 1
C_2	Cosine of Angular position of link 2		
S_2	Sin of angular position of link 2		
S_{12}	Sin of sum of angular positions of link 1 and link2		
$\ddot{\theta}_1$	Angular accelerations of the links.	$\dot{\theta}^2$	Centripetal acceleration
$\dot{\theta}_1\dot{\theta}_2$	Coriolis acceleration.	$Y(t)$	Output of PID Controller
K_P	Proportional gain	K_I	Integral gain
K_D	Derivative gain	$G(s)$	Transfer function
f	Fitness function	t	time
w_1	Weight assigned to fitness function	u	The control low
w_2	Weight assigned to fitness function	a, b	The odd integers
S	Sliding surface	K	A constant > 0
u_{lf}	Low frequency controller term	u_{hf}	High frequency controller term
$d_i(t)$	Disturbances	$e(t)$	Error
$\eta_i(t)$			
V	Lyapunov function		
Γ	A positive diagonal matrix	$\dot{V}(r)$	Derivative of lyapunov function
J_1	Objective function of ESC	J_2	Objective function of ESC
u	Control input in ESC	C	Constant in ESC

Chapter 1

Introduction

1.1 Introduction to nonlinear systems

Nonlinear systems are systems that do not have a proportional relationship between the inputs and outputs, and thus cannot be described by linear equations. Nonlinear systems have specific properties like no superposition and homogeneity, chaotic behavior, multiple equilibrium states, input sensitivity, state dependence, and multivariable interactions that distinguish them from linear systems and impact the design and implementation of control algorithms. Nonlinear systems have unpredictable dynamics that make it difficult to achieve the desired performance.

Nonlinear control systems can exhibit local stability; the system may be stable for some initial conditions but unstable for others. The challenging nature of the nonlinear dynamical systems makes designing and implementing control algorithms for nonlinear control systems complex. Therefore, specialized techniques and algorithms are often required to achieve the desired performance. Researchers have implemented various control techniques such as optimal, adaptive, robust, sliding mode, model predictive control (MPC), and other nonlinear control techniques.

A nonlinear system is expressed in the standard form as follows.

$$\dot{x} = f(x) + g(x)u \quad y = h(x) \quad (1.1)$$

f, g, h are nonlinear functions.

Another representation of the nonlinear system is expressed in eq (1.2)

$$\dot{x} = f(x, u) \quad y = h(x) \quad (1.2)$$

where x is the state vector, u is the input vector, and f is the nonlinear function describing the relationship between the state and input vectors. y is the output vector and h is a nonlinear function that relates the state and output vectors. These nonlinear functions are highly complex and difficult to analyze thus, the development and analysis of nonlinear control systems require advanced

mathematical skills and a deep understanding of control theory. The aim of nonlinear control is to make the system stable and maintain the desired behaviour in the presence of nonlinearities. Applications of nonlinear control can be found in various fields such as robotics, aerospace, electrical power systems, and chemical process control. One such robotic system, a two-link robotic manipulator has been considered for trajectory tracking problem in this thesis.

1.2 Control of nonlinear systems

Nonlinear systems are very difficult to control and almost every physical system is nonlinear in nature. These systems are inherently uncertain, unstable, and complex. The properties like no superposition and homogeneity, Chaotic behavior, Multiple equilibrium states, and Input sensitivity makes nonlinear systems challenging to control [1]. Several methods have been employed by the researchers for the analysis and control of such type of systems. Control of these systems is always an area of interest for researchers. Researchers have been developing new methods for controlling such nonlinear systems. The advent and popularity of artificial intelligence has provided a great amount of autonomy and innovations in the control of nonlinear systems.

Robotic manipulators are an example of such a system that has highly nonlinear, uncertain, and unstable dynamics [2]. With technological advancement, robotic manipulators have gained much popularity in industrial as well as medical applications. These applications are pick and place, material handling, assembling, welding, teleoperations, haptic interfaces, prosthetic limbs, and many more. Taking into consideration the non-linear characteristics and complexity of robotic manipulators, effective and optimized control is essentially required. Various methods like PID [3], sliding mode control [3], FOPID [4-5], and robust adaptive control [6-7] have been implemented to solve the problems of robotic systems. The technological advancements in the field of artificial intelligent techniques like artificial neural networks (ANN), fuzzy logic, and metaheuristic algorithms have imparted better capabilities in such systems. Researchers have implemented these techniques on robotic systems and are termed as intelligent control schemes [8].

1.3 Robotic Manipulators

Robotic manipulators are inherently complicated, nonlinear systems. As these systems are widely used in industry, efficient control of robotic manipulators becomes crucial. A robotic manipulator's dynamics refer to how it moves and responds to external forces and torques and can be described mathematically using the principles of classical mechanics. The robotic manipulator's equations of motion typically involve the displacements, velocities, and accelerations of the links and joints, as well as the external forces and torques acting on the system. The dynamics of a robotic manipulator are affected by several factors, such as the mass and geometry of the links, the type and configuration of the joints, the friction and damping in the joints, and the control algorithms used to operate the system. One important concept is the inverse dynamics problem, which involves determining the joint torques required to produce a desired end effector motion. This problem is solved by first calculating the gravitational, centrifugal, and Coriolis forces acting on the system, and then using these forces to calculate the required joint torques. Understanding the dynamics of a robotic manipulator is essential for designing, controlling, and optimizing robotic systems in a wide range of applications.

These robotic manipulators comprise actuated end-effectors, links, and joints. The links' joints are exposed to torques, and the locations of the links are tracked. It is difficult to predict the dynamics of such systems because of their inherent nonlinearities and uncertainty. To determine the dynamic of robotic manipulator systems, the Euler-Lagrange system is employed. This Lagrangian function takes the links' potential and kinetic energy into account [2]. Lagrangian equations of motion are used to this function to derive the dynamics of the system. Robotic manipulators are used in a wide range of sectors, such as process engineering in the chemical, oil & gas, space technology, and medical sciences [5]. These manipulators perform the tasks with higher speed and accuracy. Therefore, effective control of system output variables such as position and velocity are of the utmost required. A basic mechanical structure having single degree of freedom (DoF) and one link only [9] has been used

frequently for analyzing and implementing control algorithms [4]. With the increase in robotic technology, the structure with two or more DoF is used for the aforementioned purpose, they are multi-link robotic manipulators generally have two links or three links. In general, robotic manipulators fall into the categories of single-link, two-link, and three-link manipulators. Multilink robotic manipulators are manipulators with more than one link and have more than one degree of freedom. Each link has mass and angular displacement. The dynamic equations for a robotic manipulator have been presented in the next chapter.

1.4 Motivation of the work

Control and optimization of nonlinear systems are always challenging areas for researchers. Researchers are controlling these systems using various control methods like PID, robust, adaptive, and Sliding mode control. Such methods require precise knowledge of the system's dynamics. This introduces certain complexity because of nonlinearities in the system. Conventional controllers give improved performance when intelligent techniques or intelligent methods are assimilated with them. These intelligent control techniques are capable of inducing some decision-making capability that leads to improved performance of robotic systems. Metaheuristic algorithms have gained much popularity with the growth of artificial intelligence, this fascinates researchers to implement these techniques to find the optimal solutions. The use of such metaheuristic algorithms helps to obtain the optimal parameters of the control schemes. Thus, with the employment of intelligent control methods, the system will be able to perform its task efficiently with increased capability and payload capacities.

1.5 Objectives

The key objectives of this thesis work are:

1. Mathematical modeling and stability analysis of robotic manipulator.
2. Comparison and analysis of intelligent control techniques.
3. Optimization of the robotic manipulator using metaheuristic algorithms.

1.6 Thesis Organization

The completed research work is compiled in the following subsequent chapters:

1.6.1 Chapter 2: Literature Review

Previous research work focused on every aspect of controlling robotic manipulators using intelligent control methods has been presented in this chapter. It also consists of theoretical and realistic implications of these control methods on the trajectory control of robotic manipulators. In order to compile information on the earlier research carried out on trajectory control of manipulators, a comprehensive literature survey is conducted along with its complete architecture. Metaheuristic algorithms have gained popularity among researchers to solve complex problems. Implementation of these algorithms on robotic systems has been presented in this chapter. In addition to this analysis, the key features, advantages, and disadvantages have been presented in this chapter.

1.6.2 Chapter 3: Analysis of control strategies for robotic manipulator

In this chapter, the control strategies implemented for trajectory tracking have been presented. The methods like PID, FOPID, Robust, and Adaptive control with their architecture and mathematical formulation have been discussed.

1.6.3 Chapter 4: Optimization and statistical analysis of control techniques for linearized model

In this chapter, the control techniques PID and FOPID have been designed using the metaheuristic algorithms GWO, WOA, MFO, and MVO for a linear model of a two-link robotic manipulator. Also, the mathematical formulation of these algorithms has been presented. These algorithms have been used to find the optimal gains of the controller. The effectiveness of these algorithms has been evaluated by performing a statistical analysis.

1.6.4 Chapter 5: Optimization and statistical analysis of control techniques for nonlinear model

In this chapter, the control techniques PID has been designed using the recent metaheuristic algorithms ASO, AOA, SHO, and STO for a nonlinear model of a two-link robotic manipulator. Also the mathematical formulation of these algorithms has been presented. These algorithms have been used to find the optimal gains of the controller. The effectiveness of these algorithms has been evaluated by performing a statistical analysis. A novel hybrid algorithm STOPSO has been designed and presented in this chapter.

1.6.5 Chapter 6: Conclusion and Future Work

In this chapter the outcomes of the research work are concluded and presented along with its future aspects.

Chapter 2

Literature review

2.1 Introduction

The design of robotic manipulators and their control have drastically changed as a result of robotics technological innovations. Almost every industry including Electrical, mechanical, process, and medical use manipulators to reduce labor costs and increase accuracy. Effective controlling of manipulators is challenging because of their inherent complex dynamics, uncertain behavior, and nonlinearities. Researchers are developing many ways to implement effective control strategies using classical, modern, and intelligent techniques.

To perform the tasks, robotic manipulators interact with the real environment in all intended applications. Therefore, the requirement of understanding the input–output relations arise, Thus, there is a requirement for intelligent control techniques. The control of robotic manipulators has been greatly impacted by the revolutionary rise in artificial intelligence. In this chapter, various intelligent control strategies used in robotic manipulator systems are thoroughly reviewed. The intelligent control methods include ANN, FLC, expert systems, metaheuristic algorithm, and machine learning control (MLC).

The ability to emulate human intelligence makes these intelligent control methods popular for controlling robotic systems. Conventional control methods like PID SMC, robust, adaptive optimal, and FOPID have been implemented in robotic systems to achieve the control objectives. These intelligent control methods improve control performance when integrated with conventional methods under performance constraints.

2.2 Intelligent Control Methods

Intelligent control refers to the implementation of AI techniques such as fuzzy logic, neural networks, machine learning, and optimization algorithms in control strategies to attain the desired performance of physical systems [13]. The system acquires certain characteristics like learning capacity, memory, and the ability to handle unidentified or unanticipated conditions. This control makes decisions based on approximation theory, which estimates any circumstance or representation. Intelligent control techniques have been proven effective in complex systems [8].

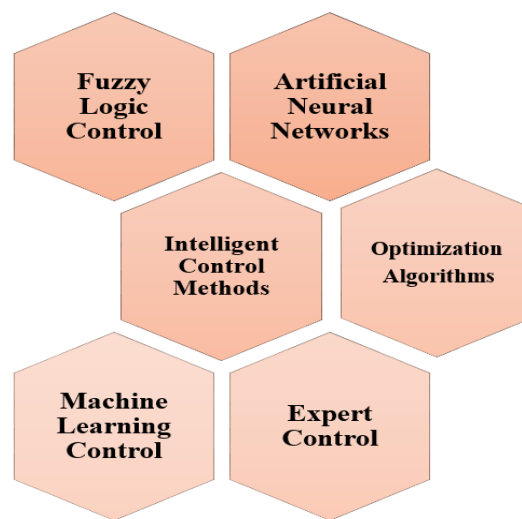


Figure 2.1 Intelligent Control Methods

Thus intelligent control is the use of AI techniques to achieve the control objectives by improving the performance of the system. These methods include FLC, expert systems, ANNs, deep learning, machine MLC, and metaheuristic algorithms for optimization, and are known as soft computing methods. The implementation of such methods in robotic systems provides the ability to achieve the intended objective effectively. Figure 2.1 presents the intelligent control methods used for the controlling the robotic systems. ANNs are inspired by the human brain neurons and inculcate the learning ability, FLC implements the fuzzy theory-based control, and the metaheuristic optimization techniques help to obtain the optimum solutions for the control law and the system's parameters.

Expert control is software-based programs that integrate the system with an inference mechanism to achieve the desired control output. Researchers have been implementing each of these techniques extensively to control the robotic manipulators for problems like trajectory tracking and path planning. Each of these technologies significantly impacts the performance of the robotic systems. All these methods have been discussed and reviewed in the subsequent subsections.

2.2.1 Artificial Neural Networks

ANN gives the biological intelligence of neurons to the systems. Input, hidden, and output layers are the three distinct layers that constitute the framework of the ANNs. Based on data relating to manipulators, these neural networks train the variable or parameters. Neural networks have various configurations, like *feed-forward*, *feedforward neural networks based on backpropagation*, and *recurrent neural networks* [14]. *Feedforward neural networks*- This network does not have any feedback mechanism or loop. *Backpropagation based feedforward neural networks*- in this type, a sigmoid function is used as an activation function. This may have multiple hidden layers. *Radial basis feedforward neural networks based on backpropagation*- It merely includes one hidden layer and a radial basis function, which is the activation function.

Recurrent Neural Networks are another popular networks- The learning of neurons is necessary for this neural network-based control mechanism; input-output mapping handles the learning and makes feedback available in a loop. Because of feedback availability, it shows the related information in a bi-directional way. This network frequently serves the purpose of controlling robotic manipulators. Neural networks are often utilised to control robotic manipulators due to their nonlinear and complicated dynamics [15].

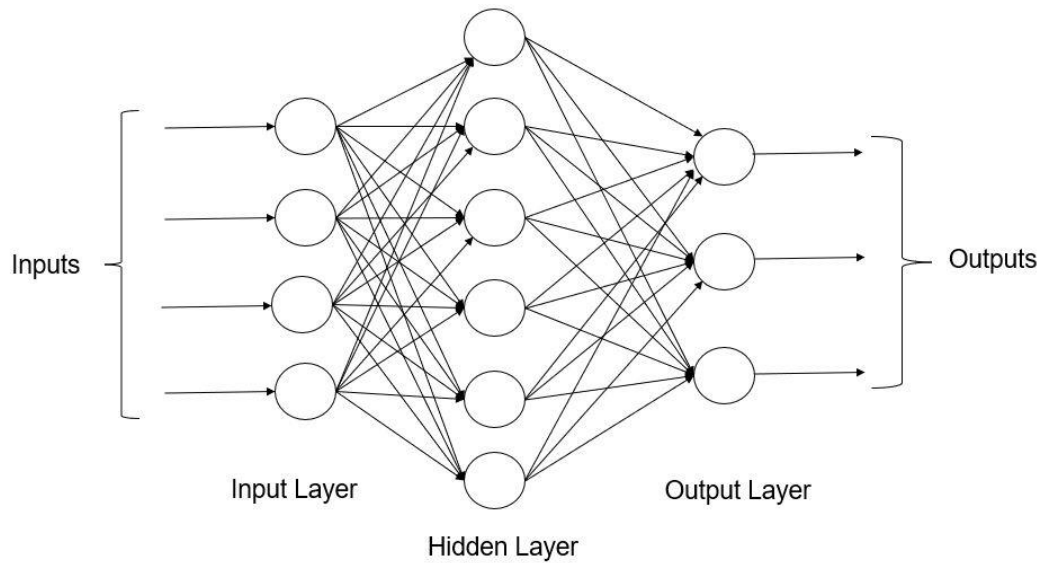


Figure 2.2 Architecture of artificial neural networks

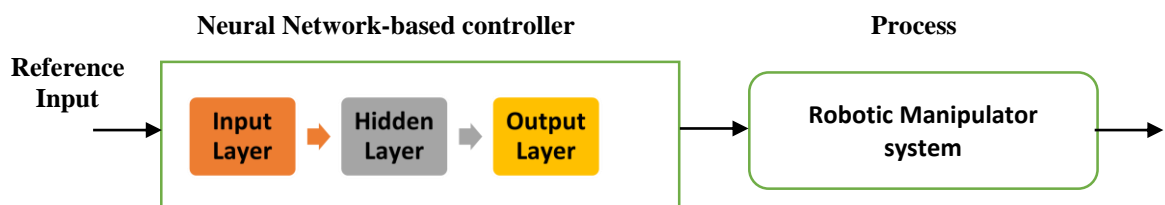


Figure 2.3 Architecture of ANN based control methods

Figure 2.2 and Figure 2.3 presents the architecture of neural networks and the control schemes based on such networks. *Bin Jin* [16] demonstrated a neural network-based backpropagation and calculated torque control. Prior knowledge was included into the control design, resulting in error convergence and quick and effective control. In [17], the authors presented chaotic neural networks based direct adaptive control of robotic systems with three axes. Since chaotic neural networks feature internal feedback loops inside their layers, they have a substantial influence on the dynamics of robotic manipulators. The proposed neural network uses a PD controller and a backpropagation method as a learning technique. Improvement in performance over recurrent neural networks is shown through simulation findings. SMC is a nonlinear control method used for the controlling a robotic manipulator system. In [18], the authors designed a

SMC for a robotic manipulator utilizing ANN for trajectory control issues. The system's uncertainties are addressed by a three-layer neural network. The suggested controller is stable, as demonstrated by Lyapunov's theorem. The proposed controller's robustness is verified by simulation, although it has a chattering effect. In [19], the authors proposed a neural network-based SMC with a chatter-free response for a robotic manipulator system. The nonlinearities and uncertainties in the system under consideration are approximated by an RBF neural network. Hence, it is a neural network-based SMC having chattered free response. The convergence of the position tracking error is demonstrated using the Lyapunov stability theorem. Robotic manipulators' nonlinear and unpredictable dynamics have been well suited to SMC. It responds more quickly and can effectively manage the innate uncertainties. When learning and estimating uncertainty in manipulators, neural networks perform effectively. The authors in [20], proposed a unique SMC for robotic manipulators employing neural networks, with the weights being set by a fuzzy supervisory controller. Combining all these techniques, it is known as fuzzy supervisory sliding mode neural network control (FSSMNNC). The Lyapunov approach validates the developed controller's steady response and error convergence.

The authors of [21] described a neural network-based adaptive controller for trajectory control of a robotic manipulator that uses a proportional derivative (PD) controller. The suggested controller performs better since the neural network simulated the system's nonlinearities and uncertainties. The performance of the controller, approximation, and tracking errors are all improved by the Lyapunov function. In [22], the authors have shown an adaptive neuro controller for a robotic manipulator system utilising a radial basis function neural network (RBFNN). The unpredictable and nonlinear dynamics of a robotic manipulator are approximated by RBFNNs because they are theoretically tractable. A Signum function-based saturation function serves as an auxiliary controller that ensures the suggested control scheme's stability and resilience in the face of system uncertainties and disruptions.

In [23], the authors have applied neural networks in the form of a nonlinear compensator to control the trajectory of robotic manipulators. The authors proposed a model learning scheme that was efficient in the effective learning of

the manipulator dynamics that provided effective control. Authors proposed a full-state feedback neural-based control for trajectory control of flexible joint manipulators which have very high uncertainties in dynamics. Lyapunov's function validates the stability and effectiveness of the proposed controller. In [24] the authors have demonstrated the shortcomings of RNN solutions for motion control, such as error accumulation and convexity restriction for robotic manipulators. To address these problems, the authors suggested two modified neural network techniques. The modification of the control law by modifying the constraints resulted in unique ways for error accumulation and convexity restriction of robotic manipulator motion control. Both rigid and flexible joint robotic manipulators can benefit from neural networks.

2.2.2 Fuzzy Logic Control (FLC)

Fuzzy control is a way for representing and implementing the knowledge of a (smart) human about how to control a system. Fuzzy logic was the invention of *Prof. L.A. Zadeh* in 1965 [25]. The application of this FLC in steam engine control by *Dr. E.H.Mamdani* [26] made this popular. The fuzzification, fuzzy inference, and defuzzification are three-step used to implement a FLC [27]. The fuzzification is known as converting the input values into fuzzy sets with the help of fuzzy rule bases. This is the initial step in the design of the FLC. Fuzzification considers the uncertainties of the systems and evaluates the conditions for the controller design. Fuzzy Inference identifies which control rules should be applied to the fuzzified inputs and outputs the controller in the form of fuzzy sets. Defuzzification is the process of getting the actual output from the fuzzy output of the controller, and it therefore interacts with the actual system or mathematical model under consideration. A FLC based controller architecture is shown in Figure 2.4 below. FLC offers consistently improved results for systems operating in a nonlinear, complicated, and unpredictable environment.

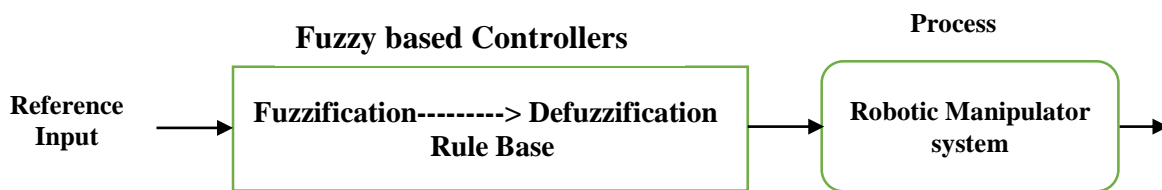


Figure 2.4 Fuzzy Logic-based control approach

There are two popular approaches to design the FLC for a nonlinear system

- i. Mamdani Approach
- ii. Takagi & Sugeno's Approach

Mamdani's technique employs linguistic fuzzy modelling, which has a high interpretability but a poor accuracy, whereas Takagi and Sugeno's method employs exact fuzzy modelling, which has a high accuracy but a low interpretability. Due of the inherent complexity, uncertainties, and nonlinearity in robotic manipulators, massive computations are required, designing the control law complex task. FLC describes the dynamic modelling and computational time constraints efficiently. It is an important part of intelligent control for controlling the robotic manipulators in an intended manner. FLC is implemented at a hierarchical level. *C.M Lim and T.Himaya* [29] presented the application of FLC in robotic manipulators. The authors illustrated how to use fuzzy logic to simplify control rules and improve a system's performance. The authors designed a PI controller using fuzzy logic, where PI control provides transients and steady-state features while fuzzy logic enhances the damping properties. By measuring the parameters and giving them a membership function value thus obtaining fuzzy sets, the control law is determined. Every rule has a value for the outcome's membership function. A linear combination of results for each rule with membership function weights yields the final control action. Simulation experiments have been used to verify this method's efficacy. In [30], the authors implemented a ordered control on a robotic manipulator using FLC. The authors designed a fuzzy PD controller for joint torques and positions whereas kinematic aspects are in supervisory mode. Thus, it has a hierarchy in control. A comparative study of fuzzy PD and conventional

PD controllers is presented, and fuzzy PD gives better performance. In [31], the authors have designed a CS algorithm tuned optimal fuzzy PID controller. Another hybrid fuzzy PID based control schemes for robotic systems have been illustrated in [32]. Multivariable and nonlinear dynamics of robotic manipulators make it difficult to design the control law. By taking this into account, the authors of [33] suggested a model-free hybrid fuzzy logic and neural network-based control method that takes into account the multi-input multi-output (MIMO) characteristics of robotic manipulators. Joint locations are controlled by a fuzzy logic controller, and the coupling between joints is controlled using backpropagation neural networks. Consequently, neural networks enhance the performance of controls. The experimental results demonstrate that this technique improves trajectory tracking. For a multi-link robotic manipulator, the authors in [34] developed a robust fuzzy model control algorithm that considers torque disturbances and measurement noise, a self-tuning adaptive fuzzy technique with a mechanism for adjusting the parameters. This method is unique because it employs a rule-base of the IF - THEN control input type. The suggested method offers increased stability and transients. The Lyapunov stability approach confirms that the tracking error is confined. The authors in [35], developed the fuzzy support vector machine (SVM) control methods for robotic manipulators in combining the genetic algorithm (GA) and the least square algorithm (LSA). SVM identified the system's FLC and nonlinearities. The controller's parameters have been tuned using the techniques for optimization GA and LSA. While GA performed the live optimization of these parameters, LAS performed the offline optimization of the SVM parameters. Studies in simulations have verified the efficacy of the suggested control technique. For a nonlinear MIMO robotic system, a PD-type fuzzy iterative learning control (ILC) [36] was developed. The gains of the PD type ILC controller design was optimized using fuzzy logic. In [37] authors presented an adaptive fuzzy state feedback design for a flexible robotic manipulator system. Fuzzy logic has approximated the uncertainties and actuator saturation. The combination of backstepping and command filtering developed a novel adaptive fuzzy tracking backstepping control.

FOPID is a reliable and proficient approach for controlling nonlinear and unpredictable systems such as robotic manipulators. Researchers are implementing this method for controlling nonlinear systems. The authors introduces a fractional order fuzzy (FOFPID) [38] controller for manipulator's trajectory control. The suggested controller is strong enough to handle the system's nonlinearities and uncertainties. Using fuzzy inference rule bases, input variables like error, its derivative, and its integration are fuzzified. The authors compared the performance of conventional PID, fuzzy PID, FOPID, and FOFPID. Integral absolute error (IAE) and integral of absolute change in controller output serve as performance indicators for the controllers (IACCO). The Cuckoo search (CSA) method has been used to fine-tune the controller's settings. As an outcome, among all the controllers compared, the FOFPID controller performed the best. In [39] the authors, have proposed An NLA-FOPID controller for a two-link robotic manipulator. The proposed controller was evaluated for trajectory tracking, disturbance rejection, sensor noise rejection, and a number of other system uncertainties. The parameters of the controller have been optimised using the backtracking search algorithm (BSA). In comparison to the nonlinear adaptive fuzzy PID controller, the suggested controller provided better and enhanced performance. In [40] the authors, presented a *fuzzy fractional order PID (FFOPID)* for a robotic manipulator aimed for tracking the trajectory. The controller manifolds PID error. The controller is calculated by a fuzzy inference system using the PID elongated error and fractional order integral. *Zafer Bingul et-al.* [41] developed the Mamdani type FLC for trajectory tracking of a robotic manipulator. The three separate cost functions used for PSO are mean of the absolute magnitude of error (MAE), mean of the squared root of error (MRSE), and reference-based error with control effort (RBECE). The suggested controller has been demonstrated as more reliable and efficient when compared to traditional PSO tuned PID. In the majority of applications, robotic manipulators are employed to repeatedly carry out different tasks. *In* [42], the authors presented a repeating learning control using adaptive fuzzy logic. After implementing an adaptive fuzzy approach, fuzzy logic is used to model the initial step's uncertainty. The proposed method is novel since it uses dynamic

rule bases and hence features self-tuning membership functions. The closed loop's boundedness is ensured via the Lyapunov approach. In [43] the authors, proposed the interval type-2 fuzzy PID integral control approach, using GA to tune the controller's parameters. The fitness function was the integral of the time square error (ITSE). The suggested technique was determined to be more reliable and effective in a comparison analysis of the proposed controller with type-1 fuzzy PID and conventional PID. Fuzzy logic has a positive impact on SMC's ability to handle the system's uncertainty. In [44], for trajectory control of robotic manipulators, the authors introduced a MIMO adaptive fuzzy terminal SMC. The suggested approach incorporates an adaptive system for terminal SMC together with the benefits of fuzzy logic. It performed better when compared to alternative ways. The convergence of the error was given by the bounded Lyapunov stability criteria. Based on hybrid PSO-GA optimization approaches, the authors of [45], developed a type-1 and type-2 FLC for robotic manipulator trajectory control. These techniques have been employed to calculate the membership function values of the proposed fuzzy controllers. Fuzzy logic is a prominent pick for robotic control systems. The literature clearly demonstrate that researchers are employing these meta-heuristics algorithms to adjust the parameters of control techniques to provide intelligence in robotic manipulator's control.

2.2.3 Expert Control Systems (ECS)

ECS are intelligent control technique that simulates the expertise of a human expert while providing effective control. They integrate control design into the system and create an inference mechanism [46]. These control methods do not require precise information and thus can handle information uncertainty effectively. Expert systems are intelligent algorithms that use knowledge and inference procedures to solve complicated problems [47]. To perform the intended tasks the systems get decision making abilities utilizing the inference mechanism based on certain rules. The rule base creates reasoning and determines the logic so that the inference mechanism can infer the conclusions from knowledge base.

An ECS expert control systems are computer-based programs that replicate human intelligence by using an inference mechanism and thus can deal the complex real-time problems effectively. In [48], the authors have presented the design, principles, classification, and some implementation issues in the expert control methods. Several challenges need to be addressed when implementing these approaches including real-time operational knowledge bases, competent online information environments, real-time intelligent controller design, parallel reasoning, and an intelligent interface between control schemes and the user. Therefore, it is relatively perplexing to implement such techniques in real time with the real rule base.

The technological advancements in AI techniques like FLC and ANN have created more opportunities to design expert control methods. The architecture of an expert control system utilizes a model of the system, an inference engine to perform a real-time control mechanism, a knowledge base, a learning mechanism, and an interface to the user. In [48], the authors have proposed the guidelines and the development method of expert control systems in six different phases. Table 2.1 illustrate the six phases of the development of an expert control system.

Table 2.1 Development model of an ECS

S. No.	Development Stages	Description of each stage
1	Feasibility analysis	Analysis of all the issues such as cost, and development time is carried out.
2	System specification	In this stage, the problem is defined, and a task assessment has been performed.
3	System design	This is the actual design stage that includes the concept and logic of control, and structure design.

4	System Construction	The system is being built with an inference mechanism and a real-time algorithm.
5	Software design	Using AI-based algorithms the software has been designed.
6	Evaluation stage	This stage evaluates the efficiency and usability of the designed control.

