

Republic of Iraq
Ministry of Higher Education and
Scientific Research
University of Technology
Control and Systems Engineering Department



Multi Objective Decision Maker for Single and Multi Robot Path Planning

**A Thesis Submitted to the Control and Systems Engineering Department,
University of Technology in a Partial Fulfillment of the Requirements for the
Degree of Master of Science in Computers Engineering.**

By

Esraa Adnan Hadi

Supervised By

Assist. Prof. Dr. Muna Mohammed Jawad

1440 A.H

2018 A.D

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

﴿وَكَلِمًا فَضِلَّ اللَّهُ عَلَيْكَ وَمَرَحْمَةً لَهْمَتْ طَائِفَةٌ مِنْهُمْ أَنْ يُضِلُّوكَ وَمَا يُضِلُّونَ إِلَّا أَنْفُسَهُمْ ۗ
وَمَا يُضُرُّونَكَ مِنْ شَيْءٍ ۗ وَأَنْزَلَ اللَّهُ عَلَيْكَ الْكِتَابَ وَالْحِكْمَةَ وَعَلَّمَكَ مَا
لَمْ تَكُنْ تَعْلَمُ ۗ وَكَانَ فَضْلُ اللَّهِ عَلَيْكَ عَظِيمًا ۙ﴾

صدق الله العظيم

سورة النساء آية (113)

Acknowledgements

*First of all, I have to express all the thanks, gratitude to **Almighty Allah** Who gives me the ability to achieve this imperfect work and all what has been done, and without His bless and support nothing can be done.*

*I would like to express my deep sense of gratitude and respect to my supervisor, **Assist. Prof. Dr. Muna Mohammed Jawad**, for her excellent guidance, encouragement and support she has provided during my time as her student. I consider myself fortunate to be her student.*

*Words cannot express how grateful I am to my **parents, brother and two sisters** who have been a source of encouragement and inspiration to me throughout my life. Their prayers for me were what sustained me. I thank them for being there for me throughout my entire study. I also would like to thank the head and all staff of the Control and Systems Engineering Department in University of Technology for their help.*

Finally, I would like to thank all of my friends and my colleagues who have offered support and advice.

Esraa Adnan Hadi

2018

Dedication

*To whom I proud to carry his name, My Father.
Thank you for keeping me going even when I thought I
could not.*

*To the tittle of love, tenderness, and hope.
To whom that have taught me to endure, no matter how the
circumstances change, My Mother.*

*To the big heart who taught me success, patience and the
emergence of a passion for knowledge*

My Dear brother (Dr.Ahmed)

My Beautiful sisters (Alaa and Asmaa)

*To baby girl, who is the frolic and joy into my life,
(Zahraa)*

“My God bless you all “

Esraa Adnan Hadi

Supervisor Certificate

I certify that the preparation of the thesis entitled “**Multi Objective Decision Maker for Single and Multi Robot Path Planning**” by “**Esraa Adnan Hadi**” was done under my supervision at the Department of Control and Systems Engineering, University of Technology in partial fulfillment of the requirements for the degree of **Master of Science in Computers Engineering**.

Signature: *m . M. AL-Nayar*

Name: **Assist. Prof. Dr. Muna Mohammed Jawad AL- Nayar**

Date: *22* / 11 / 2018

Linguistic Certificate

I certify that this thesis entitled “**Multi Objective Decision Maker for Single and Multi Robot Path Planning**” by “**Esraa Adnan Hadi**” under my linguistic supervision. Its language was amended to meet the style of English language.

Signature:



Dr. Samir Ali Amin
Assist. Professor

Name: **Assist. Prof. Dr. Samir Ali Amin AL- Rabii**

University of Technology / Department of Mechanical Engineering

Date: 02 / 10 / 2018

EXAMINATION COMMITTEE CERTIFICATE

We certify that we have read the thesis entitled “**Multi Objective Decision Maker for Single and Multi Robot path planning**” prepared by the student “**Esraa Adnan Hadi**” and as an examination committee, examined her in its contents and that in our option; it meets the standard of a thesis for the degree of Master of Science in Computers Engineering.

Signature: 

Name: **Asst. Prof. Dr. Ebtessam N. Abdullah**


University of Kufa

Computer science

Department

Data: 13/01/2019

(Chairman)

Signature: 

Name: **Lec. Dr. Eklas K. Hamza**

University of Technology

Control and Systems Engineering

Department

Data: 10/01/2019

(Member)

Signature: 

Name: **Asst. Prof. Dr. Hamid M. Hasan**

University of Technology

Control and Systems Engineering

Department

Data: 10/01/2019

(Member)

Signature: 

Name: **Asst. Prof. Dr. Muna M. Jawad**

University of Technology


Control and Systems Engineering

Department

Data: 10/01/2019

(Supervisor & Member)

Approved for the Control and Systems Engineering Department
University of Technology
Baghdad - Iraq

Signature: 

Name: **Asst. Prof. Dr. Azad Raheem Kareem**

Data: 17/01/2019

University of Technology / Head of Control and Systems Engineering Department.

ABSTRACT

Decision making (DM) includes information gathering, data mining, modeling and analysis. Path planning problem is one of decision-making applications in the robotics field; its purpose is to find a shortest path from the start position to a target position without hitting any obstacle cluttered in the environment.

This thesis presents four optimization algorithms to find the solution for the multi objective path planning for multi mobile robot. The first two algorithms are based on the standard Particle Swarm Optimization (PSO) algorithm and the developed PSO with chaotic map to form Chaotic Particle Swarm Optimization (CPSO) algorithm in order to prevent a slide into local minima. The other two algorithms are based on the standard Firefly (FF) algorithm strong technique but suffers from trapping into several local optimum problem, and the proposed algorithm, which based on the FF and CPSO algorithms to form a hybrid technique called Firefly Chaotic Particle Swarm Optimization (FFCPSO) algorithm as a global path planning. Cubic spline is used to generate a smooth path by interpolation of optimization algorithms, and the objective function is evaluated by two constraints; the first one is the path length, and the second one is the obstacles avoidance. Furthermore, based on a kinematic model for a wheeled mobile robot, the platform linear and angular velocities and linear and angular velocities of right and left wheel are calculated to direct a National Instrument (NI) mobile robot's wheel to follow a desired path to reach a predefined target.

The optimization algorithms were simulated using MATLAB (R2014a) program and the simulation results showed that the CPSO algorithm is better than the PSO algorithm with a less number of iterations and, the proposed hybrid FFCPSO algorithm is applicable to mobile robots path planning for obtaining a perfect path in the workspace as a compared with basic FF algorithm. This is demonstrated by minimizing the path length and obtaining the smoothness velocities for wheeled NI-mobile robots without exceeding the limited values (less than 0.5 m/sec). In addition, these algorithms

are compared with the other research papers to evaluate their performance. By comparing the results achieved by the proposed algorithms with the results achieved by the previous works under the same conditions, the cubic polynomial interpolation is a good feature by generating a smooth path without sharp edges during the learning process, so the mobile robot can moving smoothly and safety.

LIST OF CONTENTS

Subject		Page No.
ABSTRACT		I
LIST OF CONTENTS		III
LIST OF SYMBOLS		VI
LIST OF ABBREVIATIONS		IX
LIST OF FIGURES		XI
LIST OF TABLES		XVI
LIST OF ALGORITHMS		XVII
CHAPTER ONE: INTRODUCTION AND LITERATURE SURVEY		
1.1	Introduction	1
1.2	Mobile Robot	3
1.3	Path Planning	5
1.4	Traditional and Soft Computing Algorithms	5
1.6	Motivation	8
1.7	Thesis Objectives	9
1.5	Literature Survey	9
1.8	Thesis Organization	13
CHAPTER TWO: THEORETICAL BACKGROUND		
2.1	Introduction	14
2.2	Nonholonomic Wheeled Mobile Robot Kinematic Model	14
2.3	Autonomous Mobile Robot Path Planning	17
	2.3.1 Robot Path Planning	17
	2.3.2 Path Planning Classification	18
	2.3.3 Path Planning Mapping and Environment Modeling	19
2.4	Swarm-Based Optimization Algorithms	20
	2.4.1 Particle Swarm Optimization (PSO) Algorithm	21
	2.4.1.1 The Original PSO Algorithm	21
	2.4.1.2 The PSO Algorithm Flow	24
	2.4.2 Firefly (FF) Algorithm	26
	2.4.2.1 Firefly Mainframe	26
	2.4.2.2 The Original FF Algorithm	27

		2.4.2.3	The FF Algorithm Flow	28
CHAPTER THREE: MODELLING AND OPTIMIZATION BASED PATH PLANNING				
3.1	Introduction			30
3.2	Path Planning Based on Decision Making Systematic			30
3.3	Modeling of the Environment and Obstacles			32
3.4	Path Construction			32
3.5	Objective Function			33
	3.5.1	Obstacle Avoidance		34
	3.5.2	Minimum Path Length		35
3.6	Path Optimization Algorithms			38
	3.6.1	Improved Basic PSO Algorithm		39
	3.6.2	Firefly (FF) Algorithm		43
	3.6.3	Proposed Hybrid FFCPSO Algorithm		43
CHAPTER FOUR: SIMULATION RESULTS AND DISCUSSION				
4.1	Introduction			49
4.2	Simulation Parameters' Setting			50
	4.2.1	Parameters Setting for basic PSO and Chaotic PSO Algorithm		50
	4.2.2	Parameters Setting for basic FF Algorithm		51
	4.2.3	Parameters Setting for Hybrid FFCPSO Algorithm		51
	4.2.4	Parameters Setting of Start and Target Position		52
	4.2.5	Parameters Setting of Obstacles Positions		52
4.3	Simulation Results			53
	4.3.1	Case A: Single Wheeled NI- Mobile Robot		53
		4.3.1.1	Single Optimal Path Finding	53
		4.3.1.2	Single Mobile Robot Velocities	57
	4.3.2	Case B: Three Independent Wheeled NI- Mobile Robots		62
		4.3.2.1	Multi Optimal Path Finding	62
		4.3.2.2	Multi Mobile Robot Velocities	68
	4.3.3	Case C: Three Follow Up Wheeled NI- Mobile Robots		83
		4.3.3.1	Follower Optimal Path Finding	83
		4.3.3.2	Follower Mobile Robot Velocities	90
4.4	Performance Evaluation			105

CHAPTER FIVE :CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK		
5.1	Conclusions	110
5.2	Suggestions for Future Work	112
REFERENCES		
	References	113
APPENDIX		
	Appendix A: Parameters Setting	118
	List of Publications	123

LIST OF SYMBOLS

Symbol	Meaning	Unit
$(a_1 \text{ and } a_2)$	The random numbers with between 0 and 1.	
c	The midpoint of the wheeled mobile robot.	
c_1	The cognitive acceleration constant.	
c_2	The social acceleration constant.	
D	The number of optimization parameters.	
(D_{gx}, D_{gy})	The Cartesian distance with G_{best} .	
(D_{px}, D_{py})	The Cartesian distance with P_{best} .	
$Fitness(i)$	The fitness value of each agent	
$F(rr)$	The flash intensity.	
F_o	The standard flash intensity.	
F_s	The light intensity at the source.	
G_{best}	The best particle among all the particles in the population.	
\mathcal{H}	The obstacle zone, weight factor.	
i	The current particle.	
k	The current obstacle.	
L	The length of wheeled NI-mobile robot.	m
m	The total number of obstacles in the workspace.	
$ML(i)$	The minimum path length of each agent.	m
np	The total number of interpolation points.	cm
pop_size	The size of population.	
P_{best}	The Personal best position of each particle in the swarm.	
O	The obstacle position in the workspace.	
Q	The location of the wheeled mobile robot in the global coordinate frame.	
R	The radius of each driving wheel.	m
rr	The distance between any two fireflies.	
$r_1 \text{ and } r_2$	The random numbers between 0 and 1.	
$rand$	The random number in $[0, 1]$.	
R_{obs}	The radius of any obstacle.	cm
$Safety(i)$	The safety variable for each agent.	

S_{ref}	The reference path points.	
$S^T(Q)$	The matrix associated with the constraints	
t	The current iteration.	
T_{max}	The maximum number of iteration.	
T_s	The sampling period of the system	sec
T_v	The travelling time.	sec
U	The control parameter.	
V_a	The platform angular velocity.	rad/sec
V_{amax}	The maximum angular velocity of mobile robot platform.	rad/sec
V_{amin}	The minimum angular velocity of mobile robot platform.	rad/sec
V_i^t	The velocity of particle i at (t) iteration.	
V_i^{t+1}	The velocity of particle i at ($t+1$) iteration.	
VL	The velocity of left wheel.	m/sec
V_{nmax}	The maximum linear velocity of mobile robot platform.	m/sec
V_{nmin}	The minimum linear velocity of mobile robot platform.	m/sec
VL_{max}	The maximum linear velocity of left wheel mobile robot.	m/sec
VL_{min}	The minimum linear velocity of left wheel mobile robot.	m/sec
V_{max}	The maximum velocity bound of particles.	
V_{min}	The minimum velocity bound of particles.	
V_n	The platform linear velocity.	m/sec
VR	The velocity of right wheel.	m/sec
VR_{max}	The maximum linear velocity of right wheel mobile robot.	m/sec
VR_{min}	The minimum linear velocity of right wheel mobile robot.	m/sec
V_{ref}	Reference linear velocity	m/sec
W	The distance between the two driving wheels.	m
ω	The inertia weight factor.	
ω_{max}	The maximum value of inertia weight.	
ω_{min}	The minimum value of inertia weight.	
ω_{new}	The new inertia weight factor.	
W_{ref}	Reference angular velocity	rad/sec
WL_{max}	The maximum angular velocity of left wheel mobile robot.	rad/sec
WL_{min}	The minimum angular velocity of left wheel mobile robot.	rad/sec
WR_{max}	The maximum angular velocity of right wheel mobile robot.	rad/sec

WR_{min}	The minimum angular velocity of right wheel mobile robot.	rad/sec
X_i^t	The position of particle i at (t) iteration.	
X_i^{t+1}	The position of particle i at (t+1) iteration.	
X_{max}	The maximum position bound of particles.	
X_{min}	The minimum position bound of particles.	
x and y	The coordinates of point c in the global coordinate frame	
(X_s, Y_s)	The mobile robot initial point.	
(X_t, Y_t)	The mobile robot terminal point.	
(X_{obs}, Y_{obs})	The center of any obstacle.	cm
Z^1	The chaotic initial value.	
$\tilde{\alpha}$	The randomization parameter.	
β	The brightness of firefly.	
β_0	The brightness at $rr=0$.	
ϵ_i	The vector of random variables.	
γ	The fixed flash absorption coefficient.	
φ	The number of samples.	

LIST OF ABBREVIATIONS

Abbreviation	Meaning
2D	Two-dimensional
3D	Three-dimensional
ABC	Artificial Bee Colony
AFA	Adaptive Firefly Algorithm
AI	Artificial Intelligence
AMPSO	Adaptive Multi objective Particle Swarm Optimization
APF	Artificial Potential Field
AUVs	Autonomous Underwater Vehicles
BFO	Bacterial Foraging Optimization
CSA	Cat Swarm Algorithm
CD	Cell Decomposition
CFA	Chaotic Firefly Algorithm
CFPSO	Constriction Factor Particle Swarm Optimization
CFA-OAS	Chaotic Firefly Algorithm- Optimization Adjustment Strategy
CPSO	Chaotic Particle Swarm Optimization
CPU	Central Processing Unit
DABC	Directed Artificial Bee Colony
DM	Decision Making
FA	Fruit fly Algorithm
FF	Firefly
FFCPSO	Firefly Chaotic Particle Swarm Optimization
FL	Fuzzy Logic
GAs	Genetic Algorithms
GPP	Global Path Planning
GSA	Gravitational Search Algorithm
LMRs	Legged Mobile Robots
LPP	Local Path Planning
MATLAB	Matrix Laboratory
MRSs	Multi Robot Systems
NI	National Instrument
NNs	Neural Networks

NWMR	Nonholonomic Wheeled Mobile Robot
PC	Personal Computer
PRM	Probabilistic Road Map
PSO	Particle Swarm Optimization
PSO-w	Particle Swarm Optimization with adaptive inertia weight
RM	Road Map
RN	Robot Navigation
SGA	Slice Genetic Algorithm
SI	Swarm Intelligence
SOPSO	Second-order Oscillating Particle Swarm Optimization
TS	Tabu Search
UAVs	Unmanned Aerial Vehicles
UCAV	Unmanned Combat Aerial Vehicles
WMR	Wheeled Mobile Robot
WPSO	Inertia Weight Particle Swarm Optimization