In the very first step, a feasibility study is performed and all issues such as cost, resources, and development time need to be evaluated. The second stage focuses on system specification and defines the problem and assesses all the tasks to be completed. The third steps determine the structure of control and create the concepts and logic. This stage is the system design and is very important in the development of expert control. The next stage is the actual construction stage where the inference mechanism and real-time control law is determined. The software model utilizing AI-based algorithms has been designed in the fifth stage. Finally, the last stage is about evaluation that checks the control designed for efficiency and usability. In this stage, expert control is evaluated in all aspects.

The expert control has been designed for robotic applications by researchers. *Z. Geng and M. Jamshidi* [49] designed a real-time ECS for trajectory tracking of a robotic manipulator having two links. An expert self-learning controller for controlling the trajectory problem of the manipulator having mathematical logic and procedures has been designed. This eliminated the constraints like the requirement of suitable system dynamics and disturbances have been eliminated by imparting decision-making in the proposed self-learning controllers. Hence the precise model is not required, and a data-rule base was able to meet the system requirement. *Astrom et-al.* [50] illustrated the heuristics and logic in expert control law that makes the conventional method simpler and easier. The authors proposed the rule-based expert system architecture having the components like the rule base creating the reasoning, database, and inference engine- imparts the decision making, and user interface.

In [51], the authors have suggested an expert intelligent control algorithm for a real-time path formation of a manipulator. The proposed algorithm was able to generate three different paths having a decision-making capability to avoid and obstacle and discover the shortest path. It is evident from the literature that ECS is preferred for complex systems. The requirement of a perfect knowledge base makes these expert control methods complex. The popularity of FLC, ANN, and other AI techniques has gained more popularity in the control of robotic systems.

2.2.4 Machine Learning Control (MLC)

MLC is a model-free control technique that has gained popularity recently and effectively manages complex nonlinear dynamics [52]. It is a data-driven control methodology that suits the model-free dynamics. The main focus of MLC is on the control law, and the controller adapts to any known or unknown dynamics. MLC is a blend of machine learning techniques, complicated dynamic nonlinear systems, and feedback control [53]. It is a technique for controlling and analysing the nonlinear systems and do not require knowledge of the system's precise model.

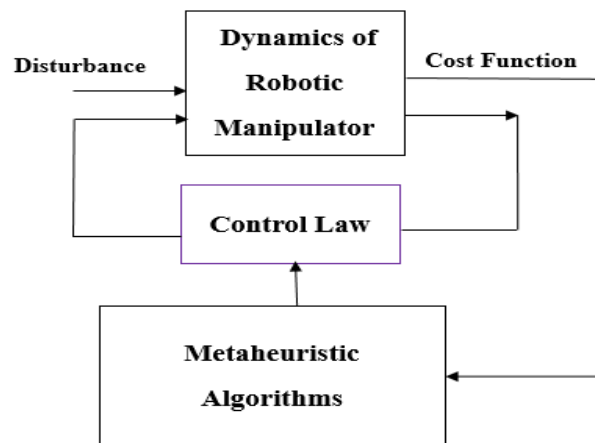


Figure 2.5 Architecture of MLC

The metaheuristic optimization algorithms such as GA, PSO, and ABC determine the control law using specific objective functions. Figure 2.5 presents the architecture of MLC. Thus tuning the parameters imbibes the intelligence in

this control scheme MLC combines the advantages of machine learning with artificial intelligence. MLC is an evolving development in the control of complex systems. The data availability provides learning and improves the performance of the systems. *N. Gautier et- al.* in [54] illustrated this model-free in flow control based utilizing the genetic algorithm. Considering the popularity of this control approach, there is an utmost requirement of for the mathematical formulation of MLC. *Akshat Diveev et-al.* [55] formulated the mathematical concepts for this control using supervised and unsupervised learning theories. The authors implemented neural networks to calculate the parameters for an unknown function that maps the input and output. *Shouyi Wang et al.* [58] reviewed the supervised, unsupervised, and reinforcement learning mechanisms in machine learning and analyzed their applications in bipedal robotic control. In [53] the authors proposed an MLC in real-time for a robotic manipulator. This control has been designed using a hybrid artificial bee colony (ABC) and a fuzzy theory-based learning algorithm. A fractional order PID MLC has been proposed where the hybrid ABC fuzzy algorithm has been implemented to tune the controller aimed for motion control a six degree of freedom articulated robotic manipulator.

Fuzzy rules, membership functions, and fractional controller gains have been computed using the ABC algorithm. The proposed FOPID MLC was tested against the PID and fuzzy PID, and found to be effective for motion control of the manipulator. This sets the stage for future research in the machine-learning control of robotic manipulators. Without having full knowledge of the system, MLC provides the control mechanism for a system. This capability paves the way for more research into robotic manipulator control using MLC.

2.2.5 Optimization Algorithms

Optimization is the process of achieving the best feasible solutions for the control design parameters by either minimizing or maximizing the computed objective function within the restrictions identified. These optimization techniques include evolutionary algorithms like a genetic algorithm (GA) and other nature-inspired algorithms based on the intelligence of a swarm, known

as swarm intelligence like particle swarm optimization (PSO), ant colony optimization (ACO), artificial bee colony(ABC), crow search optimization (CSO), and whale optimization algorithm (WOA) [59] [60].

These optimization techniques are implemented to find the optimal solutions for the controller parameters. Each of the control schemes has specific parameters which determine the functionality of the uncertain and nonlinear dynamics of the robotic manipulator in a desired manner.

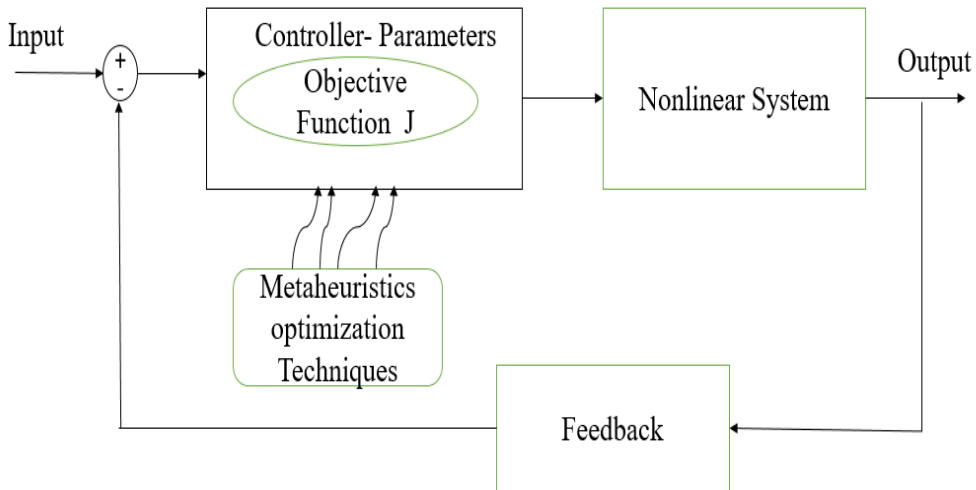


Figure 2.6. Implementation of a metaheuristic optimization technique in control

Figure 2.6 illustrates the implementation of optimization algorithms on control techniques. Thus, considering the performance of robotic systems, these metaheuristics optimization algorithms yield the optimum value of an intended objective function [61]. For tracking the trajectory these algorithms are implemented using an objective function which is formulated on the basis of the control law design. Four different types of error functions like *IAE*, *ITAE*, *ISE*, and *ITSE* have been evaluated as objective functions [62].

The feedback control's main aim is to minimize the error $e(t)$. The generalized form of the objective function is expressed below in eq. (2.1).

$$J = \int_0^{\infty} t^a [e(\theta), t]^b dt \quad (2.1)$$

$$e(\theta) = (\theta_r - \theta_v) \quad (2.2)$$

J is the cost or objective function and $(e(\theta), t)$ is the error. The difference between the desired or reference trajectory and the actually obtained trajectory is termed as an error. For different values of a and b different types of error functions are obtained.

$a=0$ and $b=1$; *Integral Absolute Error IAE*,

$a=1$ and $b=1$; *Integral Time Absolute Error ITAE*,

$a=0$ and $b=2$; *Integral Square Error ISE*,

$a=1$ and $b=2$; *Integral Square Time Weighted Error ISTE*

$a=2$ and $b=2$; *Integral Square Time Squared Weighted Error ISTSE*

In robotic applications, these metaheuristic algorithms minimize the error function by providing the optimal values of controller parameters. One of the key objectives of implementing these metaheuristic optimization algorithms in the control of robotic manipulators is to find the optimal solution for problems like trajectory control [63]. Table 2.2 shows the applications of metaheuristic algorithms in robotic manipulators. All of these algorithms pave the way for intelligent control of robotic manipulators. The controller's parameters are fine-tuned using these and better performance of the system has been achieved using these algorithms.

Table 2.2 Applications of metaheuristic algorithms in robotic manipulators

Year	Algorithm	Inspiration	Application in Robotic Control	References
1975	GA	Evolution	Adaptive FLC, Computed torque control. Optimal PID and SMC. Adaptive control, Multi-objective PID, PID controller.	[63],[66],[67], [68],[69],[78] [79]
1995	PSO	Bird flock	FOPID for a robotic wrist having two links, fuzzy PID for a two-link robotic manipulator. Multi-objective FOPID	[62], [63], [67] [77], [80], [81] [82]

2006	ACO	Ant colony	PID controller. Optimal path planning	[63], [71], [83] [84], [85]
2006	ABC	Honeybee	Hybrid ABC-GA-based fuzzy FOPID, Type 2 fuzzy PID. MLC. Optimal PID for a two-link robotic manipulator.	[53],[86] [87], [95]
2009	FFA	Fireflies' social behavior	Path planning for robotic system	[88]
2013	GWO	hunting mechanism of grey wolves	PID controller. Hybrid GWO-ABC tuned FOPID. Hybrid GWO- WOA tunes FLC.	[89],[90],[91]
2016	WOA	Social behavior of humpback whales	PID for a robotic manipulator. Sliding mode controller (SMC)	[92],[93]
2017	BAS	foraging behavior of the beetle	RNN for tracking control & impediment prevention of a robotic manipulator	[72]
2017	CSO	The behavior of the chicken swarm	Trajectory planning of manipulator	[70]
2018	CSA	Behavior of Cuckoos	FOPID for a robotic manipulator.	[76]

For trajectory tracking utilising multi-objective PSO (MOPSO), a FOPID controller is described in [62]. The suggested controller demonstrated greater performance in tracking trajectories and managing system uncertainties. The

authors in [63] illustrated trajectory control utilizing several metaheuristic optimization techniques. The location and joint objectives are part of the optimization problem's objective function. Yadav et al. [64] used the traditional Ziegler Nichols approach with optimization algorithms like GA, PSO, and ACO to create a hybrid controller for motion control of a robotic manipulator. All other methods, including the traditional Ziegler-Nichols approach, are outperformed by the GA. The majority of engineering applications use multi-objective optimization, which frequently identifies solutions that meet many objectives. Contrary to conventional methods, tuning the controller using multi-objective optimization techniques is challenging since it must exhibit high precision, energy savings, and the required response.

In [65], the authors presented a multi-objective evolutionary optimization method for the trajectory control of a two-link robotic manipulator using a non-dominated sorting GA (NSGA-II). This approach iterates to deliver the best gains for multivariable PID controllers for the best fitness functions under specific initial conditions. An effective solution selection method based on classes of dominance is implemented by the NSGA-II algorithm. The way the selection operator operates is different from GA. The suggested optimization method, which is based on NSGA-II, offers a useful way to create straightforward solutions with high reliability in closed loops. M Vijay et al. [66] employed GA to create PD and PID controllers with high dynamic characteristics, global stability, better disturbance rejection, and minimal tracking error. This is a basic direct search technique that doesn't rely on numerical or analytical gradients. As an objective function, three distinct error functions- IATE, ISE, and ISTE have been evaluated. ISTE gives the lowest value of the objective function in comparison to other methods, while a GA-tuned PID controller produces the lowest tracking error and best disturbance rejection. In [67], the authors have proposed a novel nonlinear fractional PID controller for a two-link robotic manipulator. NSGA-II is implemented to optimize the related parameters and controller gains for minimum error value and control variations. The designed NLF-PID controls the two-link robotic manipulator effectively. In [69], authors designed two PID controllers for tracking control of both links of a robotic manipulator using the GA. The

performance indices used in this case are the *integral square error (ISE)* and the *integral square of change in controller output (ISCCO)* for both links. It was statistically determined that there is always an optimum solution within a certain range by executing this optimization 2000 times.

The application of nature-inspired algorithms in optimum trajectory planning of robotic manipulators is very well evident in the literature. In [70] the authors have implemented CSO to track the trajectory of manipulators. The trajectory is generated using a B-spline. To determine the optimal trajectory, the minimum travelling time is considered as a fitness function. The experimental results support the proposed algorithm. In [71], authors designed ACO tuned PID controller to achieve optimal trajectory tracking of robotic manipulators. The authors tuned the controller gains using ACO for both scenarios, i.e. with and without external disturbances. The proposed method has been proven effective and Simulation studies shown the fast convergence of error. For tracking control and obstacle avoidance of a robotic manipulator, *Ameer Hamza Khat et al.* [72] introduced a metaheuristic control method beetle antenna olfactory RNN. This strategy combines the tracking control and obstacle avoidance problems by adding a penalty term to the cost function. The beetle's olfactory capabilities for locating food are the inspiration for this optimizer. The efficiency of the suggested control framework is confirmed by the results of the simulation studies. In [73], the authors proposed an intelligent algorithm Weighted Jacobian Rapidly exploring random tree (WJRRT) for path planning of robotic manipulators. Each node of the tree in the WJRRT algorithm has a fitness rating that aids in deciding the ultimate route. Several robotic manipulator models are simulated, and the results are compared using the Jacobian Transpose (JTRRT), Bidirectional RRT (BiRRT), and Tangent Bundle RRT (TBRRT) algorithms. Making the hybrid algorithm provides the opportunity to the use of the advantages of these metaheuristic optimizations. In [74], the authors introduced a novel hybrid artificial bee colony-genetic algorithm (ABC-GA) optimization technique-based fractional-order fuzzy pre-compensated fractional order PID (FOFP-FOPID) controller. In trajectory tracking, the ITAE function has been considered as the objective function. The intelligent control of robotic manipulators is an area that researchers are very

interested in; hence these metaheuristic algorithms are quite important. A comparison of several optimization techniques for controlling robotic manipulators was presented by *Richa Sharma et al.* [75]. The authors used GA, PSO, and simulated annealing (SA) to adjust the traditional PID controller. The simulation results demonstrate that PSO performs better than GA and SA. As a result, optimization approaches have been extensively used and are currently a prevalent method for developing control mechanisms.

2.3 Comparison of intelligent control methods

The intelligent control methods have the ability to alter the performance of the robotic manipulator systems robustly. Such methods introduce intelligence in the manipulators by imbuing decision-making. ANN-based control methods impart the learning mechanism using the data as training and test sets. ANN control method is proven efficient but requires large real-time training and test data sets. FLC has a sophisticated rule base construction yet manages the system uncertainties well. Expert systems are software program-based control methods that requiring an extensive knowledge data base which makes them complex. MLC is a model-free control approach that does not require exact mathematical modelling. The metaheuristic optimization algorithms are used in design of MLC controllers. Optimization techniques find optimal values of the controller parameters; based on these parameters, a cost function is optimized. This cost function measures the system's performance. Thus, these optimization techniques optimize the system performance for a particular problem like the trajectory control of robotic manipulators. The previous section reviews the use of several of these optimization strategies. The no free lunch (NFL) theorem states that no optimization strategy can provide the optimal outcomes for all optimization problems. As a result, researchers are continuously developing novel optimization techniques like the red fox optimization algorithm (RFO), Tunicate Swarm algorithm (TSO), Marine Predators Algorithm (MPA), Chimp optimization algorithm (COA), Water strider algorithm, and Student psychology-based optimization method (SPBO), crow search algorithm (CSO), and Owl search algorithm (OSA) [96], [97] to solve distinct complex engineering problems. Future research will be facilitated using these novel

methodologies for the intelligent control of robotic manipulators. Each of these methods has certain benefits and drawbacks. Table 2. 3 presents the important characteristics including specific advantages and disadvantages of all intelligent control methods.

Table 2.3 Key features of intelligent control techniques

S. No.	Intelligent Control Methods	Key features	Advantages	Disadvantages
1	ECS	Has an inference mechanism requiring a knowledge base. Employed in process industries.	Making decisions with a knowledge base. Rapid and consistent.	Requirement of a perfect knowledge base.
2	FLC	Fuzzification, defuzzification, and, rule based.	Simple, easy and effective scheme.	The construction of rule base is complex.
3	ANN based Control	Training of neural networks is essential in this data driven method.	Effective with sufficient data sets. Inculcates intelligence and decision making.	Huge training and test data sets are needed.
4	Optimization algorithms	Determine the best (optimal) solution. Proficient when implemented control techniques. Stochastic nature.	Simple approach, provides the optimal solution for an optimization problem.	Execution of these techniques is challenging and time-consuming.

5	MLC	No need for exact knowledge of system. Metaheuristic algorithms are implemented to determine the controller.	It is a model-free technique. The system does not need to be precisely modelled.	The controller design is complex for varying conditions
---	-----	--	--	---

AI has gained in popularity as a result of technological developments. AI enabled intelligent control graces humankind's intelligence to robotic systems. Over the past few decades, robotics has greatly increased in popularity and affected all industrial application fields. The use of the above-presented AI techniques in robotic control applications has tremendously increased over the last few decades. Intelligent control is the application of AI and optimization techniques in robotic systems to achieve the desired goals [94]. The use of intelligent control in robotic system applications has very prominent future.

Chapter 3

Analysis of control strategies for robotic manipulators

3.1 Introduction

Robotic manipulators are highly nonlinear and extremely uncertain systems. These systems have applications in different areas including medical and pharmaceuticals. Robotic systems provide assistance and help to humans by performing some tasks. Technological advances and research have improved robotics and robotic systems to a larger extent. Such systems have to interact with real-time situations so precise control is required. Researchers are continuously striving to explore new different methods to enhance the performance of robotic systems.

Various conventional methods like SMC, adaptive control, robust control, optimal control, PID, and FOPID have been implanted on robotic manipulators for controlling the manipulators for problems like trajectory tracking and path planning. Each of these control laws impacts the performance of robotic manipulators. Further, the use of artificial intelligent techniques in combination with such conventional methods has imparted decision-making in the robotic system thus named intelligent control. In this chapter, the dynamics of two-link robotic manipulators, modeling in MTALAB/Simulink, robust adaptive control, SMC, PID, FOPID, and extremum seeking control have been presented.

3.2 Dynamics of a two-link Robotic Manipulator

A two-link robotic manipulator has been used for the validation of control strategies [2]. Figure 3.1 presents a robotic manipulator having two links with masses m_1, m_2 length l_1, l_2 , and angular positions θ_1, θ_2 . The unit of mass is kilograms, length is meters, and angular position is in degrees. The trajectory control problem has been studied utilizing this model. This robotic manipulator is modelled using the Lagrangian function which considers the energies of the system.

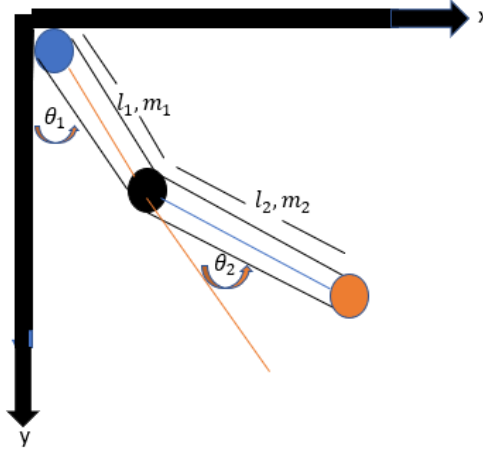


Figure 3.1 A two-link robotic manipulator

The dynamics of a robotic manipulator is obtained using the Euler-Lagrange technique. The Lagrangian function L considers the kinetic and potential energy of the links.

$$L = K - U \quad (3.1)$$

Where K is kinetic energy and U is potential energy of the system.

Total kinetic energy and potential energy of the system is as follows:

$$K = \frac{1}{2} m_2 l_1^2 \dot{\theta}_1^2 + \frac{1}{2} m_2 l_2^2 (\dot{\theta}_1^2 + \dot{\theta}_2^2 + 2\dot{\theta}_1 \dot{\theta}_2) + m_2 l_1 l_2 \cos \theta_2 (\dot{\theta}_1^2 + \dot{\theta}_1 \dot{\theta}_2) \quad (3.2)$$

$$U = -(m_1 + m_2) g l_1 \cos \theta_1 - m_2 g l_2 \cos(\theta_1 + \theta_2) \quad (3.3)$$

The Lagrangian function of the system can be represented as follows:

$$L = \frac{1}{2} m_2 l_1^2 \dot{\theta}_1^2 + \frac{1}{2} m_2 l_2^2 (\dot{\theta}_1^2 + \dot{\theta}_2^2 + 2\dot{\theta}_1 \dot{\theta}_2) + m_2 l_1 l_2 \cos \theta_2 (\dot{\theta}_1^2 + \dot{\theta}_1 \dot{\theta}_2) + (m_1 + m_2) g l_1 \cos \theta_1 + m_2 g l_2 \cos(\theta_1 + \theta_2) \quad (3.4)$$

Considering the Lagrangian function, the following equations of motions are used to obtain the dynamics of the system,

$$T_1 = \frac{d}{dt} \left(\frac{dL}{d\dot{\theta}_1} \right) - \frac{dL}{d\theta_1} \quad (3.5)$$

$$T_2 = \frac{d}{dt} \left(\frac{dL}{d\dot{\theta}_2} \right) - \frac{dL}{d\theta_2} \quad (3.6)$$

After placing the Lagrangian function from eq. (3.5 - 3.6), following dynamical expressions are obtained

$$T_1 = [(m_1 + m_2)l_1^2 + m_2l_2^2 + 2m_2l_1l_2 \cos \theta_2]\dot{\theta}_1 + [m_2l_2^2 + m_2l_1l_2 \cos \theta_2]\dot{\theta}_2 - 2m_2l_1l_2 \sin \theta_2 \dot{\theta}_1\dot{\theta}_2 - m_2l_1l_2 \sin \theta_2 \dot{\theta}_2^2 + (m_1 + m_2)gl_1 \sin \theta_1 + m_2gl_2 \sin(\theta_1 + \theta_2) \quad (3.7)$$

$$T_2 = (m_2l_2^2 + m_2l_1l_2 \cos \theta_2)\ddot{\theta}_1 + m_2l_2^2\ddot{\theta}_2 + m_2l_1l_2 \sin \theta_2 \dot{\theta}_1^2 + m_2gl_2 \sin(\theta_1 + \theta_2) \quad (3.8)$$

Utilizing the above equations, the generalized equation has been written as follows:

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) = \tau \quad (3.9)$$

These equations can be expressed in the following general matrix form.

$$\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} (m_1 + m_2)l_1^2 + m_2l_2^2 + 2m_2l_1l_2C_2 & m_2l_2^2 + m_2l_1l_2C_2 \\ (m_2l_2^2 + m_2l_1l_2C_2) & m_2l_2^2 \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} 0 & -m_2l_1l_2S_2 \\ m_2l_1l_2S_2 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{bmatrix} + \begin{bmatrix} -m_2l_1l_2S_2 & -m_2l_1l_2S_2 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1\dot{\theta}_2 \\ \dot{\theta}_2\dot{\theta}_1 \end{bmatrix} + \begin{bmatrix} (m_1 + m_2)gl_1S_1 + m_2gl_2S_{12} \\ m_2gl_2S_{12} \end{bmatrix} \quad (3.10)$$

where,

$$C_1 = \cos(\theta_1) \quad C_2 = \cos(\theta_2)$$

$$S_2 = \sin(\theta_2) \quad \text{and} \quad S_{12} = \sin(\theta_1 + \theta_2)$$

$\ddot{\theta}_1, \ddot{\theta}_2$ = angular accelerations of the links. $\dot{\theta}_1^2, \dot{\theta}_2^2$ = centripetal acceleration

$\dot{\theta}_1\dot{\theta}_2$ = Coriolis acceleration. Coriolis acceleration is prevalent because the first link serves as a rotational frame for link two. The nonlinear manipulator dynamics as shown in eq. (3.9-3.10) has been linearized for trajectory tracking using the following steps.

1. To linearize, the system states has been assumed as follows

$$X_1 = \theta_1; X_2 = \theta_2; X_3 = \dot{\theta}_1; X_4 = \dot{\theta}_2 \quad (3.11)$$

$$\dot{X}_1 = \dot{\theta}_1 = X_3; \dot{X}_2 = \dot{\theta}_2 = X_4; \dot{X}_3 = \ddot{\theta}_1; \dot{X}_4 = \ddot{\theta}_2 \quad (3.12)$$

2. The eq. (3.10) has been represented using different constants then Taylors series expansion about its equilibrium points $\frac{\pi}{2}$ and 0 has been performed.
3. By differentiating, each variable is linearized with respect to all other variables. The following state space equation model has been obtained on linearizing about the equilibrium points:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -0.4568 & -0.6196 & 0 & 0 \\ 0.2485 & -6.6174 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0.7870 & -0.0426 \\ 0.0426 & 0.1349 \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \quad (3.13)$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \quad (3.14)$$

For a stable system all the Eigen values must be negative, so on calculating the Eigen values of the linear model obtained in eq. (3.13-3.14), Eigen values are found to be complex conjugate thus making the system unstable.

$$\text{eig}(A) = \begin{bmatrix} 0.0000 + 0.6942i \\ 0.0000 - 0.6942i \\ 0.0000 + 2.5675i \\ 0.0000 - 2.5675i \end{bmatrix} \quad (3.15)$$

Both the linear and nonlinear models considered for trajectory tracking problem in the next chapters, are inherently unstable hence there is requirement of the design of effective control method. The next sections describe the control methods designed. Further metaheuristic algorithms have been implemented to design the optimal control laws. The two angular positions of both links are output trajectory that should track the reference trajectory. MATLAB/SIMULINK software has been utilized to perform mathematical modeling using the above equations. Figure 3.2 shows the flow diagram of mathematical modeling and Figure 3.3 shows the Simulink diagram of the mathematical model. A polynomial trajectory is given as a reference to two PID controllers. S_rigid is the MATLAB script having dynamical equations of the system. Two separate PID modules have been designed which are connected to

system model script. The separate error values and the combined outputs have been shown in the display.

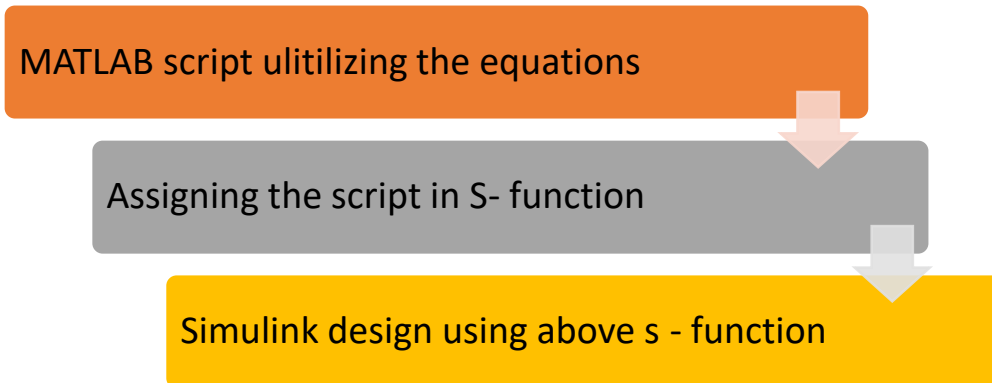


Figure 3.2 Flow diagram of mathematical modeling

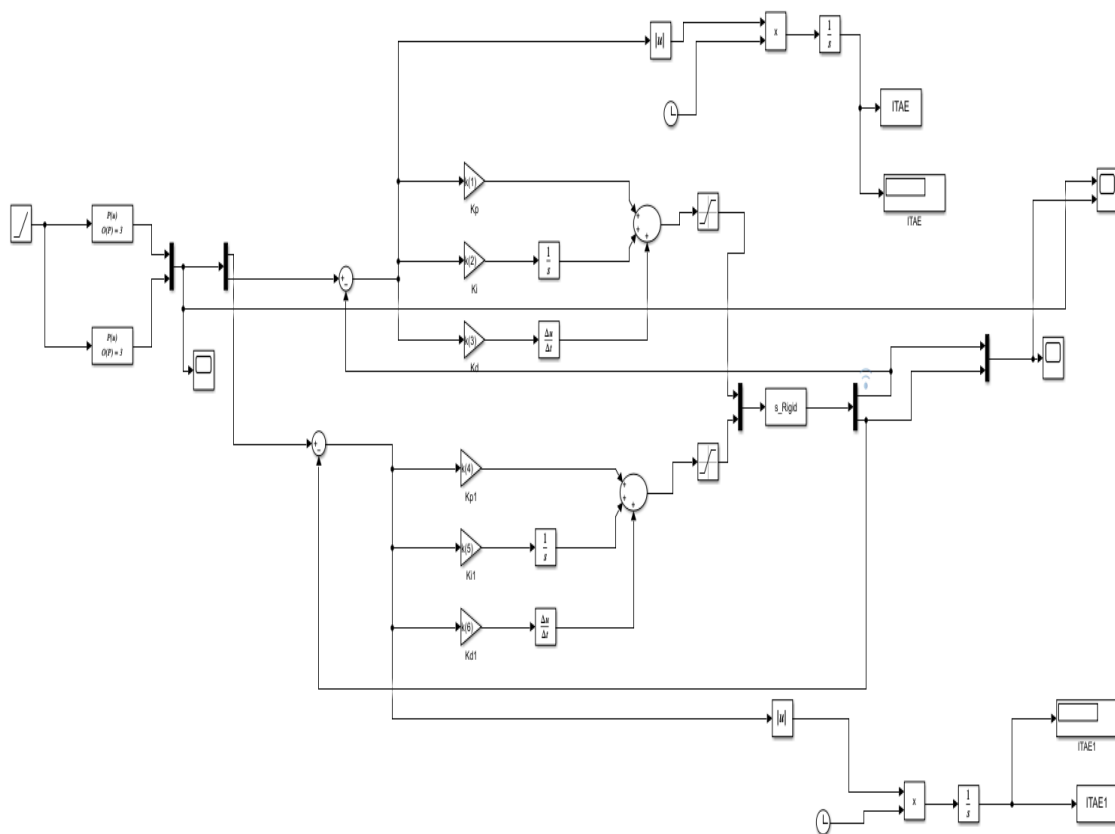


Figure 3.3 Simulink diagram of the mathematical model

3.3 PID and FOPID Control

PID controller is the most widely used industrial controller as more than 95 % of the industrial controllers are PID in nature. It has a high degree of robustness in controlling robotic manipulators due to their stability, adaptability, simplicity, and real-time control capabilities. PID controller has three gains namely, proportional, integral, and derivative. The design requires implementation of three parameters K_P proportional gain, K_I integral gain, and K_D derivative gain. Each of these gains largely impacts the performance of the system. The PID controller becomes effective when no offset error and improved speed of response is required. Bennett, S. (1993) illustrated the PID control theory and implementation to the complex systems [98]. The PID controller's equations and transfer function are written as follows:

$$Y(t) = K_P e(t) + K_I \int e(t) dt + K_D \frac{de(t)}{dt} \quad (3.16)$$

$$Y(s) = \left(K_P + \frac{K_I}{s} + sK_D \right) E(s) \quad (3.17)$$

$$G(s) = \left(K_P + \frac{K_I}{s} + sK_D \right) \quad (3.18)$$

This controller gives the output according to the error signal. Finding the values of K_P , K_I and K_D is known as tuning the controller. Ziegler Nicholas and Tyreus Luyben are the approaches used to tune the PID controllers. Numerous tuning methods are available in the literature, some include conventional approaches stated above while some employ intelligent methods such as metaheuristic algorithms to implement these control algorithms.