LIST OF FIGURES

Figure No.	Title	Page No.
1.1	The systematic of decision-making process.	2
1.2	Mobile robot types.	4
1.3	Traditional and soft computing methods.	6
2.1	The mobile robot representation.	15
2.2	Environment models.	20
2.3	Basic structure of PSO for global best approximation.	22
2.4	The flowchart of PSO algorithm.	25
2.5	The flowchart of FF algorithm.	29
3.1	Path planning based decision-making systematic.	31
3.2	Original and virtual obstacle boundaries.	32
3.3	Overall scheme of applying optimization algorithms based path planning.	36
3.4	Overall scheme of applying optimization algorithms based path planning for mobile robots.	37
3.5	The Flowchart of CPSO Algorithm based path planning.	42
3.6	The Flowchart of FF Algorithm based path planning.	45
3.7	The Flowchart of FFCPSO Algorithm based path planning.	48
4.1	The shortest path for case A based on optimization algorithms	54
4.2	Variation of objective function through the number of iterations / case A.	55
4.3	Variation of objective function through number of runs for case A.	56
4.4	Desired and actual path for case A.	58
4.5	The angular velocity of right and left wheels/ case A based on CPSO algorithm.	58
4.6	The angular velocity of right and left wheels / case A based on FF algorithm.	59
4.7	The angular velocity of right and left wheels / case A based on FFCPSO algorithm.	59
4.8	The linear velocity of right and left wheels / case A based on CPSO algorithm.	60
4.9	The linear velocity of right and left wheels / case A based on FF algorithm.	60
4.10	The linear velocity of right and left wheels / case A based on FFCPSO algorithm.	60

4.11	The platform angular and linear velocities/ case A based on CPSO algorithm.	61
4.12	The platform angular and linear velocities/ case A based on FF algorithm.	61
4.13	The platform angular and linear velocities/ case A based on FFCPSO algorithm.	62
4.14	The shortest path for each robot / case B.	63
4.15	Variation of the objective function through the number of iterations for first path / case B.	65
4.16	Variation of the objective function through the number of iterations for second path / case B.	65
4.17	Variation of the objective function through the number of iterations for third path / case B.	66
4.18	Desired and actual path for case B.	70
4.19	The angular velocity of right and left wheels for 1 st robot / case B based on CPSO algorithm.	71
4.20	The angular velocity of right and left wheels for 1 st robot / case B based on FF algorithm.	71
4.21	The angular velocity of right and left wheels for 1 st robot / case B based on FFCPSO algorithm.	72
4.22	The angular velocity of right and left wheels for 2 nd robot / case B based on CPSO algorithm.	72
4.23	The angular velocity of right and left wheels for 2 nd robot / case B based on FF algorithm.	73
4.24	The angular velocity of right and left wheels for 2 nd robot / case B based on FFCPSO algorithm.	73
4.25	The angular velocity of right and left wheels for 3 rd robot / case B based on CPSO algorithm.	74
4.26	The wheel angular velocity of right and left for 3 rd robot / case B based on FF algorithm.	74
4.27	The angular velocity of right and left wheels for 3 rd robot / case B based on FFCPSO algorithm.	74
4.28	The linear velocity of right and left wheels for 1 st robot / case B based on CPSO algorithm.	75
4.29	The linear velocity of right and left wheels for 1 st robot / case B based on FF algorithm.	76
4.30	The linear velocity of right and left wheels for 1 st robot / case B based on FFCPSO algorithm.	76

4.31	The linear velocity of right and left wheels for 2 nd robot / case B based on CPSO algorithm.	77
4.32	The linear velocity of right and left wheels for 2 nd robot / case B based on FF algorithm.	77
4.33	The linear velocity of right and left wheels for 2 nd robot / case B based on FFCPSO algorithm.	77
4.34	The linear velocity of right and left wheels for 3 rd robot / case B based on CPSO algorithm.	78
4.35	The linear velocity of right and left wheels for 3 rd robot / case B based on FF algorithm.	78
4.36	The linear velocity of right and left wheels for 3 rd robot / case B based on FFCPSO algorithm.	79
4.37	The platform angular and linear velocities for 1 st robot / case B based on CPSO algorithm.	79
4.38	The platform angular and linear velocities for 1 st robot / case B based on FF algorithm.	80
4.39	The platform angular and linear velocities for 1 st robot / case B based on FFCPSO algorithm.	80
4.40	The platform angular and linear velocities for 2 nd robot / case B based on CPSO algorithm.	81
4.41	The platform angular and linear velocities for 2 nd robot / case B based on FF algorithm.	81
4.42	The platform angular and linear velocities for 2 nd robot / case B based on FFCPSO algorithm.	82
4.43	The platform angular and linear velocities for 3 rd robot / case B based on CPSO algorithm.	82
4.44	The platform angular and linear velocities for 3 rd robot / case B based on FF algorithm.	83
4.45	The platform angular and linear velocities for 3 rd robot / case B based on FFCPSO algorithm.	83
4.46	Follower Mobile Robots / case C Based on basic PSO algorithm.	84
4.47	Follower Mobile Robots / case C Based on CPSO algorithm.	84
4.48	Follower Mobile Robots / case C Based on basic FF algorithm.	85
4.49	Follower Mobile Robots / case C Based on hybrid FFCPSO algorithm.	85
4.50	Variation of the objective function for first path through the number of iterations / case C.	87

4.51	Variation of the objective function for second path through the number of iterations / case C.	87
4.52	Variation of the objective function for third path through the number of iterations / case C.	88
4.53	Desired and actual path for case C.	92
4.54	The angular velocity of right and left wheels for 1 st robot / case C based on CPSO algorithm.	93
4.55	The angular velocity of right and left wheels for 1 st robot / case C based on FF algorithm.	93
4.56	The angular velocity of right and left wheels for 1 st robot / case C based on FFCPSO algorithm.	94
4.57	The angular velocity of right and left wheels for 2 nd robot / case C based on CPSO algorithm.	94
4.58	The angular velocity of right and left wheels for 2 nd robot / case C based on FF algorithm.	95
4.59	The angular velocity of right and left wheels for 2 nd robot / case C based on FFCPSO algorithm.	95
4.60	The angular velocity of right and left wheels for 3 rd robot / case C based on CPSO algorithm.	96
4.61	The angular velocity of right and left wheels for 3 rd robot / case C based on CPSO algorithm.	96
4.62	The angular velocity of right and left wheels for 3 rd robot / case C based on FFCPSO algorithm.	96
4.63	The linear velocity of right and left wheels for 1 st robot / case C based on CPSO algorithm.	97
4.64	The linear velocity of right and left wheels for 1 st robot / case C based on FF algorithm.	97
4.65	The linear velocity of right and left wheels for 1 st robot / case C based on FFCPSO algorithm.	98
4.66	The linear velocity of right and left wheels for 2 nd robot / case C based on CPSO algorithm.	98
4.67	The linear velocity of right and left wheels for 2 nd robot / case C based on FF algorithm.	99
4.68	The linear velocity of right and left wheels for 2 nd robot / case C based on FFCPSO algorithm.	99
4.69	The linear velocity of right and left wheels for 3 rd robot / case C based on CPSO algorithm.	100
4.70	The linear velocity of right and left wheels for 3 rd robot / case C based on FF algorithm.	100

4.71	The linear velocity of right and left wheels for 3 rd robot / case C based on FFCPSO algorithm.	101
4.72	The platform angular and linear velocities for 1 st robot / case C based on CPSO algorithm.	101
4.73	The platform angular and linear velocities for 1 st robot / case C based on FF algorithm.	102
4.74	The platform angular and linear velocities for 1 st robot / case C based on FFCPSO algorithm.	102
4.75	The platform angular and linear velocities for 2 nd robot / case C based on CPSO algorithm.	103
4.76	The platform angular and linear velocities for 2 nd robot / case C based on FF algorithm.	103
4.77	The platform angular and linear velocities for 2 nd robot / case C based on FFCPSO algorithm.	103
4.78	The platform angular and linear velocities for 3 rd robot / case C based on CPSO algorithm.	104
4.79	The platform angular and linear velocities for 3 rd robot/ case C based on FF algorithm.	104
4.80	The platform angular and linear velocities for 3 rd robot / case C based on FFCPSO algorithm.	105
4.81	The best results achieved by [8] case study 2.	106
4.82	The best results of first comparison.	106
4.83	The best results achieved by [21].	107
4.84	The best results of second comparison / Map a and Map c.	107
4.85	The best results achieved by [24] case one.	108
4.86	The best results of third comparison.	109