Fractional calculus is capable of enhancing the performance of a traditional PID controller. Including the fractional operators in PID is termed as fractional order PID (FOPID). It is designed by changing the integral and derivative term of a conventional PID to a fractional value ranging 0 to 1. Thus, it provides more design flexibility to the system by using two more terms. In [99], the authors have presented the trajectory tracking control of a three DoF robotic manipulators using PID and FOPID control methods. The FOPID control action is more effective as it provides additional gains to get the desired response [5]. The corresponding equation and transfer function of FOPID has been expressed as follows [10]:

$$T_i(t) = K_{P_i} e_i(t) + K_{I_i} \frac{d^{-\lambda_i}}{d^{-\lambda_i}} e_i(t) + K_{D_i} \frac{d^{\mu_i}}{d^{\mu_i}} e_i(t) \quad i=1,2 \quad (3.19)$$

$$G(s) = (K_{P_i} + \frac{K_{I_i}}{s^{\lambda_i}} + K_{D_i} s^{\mu_i}) E(s) \quad (3.20)$$

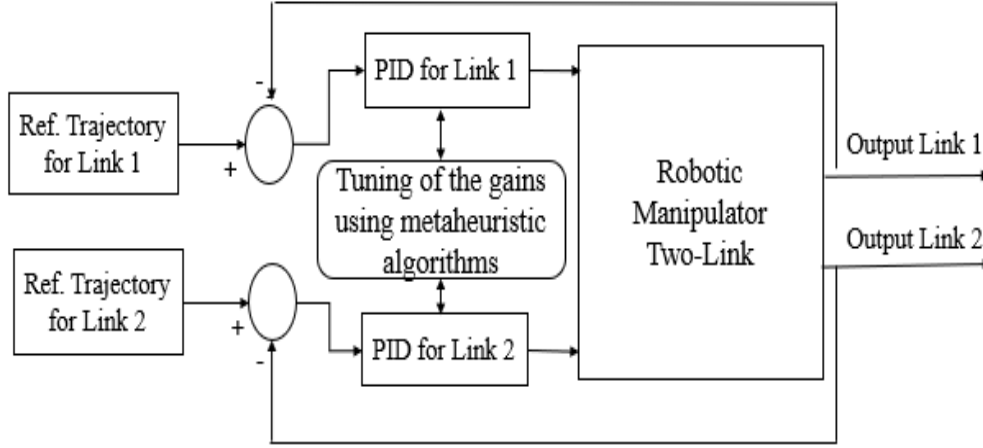


Figure 3.4 Block diagram of PID controller design.

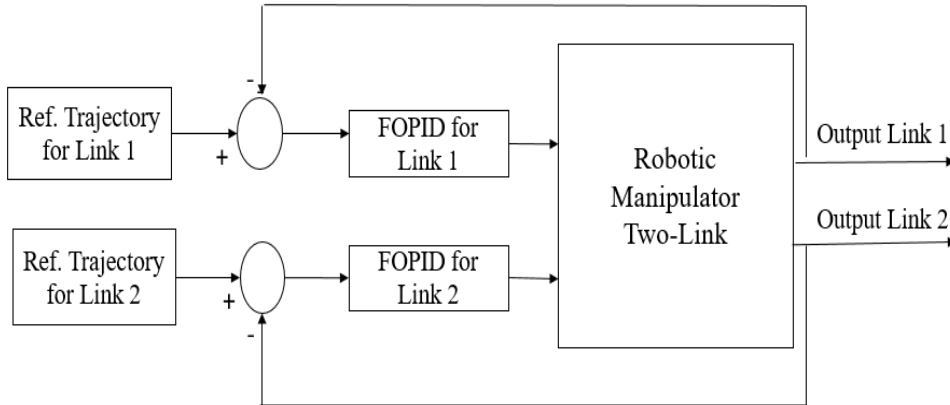


Figure 3.5 Block diagram of FOPID controller design.

Figure 3.4 and Figure 3.5 describes the implementation of the proposed PID and FOPID control for a robotic manipulator having two angular displacements. The robotic manipulator has multi-input multi-output (MIMO) characteristics so two different PID and FOPID has been designed. Efficient tuning is necessary for effective design of controllers. The metaheuristic algorithms GWO, WOA, MVO, and MFO have been implemented to tune these controllers for optimized gains. These algorithms require a performance criterion for the evaluation, so a cost or fitness function has been considered shown in eq. (3.21-3.22). The

proposed control methods are being tuned by these metaheuristic algorithms and thus, the optimum function value validates the effectiveness of these algorithms. The fitness function has been defined as the weighted sum of the *integral absolute error IAE* and the *integral time absolute error ITAE* of both links as shown in eq. (3.21-3.22)

$$f = w_1 * \int e(t)dt + w_2 * \int e_2(t)dt \quad (3.21)$$

$$f = w_1 * \int e_1(t)tdt + w_2 * \int e_2(t)tdt \quad (3.22)$$

w_1 and w_2 are the weights assigned to IAE of both the links and their values are 0.5. A cubic polynomial reference trajectory for the nonlinear model and a fixed step trajectory for the linearized model has been considered for tracking the trajectory. These reference trajectories have been illustrated in detail in the next chapters. The following steps present the methodology for achieving trajectory tracking for a robotic manipulator using PID and FOPID methods:

1. Determine the reference trajectory to be followed by each link of the robotic manipulator.
2. Design of a MIMO PID controller to control the trajectory.
3. Implementation of the recent metaheuristic algorithms to tune the designed controllers.
4. Statistical analysis and attainment of the optimal gains with minimum fitness value.
5. Design of a novel hybrid metaheuristic algorithm to enhance the performance of the designed control scheme.
6. Comparative analysis of implemented algorithms by assigning them Friedman's ranking.

The PID and FOPID controllers have been designed using the four metaheuristic algorithms GWO, WOA, MFO, and MVO for linearized model. The PID controller has been designed for trajectory tracking using the metaheuristic algorithms AOA, ASO, SHO, and STO for nonlinear models. Then a novel hybrid algorithm STOPSO has been designed and implemented using the PID controller for trajectory tracking. The detailed implementation and results have been presented in next chapters.

3.4 Robust Adaptive Sliding Mode Control

SMC consider uncertainties and robustness as a part of design process. It is also known as variable structure control on account of switching nature of control [100]. The objective of SMC is to find the control law expression in terms of the systems state x such that $x \rightarrow x_d$ asymptotically and $e = x - x_d$ approaches to zero at an exponential rate. This is done with the help of a sliding surface. Adaptive sliding mode control (ASMC) [101] is dynamic adaptation within a sliding mode and is based on the purported equivalent control, which is discovered through direct observations of the output of a first-order low-pass filter with a discontinuous control that has a particularly modified magnitude value as its input. In [101], the authors have illustrated the design of ASMC for robotic manipulators. The schematic of the proposed ASMC is presented and shown in Figure 3.6

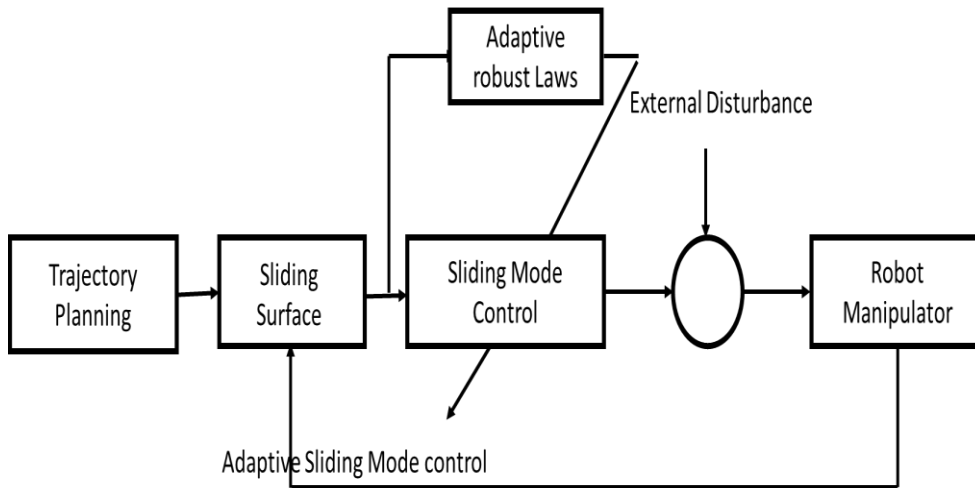


Figure 3.6 Block Diagram of the adaptive SMC controller

Consider the dynamic of n link robotic manipulator is given as

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau \quad (3.23)$$

where $q, \dot{q}, \ddot{q} \in \mathbb{R}^n$ are angle, velocity and acceleration of joints respectively. $M(q)$ is the positive definitive inertia or mass matrix, $C(q, \dot{q})$ is the centripetal coriolis matrix and $G(q)$ is the gravitational force and τ is the applied torque to the robot manipulator. In the design of a robust control system parameter are

written such that it contains a nominal value and contains parameter uncertainties or tolerance factor.

The mass matrix can be expressed as

$$M(q) = M_0(q) + \Delta M(q) \quad (3.24)$$

$$C(q) = C_0(q) + \Delta C(q) \quad (3.25)$$

$$G(q) = G_0(q) + \Delta G(q) \quad (3.26)$$

Where $M_0(q)$ is the nominal value and $\Delta M(q)$ is the parameter uncertainties or tolerance factor, means if $\Delta M(q)$ is given $\pm 0.5\%$ then mass value can vary from 99.5 to 100.5 respectively. The higher the parameter uncertainties percentage, the more robust the system will be. Similarly, $\Delta C(q)$ and $\Delta G(q)$ are the coriolis and gravitational parameter uncertainties factor. Mostly in Robust control one must deal with disturbances (input and output disturbances) so input can be bifurcated in to control law and disturbances.

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = u + d_i(t) \quad (3.27)$$

Assume that disturbances $d_i(t)$ are bounded in nature. The eq (1) can be written in the following form

$$M_0(q)\ddot{q} + C_0(q, \dot{q})\dot{q} + G_0(q) = u + \eta_i(t) \quad (3.28)$$

Where $\eta_i(t)$ is defined as

$$\eta_i(t) = -\Delta M(q) - \Delta C(q) - \Delta G(q) + d_i(t) \quad (3.29)$$

Assumption 1: Mass matrix $M(q)\ddot{q}$ and coriolis matrix $C(q, \dot{q})\dot{q}$ are bounded in nature. Their derivatives are also bounded.

Assumption 2: Disturbance $d_i(t)$ is bounded and continuous differentiable.

The robotic manipulator's trajectory tracking control can be calculated as follows. Define the error of trajectory tracking $e = q - q_d$, where q_d is the reference trajectory. Main objective is to obtain the control law u such that output trajectory follows the reference trajectory.

Consider the surface $S = e_1 + Ke^{\frac{b}{a}}$, where $e_1 = \dot{q} - \dot{q}_d$, $K > 0$, a, b are the odd integers.

The dynamic error corresponding to eq. (3.27) is

$$\begin{cases} \dot{e} = e_1 \\ \dot{e}_1 = -\ddot{q}_d - M_0(q)^{-1}(C_0 + G_0) + M_0(q)^{-1}u + M_0(q)^{-1}\eta_i(t) \end{cases} \quad (3.30)$$

The equivalent control is the sum of high-frequency control. Robust control primarily deals with low-frequency and adaptive high-frequency conditions [6]. Overall controller will be the sum of low frequency controller and high frequency controller, shown as $u = u_{lf} + u_{hf}$. The representation of the equivalent control can be found out with or without disturbances and uncertainties.

$$u_{lf} = M_0(q)(\ddot{q}_d - \frac{b}{a}K \left\{ e^{\frac{b}{a}-1} \right\} + (C_0 + G_0) \quad (3.31)$$

$$u_{hf} = -\frac{(S^T M_0(q)^{-1})^T}{\|S^T M_0(q)^{-1}\|^2} [\|S\| \|M_0(q)^{-1}\|] \quad (3.32)$$

The nominal value of $m_1 = 0.3$ and $m_2 = 1.2$ are respectively and suppose the uncertainty of $\pm 10\%$. The other system parameter are $L_1 = 1, L_2 = 0.8, J_1 = J_2 = 5 \text{ kg.m}$. The following two trajectories has been chosen for desired angular position

$$q_{d1} = 1.26 - \frac{8}{5} \exp(-t) + \frac{8}{20} \exp(-5t) \quad (3.33)$$

$$q_{d2} = 1.5 - \frac{8}{5} \exp(-t) + \frac{8}{20} \exp(-5t) \quad (3.34)$$

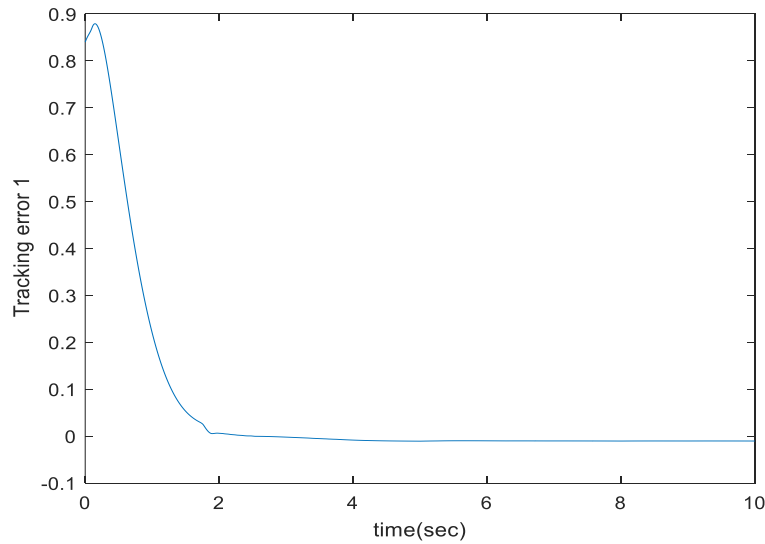


Figure 3.7 Error of link 1

Figure 3.7 and Figure 3.8 show the error value for both links and Figure 3.9 and Figure 3.10 shows the tracking error for both links. It is clear from the figures

that the proposed controller can track the reference trajectory. The initial deviations from the reference trajectory have been handled well by the proposed controller.

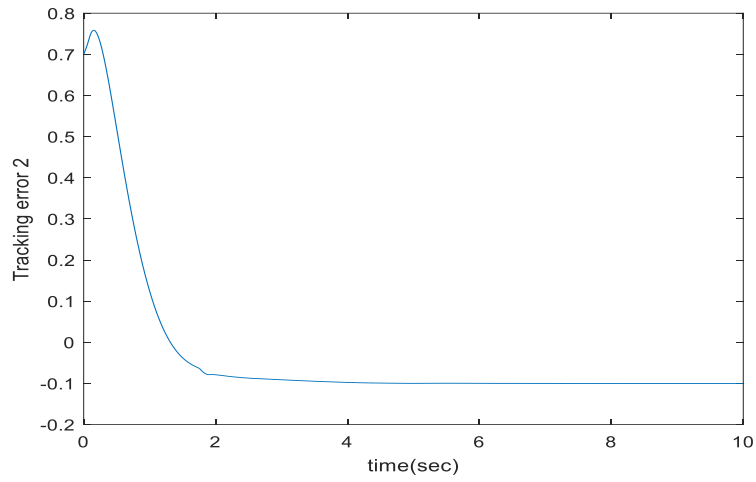


Figure 3.8 Error of the link 2

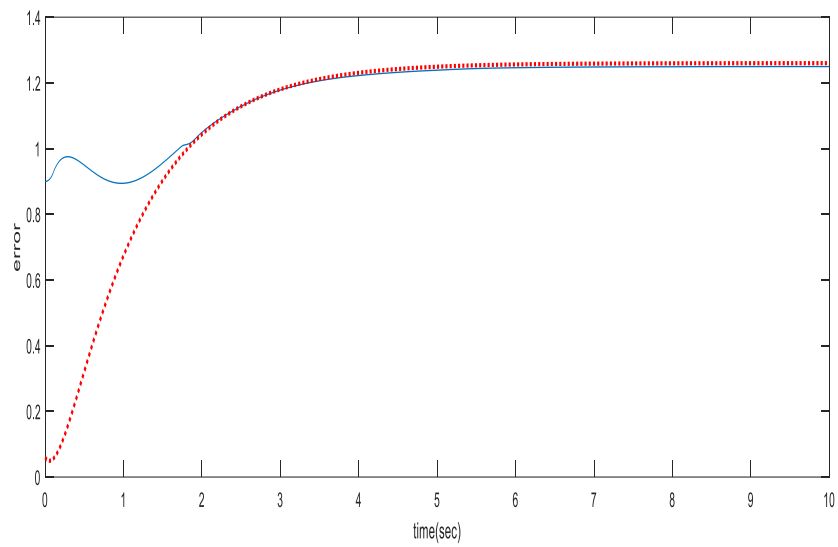


Figure 3.9 Tracking error of the first joint with ASMC

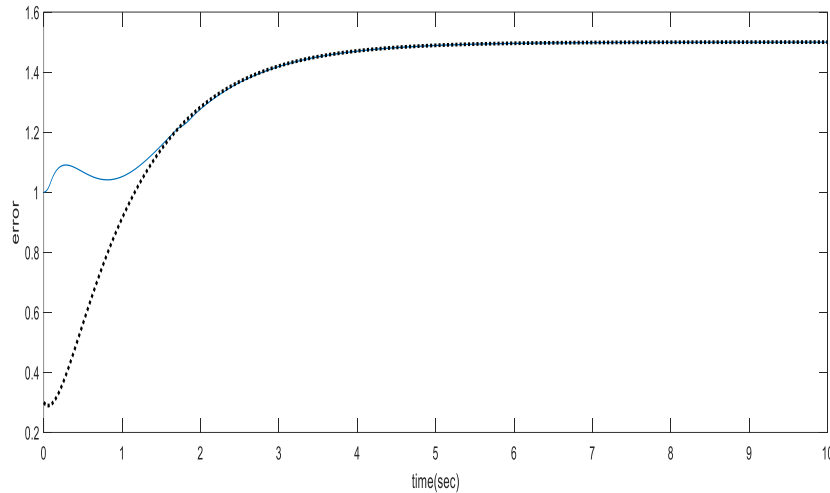


Figure 3.10 Tracking error of the second joint with ASMC

3.5 Stability Analysis

For nonlinear control one of the most widely used techniques for stability analysis is the Lyapunov stability criterion which is known as the Lyapunov theorem. Lyapunov stability theory is widely used in control theory, robotics, and other fields of engineering to analyze and design control systems. It provides a powerful tool for ensuring the stability and performance of complex systems. This stability criterion utilizes the positive or negative definite and semi-definite properties of the functions. A function $V(x)$ can be characterized as positive definite (PD), positive semidefinite (PSD), negative definite (ND), and negative semidefinite (NSD) if it satisfies the following properties.

If $V(0) = 0$, and $V(x) > 0$ for $x \neq 0$, then function is PD.

If $V(0) = 0$, and $V(x) \geq 0$ for $x \neq 0$, then function is PSD.

If $V(0) = 0$, and $V(x) < 0$ for $x \neq 0$, then function is ND.

If $V(0) = 0$, and $V(x) \leq 0$ for $x \neq 0$, then function is NSD.

According to the Lyapunov's theorem, for $x = 0$ an equilibrium point for a dynamic system $\dot{x} = g(x)$, $D \in R^n$ is the domain consisting of equilibrium point, a positive definitive Lyapunov function $V: D \rightarrow R$ a constantly differentiable function is assumed such that $V(0) = 0$, and $V(x) > 0$ for $x \neq$

0, derivative of V , $\dot{V}(x) < 0$ has to be negative definite then system would be stable [103], For $\dot{V}(x) > 0$ or $\dot{V}(x) \geq 0$ the system will be unstable.

For analyzing the stability following Lyapunov function has been considered.

$$V(r, \theta) = \frac{1}{2}mr^2 + \frac{1}{2}\theta^T\Gamma^{-1}\theta \quad (3.35)$$

Where $r = \dot{e} + ae$ and Γ is positive diagonal matrix.

$$\dot{V}(r, \theta) = r(m\ddot{q} + C\dot{q} + Gq + \alpha m\dot{e} - \tau) + \frac{1}{2}\dot{\theta}^T\Gamma^{-1}\theta + \frac{1}{2}\theta^T\Gamma^{-1}\dot{\theta} \quad (3.36)$$

$$\dot{V}(r, \theta) = r(P^T\theta - \tau) + \theta^T\Gamma^{-1}\dot{\theta} \quad (3.37)$$

Where $P^T\theta = m\ddot{q} + C\dot{q} + Gq + \alpha m\dot{e}$

$$\dot{V}(r, \theta) = r(P^T\theta - \tau) - \theta^T\Gamma^{-1}\dot{\hat{\theta}}, \dot{\hat{\theta}} = -\dot{\hat{\theta}} \quad (3.38)$$

Next is to design $\tau = \widehat{P^T\theta} + kr$, $\dot{\hat{\theta}} = \Gamma Pr$

then $\dot{V}(r) = -kr^2$, which is negative semi definite. The negative semi-definite derivative of the Lyapunov function validates the stability of the proposed controller.

3.6 Extremum Seeking Control (ESC)

With the advent of technology robotic systems have gained much popularity and have been employed in many industrial applications. Robotic systems provide ease and comfort to human lives by embedding a certain amount of autonomy. Because of the large applications in industry effective control methods are extremely important. Researchers are continuously exploring ways to control these robotic systems and provide autonomous solutions for performing industrial tasks. ESC [104] is a control strategy that tracks the varying performance function. It refers to monitoring the varying maximum or minimum of a performance function. Ariyur, K. B., & Krstic, M. (2003) presents the principles and real-time optimization using ESC [105]. ESC is a local optimizer and changes in the system dynamics are faster than the perturbations. Figure 3.11 describes the ESC method for robotic manipulator, the cost function or performance function is used to design the control law. It is an equation-free adaptive control approach that adapts to parameter changes. It is an optimization technique in which a sinusoidal perturbation is added to the control input u . Based on the control law, this method provides the optimum value of the objective function. To perform various industrial tasks robotic manipulator has to track the optimum position. ESC tracks that optimum value.

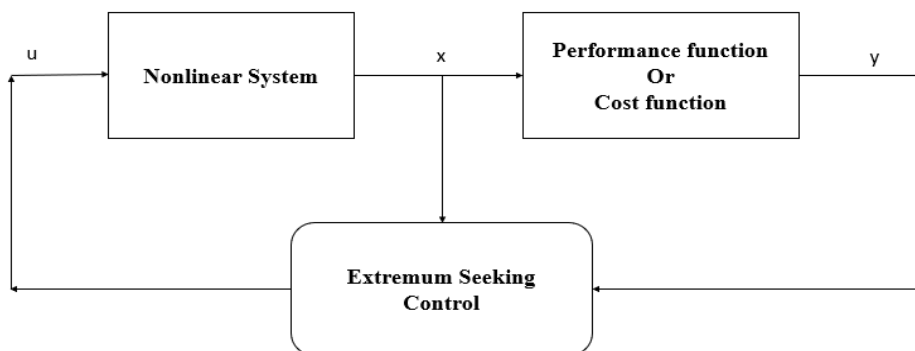


Figure 3.11 Block diagram of ESC

In [106], the authors have described each of these ESC techniques by performing a comparison and robustness analysis of each of the ESC techniques. Authors characterized these techniques as sliding mode ESC, neural network-based ESC, approximation-based ESC, and adaptive ESC. Robotic

systems have high nonlinearities, uncertain dynamics, and high disturbance, perturbation-based ESC gives robust trajectory performance. In conclusion, the authors suggested the use of approximation-based ESC when noise is not much effective, neural network ESC is preferred in the presence of significant noise while perturbation-based ESC is much more effective when a high level of noise and uncertain dynamic effects are present. Thus, perturbation-based ESC is found to be the most robust. ESC techniques are classified as shown in Figure 3.12

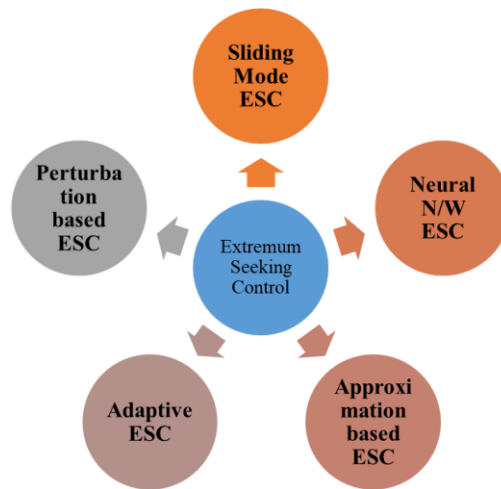


Figure 3.12 Classification of ESC Techniques

Malek, H., & Chen, Y. (2016) [107] proposed a novel fractional order ESC that improved the convergence, robustness, and performance by incorporating fractional calculus in ESC. Simulation and experimental analysis validate the proposed scheme and show better performance as compared to the classic extremum-seeking algorithm. ESC provides a robust and stable response. In [108], the authors performed a stability analysis of ESC and proved the stability of ESC by averaging method and singular perturbation analysis. Further in [109], the authors suggested inclusion of a dynamic compensation that resulted in improvement in the performance of ESC and enhanced stability. Researchers are continuously finding ways to improve the ESC. In [110], the authors designed a novel fast ESC to improve the static and dynamic performance of ESC without any steady state oscillations. ESC is suitable for the system's disturbances and variations in the parameters. ESCs have applications in the

systems that have disturbances and variations in the parameters over time. ESCs find applications in areas like process control [111], Renewable energy [112], automotive industry [113], and robotic systems.

3.6.1 Design of ESC for robotic manipulator

Robotic manipulators being exceedingly nonlinear, uncertain, and MIMO systems, two separate ESCs using two different objective functions have been designed for two different links. A cubic polynomial reference trajectory has been considered, having optimum values of $\pi/4$ and $\pi/6$ respectively. J_1 and J_2 are the objective functions used to track the optimum value of trajectories of both links.

$$J_1 = (\pi/4) - (0.89 - u)^2 + C \quad (3.39)$$

$$J_2 = (\pi/6) - (0.72 - u)^2 + C \quad (3.40)$$

u is a control input, a sinusoidal perturbation is introduced to this. C is a constant term. Figure 3.13 shows the reference trajectory considered for tracking and Figure 3.14 presents the output of the ESC designed for both links.

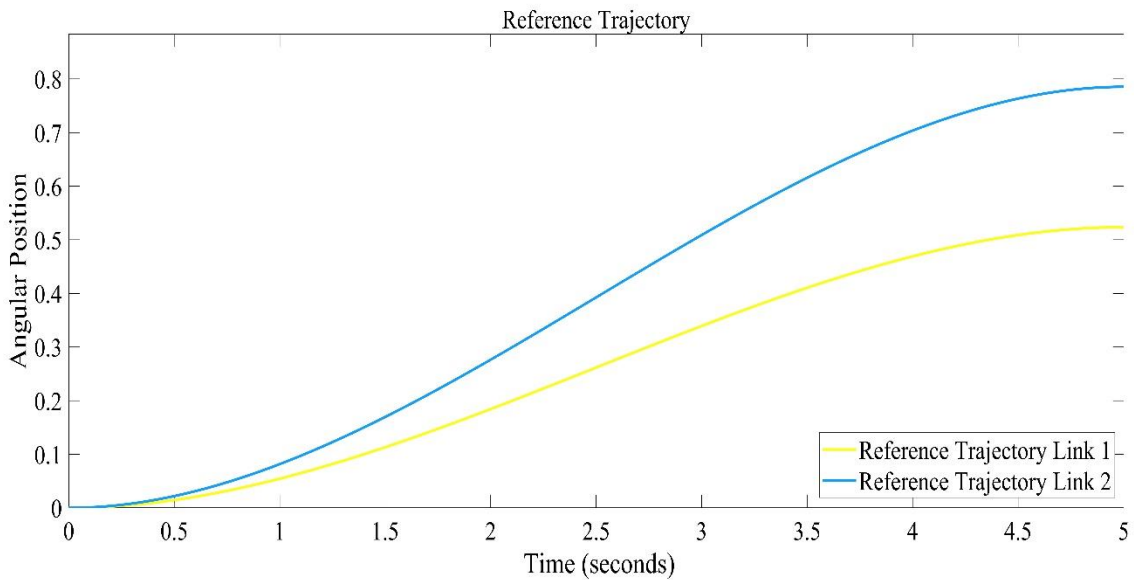


Figure 3.13 Polynomial Reference trajectory

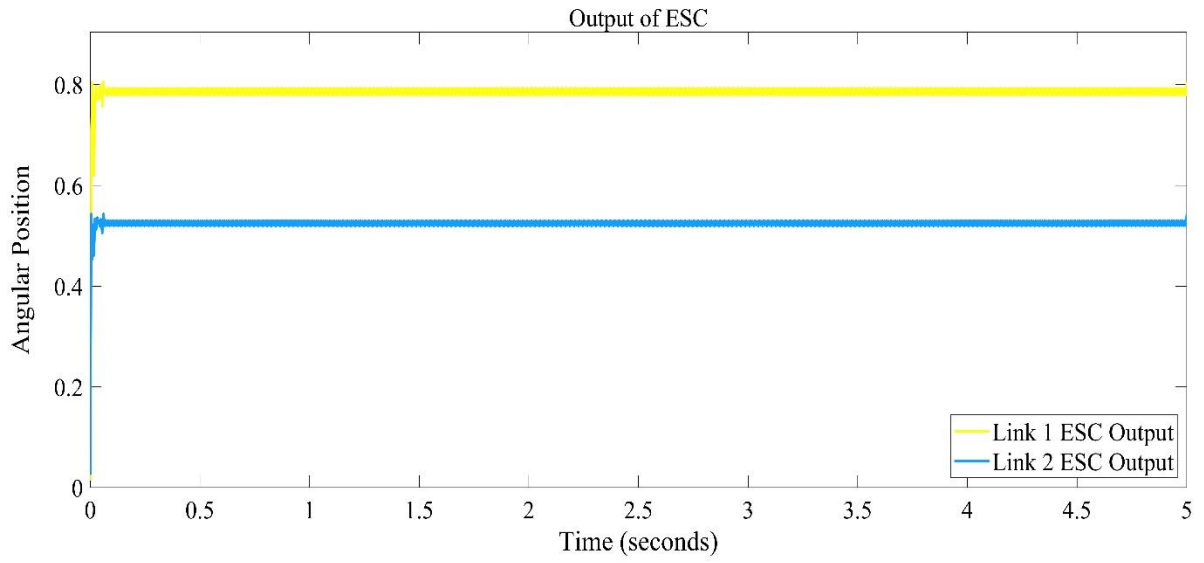


Figure 3.14 Output of ESC

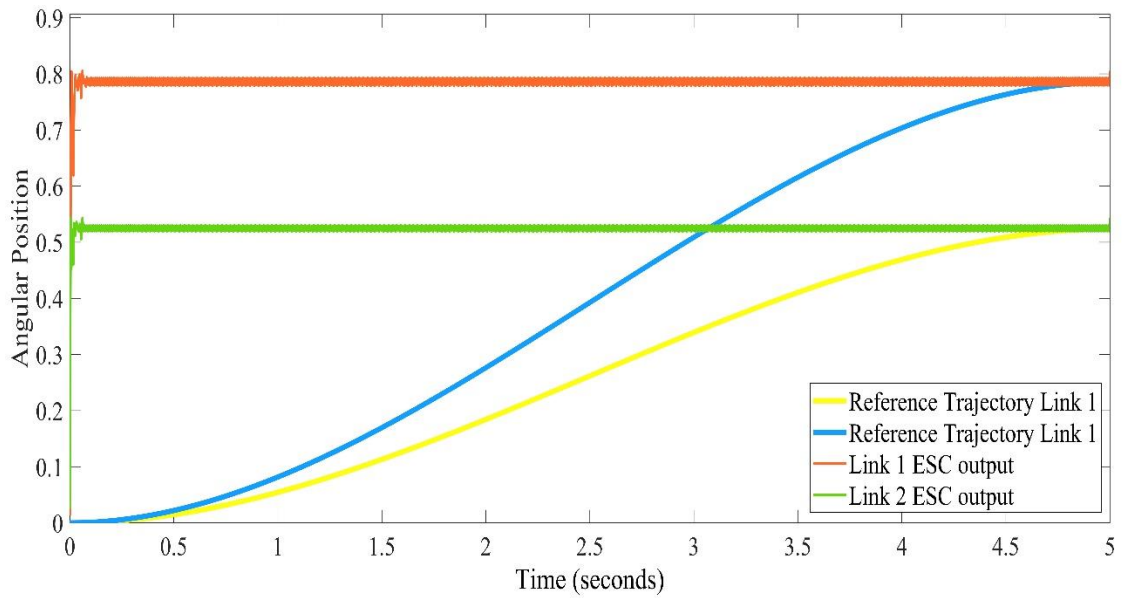


Figure 3.15 Optimum value tracking of the reference trajectory

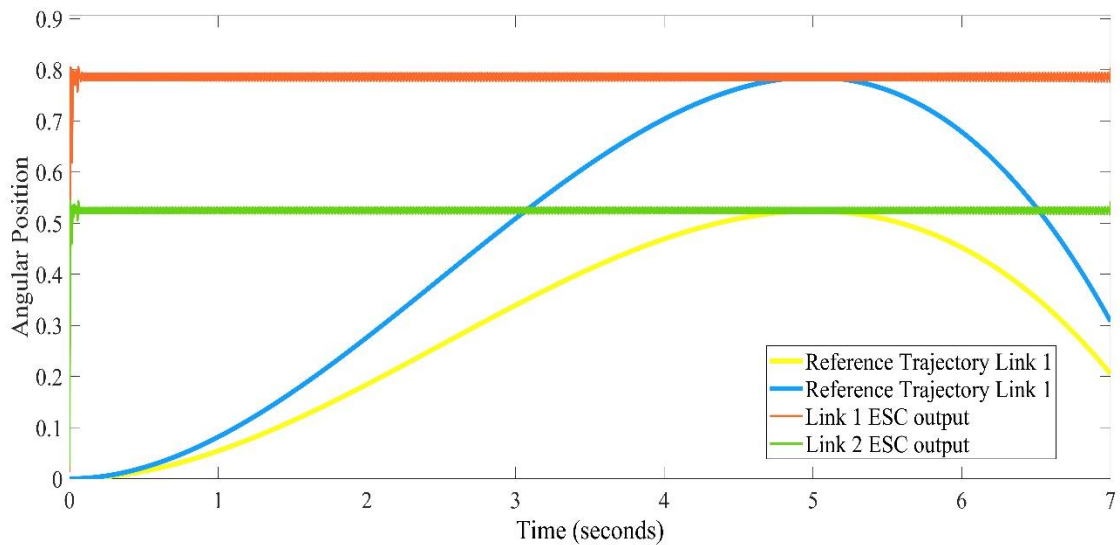


Figure 3.16 Optimum value tracking of the reference trajectory

Figure 3.15 and Figure 3.16 show the optimum value trajectory tracking for both the links, where both the objective functions have been able to track the optimum value of trajectories. It is evident from both figures that the designed controller is able to seek the optimum value in the reference trajectories.

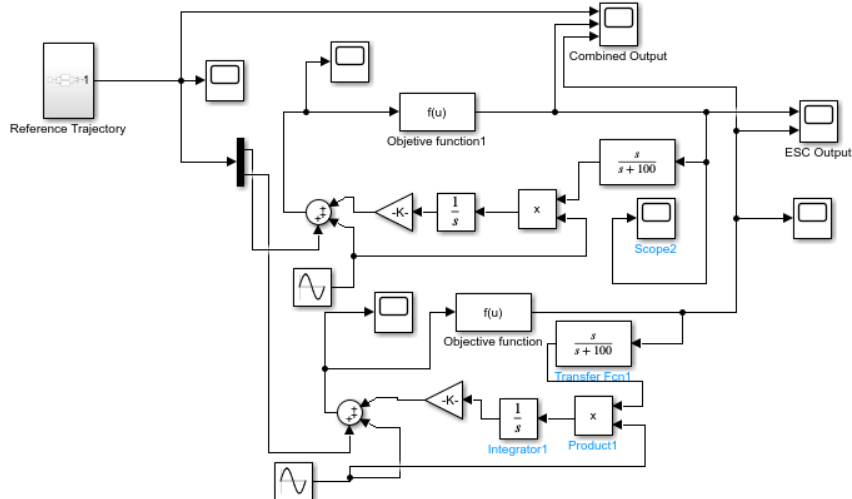


Figure 3.17 SIMULINK model of ESC for trajectory control of robotic manipulator.

The Simulink model of the ESC has been shown in Figure 3.17. The polynomial trajectory as shown in Figure 3.13 has been given as a reference. A sinusoidal perturbation has been included in the control law, this changes the control law

and estimates the best input. Two different objective functions have been incorporated each on the different links of a robotic manipulator. The output of the proposed controller shows tracking of the optimal point of given reference trajectory.

Robotic manipulators have various industrial applications that require tracking of optimum value of the desired trajectory. Because of the complex and uncertain behaviour of the robotic system, it is difficult to track the optimum value in such applications. ESC can track the optimum point trajectory satisfactorily.

Chapter 4

Optimization and statistical analysis of control techniques for linearized model

4.1 Introduction

Metaheuristic algorithms are nature-inspired methods that find the best possible solution (optimum value) to any challenging problem. Mirjalili et al. [59] presented optimization algorithms inspired from nature to solve complex problems like trajectory control of robotic manipulators. The trajectory tracking has been achieved using control schemes like PID and FOPID. To find the optimal gains of these control techniques, the following algorithms have been implemented. The fitness function or cost function is the performance indices in the terms of the error values of the achieved trajectory. The metaheuristic algorithms implemented on PID and FOPID controllers provide the optimum value of the fitness function of a linear two-link robotic manipulator.

- i. Grey wolf optimizer (GWO) algorithm
- ii. Whale optimization (WOA) algorithm
- iii. Moth flame Optimization (MFO) algorithm
- iv. Multi-verse optimization (MVO) algorithm

Each of these algorithms is inspired by swarm intelligence and nature, these algorithms can be modeled and expressed in a mathematical approach [60]. The description and mathematical approach for these algorithms and their implementation in tracking the trajectory of a linearized two-link manipulator has been presented in the next sections.

4.2 Grey Wolf Optimization (GWO)

Mirjalili et al. (2014) [114] projected an innovative stochastic swarm intelligence algorithm named GWO. The algorithm is inspired from the hunting behavior and social hierarchy of grey wolves and finds wide application to optimize complex optimization problems. Grey wolves are also referred to as idealist hearths belonging to the family of Canidae mostly lives in regions of north America. Grey wolves communicate through howling, barking and through different body language specially to build social hierarchy of packs. There are four levels in the hierarchy of wolves which are alpha (α) the leaders, beta (β) coordinating subordinates, delta (δ), omega (ω) as shown in Figure 4.1. Alpha wolves are the leaders of the pack and dictate to other wolves. Beta wolves are subordinate of alpha and guide the other wolves. Delta wolves stand third in hierarchy but dominates the omega wolves. These wolves include scouts, hunters, sentinels and are responsible for maintaining the safety of the entire pack. Omega wolves are last in the hierarchy and follow all the other dominant wolves. All four categories of grey wolves live in packs of 5 to 15 and show an important behavior of group hunting. These wolves first track and chase the prey, once the prey is chased then these packs of wolves enfold the prey and harass it until the prey is tired and finally, they attack the prey. The flow chart of GWO has presented in Figure 4.2

The mathematical approach of GWO is illustrated as:

Step 1: The position of grey wolves (search agents) initialized arbitrarily in the search space:

$$Y_i = (y_i \dots \dots \dots y_n) \quad (4.1)$$

where, n signifies the space dimension.

Step 2: The coefficients a, A, and C are initialized and expressed as:

$$A = 2 * a * r_1 - a \quad (4.2)$$

$$C = 2 * r_2 \quad (4.3)$$

where, r_1 , r_2 represents the random numbers having range [0,1] and a is a constant decreasing from a value of 2 to 0 over the iterations.

Step 3: The fitness of all grey wolves is evaluated, depending on the problem the position of first three best grey wolves is termed as Y_α , Y_β , Y_δ respectively. The position of rest of the grey wolves are Y_ω .

Step 4: Upgrade the position of each grey wolves towards the best grey wolves (Y_α , Y_β , Y_δ) based on the following equations:

$$X_\alpha = |C_1 * Y_\alpha - Y|, \quad X_\beta = |C_2 * Y_\beta - Y|, \quad X_\delta = |C_3 * Y_\delta - Y| \quad (4.4)$$

$$Y_1 = Y_\alpha - A_1 * (D_\alpha), \quad Y_2 = Y_\beta - A_2 * (D_\beta), \quad Y_3 = Y_\delta - A_3 * (D_\delta) \quad (4.5)$$

$$Y(k + 1) = \frac{Y_1 + Y_2 + Y_3}{3} \quad (4.6)$$

where, A_1, A_2, A_3 and C_1, C_2, C_3 are the coefficients of α, β, δ wolves and k denote the current iteration.

Step 5: Update the coefficient vectors A and C using eq. (4.2) and eq. (4.3).

Step 6: Initialize the position of grey wolves again that go above the defined space.

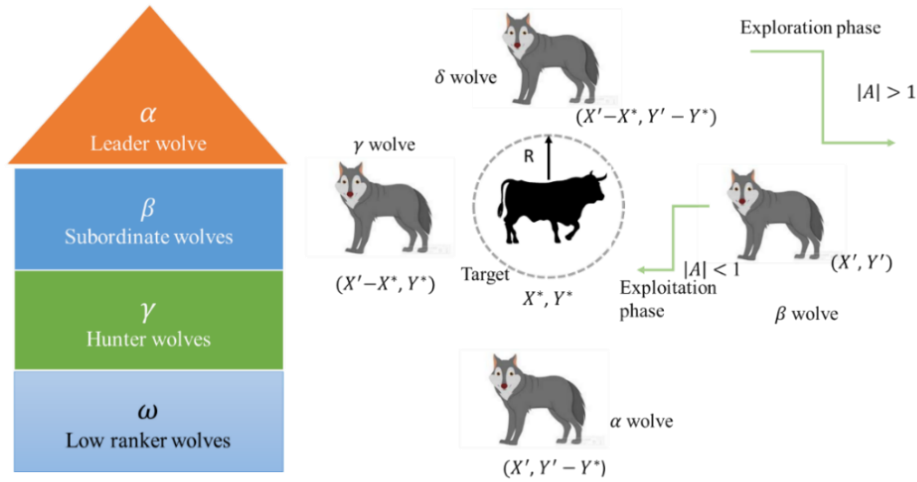


Figure 4.1 Description of GWO Algorithm

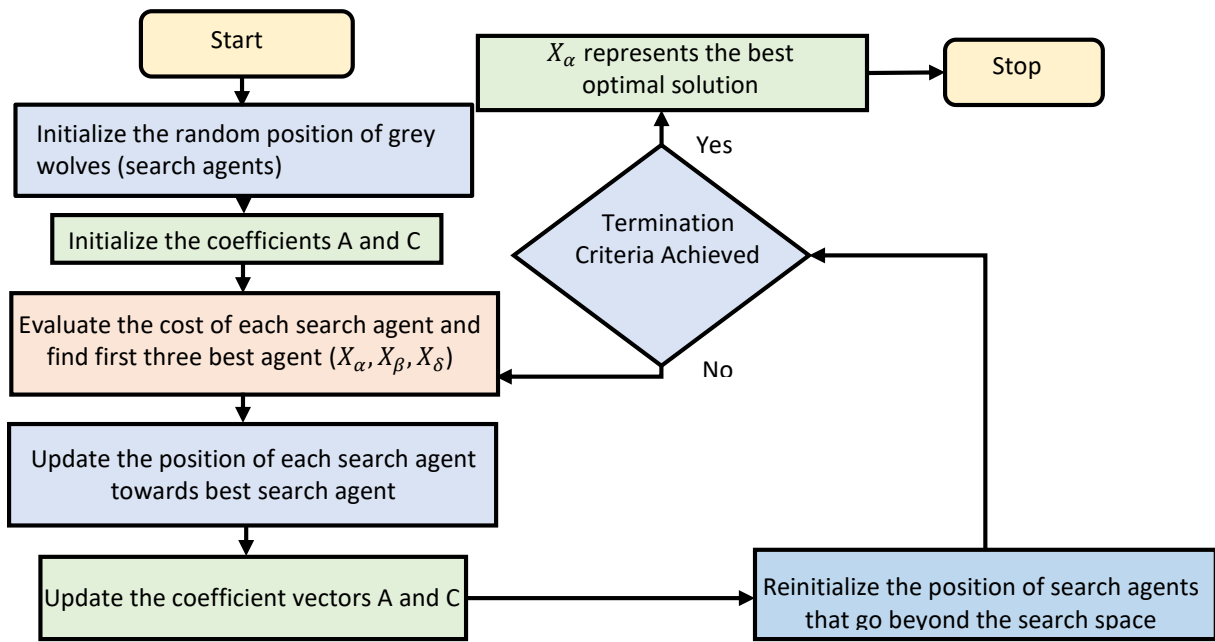


Figure 4.2 Flow diagram of GWO algorithm

4.3 Whale Optimization Algorithm (WOA)

Mirjalili et al. (2016) [115] proposed novel stochastic optimization algorithm named WOA. Humpback whales are the species of baleen whale and is one of the intelligent species having emotions. The most unique feature of these whales are that they are the biggest mammal in the world with adult length around 39-53 feet and weight 25-30 metric tons. Spindle cells named after their spindle-shaped bodies occur in brain are responsible for the social, intelligent and smart behavior of whales. Humpback whales have a most diverse hunting behavior, and the main source of their food is small fish. Whales hunt their prey with bubble-net feeding technique in which a group of whales encircle their prey and blows bubbles around them. In upward-spiral whale moves 12 meter down and form bubble in a spiral pattern towards the prey and dip up towards the surface while in double loop whale move around the prey in three different fashions, i.e., coral loop, lobtail, and capture loop. The algorithm works in three phases, i.e., searching, encircling, and hunting prey. When swimming around their prey, humpback whales can maneuver in a spiraling pattern or along a path that is gradually getting smaller and a probability factor p switches either of the two movement. Exploration and exploitation need to

be kept in balance as shown in Figure 4.3 through $|\bar{C}|$ vector decreasing from 2 to 0 over the iterations. In the initial phase when $|\bar{C}| \geq 1$ then the whales explore around the random prey while as $|\bar{C}| < 1$ then whale exploits the search space and swim around the best prey. The flow diagram of WOA is shown in Figure 4.4.

The mathematical approach inspired from spiral bubble-net feeding movement of whales around prey is illustrated as:

Step 1: Initiate the population of whales (search agents) arbitrarily within definite space:

$$Y_i = (y_i \dots \dots \dots y_n) \quad (4.7)$$

where, n signifies the space dimension.

Step 2: Evaluate the cost of each whale and depending on the problem (minimization or maximization) find the position of best whale (Y^*).

Step 3: Modify the constants G, H using the following equations:

$$G = 2 * b * r - b \quad (4.8)$$

$$H = 2 * r \quad (4.9)$$

where, r is the random number having range [0,1], b is iteratively decreasing from 2 to 0 and p is a random parameter lying between [0, 1].

Step 4: If $p < 0.5$ and $|H| \geq 1$, random position of whale (Y_{rand}) is selected in search space and position of whale is updated around it using the following equations:

$$D = |H * Y_{rand} - Y| \quad (4.10)$$

$$Y(t + 1) = Y_{rand} - G * D \quad (4.11)$$

else if, $p < 0.5$ and $|G| < 1$, then apprise the position of whale around the best search agent (X^*) using the following eq. (4.7-4.8).

$$D = |H * Y^* - Y| \quad (4.12)$$

$$Y(t + 1) = Y^* - G * D \quad (4.13)$$

else, $p > 0.5$, then update the position of whale using the following equation:

$$Y(t + 1) = D' \cdot e^{dl} \cdot \cos(2\pi l) + Y^*(t) \quad (4.14)$$

where, $D' = |Y^*(t) - Y(t)|$ is the distance between the whale and best searched prey ($Y^*(t)$), is the constant d maintains the logarithmic spiral shape and l is the

arbitrary number confined in the range $[-1,1]$, \cdot is element by element multiplication.

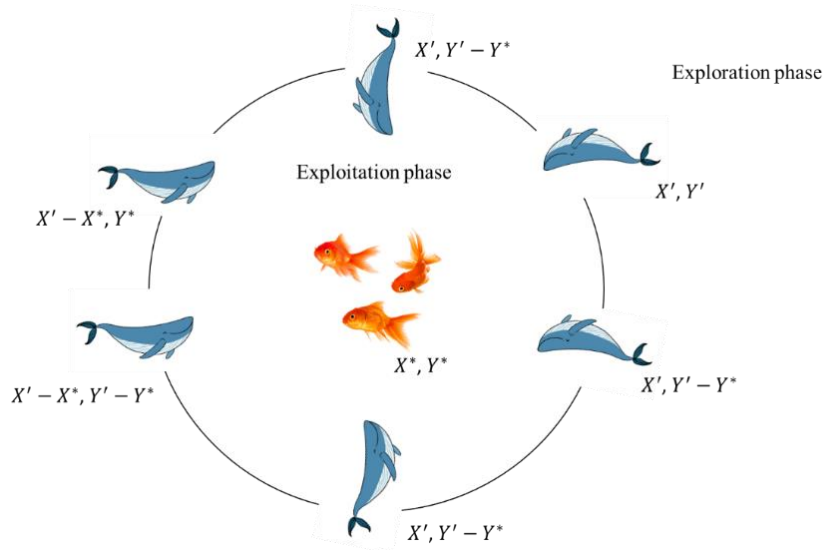


Figure 4.3 Description of WOA

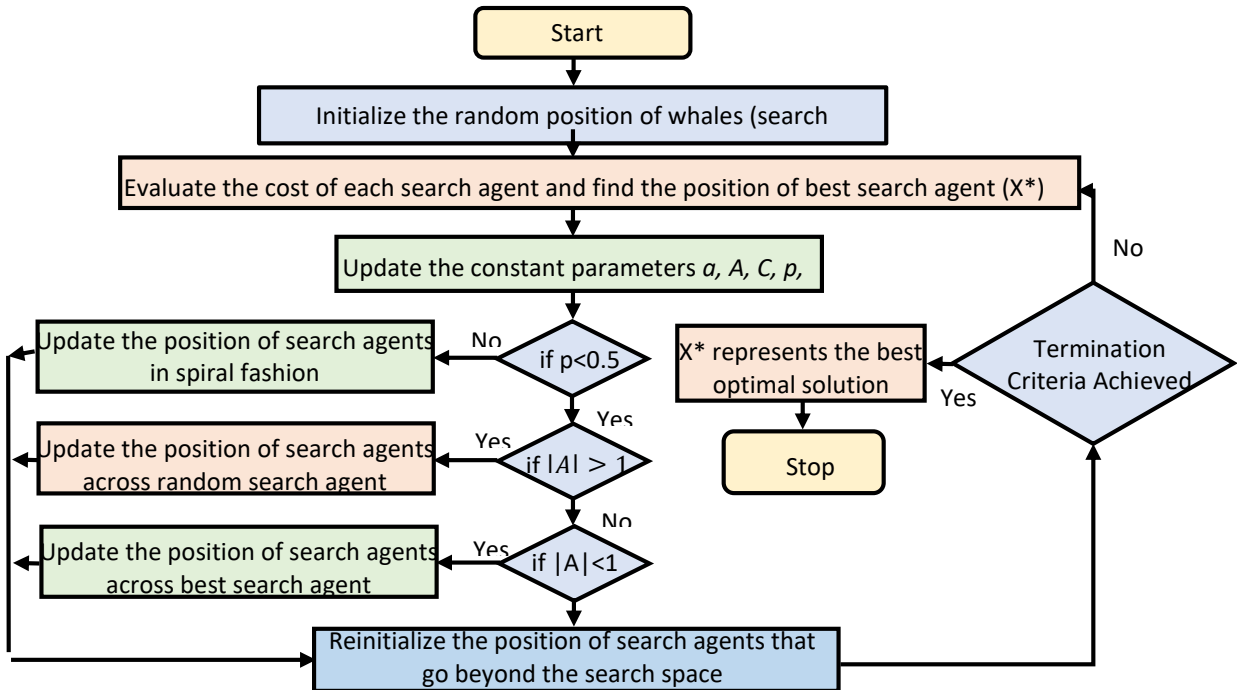


Figure 4.4 Flow diagram of WOA

4.4 Moth Flame Optimization (MFO)

Mirjalili et al. (2015) [116] proposed a novel population-based nature-inspired swarm intelligence algorithm named MFO. Moths are a paraphyletic group of insects with characteristics resembling those of butterflies. Around 160,000 species of moths, including nocturnal, crepuscular, and diurnal species, are known to occur in nature. Some moth larvae dig burrows in the earth and dwell there until they are adults, while others grow up in cocoons. By maintaining a constant angle with respect to the moon, the moth travels over a great distance in a straight line. However, the moth eventually converges to it after being entangled in a swirling path across the artificial lights.

Figure 4.5 illustrates how the MFO algorithm simulates the movement of moths in a logarithmic spiral way across the flame to get the best solution. The search space is initially filled with a random population of moths, and their positions are spirally updated with regard to the flame while keeping in mind that the moth movement shouldn't go outside the search space. Moths can be thought to move in a hyper ellipse across the flame in every direction. The moth's movement in the direction of the flame causes the algorithm to become locked in local optimum and the position of each moth is updated in relation to its corresponding flame. This causes each moth to travel around diverse flames and lowers the likelihood of local optima stasis.

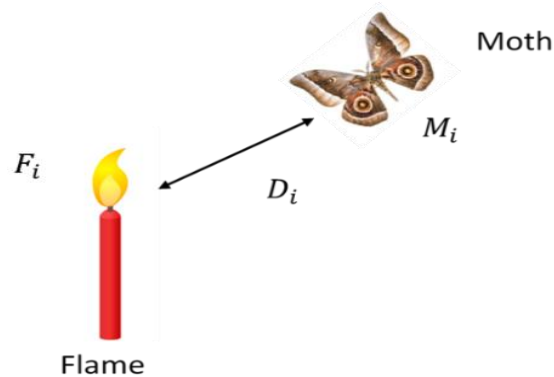


Figure 4.5 Description of MFO algorithm

The position of flame is also modified during each iteration with regard to the optimal answer, which improves the algorithm's exploration capacity. The moth's movement in several locations within the confined search space improves the level of exploration but reduces the exploitation capability. Exploration and exploitation need to be balanced in each optimization technique. To increase the algorithm's exploitation capabilities, an adaptive approach for estimating the number of flames is provided. The number of flames is reduced adaptively throughout an iteration ensuring that moths update their position with regard to the best updated flame in the preceding iterations. The flow diagram of MFO is shown in Figure 4.6

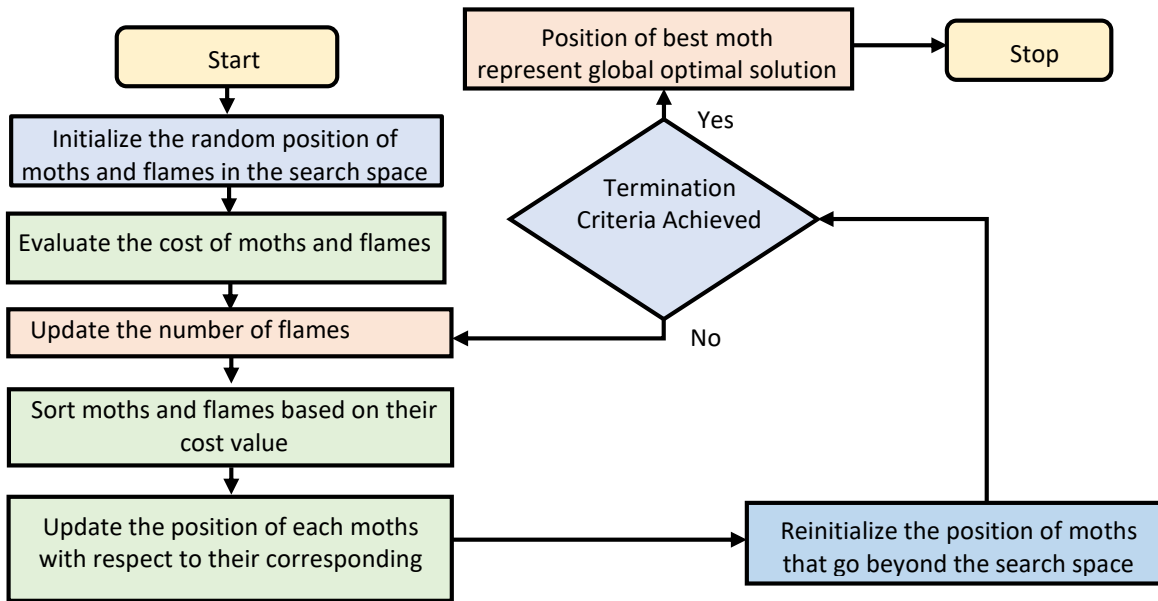


Figure 4.6 Flow diagram of MFO algorithm

The mathematical approach of MFO is illustrated as:

Step 1: Initializing the population of the moths (search agents) arbitrarily in the definite space as:

$$P = \begin{bmatrix} p_{1,1} & p_{1,2} & \dots & \dots & p_{1,l} \\ p_{2,1} & p_{2,2} & \dots & \dots & p_{2,l} \\ p_{m,1} & p_{n,2} & \dots & \dots & p_{n,l} \end{bmatrix} \quad (4.15)$$

where, m is the moth numbers and l are the dimensions.