LIST OF TABLES

Table No.	Title	Page No.
1.1	Decision Making Applications.	3
2.1	Kinematic model symbols.	15
4.1	The Hardware and Software specifications.	49
4.2	Wheeled NI- Mobile Robot parameters [48].	50
4.3	PSO and CPSO parameters.	51
4.4	FF parameters.	51
4.5	FFCPSO parameters.	52
4.6	Start and target definition for all cases.	52
4.7	Obstacles definition for all cases	53
4.8	Waypoints coordination for case A.	54
4.9	Comparison results for case A.	55
4.10	Percentages of the objective function between algorithms for case A	56
4.11	Waypoints coordination for case B.	64
4.12	Comparison results for case B.	67
4.13	Percentages of the objective function between algorithms for case B.	67
4.14	The distance error for case B.	70
4.15	Waypoints coordination for case C.	86
4.16	Comparison results for case C.	89
4.17	Percentages of the objective function between algorithms for case C.	89
4.18	The distance error for case C.	92
4.19	Results comparison with [7].	106
4.20	Results comparison with [21].	108
4.21	Results comparison with [24].	109

LIST OF ALGORITHMS

Algorithm No.	Title	Page No.
Algorithm 1	Particle Swarm Optimization Algorithm.	21
Algorithm 2	Firefly Algorithm.	26
Algorithm 3	Chaotic Particle Swarm Optimization Algorithm.	39
Algorithm 4	Firefly Chaotic Particle Swarm Optimization Algorithm.	43

CHAPTER ONE

INTRODUCTION AND LITERATURE SURVEY

CHAPTER ONE

INTRODUCTION AND LITERATURE SURVEY

1.1 Introduction

The decision making (DM) is a process of decreasing uncertainty and doubt about options to allow a best choice to be made from among them. It is more complicated and difficult because the number of available alternatives is much larger today than ever before. Due to the availability of information technology and communication systems, especially the availability of the Internet and its search engines, we can find more information quickly and therefore generating more alternatives. Second, the cost of making errors can be very large because of the complexity of operations and automation. Third, there are continuous changes in the fluctuating environment and more uncertainties in impacting elements, including information sources and information itself. More importantly, the rapid change of the decision environment requires decisions to be made quickly. All these reasons cause that the decision makers need techniques to support and help making high quality decisions [1].

The DM is a cognitive process, which leads to select a course of action among a number of alternatives. Generally, a process of decision starts when one needs to find a solution, but one does not know the solution that is accepted by the decision makers or not. Simon invented the systematic of decision-making process in year 1977, which involves four phases: Intelligence, Design, Choice and Implementation phases [1].

In intelligence phase, identify the problem that need to be solved includes good understanding on problem assumptions, boundaries and any related initial and desired conditions. Then, obtain requirements by collecting data and analyzing the

decision situation. The requirements are conditions in which any acceptable solution must meet in order to solve the problem. In design phase, construct model that represents the system by making assumptions and by writing down the relationships between all variables. Then generate potential alternative solutions and set criteria in order to evaluate alternatives. In choice phase, select a suitable solution to the model. Once the solution seems to be reasonable, we are ready for the implementation phase [1, 2].

From these phases, it is clearly that the decision is a choice among various options, so decision makers will go through all these steps in the process in order to reach the final choice. In addition, these steps are abstracted in Figure (1.1) [2].

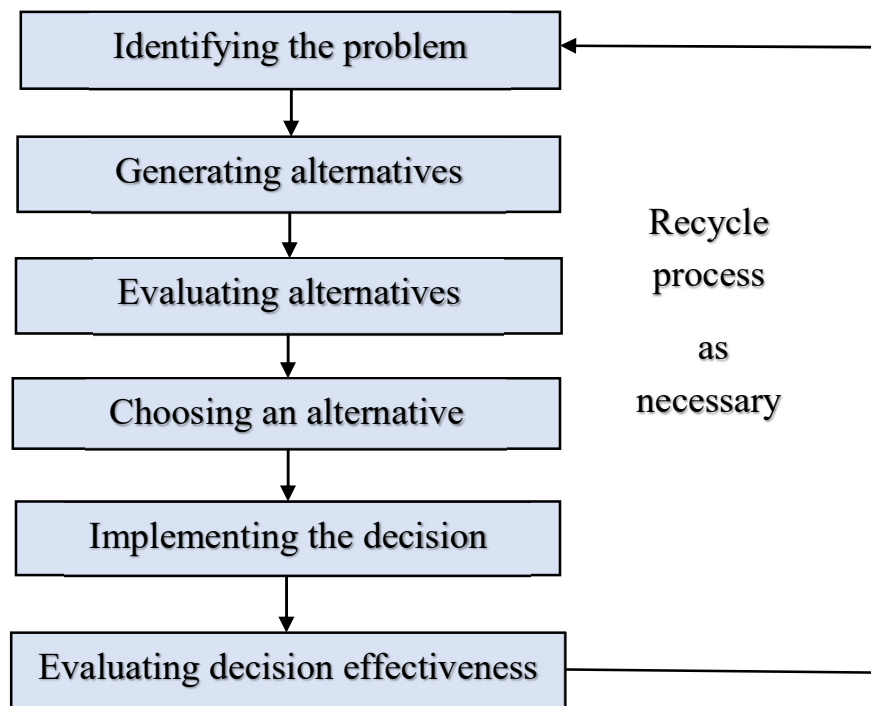


Figure (1.1): The systematic of decision-making process [2].

There are wide applications of the decision-making process systematic in many areas including the items in Table (1.1) [3]:

Table (1.1): Decision Making Applications [3].

Field	Application
Computers	Task scheduling, memory allocation and hardware management.
Robotics	Mission planning, path planning and robot navigation.
Medicine	Health monitoring and medical diagnostic systems.
Science	Automated interpretation of experimental data.
Traffic Systems	Routing and signal switching.
Process Industries	Performance assessment, monitoring and failure diagnosis.
Manufacturing	Resource allocation routing, scheduling and planning materials flow and machine and equipment design.

1.2 Mobile Robot

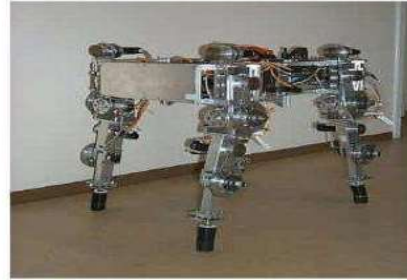
Lexically, the term “Robot” is originally derived from the Czech word “Robota” which means “to make things manually”. The Oxford dictionary describes “Robot” as a machine that looks like a human and enables to duplicate certain human functions and movements in an automatic way [4]. Robots can be divided into groups; the first group is fixed robots, while the second group is mobile robots. Fixed robots are mounted on a fixed land, and materials are transported to the workspace close to the robot. A mobile robot, from its name, is a robot that can be moved from one place to another place autonomously and without human interaction [5].

Nowadays, mobile robots can change position safely in cluttered environments, recognize real objects, understand natural speech, path planning, locate themselves, and generally think by themselves. Intelligent mobile robots are designed to employ the technologies and methodologies of cognitive, intelligence, and behavior based control. Mobile robots should maximize the flexibility of performance subject to minimal computational complexity and minimal input dictionary. Ground mobile robots are classified into classes namely, wheeled mobile robots (WMRs) and legged mobile robots (LMRs). Furthermore, mobile robots also include additional classes,

such as autonomous underwater vehicles (AUVs) and unmanned aerial vehicles (UAVs) as shown in Figure (2.1). WMRs are very popular ones because they are suitable for typical applications with relatively low mechanical complexity and energy consumption. LMRs are appropriate for tasks in nonstandard environments, stairs, heaps of rubble and so on [5].



A: Wheeled mobile robot.



B: Legged mobile robot.



C: Underwater mobile robot.



D: Aerial mobile robot.

Figure (1.2): Mobile robot types [5].

The Multi Robot Systems (MRSs) can be described as a group of robots working in the same environment. However, the range of robotic systems starts from simple sensors, processing and acquiring data, to complex humans such as machines, which are able to interact with environments in complex ways. Multi-robot systems have been widely applied in rescuing, industry, exploration of outer space areas, due to their characteristics of reliability, robustness, and economy [6].

1.3 Path Planning

Path planning is one of the decision-making applications in robotics field that was prefaced in late 60's. Path planning is vitally important problem in mobile robot navigation, which is formulated by giving the model of mobile robot and environment description and then planning a path between two predefined points, start and target in order to accomplish different tasks. In general, there are many paths for mobile robot to reach the goal, but actually, the superior path is adopted based on some optimization criteria, such as least energy consuming, shortest distance or shortest distance and shortest time that are most adopted criteria. Path planning is considered as NP-hard (non-deterministic polynomial time) problem; that means the computational time that is required in order to solve such a problem, which rises dramatically, while the size of the problem rises. According to this definition, path-planning problem is classified as an optimization problem [7, 8].

1.4 Traditional and Soft Computing Algorithms

Path planning algorithms have been grown from one generation to another during the past 50 years. Since the last 60's, many papers have been proposed in order to solve the problem of path planning for mobile robots. More than 1400 research covering sufficient approaches in robot path planning for the time span of 1973-2007 were surveyed in [9]. The path planning involves two approaches namely, traditional or conventional methods and soft computing methods. From Figure (1.3), it is clearly to say that all path-planning methods were classical solution until 1983. Only (3.13%) dealt with soft computing solution until 1987, but this percentage was increased at the end of the eighties, especially after 1992 until now.

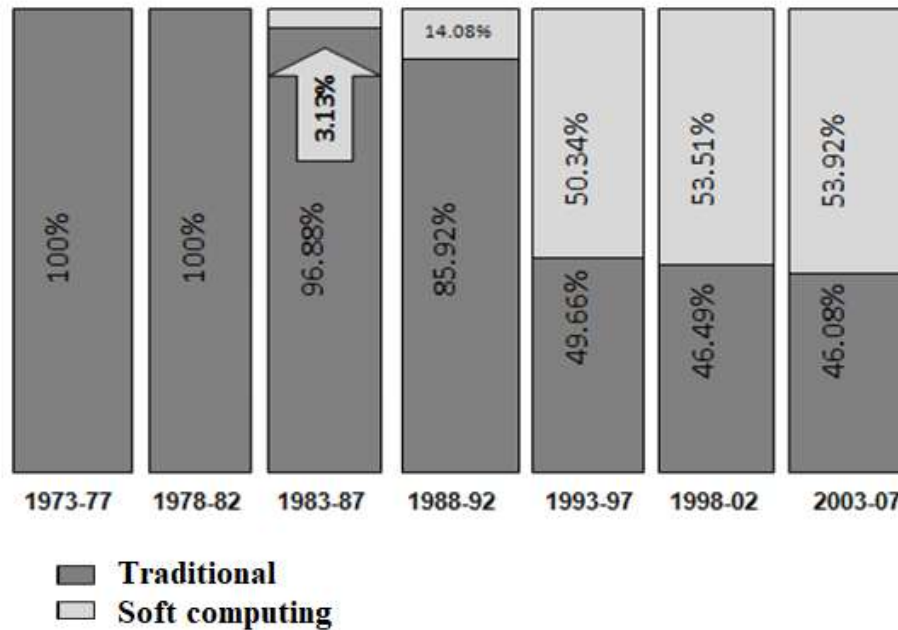


Figure (1.3): Traditional and soft computing methods [9].

The traditional methods does not enforce intelligence into path planning and it includes Road Map (RM), Cell Decomposition (CD) and Artificial Potential Field (APF).

- In Road Map (RM) approach [10], there are two phases, namely constructional phase, and a query phase. In the constructional phase, the path graph is built from the source to destination along with the obstacles. In query phase, only the source and the destination location are provided, and information about the obstacles or any interruption in the middle are not specified.
- In Cell Decomposition (CD) approach [9], the search space is decomposed into a set of simple cells and the relations between these cells are calculated for finding the path between the start and goal configuration by identifying the start and the goal cell and then connecting them with a series of in between cells.
- In Artificial Potential Field (APF) approach [10], two forces play a major role in achieving the optimal path plan from a particular source of the goal. A mobile robot moves towards the goal with the help of an attractive force, which proceeds

with the negative charge. If the robot moves towards the obstacle, it is pushed back with the help of a repulsive force by the positive charge. After that, the potential field is calculated from the robot position and then calculates the induced force from the field. This method suffers from the local minima problem and high complexity.

On the other hand, the soft computing methods are developed to overcome the disadvantages of traditional methods such as trapping in local minima and contain Neural Networks (NNs), Fuzzy Logic (FL) and Genetic Algorithms (GAs) as follows:

- Artificial Neural Network (ANN) [11]: This method uses neural networks for pattern recognition, objects tracking and improvement of process for both static and dynamic environments.
- Fuzzy Logic (FL) [10]: is widely used in controlling mobile robots and deals with neither completely true nor completely false, it represents a partial solution when a perfect solution cannot be predicted and used to solve when the pattern recognition problems arise in robotic tasks with more robust. The FL converts the human natural language into machine understanding control strategies. In mobile robot, FL is used to track a visual object by representing a color in a particular destination with the help of a sensor.
- Genetic Algorithm (GA) [12]: An appropriate “chromosome” representation of the path and representation of the environment are essential for finding out the path.

In the past decades, biologists and natural scientists studied the behaviors of social insects because of the amazing efficiency of these natural swarm systems. In the late-80s, computer scientists proposed the scientific insights of these natural

swarm systems to the field of Artificial Intelligence (AI) [13]. In 1989, the expression "Swarm Intelligence" was first introduced by G. Beni and J. Wang in the global optimization framework as a set of algorithms for controlling robotic swarm [14]. More specifically, the following two population-based optimization algorithms use the analogy of the social behavior and swarming principles in nature. Swarm intelligence (SI) has been adopted to solve various problems of engineering and mobile robotics including the path-planning problem.

- Particle Swarm Optimization (PSO) algorithm was introduced by Dr. James Kennedy and Dr. Eberhart in 1995 based on an inspiration from the social behavior of bird flocks and fish schools. PSO uses a population of particles (individuals) that are moving in the search space. During iterations, each particle is memorized the coordinates of position in the search space associated with better fitness value achieved so far. PSO also stores the position of the best value form the whole particles [15, 16].
- Firefly (FF) algorithm is now one of the most greatly used. Xin-She Yang developed FF algorithm in 2009 based on an inspiration from the natural behavior of tropical fireflies. The FF algorithm tries to simulate the attraction behavior of fireflies and lighting pattern [17].

1.5 Motivation

In the last years, mobile robot path planning has been an evolving area, so that, many techniques have been proposed to challenge this problem. The main motivation behind this thesis is that the mobile robots could be used in hazardous industrial applications. These applications could be safer if the mobile robots were to replace the human operator aspect, and at the same time, they achieve better results with high precision. This work is facing on path planning problem because it is considered a critical part in the field of robotics. This problem needs to find an

appropriate path for some mobile robots in order to move from start point to terminal point in static or dynamic environments that has obstacles, such as disasters places, planetary exploration, battlefield and so on. In path planning, there is a number of optimization criteria that must be adopted, such as distance, smoothness, energy and time in order to get a feasible or near to a feasible path.

1.6 Thesis Objectives

The general objective of this thesis is to study the single and multi-robot path-planning problem and to propose two perfect nature inspired algorithms (PSO and FF algorithms) that may solve it. Finally, the research work will be achieved as follows:

1. Applying path-planning problem based on decision-making strategy.
2. Enhancement of the original PSO algorithm to find the optimal path. Then, a hybridization of the enhanced PSO algorithm with FF algorithm to find the optimal path for single and multiple mobile robot in static known environment.
3. Verification of the collected results by making a comparison with previous research's works.

1.7 Literature Survey

Different kinds of path planning approaches and Natures-Inspired algorithms have been proposed and much has been written on solving the problem of trajectory planning for wheeled mobile robot. This survey gives a sight into some available researches that are recently discussing the path-planning issue.

C. Liu et al. (2012) [18] suggested a new firefly algorithm that has been used gradually in solving planning problems. This algorithm was designed by adaptive both random and absorption parameters after the analysis of the details of standard

firefly algorithm in order to improve the convergence speed and solution of quality. The simulation results verified the effectiveness of the proposed algorithm and path planning feasibility based on firefly technique.