Step 2: Every individual moth's position vector needs to be placed in the cost function represented by OP to calculate the cost of moth.

$$OP = \begin{bmatrix} OP_1 \\ OP_2 \\ \cdot \\ \cdot \\ OP_n \end{bmatrix} \quad (4.16)$$

Step 3: Initialize the arbitrary matrix of flames identical to moths as follows:

$$E = \begin{bmatrix} e_{1,1} & e_{1,2} & \dots & \dots & e_{1,l} \\ e_{2,1} & e_{2,2} & \dots & \dots & e_{2,l} \\ \dots & \dots & \dots & \dots & \dots \\ e_{m,1} & e_{m,2} & \dots & \dots & e_{m,l} \end{bmatrix} \quad (4.17)$$

Step 4: Place the flame's location vector in the cost function to calculate the cost of each flame and is expressed as:

$$OE = \begin{bmatrix} OE_1 \\ OE_2 \\ \cdot \\ \cdot \\ OE_m \end{bmatrix} \quad (4.18)$$

Step 5: Sorting the moths and flames according to their cost.

Step 6: Use the following equation to update the number of flames:

$$flame\ no. = round\left(R - j * \frac{R-1}{T}\right) \quad (4.19)$$

where T is the maximum iteration, j denotes the current iteration, and R represents the maximum flames.

Step 7: Update the position of every individual moth according to the corresponding flame in a spiral fashion as:

$$P_i = S(P_i, E_j) \quad (4.20)$$

where, P_i represents the i -th moth and E_j represents the j -th flame and S is the spiral function and a logarithmic function that is defined as:

$$S(P_i, E_j) = D_i * e^{bt} * \cos(2\pi t) + E_j \quad (4.21)$$

where D_i is the distance between the i -th moth and the j -th flame, b is a constant that determines the form of the logarithmic spiral, and t is a random value between $[-1, 1]$

$$D_i = |E_j - P_i| \quad (4.22)$$

Step 8: Identify the moths leaving the search space and reposition them within the limits.

Step 9: The algorithm terminates when the minimal error stopping criteria or maximum number of iterations is reached. Alternatively, repeat steps (2) to (9).

Step 10: The best moth location reflects the universal ideal solution.

4.5 Multi Verse Optimization (MVO)

Mirjalili et-al. (2015) [117] presented MVO a nature-inspired stochastic optimization technique. MVO algorithm mathematically models the multi-verse theory of physics. As per the theory, there are multiple big bangs, and each individual big bang creates a new universe. Thus, there exists more than one universe along with the actual universe. The three main components of MVO algorithm are white, black, and worm holes. White holes are formed through the collision between parallel universes, black holes work differently with respect to white hole and absorbs most of the things with quite high gravitational force while wormholes join distinct parts of the universe. Multi-verse theory states that different universes interact with white, black and wormholes to attain stability. It has been observed that each universe has varying rise rates which control its extension throughout space. Traditionally, all optimization algorithms are governed through their exploration and exploitation capability. MVO utilizes white and black holes to better explore the search space while wormholes aid to exploit the search space. The algorithm initializes by creating a random population in the search space and over the course of iterations the objects in universe with high rate of inflation moves to the universe having low inflation rate through white/black holes. Although, all universe objects experiences teleportation via wormholes in direction

of best universe. Figure 4.7 shows the flow chart of MVO algorithm. The following steps mathematical approach of MVO algorithm is illustrated as follows:

Step 1: Initialize the arbitrary population of universe (U) in the search space:

$$U = \begin{bmatrix} y_1^1 & y_1^2 & \cdot & y_1^d \\ y_2^1 & y_2^2 & \cdot & y_2^d \\ \cdot & \cdot & \cdot & \cdot \\ y_n^1 & y_n^2 & \cdot & y_n^d \end{bmatrix} \quad (4.23)$$

where, n signifies the number of universe and d denotes the parameters.

Step 2: Initialize the parameters wormhole existence probability (WER) and travelling distance rate (TDR).

Step 3: Sort the universe (SU) and evaluate the inflation rate (fitness) of all the universe and based on the inflation rate choose the best universe.

Step 4: Normalize the inflation rate (NI) of the space.

Step 5: Update the parameters WER and TDR with the following set of equations:

$$WER = min + l * \left(\frac{max-min}{L} \right) \quad (4.24)$$

$$TDR = 1 - \frac{l^{1/p}}{L^{1/p}} \quad (4.25)$$

where, min and max are the constant parameters, l and L are the current iteration and the maximum iterations, p is a constant which defines the exploitation accuracy of the algorithm.

Step 6: Use a roulette wheel selection method to move objects from a high inflation rate universe to a low inflation rate universe.

Step 7: Transfer the objects from universe to best universe through wormhole tunnels.

Step 8: Determine the universes that go beyond the definite search space and reinitialize their positions.

Step 9: Repeat step 3 until the stopping criteria is assured.

Step 10: The position of best universe epitomizes the universal optimal solution.

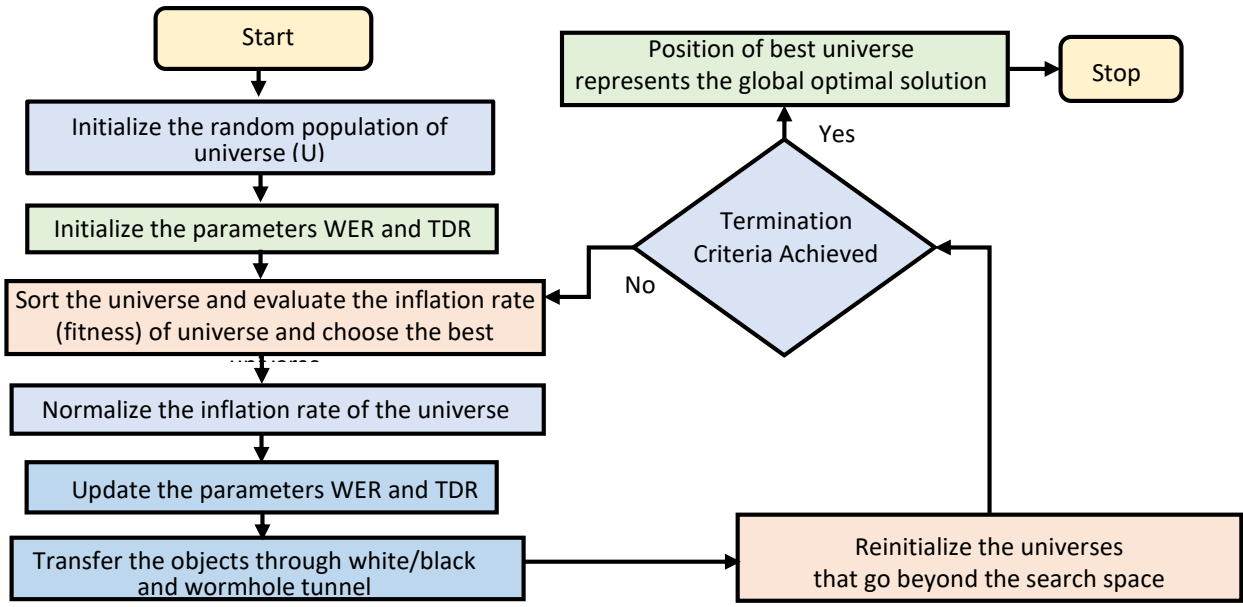


Figure 4.7 Flow chart of MVO algorithm

4.6 Results and Discussions

All these metaheuristic techniques have been implemented to design PID and FOPID controllers for the trajectory tracking of a two-link robotic manipulator. The fixed step trajectory of step value 2 has been considered for tracking. A Weighted sum of the integral absolute error IAE of both links has been considered as the fitness function as shown in eq. (4.26)

$$f = w_1 * \int e_1(t)dt + w_2 * \int e_2(t)dt \quad (4.26)$$

w_1 and w_2 are the weights assigned to the IAE of both links with the values of 0.5 each and $e = \theta - \theta_d$, the difference between the actual and reference or desired trajectory. These metaheuristic algorithms are stochastic in nature they may yield different results for each run, thus statistical analysis is required. The statistical analysis has been performed by running each algorithm 10 times.

Some parameters need to be considered for the implementation of the above-mentioned metaheuristic algorithms to optimize the control techniques; these parameters are listed in table 4.1. The upper and lower bounds have been defined.

Dimensions for PID are 6 and for FOPID it has been taken as 8. In PID controller we need to optimize the controller gains (K_{P1} , K_{I1} , K_{D1} and K_{P2} , K_{I2} , K_{D2}) and in FOPID the controller gains (K_{P1} , K_{I1} , K_{D1} , K_{P2} , K_{I2} , K_{D2} , μ , λ) thus have dimensions 6 and 8 respectively. The number of search agents has been considered as 30 and the number of iterations is 100.

Table 4.1 Parameters for metaheuristic algorithms

S. No	Algorithm parameters	PID Values	FOPID Values (λ_i, μ_i)
1	Dimensions	6	8
2	Upper Bounds	[500,300,200,500,300,200]	[500,300,200,500,300,200,1,1]
3	Lower Bounds	[0.5,0.5,0.5,0.5,0.5,0.5]	[0.5,0.5,0.5,0.5,0.5,0.5,0,0]
4	No of search agent	30	30
5	No of iterations	100	100
6	No of runs	10	10

Every run returns the optimum value of the controller gains with minimum fitness function. For these values, the lowest fitness value is considered as the optimum solution. Table 4.2 presents the gains of the PID controller for both links, the error value for each link, and the optimum fitness value. Statistical analysis has been carried out using statistical parameters like minimum, maximum, mean, median, and standard deviation of fitness value. Table 4.3 presents this statistical analysis expressing the values of these parameters. It is clear from these values the MFO has shown zero standard deviation. The standard deviation shown by MVO and GWO is 0.0001. WOA has shown a high standard deviation of 0.00165. Thus, the performance of MFO is found to be superior as compared to other algorithms.

Table 4.2. PID Controller gains, error values and objective function values for various algorithms

S. No.	Algorithm	Link 1 PID controller gains	Link 2 PID controller gains	IAE	IAE1	Objective function
1	GWO	[499.9380, 63.5886, 30.8274]	[500, 296.0716, 81.8123]	0.1671	0.3771	0.2721
2	WOA	[500, 65.5025, 30.1539]	[500, 300, 80.7489]	0.1673	0.3772	0.2723
3	MFO	[499.9877, 64.8214, 30.2722]	[499.9898, 300, 80.9017]	0.1671	0.3770	0.2721
4	MVO	[500, 65.0899, 30.1600]	[499.5655, 300, 80.7833]	0.1671	0.3773	0.2722

Table 4.3 Statistical parameters for PID Controller

S. No.	Algorithm	Minimum	Maximum	Mean	Median	Standard Deviation
1	GWO	0.2721	0.2724	0.2722	0.2722	0.0001
2	WOA	0.2723	0.3185	0.2914	0.2930	0.0165
3	MFO	0.2721	0.2721	0.2721	0.2721	0.0000
4	MVO	0.2722	0.2724	0.2723	0.2723	0.0001

FOPID provides more design flexibility by integrating fraction operators in derivative and control mode. Therefore, each FOPID has five gains to optimize using metaheuristic algorithms. Two different FOPID controllers have been designed for tracking the trajectory manipulator because of MIMO dynamics. Table 4.4 presents the gains of FOPID controller for both links, the error value for each link, fractional term values, and the optimum fitness value.

Table 4.4 FOPID Controller gains, error values and objective function values for various algorithms.

S. No.	Algorithm	Link 1 FOPID controller gains	Link 2 FOPID controller gains	$[\lambda_i, \mu_i]$	IAE	IAE1	Objective function value
1	GWO	[8.4380, 300, 134.6961]	[492.2880, 300, 200]	[1,1]	0.02846	0.3213	0.1749
2	WOA	[222.7478, 300, 200]	[500, 300,200]	[1,0.99 99]	0.03593	0.3199	0.1779
3	MFO	[2.1608, 299.9985, 133.6903]	[499.8116, 300, 200]	[1,1]	0.02657	0.3207	0.1736
4	MVO	[0.5, 300, 132.8533]	[499.9307, 300, 199.7294]	[1,0.99 96]	0.02672	0.3209	0.1738

In FOPID the Statistical analysis has been carried out using the statistical parameters like minimum, maximum, mean, median, and standard deviation of the error values. Table 4.5 presents this statistical analysis expressing the values of these parameters

Table 4.5 Statistical parameters for FOPID Controller

S. No.	Algorithm	Minimum	Maximum	Mean	Median	Standard Deviation
1	GWO	0.1749	0.1793	0.1787	0.1791	0.0013
2	WOA	0.1779	0.1796	0.1794	0.1796	0.0005
3	MFO	0.1736	0.1784	0.1767	0.1779	0.0020
4	MVO	0.1738	0.1792	0.1770	0.1788	0.0025

It is clear from these values the MFO has given the lowest fitness value and WOA has returned the highest fitness value. Considering the fitness value, the performance of MFO is found to be superior as compared to other algorithms. A non-parametric statistical test known as Friedman’s test has been performed to evaluate the performance of these metaheuristic optimization algorithms [118]. Each of these algorithms has been assigned a Friedman ranking utilizing that final ranking has been assigned [119]. Table 4.6 shows the Friedman ranking and final ranking of every algorithm implemented on PID and FOPID controllers.

Table 4.6 Ranking of the metaheuristic algorithms on PID and FOPID controller designed according to the Friedman test.

S. No.	Algorithms	PID Controller		FOPID Controller	
		Friedman Ranking	Final Ranking	Friedman Ranking	Final Ranking
1	GWO	2.4	2	2.8	3
2	WOA	3.6	4	3.9	4
3	MFO	1	1	1.4	1
4	MVO	3	3	1.9	2

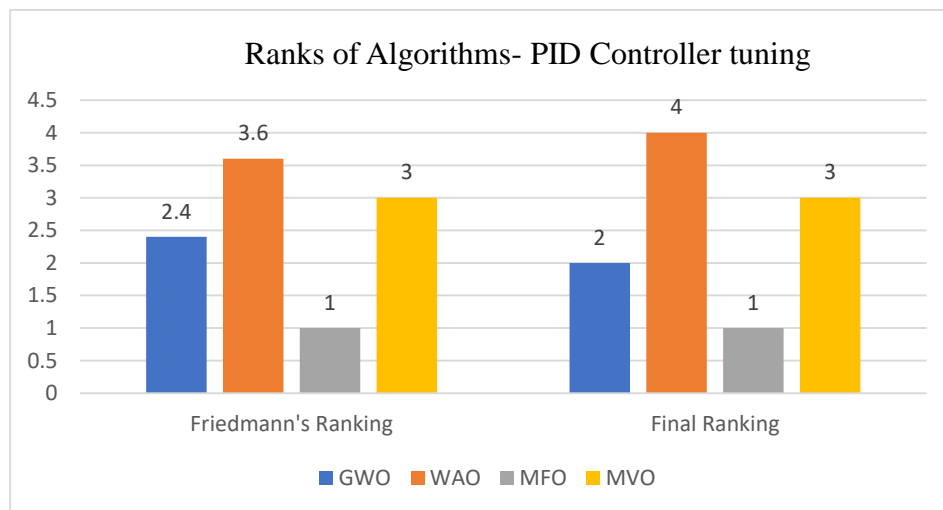


Figure 4.8 Friedman’s Ranking for PID controller

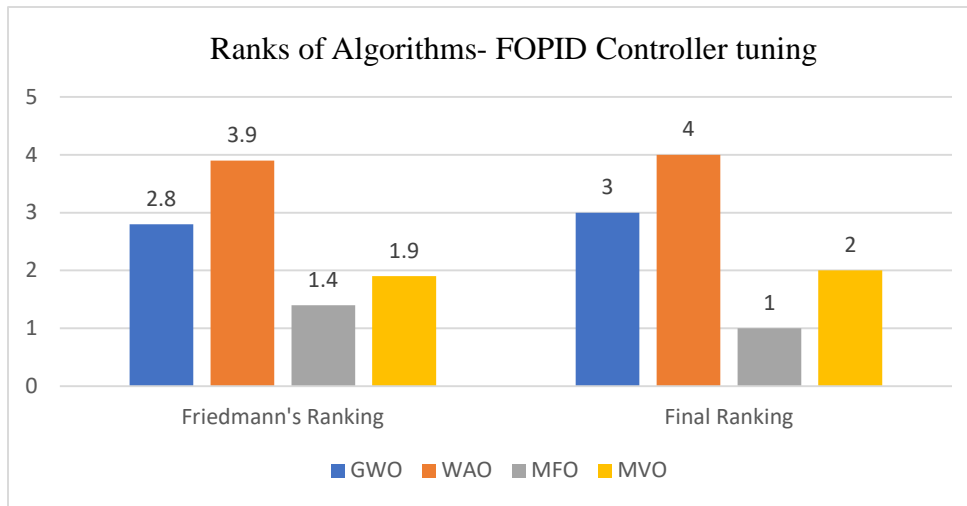


Figure 4.9 Friedman’s Ranking for FOPID Controller

Figure 4.8 and Figure 4.9 present Friedman’s ranking of the metaheuristic algorithms implemented in tuning the PID and FOPID controllers for tracking the trajectory of a robotic manipulator. The obtained Friedman’s ranking has been approximated and a final ranking has been assigned to each algorithm. It is evident from these rankings that the algorithm MFO performs the best and algorithm WOA performs the worst for both PID and FOPID control schemes. So MFO attains a rank of 1 and WOA attains 4. The performance of GWO is superior to MVO in the PID controller, thus GWO attains a rank of 2 and MVO achieves the rank of 3 as shown in Figure 4.8. The performance of MVO is superior to GWO in the FOPID controller; thus MVO attains a rank of 2 and GWO achieves the rank of 3 as shown in Figure 4.9. These ranks are clearly indicative of the performance of the algorithm on tuning the controllers to achieve the reference trajectory.

The convergence curve depicts the optimal cost function value achieved with the iterations. Figure 4.10 shows the convergence curve of these implemented metaheuristic algorithms for the PID control technique. Figure 4.11 shows the

convergence curve of these implemented metaheuristic algorithms for the FOPID control technique.

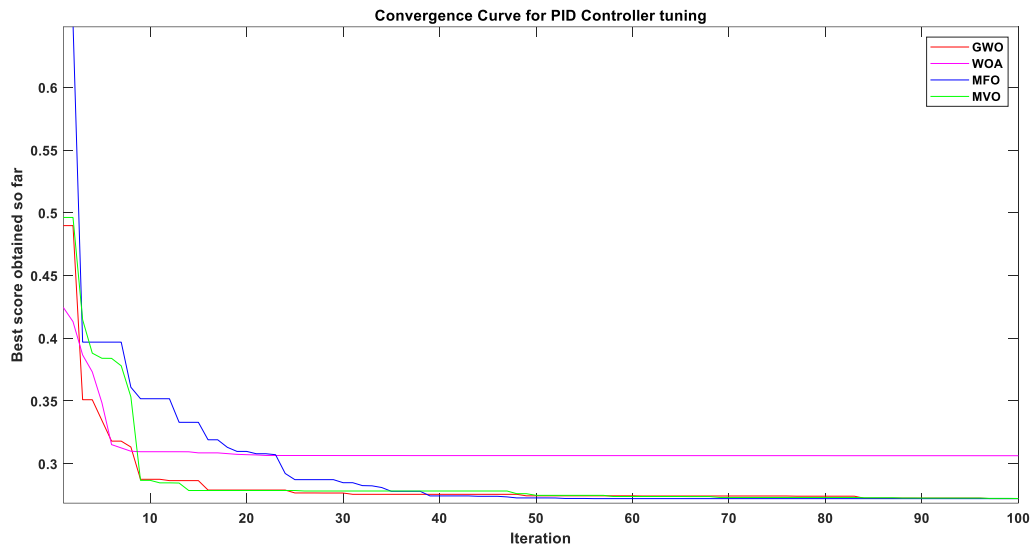


Figure 4.10 Convergence curve of the algorithms for PID controller

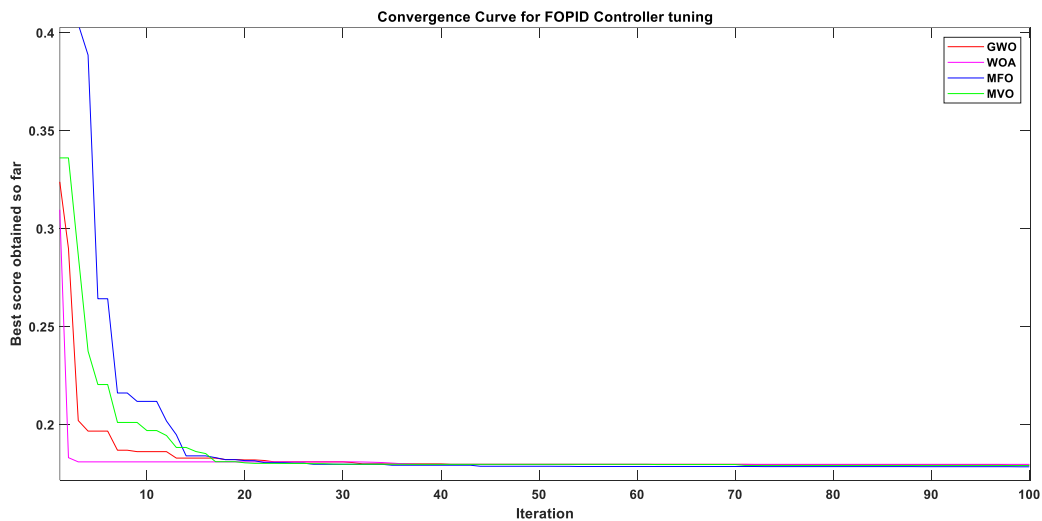


Figure 4.11 Convergence curve of the algorithms for FOPID controller

In the PID controller convergence curve, the MFO converges to the lowest fitness value while WOA converges to the highest fitness value, this signifies the best

performance of MFO and worst performance of WOA in the tracking the trajectory of a robotic manipulator using metaheuristic algorithms based PID controller. In the FOPID controller convergence curve, the MFO converges to the lowest fitness value while WOA converges to the highest fitness value, but the performance of WOA has been improved in comparison to PID control. This validates the best performance of MFO and worst performance of WOA in the trajectory control of robotic manipulators using metaheuristic algorithms based FOPID controller.

A fixed step of value 2 has been given as a reference to track the trajectories of both links of a two-link robotic manipulator. Fast settling and no overshoots in the trajectories have been the desired characteristics in the tracked trajectories of both links. Figure 4.12 shows the trajectory control by the PID controller and FOPID controller using GWO. This algorithm has tracked the reference trajectory in both control schemes, but PID has shown overshoots. It is clear from the figures that FOPID has shown a low error value in the positions of both the links as compared to the PID controller and has completely reduced overshoot in link 2 and reduced it in link 1.

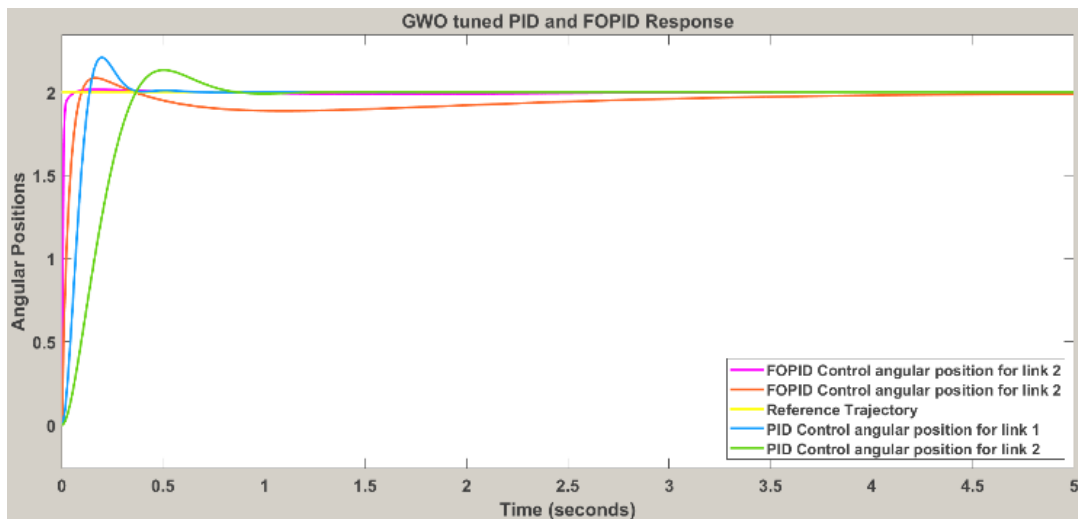


Figure 4.12 GWO tuned PID and FOPID controller response

Figure 4.13 depicts the trajectory control by the PID and FOPID controller using WOA. For link 1 PID has shown more overshoot as compared to GWO. The FOPID has reduced the overshoots. WOA has been able to track the reference trajectory but has shown a high fitness function value.

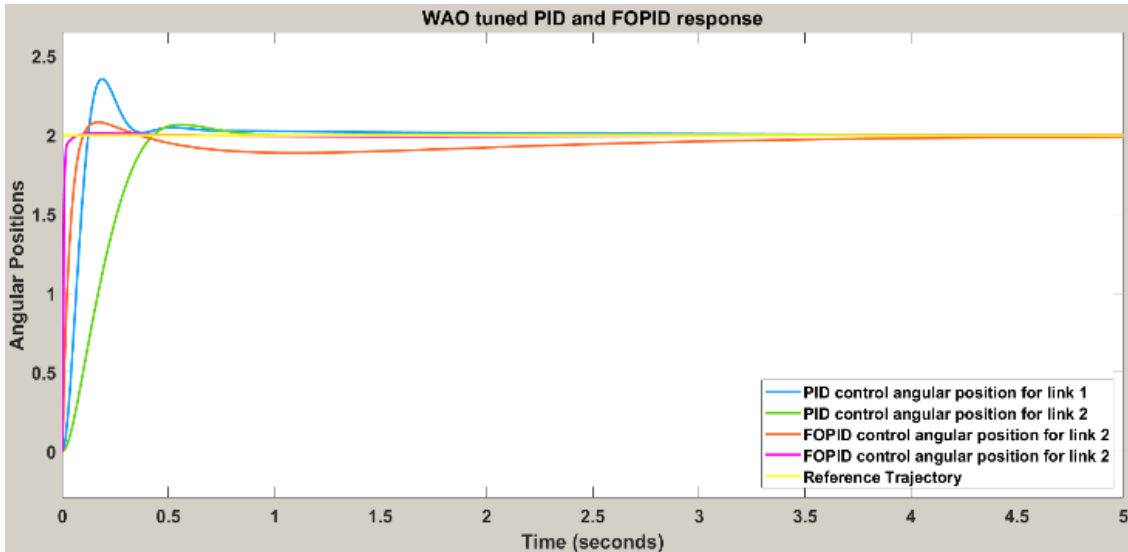


Figure 4.13 WOA tuned PID and FOPID controller response

Figure 4.14 shows the trajectory control by PID and FOPID controller using MFO and Figure 4.15 presents the trajectory control by PID and FOPID controller using MVO. The performance of these algorithms is better for FOPID as compared to PID. The overshoots in the trajectory of the links of robotic manipulators have been removed by the FOPID controller but the settling time is increased a bit for the second link while link 1 is able to settle instantly. PID and FOPID controllers are robust but gain setting for complex systems like robotic manipulator is challenging. The use of metaheuristic algorithms facilitates this problem in the design of efficient control. Considering the stochastic nature of such algorithms, different runs have been taken for finding the optimum solution. FOPID has reduced the error but increased the settling time of the angular position of the second link of the robotic manipulator. So, there is a trade-off between the error value and settling time for the angular displacement of link 2.