C. Purcaru et al. (2013) [19] proposed a new optimal path planning algorithm based on hybridization between a PSO and GSA algorithms. The hybrid PSO-GSA generates optimal paths by maximizing the distance between the generated paths and the dangerous zones that exist in the environment and by minimizing the length of the path that is needed by the mobile robot to reach the target. The adopted algorithm was validated by the running of several experiments with robots in different environments with the presence of multi obstacles and multi dangerous zones.

E. Masehian and D. Sedighizadeh (2013) [20] presented a heuristic methods for solving a multi-robot problem. Here, the method is based on the new improved variant of the PSO algorithm, which serves as a global planner. Alternatively, for locale planning and for avoiding obstacles in narrow passages, the Probabilistic Roadmap Method (PRM) was employed. The local and global planners act sequentially until all robots reach their goals. The algorithm iteratively and simultaneously finds the minimum of two objectives, smoothness and shortness of the robot's path.

N. H. Abbas and F. M. Ali (2014) [7] presented a comparative study between standard version of Artificial Bee Colony (ABC) and Directed Artificial Bee Colony (DABC) algorithms for the problem of offline autonomous mobile robot path planning. The simulation results showed that the proposed (DABC) algorithm was more effective and got satisfactory results than (ABC) algorithm. Additionally, the obstacles are irregular shape, the radius be one-half of the longest side of the obstacle, and this may cause wasted space.

B. Li et al. (2014) [21] presented a novel planning algorithm based on Firefly (FF) algorithm and Bezier curve in order to locate the collision free path. FF

algorithm was employed to optimize the control points of Bezier curve and this proposed a method which was tested in benchmark functions on different static environments. Then, it was compared with two population-based algorithms namely, GA and PSO with adaptive inertia weight factor (PSO-w). The simulation results revealed that FF outperformed the GA and PSO-w in success rate, while PSO-w offered a feasible path with acceptable length.

M. S. Alam et al. (2015) [8] proposed a path planning approach based on Particle Swarm Optimization (PSO) algorithm to compute a minimum distance with obstacles avoidance for a mobile robot in static, known environment. The proposed path planner performed random sampling on grid lines that were generated between start and target locations and found the feasible waypoints on these grid lines without exhaustive search and high computations. The simulation results depicted the efficiency of the proposed algorithm in different static environments.

N. H. Abbas and J. A. Abdulsaheb (2016) [6] proposed an adaptive multi objective particle swarm optimization (AMOPSO) algorithm based on a path tracking problem for two tests. In the first test, a single mobile robot was needed to move from its start point to its target point in static, known environment, which contains two dangerous sources and two obstacles. In the second test, the AMOPSO was used to improve the performance of the mobile robots to move from different start points to different target points with a minimum distance and without any collision between them. Furthermore, test functions are applied in order to make a comparison between standard version of PSO and the proposed AMOPSO algorithms. The simulation results showed that the AMOPSO was better than MOPSO and standard PSO algorithms to get from local minima and with quickest convergence.

M. R. Panda et al. (2016) [22] suggested a new hybridization between Particle Swarm Optimization (PSO) and Tabu Search (TS) algorithms in order to improve

the performance of a multi mobile robot path planning in workspace where the start and target location for each mobile robot is predefined. The simulation results showed that the new hybrid PSO-TS algorithm overcomes the original PSO and TS in terms of computation times and quality of solution when the obstacles are static relative to the mobile robots, while each mobile robot is dynamic relative to other mobile robots.

E. Cholodowicz and D. Figuirowski (2017) [23] introduced a mobile robot path planning with obstacle avoidance based on PSO algorithm which was analyzed in both static and dynamic environments. Cubic splines were used in order to generate a smooth path by the interpolation of optimization solution and the objective function was evaluated by two constrains; the first one is the path length, and the second one is the obstacle avoidance. The simulation results proved that the PSO algorithm is applicable to robotics field for obtaining reasonable route in 2-D workspace.

D. Pang et al. (2017) [24] presented an adaptive firefly algorithm (AFA) in order to solve the local minima problem. Then, a chaotic firefly algorithm (CFA) that utilizes chaotic sequence to tune the control parameters was developed. The CFA was enhanced to take the advantage of the optimization adjustment strategy (OAS) with the Gauss disturbance to maintain the search capability. The simulation results were compared with AFA and CFA-OAS algorithms, and demonstrated that the proposed CFA-OAS outperforms AFA in terms of path length and convergence speed.

A. Tharwat et al. (2018) [25] proposed a Chaotic Particle Swarm Optimization (CPSO) algorithm to optimize the control points of Bezier curve based on two variants namely, CPSO-I and CPSO-II, by modifying the random parameters (r_1 & r_2) with chaotic maps during iterations. To evaluate the performance of CPSO algorithm, the results of the CPSO-I and CPSO-II techniques were compared with

the basic form of PSO algorithm. Furthermore, the CPSO was tested against different numbers of objects and control points, and the CPSO achieved competitive results.

1.8 Thesis Organization

In addition to this chapter, this thesis includes the following chapters:

- **Chapter Two:** It introduces an overview of the kinematic mathematical model of the differential drive wheeled mobile robot, path planning concept, classification and mapping types. Moreover, it presents in details the standard Particle Swarm Optimization (PSO) and Firefly (FF) Algorithms.
- **Chapter Three:** It describes the path planning systematic based on decision-making process. Then, the developed PSO and proposed hybrid FFCPSO algorithms are illustrated.
- **Chapter Four:** It presents the simulation results that obtained by applying four optimization algorithms including, PSO, CPSO, FF and FFCPSO algorithms on mobile robot path planning with different cases. In addition, calculations of the mobile robots velocities on the optimum path are explained. Furthermore, this chapter also provides the discussion of the simulation results by comparing them with other previous research's works.
- **Chapter Five:** It concerns with the overall results of this work, reports conclusions and gives suggestions for the future work.

CHAPTER TWO

THEORTICAL BACKGROUND

CHAPTER TWO

THEORETICAL BACKGROUND

2.1 Introduction

This chapter describes, in details, the mathematical model of the kinematic wheeled mobile robot under pure rolling and without slipping nonholonomic constraints. Thereafter, it gives a theoretical background of the path planning problem, map construction and different types of workspace that planning issues may implement are discussed. Moreover, two metaheuristic optimization algorithms, such as standard PSO and FF algorithms are introduced, in details, to explain the way for applying the proposed algorithms.

2.2 Nonholonomic Wheeled Mobile Robot Kinematic Model

Kinematics, as a field of study, is the science of motion that refers to the behavior of mechanical systems, which deals with the geometric relationships that govern the system and studies of the mathematics of motion without considering the affecting forces. In mobile robotics, the mechanical behavior of the robot must be known, both into design proper mobile robots for tasks and to understand how to generate control software, for instance, mobile robot hardware. Design, development, modification and control of a mechatronic system require an understanding and a suitable representation of a system; specifically, a “model” of the system is required. Any model is an idealization of the actual system. The goal of the robot kinematic modeling is to find the robot speed in the inertial frame as a function of the wheels speeds and the geometric parameters of the robot [26].

In this study, the model of the non-holonomic wheeled mobile robot (WMR) is used, as shown in Figure (2.1). This model consists of right and left wheel for motion on the same axis and an omni-directional castor in face of cart in order to make

mobile robot more stable. Each wheel has a radius indicated by (R), and (W) indicates the distance between the left and right wheels (mobile robot width), while the midpoint between the wheels is indicated by (c) [27].

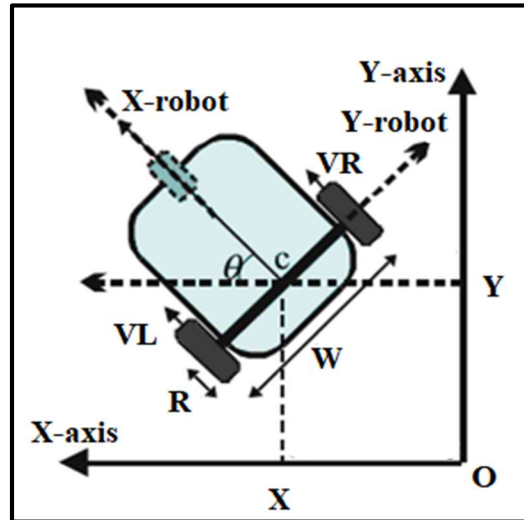


Figure (2.1): The mobile robot representation [27].