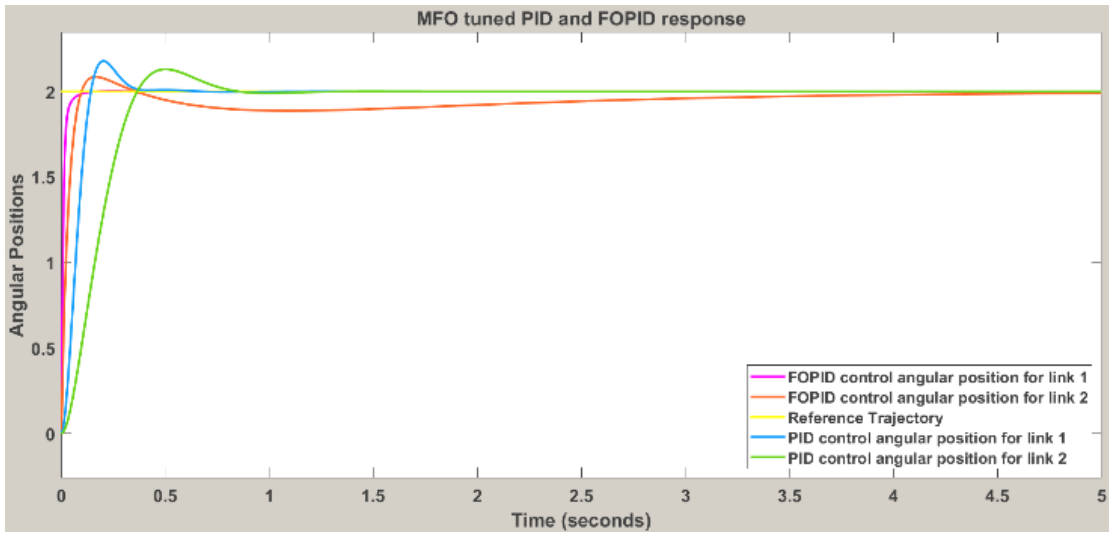


Figure 4.14 MFO tuned PID and FOPID controller response

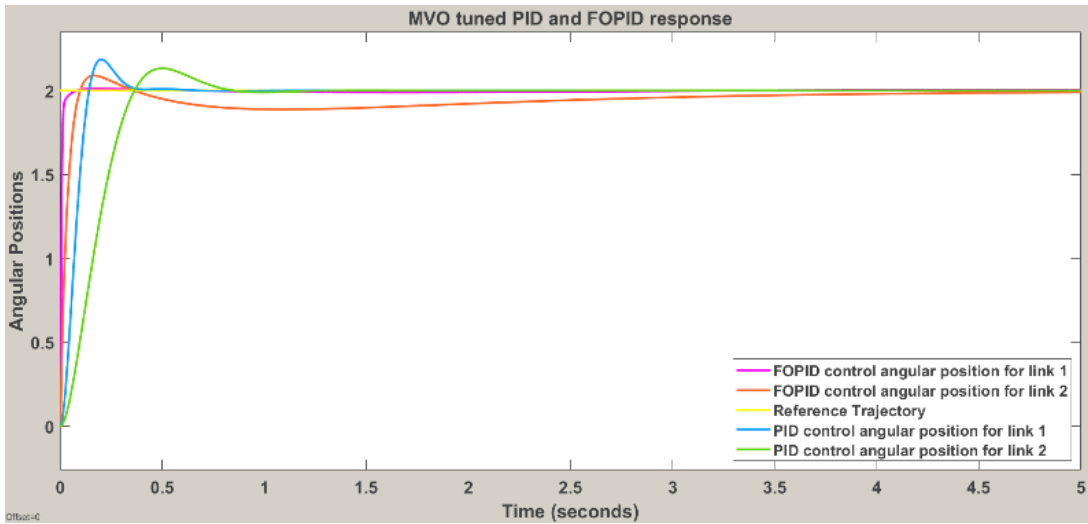


Figure 4.15 MVO tuned PID and FOPID controller response

Chapter 5

Optimization and statistical analysis of control techniques for nonlinear model

5.1 Introduction

The optimization algorithms have given more flexibility in the design of effective control schemes to the researchers. These techniques help in optimizing the control laws. Because of the stochastic nature of these algorithms statistical analysis is required to assess the effectiveness of such algorithms. In this chapter the trajectory tracking has been achieved using PID controller and following recent metaheuristic algorithms have been employed to optimize the controller parameters providing a minimum fitness function value.

- Arithmetic Optimization Algorithm (AOA)
- Atom Search optimization (ASO)
- Spotted Hyena Optimizer (SHO)
- Sooty Tern Optimization (STO)

Further, STO algorithms has been modified by combining it with PSO and a novel hybrid algorithm STOPSO has been proposed to expand the exploitation capability of STO. Each of these algorithms is inspired by a phenomenon existing in nature like swarm intelligence, mathematical operators, and the atomic structure of molecules. Taking this inspiration in consideration these algorithms can be modeled and expressed in a mathematical approach [60]. The description and mathematical approach for these algorithms and their implementation in tracking the trajectory of a double-link robotic manipulator has been presented in the next sections.

5.2 Arithmetic Optimization Algorithm (AOA)

Laith Abualigah et.al. (2021) [120] presented a novel optimization algorithm named AOA inspired by fundamental operations in mathematics. It makes use of mathematical operations like multiplication, division, addition, and subtraction's distribution pattern. The utilization of different basic arithmetic operations in unravelling the complicated arithmetic problems is the main inspiration of AOA. It is implemented in three stages: Initialization phase: Each iteration results in the best answer, starting with a collection of randomly initiated solutions. The fitness value at the t^{th} iteration is obtained using the mathematical optimizer function (MOA). In the exploration phase the division (D) and multiplication (M) mathematical operators search for the best answers. The fitness value is determined by the Math Optimizer Probability (MOP). In the exploitation phase the addition (A) and subtraction (S) mathematical operators exploit the solution for global optimum answer. The flow chart of AOA has been shown in Figure 5.1.

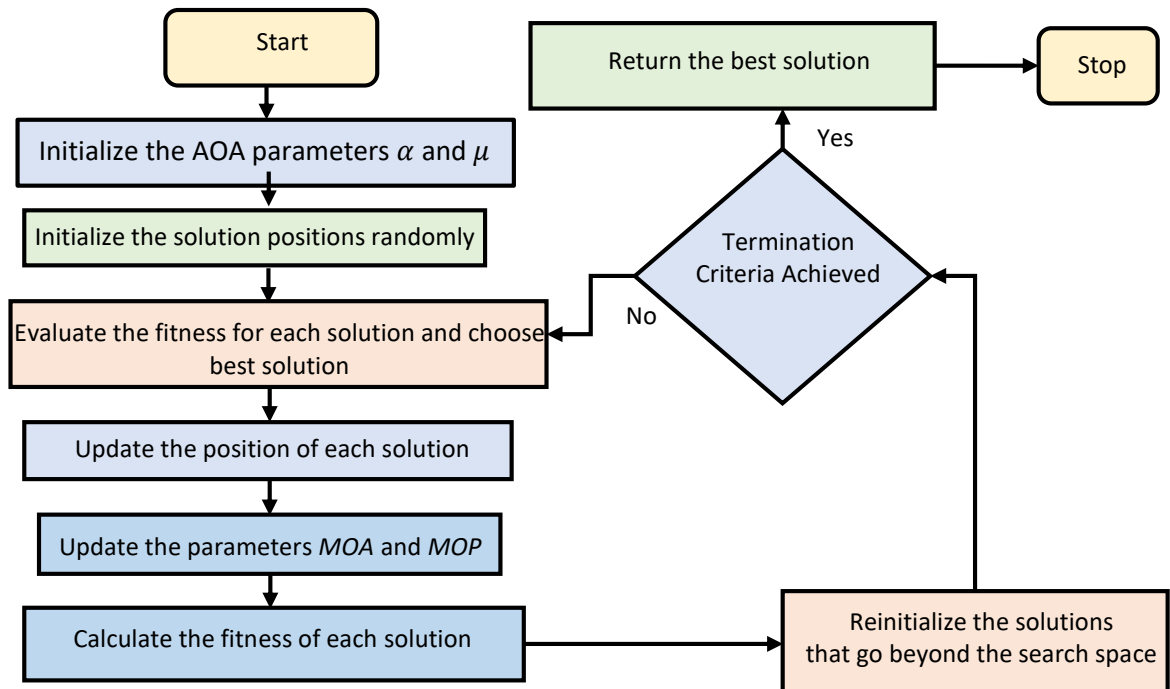


Figure 5.1 Flow chart of AOA

The mathematical modelling of AOA algorithm is outlined in the following steps.

Step 1: Initiate the AOA parameters α , μ , and the randomly generated candidate solution (X) in the search space:

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdot & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdot & x_{2,n} \\ \cdot & \cdot & \cdot & \cdot \\ x_{N,1} & x_{N,2} & \cdot & x_{N,n} \end{bmatrix} \quad (5.1)$$

where, n signifies the number of solutions.

Step 2: Appraise the fitness value of each function using the functions MOA and MOP.

Step 3: Calculate and update the MOA function.

$$MOA(C_I) = Min + C_I \left(\frac{Max - Min}{M_I} \right) \quad (5.2)$$

Where, C_I is the running iteration, M_I is the maximum iteration, Max and Min are the maximum and minimum values of MOA function.

Step 4: Calculate and update the MOP function.

$$MOP(C_I) = 1 - C_I \left(\frac{1}{M_I^\alpha} \right) \quad (5.3)$$

Where, C_I is running t iteration, M_I is maximum iteration, and α is a constant used to exploit the solutions.

Step 5: Generate the random values of variables r_1 , r_2 , r_3 in between [0, 1]. These variables determine the exploration and exploitation phase.

if $r_1 > MOA$

The exploration takes place

if $r_2 > 0.5$ then

(1) The mathematical divide operation D needs to be applied

Implement rule 1 in eq. (5.4) to update i_{th} solution

else

(2) The mathematical multiplication operator M is applied

Implement rule 2 in eq. (5.4) to update i_{th} solution

end if

else

The exploitation takes place

if $r_3 > 0.5$ then

(1) The mathematical subtraction operator S is applied

Implement rule 1 in eq. (5.5) to update i_{th} solution

else

(2) The mathematical addition operator A is applied

Implement rule 2 in eq. (5.5) to update i_{th} solution

end if

end if

end for

$C_I = C_I + 1$

end while

Step 6: Update the solutions using the following set of equations:

$$x_{i,j}(C_I + 1) = \left\{ \begin{array}{l} \text{best}(x_j) \div (MOP + \varepsilon) \times ((UB_j - LB_j) \times \mu + LB_j), \quad r_2 < 0.5 \\ \text{best}(x_j) \times MOP \times ((UB_j - LB_j) \times \mu + LB_j), \quad \text{otherwise} \end{array} \right\} \quad (5.4)$$

$$x_{i,j}(C_I + 1) = \left\{ \begin{array}{l} \text{best}(x_j) - MOP \times ((UB_j - LB_j) \times \mu + LB_j), \quad r_3 < 0.5 \\ \text{best}(x_j) + MOP \times ((UB_j - LB_j) \times \mu + LB_j), \quad \text{otherwise} \end{array} \right\} \quad (5.5)$$

Where the values UB_j and LB_j is the upper and lower bound value of the j_{th} position.

Step 7: Reiterate step 6 to either the maximum iterations are reached, or closure criteria is satisfied.

Step 8: The candidate's best solution represents the global optimal solution.

5.3 Atom Search optimization (ASO)

Weiguo Zhao *et.al.* (2019) [121] proposed a novel, physics-based metaheuristic algorithm ASO inspired by dynamics of basic molecular structure in atoms. ASO begins the optimization process by producing a set of solutions randomly. During each iteration the positions and velocities of each atoms including the best atom, is updated. There are two forces that cause the atom's acceleration. The first one is L-J potential's interaction that is the vector summation of the attraction and repulsion forces between atoms. Second is the bond-length potential's constraint force that is defined as the difference in weighted positions between every atom and the optimum atom. The entire updating and computation process is carried out interactively up till a stopping criteria is fulfilled. The best atom's fitness and position represents the optimum value. An initial set of atoms solutions with their velocities are produced at random in an ASO. According to its mass, each atom's location within the search area points to a solution. As per the distance between each atom in the population there is tendency of either attracting or repelling one another, which will cause the lighter atoms to gravitate towards the heavier ones. The flow chart of ASO has been shown in Figure 5.2.

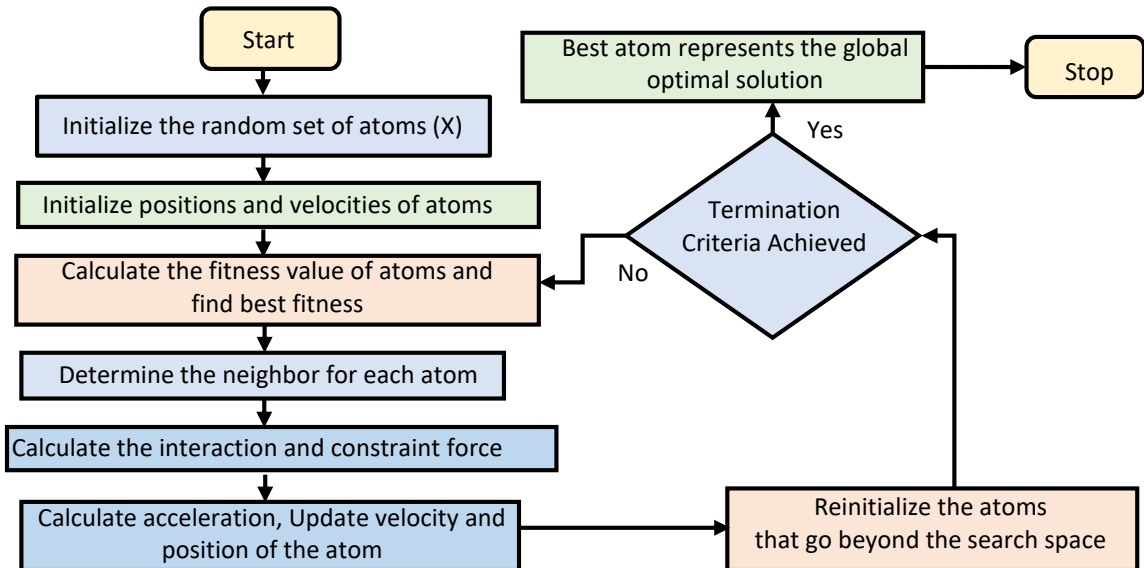


Figure 5.2. Flow chart of ASO

The mathematical modelling of the ASO is outlined in the following steps.

Step 1: Initiate a random set of atoms solution (X) within the search space with velocity v .

$$x_i = (x_i \dots \dots \dots x_n) \quad (5.6)$$

Where n denotes the number of atoms.

Step 2: Evaluate the fitness value of each function using the following equations.

$$F_{ij}(t) = -\eta(t)[2(h_{ij}(t))^{13} - (h_{ij}(t))^7] \quad (5.7)$$

Where $\eta(t)$, depth function that adjusts the attractive or repulsive force region, defined as following.

$$\eta(t) = \alpha(1 - \frac{t-1}{T})^3 e^{-\frac{20t}{T}} \quad (5.8)$$

Where α is the depth weight and T represents the maximum number of iterations considered.

Step 3: Compare the fitness value obtained with $F_{it}best$.

If $F_{it}i < F_{it}best$ then

$$F_{it}i = F_{it}best$$

$$X_{best} = X_i$$

End if

Step 4: Calculate the mass equations and determine the neighbors using the following equations.

$$M_i(t) = e^{-\frac{F_{it}i - F_{it}best}{F_{it}worst - F_{it}best}} \quad (5.9)$$

$$m_i(t) = \frac{M_i(t)}{\sum_{j=1}^N M_j(t)} \quad (5.10)$$

$$P(t) = N - (N - 2)\sqrt{\frac{t}{T}} \quad (5.11)$$

Step 5: Calculate the forces (interaction and constraint) using the following equations.

$$F_i(t) = \sum_{j \in K_{best}} rand_j F_{ij}(t) \quad (5.12)$$

$$G^d_i(t) = \gamma(t)(x_{best}^d(t) - x_i^d(t)) \quad (5.13)$$

$$\gamma(t) = \beta e^{-\frac{20t}{T}} \quad (5.14)$$

Step 6: Apprise the positions and velocities of atoms using the following equations.

$$v_i^d(t+1) = rand_i^d v_i^d(t) + a_i^t(t) \quad (5.15)$$

$$x_i^d(t+1) = x_i^d + v_i^t(t) \quad (5.16)$$

Step 7: Reiterate till step 6 until the dissolution criteria is satisfied.

Step 8: The atom's best solution represents the global optimal solution.

5.4 Spotted Hyena Optimizer (SHO)

Gaurav Dhiman et.al. (2017) [122] presented SHO, a metaheuristic algorithm simulating the hunting strategy, social interaction and accommodating behavior of spotted hyenas. Searching for prey, surrounding, and confronting prey are the three basic actions of SHO, which are represented mathematically. The intended action or goal in an encircling activity is the best solution, while the remaining search agents can alter their placements in view of the discovered best solution. Spotted hyenas are able to locate and encircle their prey. Spotted hyenas have a tendency to detect the location of their and encircle it. Target prey's location is considered as the current optimum solution. According to this current best solution, other elements update their positions. The spotted hyenas use the best search agent as a benchmark for their hunting approach, and they adjust the placements of other search agents in a cluster of optimum solutions. The process flow of ASO is shown in Figure 5.3. The mathematical description and modelling of the SHO algorithm is outlined in the below steps.

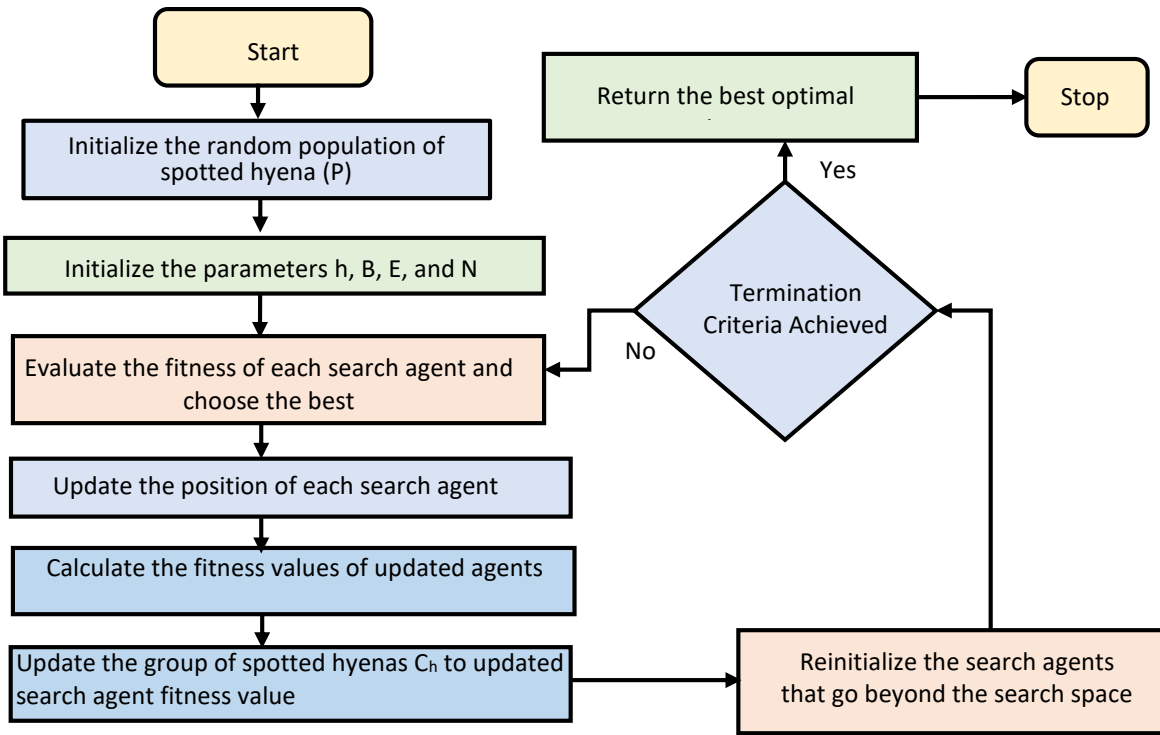


Figure 5.3. Flow chart of SHO

Step 1: The position of spotted hyenas (search agents) is initialized randomly in the search space:

$$P_i = (p_i \dots \dots \dots p_n) \quad (5.17)$$

where n represents the space dimension. $i = 1, 2, \dots, n$

Step 2: Determine the position of prey and spotted hyenas using the following equations. The coefficients h , B , N , and E are initialized.

$$D_h = |G \cdot P_p(j) - P(j)| \quad (5.18)$$

$$P(j + 1) = P_p(j) - E \cdot D_h \quad (5.19)$$

D_h - Prey and spotted hyena's distance

G, E - Coefficient vectors

j - Current iteration

P_p, P - Position of prey and spotted hyena.

$$G = 2 \cdot r_1 \quad (5.20)$$

$$E = 2h \cdot r_2 - h \quad (5.21)$$

r_1, r_2 are random vectors in between [0,1]

$$h = 5 - \left[Itr \times \left(\frac{5}{M_{Itr}} \right) \right] \quad (5.22)$$

To maintain a proper equilibrium between two main processes of optimization i.e. the exploration and exploitation, h is decreased linearly from a constant value to 0.

Step 3: The fitness of all spotted hyenas is evaluated, and the search agent closed to optimum or nearest to prey is explored. To obtain satisfactory performance the clusters are defined using the following equations.

$$C_h = P_k + P_{k+1} + \dots P_{k+N} \quad (5.23)$$

$$N = Count_n(P_h, P_{h+1}, P_{h+2}, \dots \dots (P_{h+M})) \quad (5.24)$$

C_h is a cluster formed by a group of spotted hyenas

N is a number of spotted hyenas.

P_h is the position of the first optimal-spotted hyena.

P_k is the position of various other spotted hyenas.

Step 4: Apprise the positions of the search agents using the below equation.

$$P(j + 1) = \frac{C_h}{N} \quad (5.25)$$

Step 5: Reinitiate the position of spotted hyenas that go beyond the defined space.

Step 6: Repeat till step 4 if termination criteria is not satisfied.

Step 7: The best position of spotted hyenas represents the global optimal solution.

5.5 Sooty Tern Optimization (STO)

Gaurav Dhiman et.al. (2019) [123] proposed a novel bio-inspired stochastic optimization algorithm STO that replicates the sooty tern's natural movement and attacking patterns. Sooty terns migrate in groups during migration. These sooty terns have a specific behavior that avoids collision among them and provides

guidance while migrating. The sooty terns' starting locations are distinct to prevent collisions. A sooty tern having low fitness as compared to other sooty terns leads a group of individual's sooty terns in the direction of the sooty tern with the best survival. Other sooty terns adjust their starting positions based on the position of the best fit sooty tern. Migration and attacking manners are the two main components of the STO algorithm. In the exploration behavior sooty tern satisfies the conditions like collision avoidance utilizing a function S_A , Convergence in the best neighbor's direction and updating their positions as per the optimum search agent. In the attacking approach (exploitation) while attacking on the prey, sooty terns generate the spiral behavior in the air. Figure 5.4 presents the flow chart of STO.

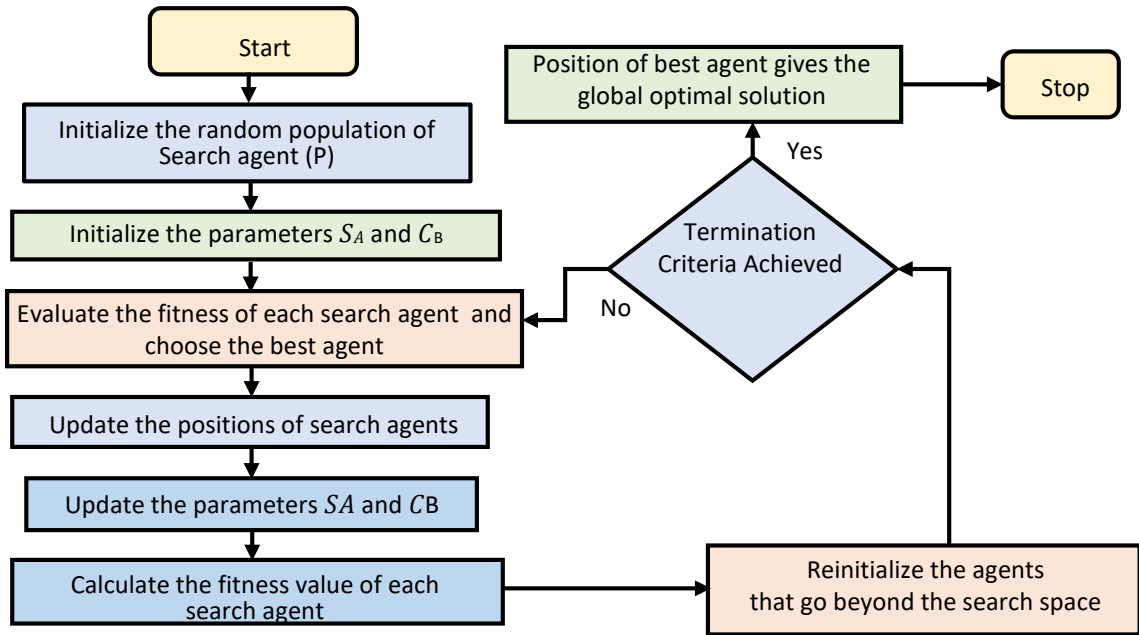


Figure 5.4. Flow chart of STO

The migration and attacking pattern of sooty terns can be mathematically modeled by the following steps.

Step 1: The position of sooty terns (search agents) is initialized randomly in the search space:

$$\vec{Y}_{st} = (y_1, y_2, \dots, y_n) \quad (5.26)$$

where n represents the space dimension. $i = 1, 2, \dots, n$

Step 2: Determine the position of sooty terns and initialize the coefficients and parameters S_a and C_b .

$$\vec{C}_{st} = S_a \times \vec{Y}_{st}(z) \quad (5.27)$$

$$S_a = C_f - \left(z \times \left(\frac{C_f}{M_{itr}} \right) \right) \quad (5.28)$$

\vec{C}_{st} position of sooty terns having collision avoidance.

$\vec{Y}_{st}(z)$ - current position of sooty terns.

C_f – a variable used to avoid collision is decreased to 0.

$Z = 1, 2, \dots, M_{itr}$.

Step 3: Evaluate the fitness of every search agent using the following equations.

$$\vec{M}_{st} = C_b \times \left(\vec{Y}_{bst}(z) - \vec{Y}_{bst}(z) \right) \quad (5.29)$$

$$C_b = 0.5 \times r \quad (5.30)$$

\vec{M}_{st} expresses the search agent's location, C_b is a constant used to improve the exploration, r is a random number between [0,1].

$$\vec{D}_{st} = \vec{C}_{st} + \vec{M}_{st} \quad (5.31)$$

\vec{D}_{st} shows the distance between search agent and best fit agent.

Step 4: Each search agent's position can be updated using the following equations.

$$x_1 = r_{adius} \times \sin(l) \quad (5.32)$$

$$x_2 = r_{adius} \times \cos(l) \quad (5.33)$$

$$x_3 = r_{adius} \times l \quad (5.34)$$

$$r = u \times e^{kv} \quad (5.35)$$

r_{adius} – radius of spiral turn

l – variable in the range of $[0 \leq k \leq 2\pi]$

u, v – constant used to define the spiral shape.

$$\vec{Y}_{st}(z) = \left(\vec{D}_{st} \times (x_1 + x_2 + x_3) \right) \times Y_{bst}(z) \quad (5.36)$$

Step 5: Update the parameters S_a and C_b .

Step 6: Repeat steps 2 to 6 until the stopping criterion is satisfied.

Step 7: Best position of the sooty tern gives the optimal solution.

5.6 Novel Hybrid STOPSO Algorithm.

5.6.1 Novelty of Work

The main contribution of the proposed study can be described as follows:

- For controlling the trajectory of a robotic manipulator, a hybrid algorithm STOPSO algorithm is proposed.
- By combining the exploitation capacity of PSO, the STO's exploitation capability is considerably enhanced. As a consequence, controller parameters are convergent to actual values with the least amount of error.
- Convergence analysis, robustness, reliability, and statistics analysis are the parameters to evaluate the trajectory tracking performance of proposed STOPSO algorithm for a robotic manipulator and compared with the previous algorithms existing in the literature.

5.6.2 Hybrid STOPSO Algorithm.

This section describes the proposed hybrid STOPSO algorithm briefly. The performance of any metaheuristic algorithm is dependent on its capabilities of exploring the solutions discovering the global optimal solution. As per NFL theorem none of the metaheuristic algorithms can offer the best solution for every challenging and complex problem. Some algorithms have a tendency to get stuck in local best solution while some algorithms have a poor rate of convergence. Maintaining a balance between exploration and exploitation for the optimization algorithms is a very difficult task. STO have high exploration abilities, and sooty

terns do this by adjusting their positions in relation to the positions of other birds to prevent collisions as they hunt for the optimal solution. In the exploitation, algorithms have tendency to get detained in local optimal solution because sooty terns create a spiral path in air for attacking the prey. PSO has better exploitation capability and poor exploration capability. So, by integrating the STO algorithm with the PSO algorithm, the exploitation potential of STO can be increased by combining the qualities of two algorithms in the hybrid form. The STO is employed initially to identify the optimal solution, and then the PSO algorithm's exploitation capabilities are used to further enhance the outcomes and get the optimal solution overall. The proposed hybrid STOSPO algorithm has been mathematically modeled as follows:

Step 1: Set the sooty terns' initial positions in the search area at random.

$$\vec{X}_s = (\vec{x}_1, \vec{x}_2, \dots \dots \dots \dots \dots \dots \vec{x}_n) \quad (5.37)$$

where, n signifies the space dimension.

Step 2: Initialize the velocities of search agents in the random search space.

$$\vec{V}_s = (\vec{v}_1, \vec{v}_2, \dots \dots \dots \dots \dots \dots \vec{v}_n) \quad (5.38)$$

Step 3: The position of best sooty tern (\vec{x}_{bs}) that indicates the best search agent, is determined by evaluating the fitness of all search agents in terms of minimization or maximization, respectively.

Step 4: The parameters S_A , C_B , w are initialized that permits the search agents to travel in the search space. These terms are defined as follows:

$$S_A = C_f - \left(z * \left(\frac{C_f}{Max_{iterations}} \right) \right) \quad (5.39)$$

$$w = w_{min} - (w_{max} - w_{min}) * \frac{iter}{Max_{iterations}} \quad (5.40)$$

where, w_{min} , w_{max} are the minimum and maximum value of inertia weight, C_f is the controlling variable that is decreased linearly from C_f to zero, $iter$ is the current iteration and $Max_{iterations}$ is the maximum number of iterations.

$$z = 0, 1, 2, 3, \dots \dots \dots \dots \dots \dots, Max_{iterations}.$$

$$C_B = 0.5 * R_{and} \quad (5.41)$$

where, R_{and} is the random number in the range [0,1].

Step 5: Using the following equations, search agents' positions are updated:

$$x' = R_{adious} * \sin(i) \quad (5.42)$$

$$y' = R_{adious} * \cos(i) \quad (5.43)$$

$$z' = R_{adious} * i \quad (5.44)$$

$$r = u * e^{kv} \quad (5.45)$$

where, R_{adious} is the radius of the spiral movement, i is the variable ranging $[0 \leq k \leq 2\pi]$, u and v are the constant terms.

$$\vec{C}_s = S_A * x_s \quad (5.46)$$

$$\vec{M}_s = C_B * (\vec{x}_{bs} - \vec{x}_s) \quad (5.47)$$

$$\vec{D}_s = \vec{C}_s + \vec{M}_s \quad (5.48)$$

$$\vec{x}_s = (\vec{D}_s * (x' + y' + z')) * \vec{x}_{bs} \quad (5.49)$$

Step 6: The velocity of search agents is adjusted depending on the position of the optimum search agent using the following equations:

$$\vec{V}_s(iter + 1) = w * \vec{V}_s(iter) + c_1 * r_1 * (\vec{x}_{bs} - \vec{X}_s(iter)) \quad (5.50)$$

where, c_1 is acceleration parameter and r_1 is the random number ranging in [0, 1]

Step 7: Apprise the position of search agents as follows:

$$\vec{X}_s(iter + 1) = \vec{X}_s(iter) + \vec{V}_s(iter + 1) \quad (5.51)$$

Step 8: S_A and C_B the dynamic terms are updated.