The WMR symbols that will be used in this thesis are illustrated in Table (2.1) [27]:

Table (2.1): Kinematic model symbols [27].

Symbol	Description	Unit
W	The distance between right and left wheel.	m
R	The radius of each wheel.	m
VR	The velocity of right wheel.	m/sec
VL	The velocity of left wheel.	m/sec
c	The midpoint of axis between right and left wheel.	
V_a	The platform angular velocity.	rad/sec
V_n	The platform linear velocity.	m/sec
[X,O,Y]	The global coordinate frame.	
$(X, Y, \theta)^T$	The current position and orientation of the WMR	

Generally speaking, the pose (position / orientation) vector for NWMR as in equation (2.1) [27] and the location in the global coordinate frame are defined as [X, O, Y].

$$Q = [X, Y, \theta]^T \quad (2.1)$$

Where, X and Y are specified in the middle axis of wheels that act as the real position of the NWMR, while θ is acting the orientation of NWMR. Based on nonholonomic constraints as in equation (2.5), the kinematic equations for mobile robot in global coordinate frame can be written as in equations (2.2), (2.3) and (2.4) after satisfied two conditions; the first one is a pure rolling wheel, while the second one is without skidding wheel [27] as follows:

$$\dot{X}(t) = v_n(t) \cos \theta(t) \quad (2.2)$$

$$\dot{Y}(t) = v_n(t) \sin \theta(t) \quad (2.3)$$

$$\dot{\theta}(t) = v_a(t) \quad (2.4)$$

$$-\dot{X}(t) \sin \theta(t) + \dot{Y}(t) \cos \theta(t) = 0 \quad (2.5)$$

The reference linear velocity and the angular velocity for the desired path are given by equations (2.13) and (2.14) [27], respectively.

$$V_{ref} = \sqrt{(\dot{X}_{rr})^2 + (\dot{Y}_{rr})^2} \quad (2.13)$$

$$W_{ref} = \frac{\ddot{Y}_{rr} \dot{X}_{rr} - \ddot{X}_{rr} \dot{Y}_{rr}}{(\dot{X}_{rr})^2 + (\dot{Y}_{rr})^2} \quad (2.14)$$

Hence, the wheel linear velocity of right and left based on reference linear and angular velocities on last equations are given by equation (2.15) [27] as follows:

$$\begin{bmatrix} V_R \\ V_L \end{bmatrix} = \begin{bmatrix} V_{ref} + \frac{W}{2} W_{ref} \\ V_{ref} - \frac{W}{2} W_{ref} \end{bmatrix} \quad (2.15)$$

The wheel angular velocity of right and left based on right and left linear velocity on equation (2.15) are given by equation (2.16) and (2.17) [28], respectively.

$$WR = VR \times R \quad (2.16)$$

$$WL = VL \times R \quad (2.17)$$

Finally, the linear and angular velocities in terms of right and left wheels linear velocities can be written as in equations (2.18) and (2.19) [27], respectively.

$$V_n(t) = 0.5 [VR(t) + VL(t)] \quad (2.18)$$

$$V_a(t) = \frac{1}{w} [VR(t) - VL(t)] \quad (2.19)$$

2.3 Autonomous Mobile Robot Path Planning

An autonomous robot is programmed to do a job without human intervention, and with the help of embodied Artificial Intelligence (AI), it can perform and live inside its surroundings [5].

2.3.1 Robot Path Planning

Path planning enables the identification and selection of appropriate path for the robot to traverse in the working arena and, in addition, the main scope of this problem involves both the efficiency and safety points. The efficiency means that the algorithm must find the minimum path in length with acceptable time by not letting the robot take unnecessary steps or stop and turn several times, which may result in a waste of time and energy consumption. While, the safety is another critical point of this problem. Therefore, the determination of an obstacle-free path between two pre-defined points through obstacles cluttered in a working area is central to the design of an autonomous robot path planning. Path planning application covers a wide area of robotics researches because it enhances robotic navigation systems in both static and dynamic environments. With the perfect path planning system, mobile robots can navigate by itself without human intervention to reach the targeted destination [29].

From an engineering point of view, the main fundamental requirement for a mobile robot would be to reach its assigned destination safely. In order to do that, any obstacle collisions must be avoided and prevented. After obstacle collision is identified as the primary requirement, secondary requirements could be identified. The path length should be taken into consideration. This would imply that the shorter the path, the more plausible the algorithm will be. Another secondary requirement for the algorithm is its efficiency. In this case, efficiency refers to the computational cost, which the algorithm needs in order to perform its assigned task. While taking obstacle collision and path length into consideration, the computational cost of the algorithm has to be taken into account. If the algorithm is computationally expensive, but generates a path, which is not significantly better than its competition, then it loses its advantage. A plausible algorithm should be balanced in terms of the time it takes to execute and the quality of the results that it produces [30].

2.3.2 Path Planning Classification

Path planning problem is organized based on two factors, as follows:

- Environment type, such as static or dynamic [31]:
 1. Robot path planning in static environment that has fixed obstacles and does not contain any moving obstacles, other than a navigating robot.
 2. Robot path planning in dynamic environment that has both fixed and dynamic moving obstacles such as moving machines, human beings and moving robots.
- Planning type, such as global or local:
 1. Global path planning (GPP) is a Map-based system in which the robot has a complete knowledge about the search environment (known the positions and sizes of the objects) before starting to move. In other words, the global path planning can be planned offline. The GPP limitation is the cost of changing

environment in global navigation, especially in dynamic environments is very expensive because the setting up of a new map is difficult [32, 33].

2. Local path planning (LPP) is a Sensor-based system in which means that the path planning is implemented during the robot navigating because of a complete information about the search environment which is not available in advance. In other word, the LPP has the ability of producing a new route corresponding to environmental changes [32, 33]. There are several limitations of LPP such as error in sensor readings, error in location estimation, changing environment and robot dynamics. Therefore, the LPP may fail in finding the route to the target location in complex environments. Mostly, this happens because the sensors will not provide the sufficient information that is required by the mobile robot to drive it out to the wanted location [12].

2.3.3 Path Planning Mapping and Environment Modeling

Mapping is a process of building a form of the environment, the suitable representation of the terrain is needed to generate a sufficiently complete map of the given surroundings that the robot will encounter along its route. There are several designs of environment forming, including the Grid-based model and the Continuous-based model where each model has its features and limitations, as shown in Figure (2.2) [34].

Some of researchers use Grid-based model, which is a grid with cells. These cells may be occupied to represent an obstacle or empty to represent a free space when the robot can travel freely. The features of this model are the simple representation, suitable for dynamic environment and local changes (only local effects). Limitation of Grid-based model requires a large memory size when increases the complexity. Others may use a Continuous-based model with the objects being either polygons or any other shape. The features of this model are the simple

shape, efficient memory and represent obstacles with a virtual circle. While, the limitations are the complex code and wasted space [34, 35].

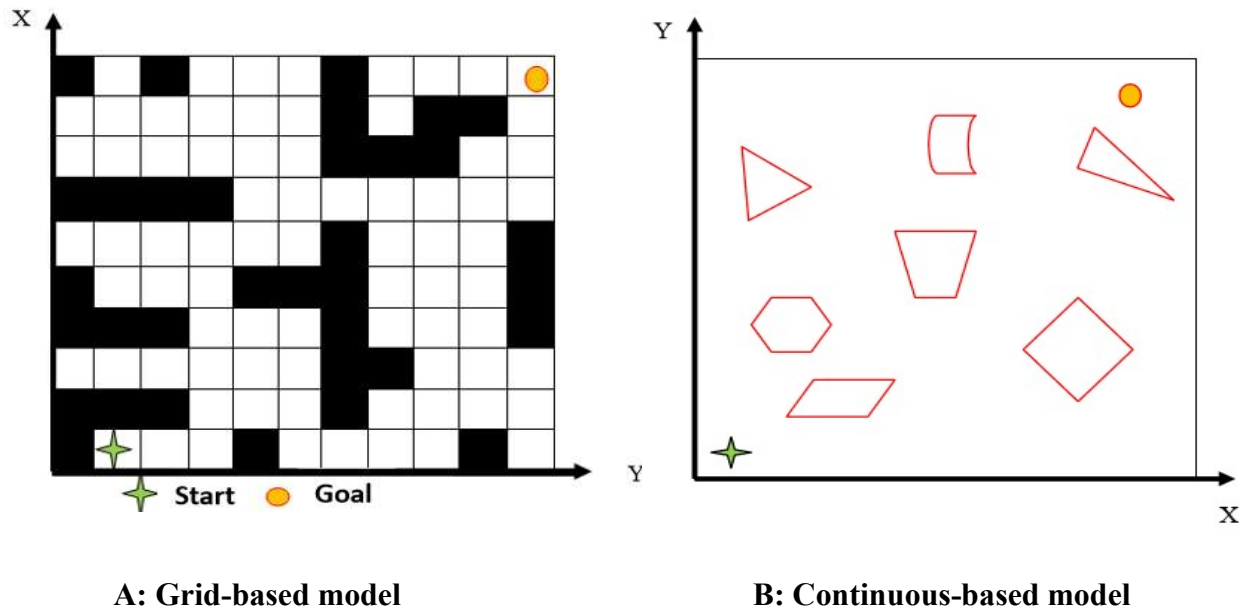


Figure (2.2): Environment models [34].