Step 9: The search agents whose positions are going beyond the search space need to be reinitialized.

Step 10: If the termination requirements, such as the minimal error or total number of iterations, are met, the algorithm is ended. Instead, repeat step (3) (9).

Step 11: The best search agent's position (\vec{x}_{bs}) denotes the global optimal solution.

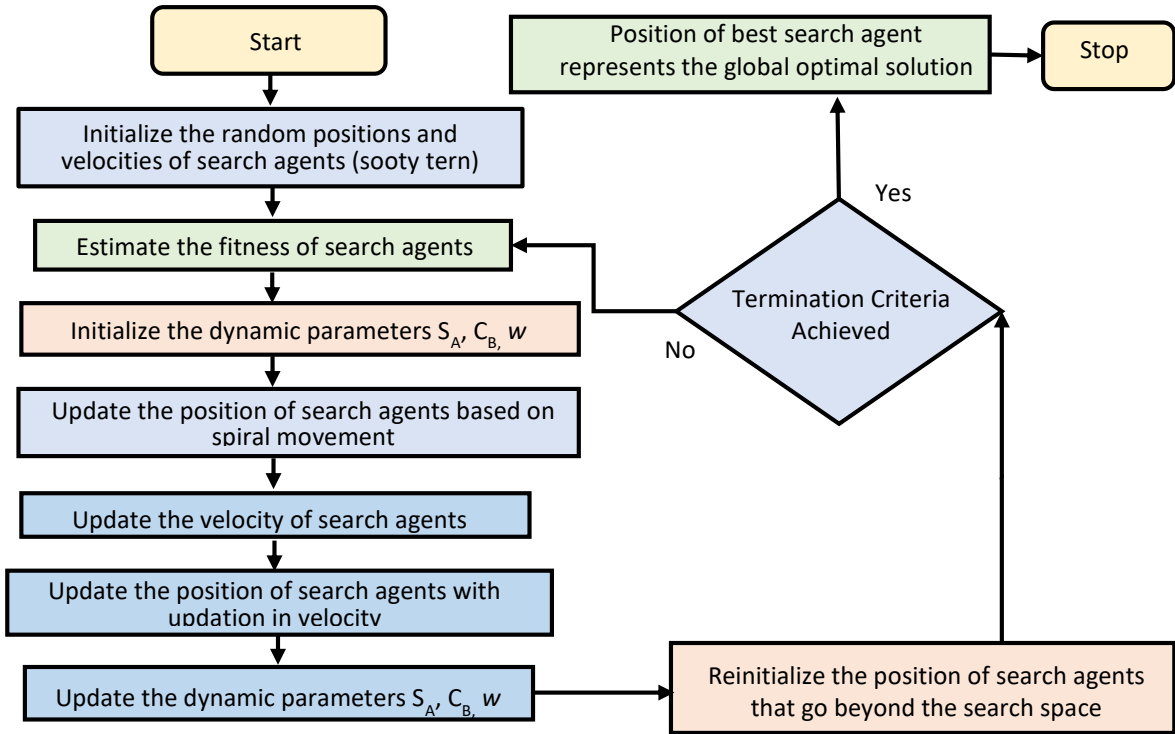


Figure 5.5 Flow chart of proposed hybrid STOPSO algorithm

5.7 Results and Discussion

For trajectory tracking of a nonlinear robotic manipulator having two links using PID controller, all the above metaheuristic algorithms (ASO, AOA, STO, SHO, and Hybrid STOPSO) have been employed and tuned the controller parameters to an optimum value. Because the robotic manipulator's MIMO dynamics two distinct PID controllers have been designed. Each of these techniques has optimized the cost function and successfully tracked the reference trajectory. The weighted sum of ITAE shown in eq. (5.52) has been considered as the performance index.

$$f = w_1 * \int e_1(t)tdt + w_2 * \int e_2(t)tdt \quad (5.52)$$

w_1 and w_2 are the weights assigned to fitness of both the links having values 0.5. The aim of implementation of these algorithms is to tune the PID controller for reference trajectory tracking. Thus, these algorithms return the optimum controller gains and minimized errors and fitness value. For tracking a trajectory, a reference

is required, hence a cubic polynomial trajectory as shown in eq. (5.53) has been considered as the reference.

$$\theta(t) = a_0 + a_1t + a_2t^2 + a_3t^3 \quad (5.53)$$

Figure 5.6 and Figure 5.7 shows the reference trajectory and trajectory generation in SIMULINK.

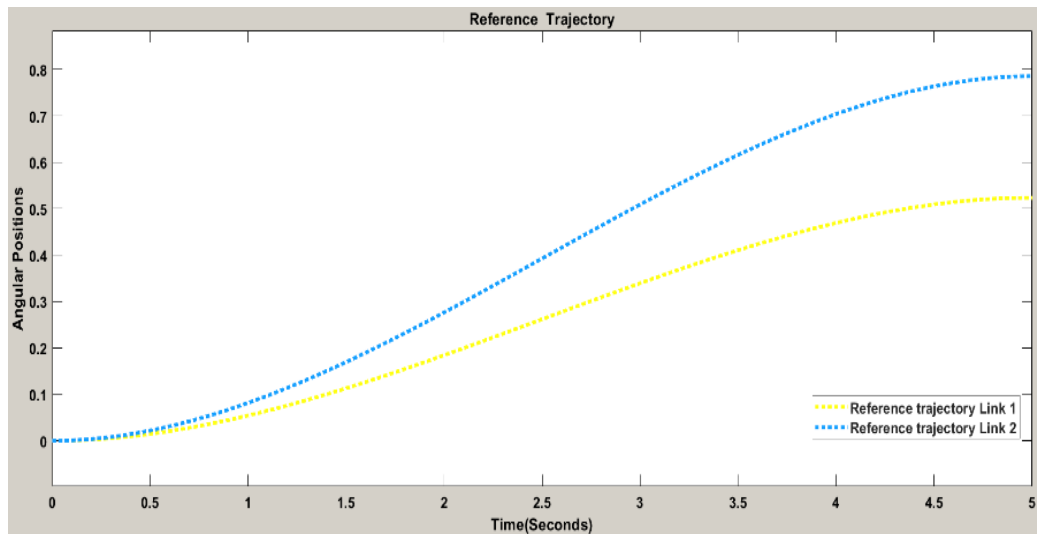


Figure 5.6. Polynomial Reference trajectory

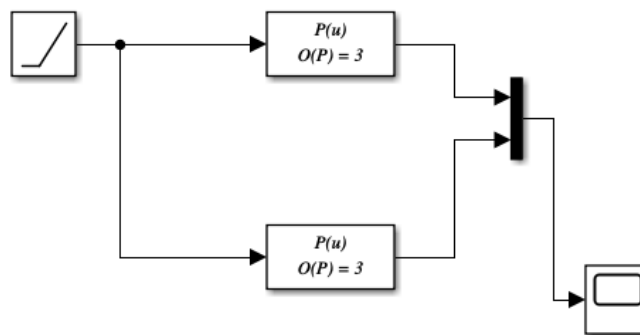


Figure 5.7 Reference trajectory generation in SIMULINK

Table 5.1 below presents the parameters considered for the implementation of the above metaheuristic algorithms. Each of these techniques has 30 search agents, and

100 iterations. Two separate PID controllers have been designed so 6 gain parameters need to tune thus, the dimension in the algorithm has been considered as 6. The upper and lower bounds have been assigned to each of the gain parameters.

Table 5.1 Values of the parameters considered for simulation

S.No	Algorithm parameters	Description	PID Values
1	Dimensions	Number of variables	6
2	Upper Bounds	Upper constraint on gain	[200,100,50,100,100,10]
3	Lower Bounds	Lower constraint on gain	[2, 2, 2, 2, 2, 2]
4	No of search agent	Population size	30
5	No of iterations	Iterations taken	100
6	Number of Runs	Number of algorithm's run	10

Every run returns the optimum value of the controller gains with minimum fitness function. For these values, the lowest fitness value is considered as the optimum solution. Table 5.2 presents the PID controller's gain for both links, the error value for each link, and the optimum fitness value.

Table 5.2 Controller gains and objective function values for metaheuristic algorithms

S. No.	Algorithm	Link 1 PID controller gains	Link 2 PID controller gains	Objective function value
1	ASO	[197.08264,99.8055, 3.3841]	[95.4870,92.845, 4.636]	0.04970
2	AOA	[200, 100, 2]	[100, 100, 8.74633118938909]	0.04607
3	STO	[200, 100, 2]	[99.9750, 90.9847, 8.8450]	0.04573

4	SHO	[186.648660502925, 100, 2]	[100, 100, 2]	0.04944
5	Hybrid STOPSO	[200, 100, 2]	[100, 100, 8.80909]	0.04541

Because of the stochastic nature of such metaheuristic algorithms, a statistical analysis has been performed by running every algorithm 10 times. Statistical analysis has been carried out using a measure of central tendencies. Table 5.3 presents these measures obtained during the statistical analysis. The ASO algorithm has shown a minimum fitness value of 0.04970 and a standard deviation of 0.09732, the AOA algorithm has shown a minimum fitness of 0.04607 with a standard deviation of 0.06423. The algorithm STO has shown an improved fitness value of 0.04573 and a standard deviation of 0.08570. The SHO has shown a minimum value of 0.0494 with the highest standard deviation of 0.28495. Further to improve the controller's performance, the proposed STOPSO algorithm has been implemented that improved the fitness value with the value of standard deviation as 0.0002. This statistical analysis validates the performance of these algorithms and shows that the hybrid STOPSO has performed the best and returned the minimum fitness value.

Table 5.3 Statistical Analysis of the fitness function in 10 runs.

S.No	Algorithm	Minimum	Maximum	Mean	Median	Standard Deviation
1	ASO	0.04970	0.05264	0.05126	0.05105	0.09732
2	AOA	0.04607	0.04821	0.04800	0.04822	0.06423
3	STO	0.04573	0.04822	0.04659	0.04629	0.08570
4	SHO	0.04944	0.05930	0.05439	0.05435	0.28495
5	Hybrid STOPSO	0.04541	0.0461	0.04601	0.0460	0.0002

A non-parametric statistical test known as Friedman's test has been carried out to assess the performance of various metaheuristic optimization algorithms. The obtained Friedman's ranking has been approximated and a final ranking has been assigned to each algorithm.

Table 5.4 presents each algorithm used on PID controllers along with its final ranking and Friedman ranking. As per this ranking the hybrid STOPSO performs the best thus, attains a rank of 1 and SHO performs the worst and attains a rank of 5. The algorithm STO achieves a rank of 2 followed by AOA rank 3 and ASO having rank 4.

Table 5.4 Ranking of the metaheuristic algorithms on PID controller designed according to the Friedman's Test

Algorithm	Friedman's Ranking	Final Ranking
ASO	4.3	4
AOA	2.8	3
SHO	4.7	5
STO	2	2
Hybrid STOPSO	1.1	1

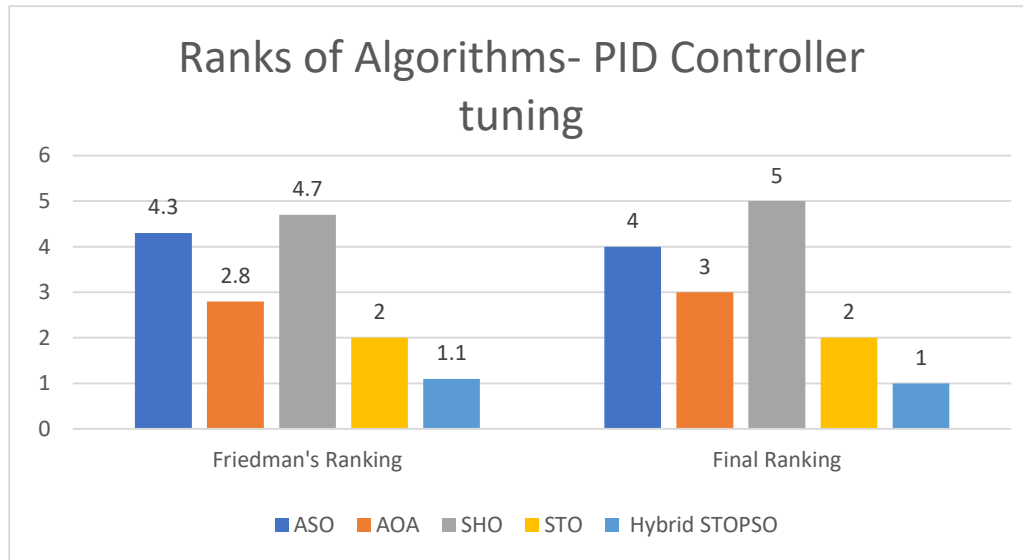


Figure 5.8 Friedman’s ranking of the metaheuristic algorithms on PID controller

Evidently the proposed algorithm hybrid STOPSO gives the best performance and SHO performs the worst in robotic manipulator’s trajectory control. The SHO achieves the rank of 5 and shows the poor statistical value of standard deviation and substantially high fitness value.

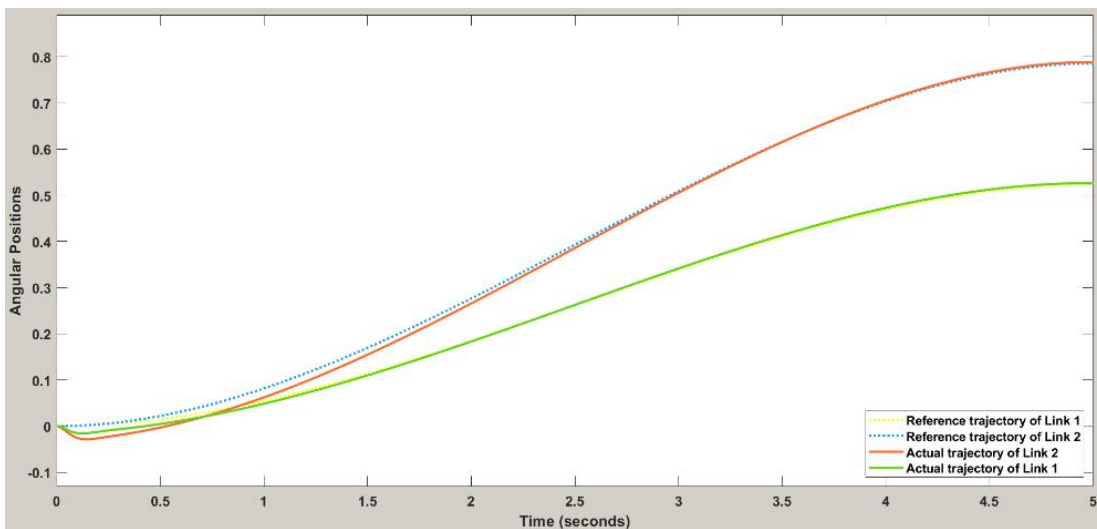


Figure 5.9 Trajectory tracking using ASO tuned PID

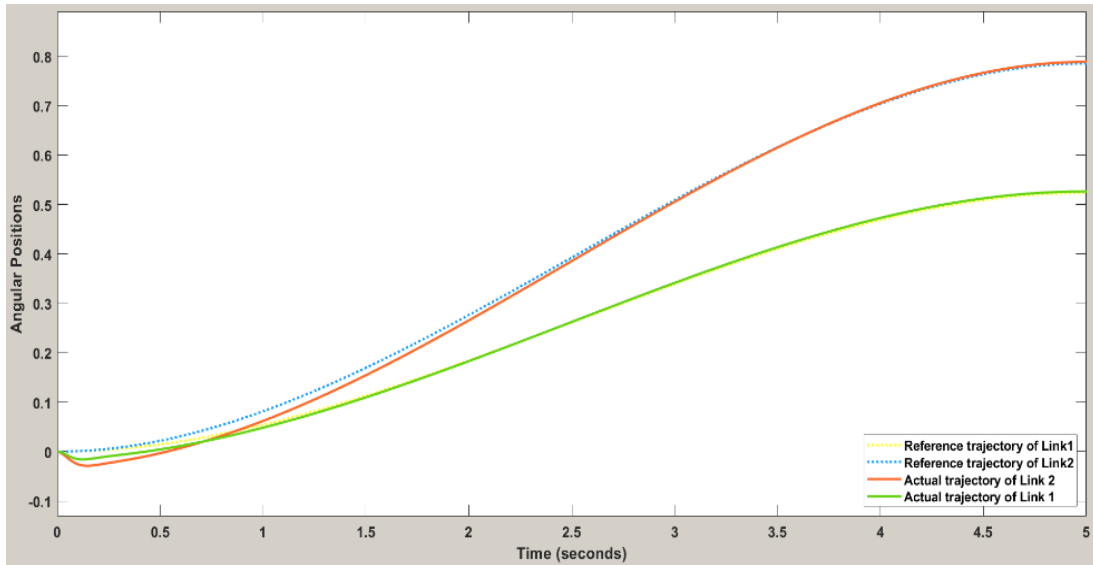


Figure 5.10 Trajectory tracking using AOA tuned PID

Figure 5.9 shows the trajectory tracking using ASO tuned PID, this provided value of objective function to be 0.04970. Initially their deviations from reference in trajectory but soon it attained the trajectory.

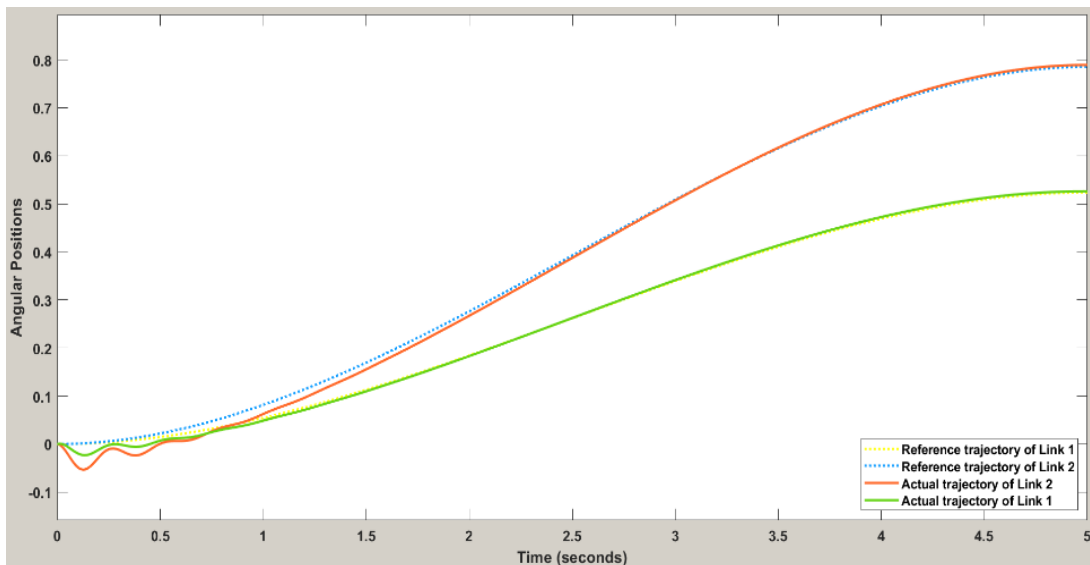


Figure 5.11 Trajectory tracking using SHO tuned PID

Figure 5.10 and Figure 5.11 present the trajectory tracking using AOA and ASO tuned PID controller. AOA has obtained value of objective function to be 0.04607 while SHO provided 0.04944. In AOA tuned PID controller the first link has some overshoots in tracking the reference trajectory while link 2 has no overshoots. SHO has shown initial overshoots in the trajectory of both links. These overshoots are not desirable for the satisfactory tracking of trajectory in real-time implementation. Statistical investigation has revealed that SHO's performance is the worst. Figure 5.12 shows the trajectory tracking using STO-tuned PID controller and returned the value of the objective function to be 0.04573. Compared to the earlier algorithms, STO has greatly improved the fitness value and demonstrated good trajectory tracking. The oscillatory behavior of the actual trajectory of link 1 is improved.

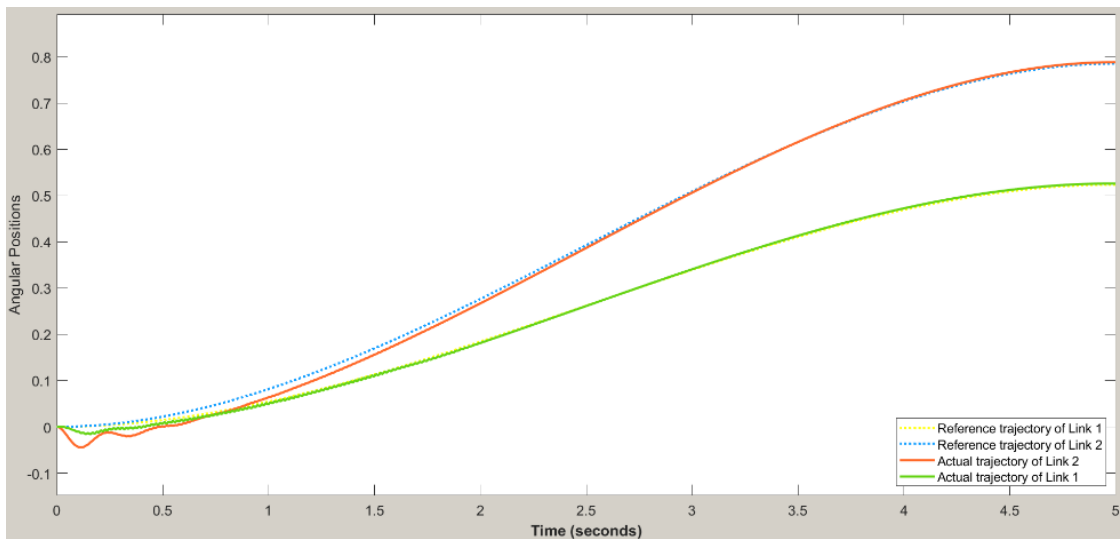


Figure 5.12 Trajectory tracking using STO tuned PID

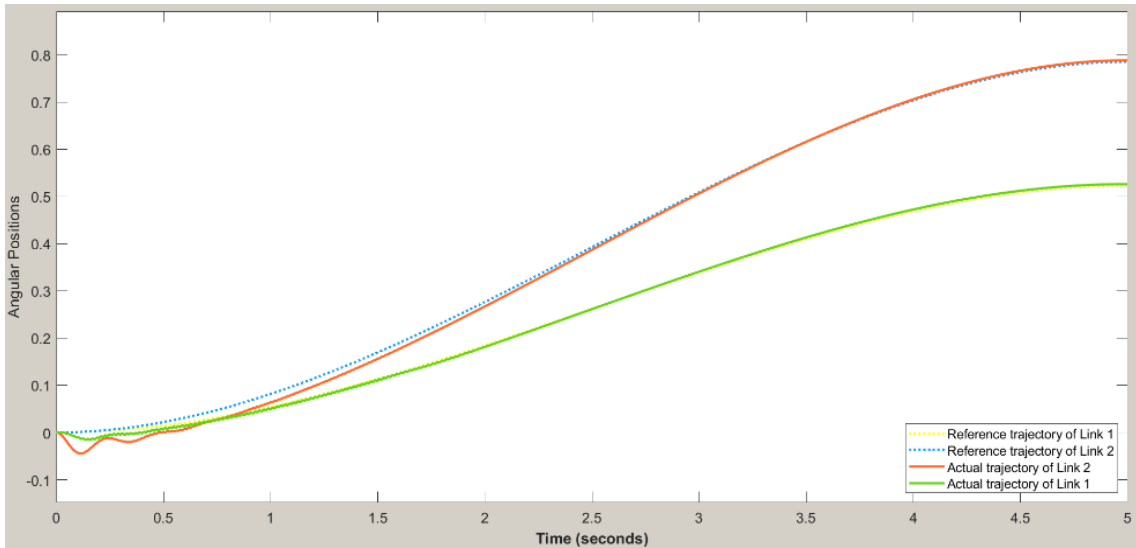


Figure 5.13 Trajectory tracking using hybrid STOPSO tuned PID

To improve the exploitation capabilities of STO, a new hybrid algorithm STOPSO has been proposed. Further the hybrid STOPSO shown in Figure 5.13 has enhanced the fitness value. It attains the best ranking in Friedman’s test.

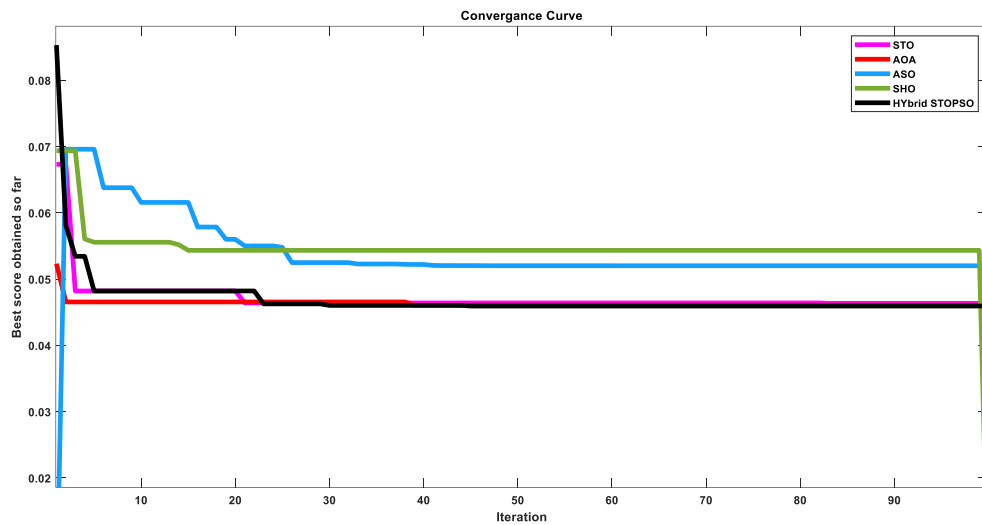


Figure 5.14 Convergence curve of all the metaheuristic algorithms.

Figure 5.14 shows the convergence curve of all the algorithms. The convergence curve clearly shows that the algorithms AOA, STO, and hybrid STOPSO converge

to lower error values, out of which the hybrid STOPSO algorithm converges to the least error value, while ASO and SHO converge to bigger values of error. Thus the designed novel hybrid STOPSO shows the convergence to minimum fitness value. Table 5.5 shows the comparative study of the STOPSO algorithm with the other algorithms implanted for trajectory tracking of a robotic manipulator utilizing the PID controller. In [71] and [92], the authors have employed ACO and WOA for trajectory tracking problem of a robotic manipulator having two links by tuning a PID controller. The performance of the controller has been assessed for IATE error, and fitness values of 0.1648 and 3.102, respectively, have been achieved. Table 5.5 shows the fitness value attained using the AOA, ASO, SHO, STO, and hybrid STOPSO algorithm. In comparison to ACO and WOA, the obtained fitness values using the proposed algorithms are significantly lower.

Table 5.5 Comparative study of the proposed algorithm

S. No	Technique Implemented	Fitness function value	Technique Implemented	Fitness function value
1.	ACO [71]	0.1648	ASO	0.04970
2.	WOA [92]	3.102	AOA	0.04607
			STO	0.04573
			SHO	0.04944
			Hybrid STOPSO	0.04541

Chapter 6

Conclusion and Future Scope of the Work

6.1 Conclusion

There are many applications of robotic manipulators in the household, industry, and medical fields. From the perspective of control, these systems are considerably more complicated and uncertain. Researchers are discovering many innovative ways to control these systems so that they can perform the intended tasks with increased efficiency. Trajectory tracking and path planning have extensive importance in robotic applications. The conventional and intelligent both control techniques have been implemented on robotic systems for various applications. Adaptive control, optimum control, and PID control are the conventional approaches, whereas intelligent control includes the application of artificial intelligence techniques such metaheuristic optimization algorithms. The intelligent control methods enhance the task handling capabilities of robotic systems.

In this work, the trajectory control problem of robotic manipulator has been addressed using adaptive sliding mode control (ASMC), PID and FOPID control, extremum seeking control, and then implementation of various metaheuristic algorithms (GWO, WOA, MFO, and MVO) on the controller designed for a linearized model. Furthermore, the metaheuristic algorithms (ASO, AOA, SHO, and STO) have been implemented on a nonlinear model of a robotic manipulator. The performance indices are weighted sum of IAE and ITAE for implementation of these algorithms. Because of the stochastic nature of such algorithms, a statistical analysis has been performed by taking 10 runs for each algorithm. As a result, all the above algorithms have attained good tracking of the trajectory under the constraints. Afterwards, to enhance the exploitation of the implemented algorithm STO a novel hybrid algorithm STOPSO has been designed and tested for tracking the trajectory of a robotic manipulator. The novel hybrid algorithm has attained the trajectory tracking by considerably improving the exploitation capability of STO

with incorporation of PSO. A nonparametric statistical test called the Friedman anova has been carried out to assess the efficacy of the presented algorithms, and a rank has been given to each algorithm. According to this test, MFO attains the best ranking and WOA takes the worst ranking in the trajectory control on a linearized model of robotic manipulator, and for the nonlinear model analysis hybrid STOPSO attains a rank of 1 followed by STO and SHO attains the worst rank. Hybrid STOPSO performs the best and SHO performs the worst. Further, a perturbation-type ESC has been designed to track the optimum point of the defined trajectory. The following points summarize the findings of the work.

1. The designed adaptive sling mode control and extremum -seeking control techniques can achieve the reference trajectory effectively.
2. For PID and FOPID implementation using the algorithms GWO, WOA, MFO, and MVO, the algorithm MFO performs the best with the cost function values 0.2721 and 0.1736 respectively, and WOA performs the worst by returning the cost function value 0.2723 and 0.1779.
3. For PID implementation using the algorithms AOA, ASO, SHO, and STO, the algorithms STO performs the best and provides the cost function value as 0.04573, and the algorithms ASO performs the worst with the cost function value as 0.04970.
4. To enhance the performance of STO a novel hybrid algorithm STOPSO has been designed and applied on PID for trajectory tracking. This algorithm provides the cost function value as 0.04541 which is improved as compared to STO.
5. A nonparametric test of Friedman's anova has been carried out and a ranking has been given to each of these algorithms.

6.2 Future Scope

Manipulators are extremely popular robotic systems in performing industrial tasks. Intelligent control techniques enhance the task handling ability of such systems. The tracking performance of the robotic manipulators can be explored for different hybrid algorithms. The application of STOPSO in a variety of other applications, such as path planning, joint angle orientation, and tuning of other conventional controllers, may also be of interest to research scholars and scientists around the world in the future. These algorithms are recent and have a wide possibility of implementation in the control design for various other robotic systems, further, this work provides a guideline or framework to implement such metaheuristic techniques on the hardware model of the robotic systems.

Bibliography

1. Vidyasagar, M. (2002). *Nonlinear systems analysis*. Society for Industrial and Applied Mathematics.
2. Niku, S. B. (2020). *Introduction to robotics: analysis, control, applications*. John Wiley & Sons.
3. Mohammed, A. A., & Eltayeb, A. (2018, August). Dynamics and control of a two-link manipulator using PID and sliding mode control. *International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)* (pp.1-5). IEEE. doi:10.1109/ICCCEEE.2018.8515795/
4. Sharma, R., Gaur, P., & Mittal, A. P. (2015). Performance analysis of two-degree of freedom fractional order PID controllers for robotic manipulator with payload. *ISA transactions*, 58, 279-291. doi: 10.1016/j.isatra.2015.03.013
5. Mohan, V., Chhabra, H., Rani, A., & Singh, V. (2019). An expert 2DOF fractional order fuzzy PID controller for nonlinear systems. *Neural Computing and Applications*, 31, 4253-4270.
6. Koo, K. M., & Kim, J. H. (1994). Robust control of robot manipulators with parametric uncertainty. *IEEE Transactions on Automatic Control*, 39(6), 1230-1233.
7. Zhang, D., & Wei, B. (2017). A review on model reference adaptive control of robotic manipulators. *Annual Reviews in Control*, 43, 188-198, doi: 10.1016/j.arcontrol.2017.02.002.
8. Castillo, O., & Melin, P. (2003). Intelligent adaptive model-based control of robotic dynamic systems with a hybrid fuzzy-neural approach. *Applied Soft Computing*, 3(4), 363-378.

9. Tokhi, M. O., & Azad, A. K. (1996). Modelling of a single-link flexible manipulator system: theoretical and practical investigations. *Robotica*, *14*(1), 91-102.
10. Chhabra, H., Mohan, V., Rani, A., & Singh, V. (2016). Multi objective PSO tuned fractional order PID control of robotic manipulator. In *Intelligent Systems Technologies and Applications 2016* (pp. 567-572). Springer International Publishing.
11. Kathuria, T., Kumar, V., Rana, K. P. S., & Azar, A. T. (2018). Control of a three-link manipulator using fractional-order pid controller. In *Fractional Order Systems* (pp. 477-510). Academic Press.
12. Gupta, M. K., Sinha, N., Bansal, K., & Singh, A. K. (2016). Natural frequencies of multiple pendulum systems under free condition. *Archive of Applied Mechanics*, *86*, 1049-1061.
13. Nagrath, I. J., Shripal, P. P., & Chand, A. (1995, January). Development and implementation of intelligent control strategy for robotic manipulator. In *Proceedings of IEEE/IAS International Conference on Industrial Automation and Control* (pp. 215-220). IEEE. doi: 10.1109/iacc.1995.465840.
14. Jin, L., Li, S., Yu, J., & He, J. (2018). Robot manipulator control using neural networks: A survey. *Neurocomputing*, *285*, 23-34.
15. Kim, Y. H., Lewis, F. L., & Dawson, D. M. (2000). Intelligent optimal control of robotic manipulators using neural networks. *Automatica*, *36*(9), 1355-1364.
16. Jin, B. (1993, October). Robotic manipulator trajectory control using neural networks. In *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)* (Vol. 2, pp. 1793-1796). IEEE. doi: 10.1109/ijcnn.1993.717002.
17. Kim, S. H., Jang, C. W., Chai, C. H., & Choi, H. G. (1997, June). Trajectory control of robotic manipulators using chaotic neural networks. In *Proceedings of international conference on neural networks*

- (*ICNN'97*) (Vol. 3, pp. 1685-1688). IEEE. doi: 10.1109/ICNN.1997.614148.
18. Wang, L., Chai, T., & Zhai, L. (2009). Neural-network-based terminal sliding-mode control of robotic manipulators including actuator dynamics. *IEEE Transactions on Industrial Electronics*, 56(9), 3296-3304. doi: 10.1109/TIE.2008.2011350.
 19. Zhu, Q. G., Chen, Y., & Wang, H. R. (2009, July). The RBF neural network control for the uncertain robotic manipulator. In *2009 International Conference on Machine Learning and Cybernetics* (Vol. 3, pp. 1266-1270). IEEE. doi: 10.1109/ICMLC.2009.5212337
 20. Hu, H., & Woo, P. Y. (2006). Fuzzy supervisory sliding-mode and neural-network control for robotic manipulators. *IEEE Transactions on Industrial Electronics*, 53(3), 929-940. doi: 10.1109/TIE.2006.874261.
 21. Rahmani, B., & Belkheiri, M. (2016, November). Robust adaptive control of robotic manipulators using neural networks: Application to a two link planar robot. In *2016 8th International conference on modelling, identification and control (ICMIC)* (pp. 839-844). IEEE. doi: 10.1109/ICMIC.2016.7804231.
 22. Lee, M. J., & Choi, Y. K. (2004). An adaptive neurocontroller using RBFN for robot manipulators. *IEEE Transactions on Industrial Electronics*, 51(3), 711-717. doi: 10.1109/TIE.2004.824878.
 23. Ozaki, T., Suzuki, T., Furuhashi, T., Okuma, S., & Uchikawa, Y. (1991). Trajectory control of robotic manipulators using neural networks. *IEEE Transactions on Industrial Electronics*, 38(3), 195-202. doi: 10.1109/41.87587.
 24. Li, S., Zhang, Y., & Jin, L. (2016). Kinematic control of redundant manipulators using neural networks. *IEEE transactions on neural networks and learning systems*, 28(10), 2243-2254 doi:10.1109/TNNLS.2016.2574363.

25. Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, 8(3), 338-353.
26. Mamdani, E. H., & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International journal of man-machine studies*, 7(1), 1-13.
27. Bai, Y., & Wang, D. (2006). Fundamentals of fuzzy logic control—fuzzy sets, fuzzy rules and defuzzifications. *Advanced fuzzy logic technologies in industrial applications*, 17-36.
28. Lim, C. M., & Hiyama, T. (1991). Application of fuzzy logic control to a manipulator. *IEEE Transactions on Robotics and Automation*, 7(5), 688-691.
29. de Silva, C. W. (1995). Applications of fuzzy logic in the control of robotic manipulators. *Fuzzy Sets and Systems*, 70(2-3), 223-234. doi: 10.1016/0165-0114(94)00219-W.
30. Kumbala, K. K., & Jamshidi, M. (1994, June). Control of robotic manipulator using fuzzy logic. In *Proceedings of 1994 IEEE 3rd International Fuzzy Systems Conference* (pp. 518-523). IEEE. doi: 10.1109/fuzzy.1994.343731.
31. Karahan, O., & Ataşlar-Ayyıldız, B. (2019). Optimal design of fuzzy PID controller with CS algorithm for trajectory tracking control. In *Intelligent Computing: Proceedings of the 2018 Computing Conference, Volume 1* (pp. 174-188). Springer International Publishing.
32. Er, M. J., & Sun, Y. L. (2001). Hybrid fuzzy proportional-integral plus conventional derivative control of linear and nonlinear systems. *IEEE Transactions on Industrial Electronics*, 48(6), 1109-1117.
33. Huang, S. J., & Lian, R. J. (1997). A hybrid fuzzy logic and neural network algorithm for robot motion control. *IEEE Transactions on Industrial Electronics*, 44(3), 408-417.
34. Tsai, C. H., Wang, C. H., & Lin, W. S. (2000). Robust fuzzy model-following control of robot manipulators. *IEEE Transactions on fuzzy systems*, 8(4), 462-469. doi: 10.1109/91.868952.

35. Zhu, D., Mei, T., Luo, M., & Guan, K. (2009, August). Fuzzy SVM controller for robotic manipulator based on GA and LS algorithm. In *2009 Sixth international conference on fuzzy systems and knowledge discovery* (Vol. 6, pp. 263-266). IEEE. doi: 10.1109/FSKD.2009.190.
36. Norouzi, A., & Koch, C. R. (2019, May). Robotic manipulator control using PD-type fuzzy iterative learning control. In *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)* (pp. 1-4). IEEE. doi: 10.1109/CCECE.2019.8861721.
37. Yu, J. P., Ma, Y., Chen, B., & Yu, H. S. (2011). Adaptive fuzzy backstepping position tracking control for a permanent magnet synchronous motor. *Int. J. Innov. Comput. Inf. Control*, 7(4), 1589-1602., doi: 10.1007/978-3-030-67723-7_8.
38. Sharma, R., Rana, K. P. S., & Kumar, V. (2014). Performance analysis of fractional order fuzzy PID controllers applied to a robotic manipulator. *Expert systems with applications*, 41(9), 4274-4289. doi: 10.1016/j.eswa.2013.12.030.
39. Kumar, V., & Rana, K. P. S. (2017). Nonlinear adaptive fractional order fuzzy PID control of a 2-link planar rigid manipulator with payload. *Journal of the Franklin Institute*, 354(2), 993-1022., doi: 10.1016/j.jfranklin.2016.11.006.
40. Muñoz-Vázquez, A. J., Gaxiola, F., Martínez-Reyes, F., & Manzo-Martínez, A. (2019). A fuzzy fractional-order control of robotic manipulators with PID error manifolds. *Applied soft computing*, 83, 105646. doi: 10.1016/j.asoc.2019.105646.
41. Bingül, Z., & Karahan, O. (2011). A Fuzzy Logic Controller tuned with PSO for 2 DOF robot trajectory control. *Expert Systems with Applications*, 38(1), 1017-1031. doi: 10.1016/j.eswa.2010.07.131.
42. Yilmaz, B. M., Tatlicioglu, E., Savran, A., & Alci, M. (2021). Adaptive fuzzy logic with self-tuned membership functions based repetitive learning

- control of robotic manipulators. *Applied Soft Computing*, 104, 107183. doi: 10.1016/j.asoc.2021.107183.
43. Kumar, A., & Kumar, V. (2017). Evolving an interval type-2 fuzzy PID controller for the redundant robotic manipulator. *Expert Systems with Applications*, 73, 161-177. doi: 10.1016/j.eswa.2016.12.029.
 44. Li, T. H. S., & Huang, Y. C. (2010). MIMO adaptive fuzzy terminal sliding-mode controller for robotic manipulators. *Information Sciences*, 180(23), 4641-4660. doi: 10.1016/j.ins.2010.08.009.
 45. Martínez-Soto, R., Castillo, O., & Aguilar, L. T. (2014). Type-1 and Type-2 fuzzy logic controller design using a Hybrid PSO–GA optimization method. *Information Sciences*, 285, 35-49. doi: 10.1016/j.ins.2014.07.012
 46. Lee, T. H. et al. (2020). *Expert control system*. AccessScience, McGraw-Hill Education.
 47. Feigenbaum, E. A. (1981). Expert systems in the 1980s. *State of the art report on machine intelligence. Maidenhead: Pergamon-Infotech*.
 48. Linkens, D. A., & Chen, M. Y. (1995). Expert control systems—2. Design principles and methods. *Engineering Applications of Artificial Intelligence*, 8(5), 527-537.
 49. Geng, Z., & Jamshidi, M. (1988, December). Expert self-learning controller for robot manipulator. In *Proceedings of the 27th IEEE Conference on Decision and Control* (pp. 1090-1095). IEEE. doi: 10.1109/CDC.1988.194486.
 50. Åström, K. J., Anton, J. J., & Årzén, K. E. (1986). Expert control. *Automatica*, 22(3), 277-286.
 51. Teoh, E. K., & Wong, C. Y. (1991, June). An expert system for real-time control of the sir-3 robotic system. In *1991., IEEE International Symposium on Circuits and Systems* (pp. 2709-2712). IEEE.
 52. Duriez, T., Brunton, S. L., & Noack, B. R. (2017). *Machine learning control-taming nonlinear dynamics and turbulence* (Vol. 116). Cham, Switzerland: Springer International Publishing.

53. Huang, H. C., & Chuang, C. C. (2020). Artificial bee colony optimization algorithm incorporated with fuzzy theory for real-time machine learning control of articulated robotic manipulators. *IEEE Access*, 8, 192481-192492.
54. Gautier, N., Aider, J. L., Duriez, T. H. O. M. A. S., Noack, B. R., Segond, M., & Abel, M. (2015). Closed-loop separation control using machine learning. *Journal of Fluid Mechanics*, 770, 442-457..
55. Diveev, A., Konstantinov, S., Shmalko, E., & Dong, G. (2021). Machine learning control based on approximation of optimal trajectories. *Mathematics*, 9(3), 265.
56. Sathya, R., & Abraham, A. (2013). Comparison of supervised and unsupervised learning algorithms for pattern classification. *International Journal of Advanced Research in Artificial Intelligence*, 2(2), 34-38.
57. Kostov, A., Andrews, B. J., Popovic, D. B., Stein, R. B., & Armstrong, W. W. (1995). Machine learning in control of functional electrical stimulation systems for locomotion. *IEEE Transactions on Biomedical Engineering*, 42(6), 541-551.
58. Wang, S., Chaovaitwongse, W., & Babuska, R. (2012). Machine learning algorithms in bipedal robot control. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(5), 728-743.
59. Mirjalili, S., Dong, J. S., & Lewis, A. (2020). Nature-inspired optimizers. *Studies in Computational Intelligence*, 811, 7-20.
60. Sharma, A., Sharma, A., Pandey, J. K., & Ram, M. (2022). *Swarm intelligence: foundation, principles, and engineering applications*. CRC Press.
61. Chakraborty, A., & Kar, A. K. (2017). Swarm intelligence: A review of algorithms. *Nature-inspired computing and optimization: Theory and applications*, 475-494.
62. Chhabra, H., Mohan, V., Rani, A., & Singh, V. (2016). Multi objective PSO tuned fractional order PID control of robotic manipulator. In *Intelligent*

- Systems Technologies and Applications 2016* (pp. 567-572). Springer International Publishing. doi: 10.1007/978-3-319-47952-1_45.
63. Lopez-Franco, C., Diaz, D., Hernandez-Barragan, J., Arana-Daniel, N., & Lopez-Franco, M. (2022). A Metaheuristic Optimization Approach for Trajectory Tracking of Robot Manipulators. *Mathematics*, 10(7), 1051
 64. Yadav, S., Kumar, S., Goyal, M., PID Tuning and Stability Analysis of Hybrid Controller for Robotic Arm Using ZN, PSO, ACO, and GA, (2022) *International Review of Mechanical Engineering (IREME)*, 16 (5), pp. 257-264. doi:https://doi.org/10.15866/ireme.v16i5.21982
 65. Ayala, H. V. H., & dos Santos Coelho, L. (2012). Tuning of PID controller based on a multiobjective genetic algorithm applied to a robotic manipulator. *Expert Systems with Applications*, 39(10), 8968-8974. doi: 10.1016/j.eswa.2012.02.027.
 66. Vijay, M., Jena, D., & Member, I. E. E. E. (2014). GA based adaptive controller for 2DOF robot manipulator. *IFAC Proceedings Volumes*, 47(1), 670-675.
 67. Mohan, V., Chhabra, H., Rani, A., & Singh, V. (2018). Robust self-tuning fractional order PID controller dedicated to non-linear dynamic system. *Journal of Intelligent & Fuzzy Systems*, 34(3), 1467-1478. doi: 10.3233/JIFS-169442.
 68. Vijay, M., & Jena, D. (2014, December). Optimal GA based SMC with adaptive PID sliding surface for robot manipulator. In *2014 9th International Conference on Industrial and Information Systems (ICIIS)* 1-6. IEEE.
 69. Sharma, R., Rana, K. P. S., & Kumar, V. (2014, February). Statistical analysis of GA based PID controller optimization for robotic manipulator. In *2014 International conference on issues and challenges in intelligent computing techniques (ICICT)* (pp. 713-718). IEEE.
 70. Mu, Y., Zhang, L., Chen, X., & Gao, X. (2016, August). Optimal trajectory planning for robotic manipulators using chicken swarm optimization.

- In *2016 8th International conference on intelligent human-machine systems and cybernetics (IHMSC)* (Vol. 2, pp. 369-373). IEEE. doi: 10.1109/IHMSC.2016.107.
71. Singh, R., & Prasad, L. B. (2018, November). Optimal trajectory tracking of robotic manipulator using ant colony optimization. In *2018 5th IEEE Uttar Pradesh section international conference on electrical, electronics and computer engineering (UPCON)* (pp. 1-6). IEEE. doi: 10.1109/UPCON.2018.8597087.
72. Khan, A. H., Li, S., & Luo, X. (2019). Obstacle avoidance and tracking control of redundant robotic manipulator: An RNN-based metaheuristic approach. *IEEE transactions on industrial informatics*, *16*(7), 4670-4680. doi: 10.1109/TII.2019.2941916.
73. Khan, A. T., Li, S., Kadry, S., & Nam, Y. (2020). Control framework for trajectory planning of soft manipulator using optimized RRT algorithm. *IEEE Access*, *8*, 171730-171743.
74. Kumar, A., & Kumar, V. (2017). Hybridized ABC-GA optimized fractional order fuzzy pre-compensated FOPID control design for 2-DOF robot manipulator. *AEU-International Journal of Electronics and Communications*, *79*, 219-233. doi: 10.1016/j.aeue.2017.06.008.
75. Sharma, R., Rana, K. P. S., & Kumar, V. (2014). Comparative study of controller optimization techniques for a robotic manipulator. In *Proceedings of the Third International Conference on Soft Computing for Problem Solving* (pp. 379-393). Springer, New Delhi.
76. Sharma, R., Gaur, P., & Mittal, A. P. (2015, March). Performance evaluation of cuckoo search algorithm based FOPID controllers applied to a robotic manipulator with actuator. In *2015 International conference on advances in computer engineering and applications* (pp. 356-363). IEEE.
77. Cruz-Bernal, A. (2013). Meta-heuristic optimization techniques and its applications in robotics. *Recent Advances on Meta-Heuristics and Their Application to Real Scenarios*, *53*.

78. Yeasmin, S., & Shill, P. C. (2017, December). GA-based adaptive fuzzy logic controller for a robotic arm in the presence of moving obstacle. In *2017 3rd International conference on electrical information and communication technology (EICT)* (pp. 1-6). IEEE.
79. Al-Dois, H., Jha, A. K., & Mishra, R. B. (2014, November). GA-based control of a robot manipulator in a foundry workcell. In *Asia-Pacific World Congress on Computer Science and Engineering* (pp. 1-8). IEEE.
80. Zennir, Y., Mechhoud, E. A., Seboui, A., & Bendib, R. (2017, October). Multi-controller approach with PSO-PI λ D μ controllers for a robotic wrist. In *2017 5th International conference on electrical engineering-boumerdes (ICEE-B)* (pp. 1-7). IEEE.
81. Liu, Y., Jiang, D., Yun, J., Sun, Y., Li, C., Jiang, G., ... & Fang, Z. (2022). Self-tuning control of manipulator positioning based on fuzzy PID and PSO algorithm. *Frontiers in Bioengineering and Biotechnology*, 9, 1443.
82. Chhabra, H., Mohan, V., Rani, A., & Singh, V. (2016). Multi objective PSO tuned fractional order PID control of robotic manipulator. In *Intelligent Systems Technologies and Applications 2016* (pp. 567-572). Springer International Publishing. doi: 10.1007/978-3-319-47952-1_45.
83. Brand, M., Masuda, M., Wehner, N., & Yu, X. H. (2010, June). Ant colony optimization algorithm for robot path planning. In *2010 international conference on computer design and applications* (Vol. 3, pp. V3-436). IEEE..
84. Liu, J., Yang, J., Liu, H., Tian, X., & Gao, M. (2017). An improved ant colony algorithm for robot path planning. *Soft computing*, 21, 5829-5839.
85. Baghli, F. Z., & Lakhali, Y. (2017). Optimization of arm manipulator trajectory planning in the presence of obstacles by ant colony algorithm. *Procedia Engineering*, 181, 560-567.
86. Kumar, A., & Kumar, V. (2017, November). Artificial bee colony based design of the interval type-2 fuzzy PID controller for robot manipulator. In *TENCON 2017-2017 IEEE Region 10 Conference* (pp. 602-607). IEEE.

87. Elkhateeb, N., & Badr, R. I. (2017). Novel PID tracking controller for 2DOF robotic manipulator system based on artificial bee colony algorithm. *The Scientific Journal of Riga Technical University-Electrical, Control and Communication Engineering*, 13, 55-62.
88. Patle, B. K., Pandey, A., Jagadeesh, A., & Parhi, D. R. (2018). Path planning in uncertain environment by using firefly algorithm. *Defence technology*, 14(6), 691-701.
89. Tripathi, S., Shrivastava, A., & Jana, K. C. (2020). GWO based PID controller optimization for robotic manipulator. In *Intelligent Computing Techniques for Smart Energy Systems: Proceedings of ICTSES 2018* (pp. 943-951). Springer Singapore.
90. Gaidhane, P. J., & Nigam, M. J. (2018). A hybrid grey wolf optimizer and artificial bee colony algorithm for enhancing the performance of complex systems. *Journal of computational science*, 27, 284-302.
91. Obadina, O. O., Thaha, M. A., Mohamed, Z., & Shaheed, M. H. (2022). Grey-box modelling and fuzzy logic control of a Leader–Follower robot manipulator system: A hybrid Grey Wolf–Whale Optimisation approach. *ISA transactions*, 129, 572-593.
92. Loucif, F., Kechida, S., & Sebbagh, A. (2020). Whale optimizer algorithm to tune PID controller for the trajectory tracking control of robot manipulator. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 42(1), 1.
93. Du, M., Guo, Z., & Meng, C. (2019, August). Tuning of SMC parameters for robotic manipulator based on whale optimization algorithm. In *2019 WRC symposium on advanced robotics and automation (WRC SARA)* (pp. 248-253). IEEE.
94. Rodríguez-Molina, A., Mezura-Montes, E., Villarreal-Cervantes, M. G., & Aldape-Pérez, M. (2020). Multi-objective meta-heuristic optimization in intelligent control: A survey on the controller tuning problem. *Applied Soft Computing*, 93, 106342. doi: 10.1016/j.asoc.2020.106342.

95. Kumar, A., & Kumar, V. (2017). Hybridized ABC-GA optimized fractional order fuzzy pre-compensated FOPID control design for 2-DOF robot manipulator. *AEU-International Journal of Electronics and Communications*, 79, 219-233. doi: 10.1016/j.aeue.2017.06.008.
96. Kaur, S., Awasthi, L. K., Sangal, A. L., & Dhiman, G. (2020). Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Engineering Applications of Artificial Intelligence*, 90, 103541.
97. Jain, M., Maurya, S., Rani, A., & Singh, V. (2018). Owl search algorithm: a novel nature-inspired heuristic paradigm for global optimization. *Journal of Intelligent & Fuzzy Systems*, 34(3), 1573-1582.
98. Bennett, S. (1993). Development of the PID controller. *IEEE Control Systems Magazine*, 13(6), 58-62.
99. Dumlu, A., & Erenturk, K. (2013). Trajectory tracking control for a 3-dof parallel manipulator using fractional-order PID control. *IEEE Transactions on Industrial Electronics*, 61(7), 3417-3426.
100. Zhihong, M., Paplinski, A. P., & Wu, H. R. (1994). A robust MIMO terminal sliding mode control scheme for rigid robotic manipulators. *IEEE transactions on automatic control*, 39(12), 2464-2469.
101. Neila, M. B. R., & Tarak, D. (2011). Adaptive terminal sliding mode control for rigid robotic manipulators. *International Journal of Automation and Computing*, 8, 215-220.
102. Baek, J., Jin, M., & Han, S. (2016). A new adaptive sliding-mode control scheme for application to robot manipulators. *IEEE Transactions on industrial electronics*, 63(6), 3628-3637.
103. Islam, S., & Liu, X. P. (2010). Robust sliding mode control for robot manipulators. *IEEE Transactions on industrial electronics*, 58(6), 2444-2453.

104. Zhang, C., & Ordóñez, R. (2011). *Extremum-seeking control and applications: a numerical optimization-based approach*. Springer Science & Business Media.
105. Ariyur, K. B., & Krstic, M. (2003). *Real-time optimization by extremum-seeking control*. John Wiley & Sons.
106. Calli, B., Caarls, W., Jonker, P., & Wisse, M. (2012, October). Comparison of extremum seeking control algorithms for robotic applications. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 3195-3202). IEEE.
107. Malek, H., & Chen, Y. (2016). Fractional order extremum seeking control: performance and stability analysis. *IEEE/ASME Transactions on Mechatronics*, 21(3), 1620-1628.
108. Krstic, M., & Wang, H. H. (2000). Stability of extremum seeking feedback for general nonlinear dynamic systems. *Automatica-Kidlington*, 36(4), 595-602.
109. Krstić, M. (2000). Performance improvement and limitations in extremum seeking control. *Systems & Control Letters*, 39(5), 313-326
110. Wang, L., Chen, S., & Ma, K. (2016). On stability and application of extremum seeking control without steady-state oscillation. *Automatica*, 68, 18-26.
111. Dochain, D., Perrier, M., & Guay, M. (2011). Extremum seeking control and its application to process and reaction systems: A survey. *Mathematics and Computers in Simulation*, 82(3), 369-380.
112. Zhang, C., & Ordóñez, R. (2007). Numerical optimization-based extremum seeking control with application to ABS design. *IEEE Transactions on Automatic Control*, 52(3), 454-467.
113. Brunton, S. L., Rowley, C. W., Kulkarni, S. R., & Clarkson, C. (2010). Maximum power point tracking for photovoltaic optimization using ripple-based extremum seeking control. *IEEE transactions on power electronics*, 25(10), 2531-2540.

114. Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, 69, 46-61.
115. Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in engineering software*, 95, 51-67.
116. Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, 89, 228-249.
117. Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016). Multi-verse optimizer: a nature-inspired algorithm for global optimization. *Neural Computing and Applications*, 27(2), 495-513.
118. Sharma, A., Sharma, A., Averbukh, M., Jatly, V., & Azzopardi, B. (2021). An effective method for parameter estimation of a solar cell. *Electronics*, 10(3), 312.
119. Sharma, A., Sharma, A., Averbukh, M., Rajput, S., Jatly, V., Choudhury, S., & Azzopardi, B. (2022). Improved moth flame optimization algorithm based on opposition-based learning and Lévy flight distribution for parameter estimation of solar module. *Energy Reports*, 8, 6576-6592.
120. Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer methods in applied mechanics and engineering*, 376, 113609.
121. Dhiman, G., & Kaur, A. (2019). STOA: a bio-inspired based optimization algorithm for industrial engineering problems. *Engineering Applications of Artificial Intelligence*, 82, 148-174.
122. Dhiman, G., & Kumar, V. (2017). Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. *Advances in Engineering Software*, 114, 48-70.
123. Zhao, W., Wang, L., & Zhang, Z. (2019). Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. *Knowledge-Based Systems*, 163, 283-304.

List of Publications

1. Devendra Rawat, Mukul Kumar Gupta and Abhinav Sharma. (2023). Trajectory Control of Robotic Manipulator using Metaheuristic Algorithms. *International Journal of Mathematical, Engineering and Management Sciences*, 8(2), 264-281.
2. Devendra Rawat, Mukul Kumar Gupta and Abhinav Sharma. (2022). Intelligent Control of Robotic Manipulators: A Comprehensive Review, *Spat. Inf. Res.* (2022).
3. Devendra Rawat, Mukul Kumar Gupta and Abhinav Sharma. (2022). Optimum point trajectory tracking of a robotic manipulator system using Extremum Seeking control ICICCD Nov 2022.
4. Devendra Rawat, Mukul Kumar Gupta and Abhinav Sharma. (2022). Metaheuristic Algorithms based Optimization of Robotic Manipulator for Trajectory Tracking. *International Journal of Advanced Technology and Engineering Exploration (IJATEE)* (Under Review)

Plagiarism Report

Thesis		
ORIGINALITY REPORT		
8%	5%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS
		1%
		STUDENT PAPERS
PRIMARY SOURCES		
1	www.researchgate.net Internet Source	1%
2	Abhishek Sharma, Abhinav Sharma, Moshe Averbukh, Vibhu Jately, Brian Azzopardi. "An Effective Method for Parameter Estimation of a Solar Cell", Electronics, 2021 Publication	<1%
3	www.um.edu.mt Internet Source	<1%
4	"Soft Computing for Problem Solving", Springer Science and Business Media LLC, 2019 Publication	<1%
5	d-nb.info Internet Source	<1%
6	Abhishek Sharma, Abhinav Sharma, Moshe Averbukh, Shailendra Rajput, Vibhu Jately, Sushabhan Choudhury, Brian Azzopardi. "Improved moth flame optimization algorithm based on opposition-based learning and Lévy	<1%