2.4 Swarm-Based Optimization Algorithms

Essentially, optimization is important to any problem including decision making, whether in engineering or economic. The decision-making task entails choosing between various options. The present study is governed by this choice in order to make the optimum decision. The goodness measure of the alternatives is described by the fitness function or performance index [36]. Optimization approaches can be classified into Deterministic and Nondeterministic (stochastic) algorithms. The deterministic algorithms depend on the mathematical nature of the problem, while the nondeterministic algorithms do not depend on the mathematical properties of a given function and are hence more appropriate for finding the global optimal solutions for any type of objective function [37].

Swarm-based algorithms have recently emerged as a family of metaheuristic (population-based) algorithms that are capable of producing low cost, fast, and

robust solutions to several optimization problems. The Swarm- based algorithms, which are used as multiple solutions in order to move through the search space during the optimization process, are known as optimization algorithms. Some of the effective algorithms that simulate the social behavior of animals, such as birds, fish, bees, ants, flies and even germs are called Nature-Inspired Algorithms [13].

In this thesis, two types of Natural-Inspired Algorithms are applied; the first is Particle Swarm Optimization (PSO) algorithm, and the second is Firefly (FF) algorithm.

2.4.1 Particle Swarm Optimization (PSO) Algorithm

Particle swarm optimization (PSO) algorithm is an optimization algorithm that was invented by Dr. James Kennedy, a social psychologist, and Dr. Russell Eberhart, an electrical engineer, in 1995 [15, 16]. The PSO simulates the social behavior of schools of fish and flocks of birds. When a fish or bird looking for food finds a good path to the food. Immediately, it transfers the information to the whole individuals. Then, the rest of the swarm becomes slow and takes a fancy to the food in gradual way [8]. Similarity to other evolutionary computation techniques such as genetic algorithms (GAs), PSO is a population-based algorithm, where each individual is called (particle) and each particle is a possible solution to the optimized problem. However, unlike GAs the PSO does not have cross over and mutation operators. PSO implements the simulation of a social behavior instead of implementing the survival of the fittest individuals [38].

2.4.1.1 The Original PSO Algorithm

The space of solution is searched with multiple particles (individuals) where by every particle is directed based on its own experience and the experience of the whole swarm. The basic variables of this algorithm are as follows: position of

particle that represents the potential solution, velocity of the particle that represents the change of position in the current iteration and the objective function that is the measure of success of the particle [38].

In the PSO work, a random position and velocity of each particle are existed, and the particles start to fly around the search space with uniform numbers of $[V_{min}, V_{max}]$ and $[X_{min}, X_{max}]$ respectively as in equations (2.20) and (2.21) [8].

$$V_i = V_{min} + a_1(V_{max} - V_{min}) \quad (2.20)$$

$$X_i = X_{min} + a_2(X_{max} - X_{min}) \quad (2.21)$$

Where, a_1 and a_2 are random numbers between [0-1].

Now, particles mutually shared their experience and they will approximate to one global best position ever visited by all particles, as shown in Figure (2.3) [39].

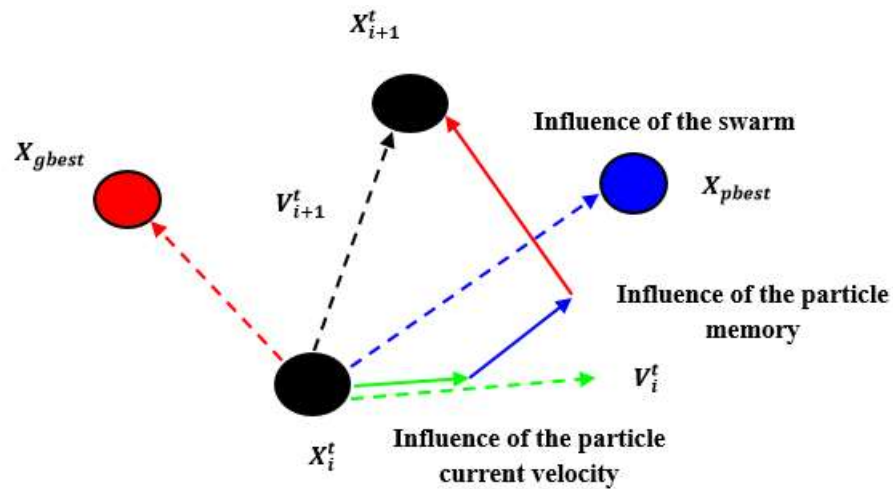


Figure (2.3): Basic structure of PSO for global best approximation [39].

Mathematically, equation (2.22) is used to update the speed of each particle, while equation (2.23) represents the update of position according to its previous velocity and position. In a gradual way, particles reach the global best positions by

communicating the personal best (X_{pbest}) and global best (X_{gbest}) to each other [8, 38].

$$V_i^{t+1} = \omega V_i^t + c_1 r_1 [X_{pbesti} - X_i^t] + c_2 r_2 [X_{gbest} - X_i^t] \quad (2.22)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (2.23)$$

Where:

V_i^t represents the rate of the position change (velocity) of the i particle.

X_i^t represents the position of the i particle.

t and $t+1$ are denoted by the actual and next iteration respectively.

X_{pbest} represents the best weight of the particle.

X_{gbest} represents the best particle among all the particles in the swarm.

c_1 indicates the individual-learning rate, while c_2 indicates the group-learning rate.

These parameters reveal the relative importance of the particle's own best position to its neighbor's best position. In other words, they are responsible for varying the speed of individual towards X_{pbest} and X_{gbest} . In spite of constants c_1 and c_2 are not critical parameters for determining the convergence of PSO algorithm, a correct setting may increase the algorithm convergence.

r_1 & r_2 are uniform distributed random numbers in the range between [0-1].

Additionally, Shi and Eberhart proposed inertia weight (ω) in 1998 [40]. This symbol is responsible for dynamically adjusting the speed of particles in order to allow the individuals to converge more efficiently and accurately. Therefore, it is responsible for balancing between global and local search, then needing less number of iterations for PSO algorithm to converge. A high value of inertia weight leads to a global search, on another hand, a small value implies in a local search. A balance

between local and global search can be achieved by using linearly decreasing inertia weight strategy as in equation (2.24) [8, 38].

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{T_{max}} * t \quad (2.24)$$

Where, ω_{max} and ω_{min} are maximum and minimum values of inertia weight factor (ω), respectively, while T_{max} is the maximum number of iterations.

2.4.1.2 The PSO Algorithm Flow

In summary, the PSO process is as follows [6]:

Step 1: Initialize the position and velocity randomly for each particle in the swarm.

Step 2: Evaluate the objective function for each particle in the swarm.

Step 3: Check, if the objective value is better than the personal best (X_{pbest}) objective value in history, the current objective value set as a new (X_{pbest}).

Step 4: From all the individuals or neighborhood, choose the particle with the best objective value and set it as (X_{gbest}).

Step 5: For each particle in the swarm:

- Update the particle velocity as in equation (2, 22).
- Update the particle position as in equation (2, 23).

Step 6: Repeat to step two *until* stopping criteria is satisfied.

Figure (2.4) shows the general PSO process [41].

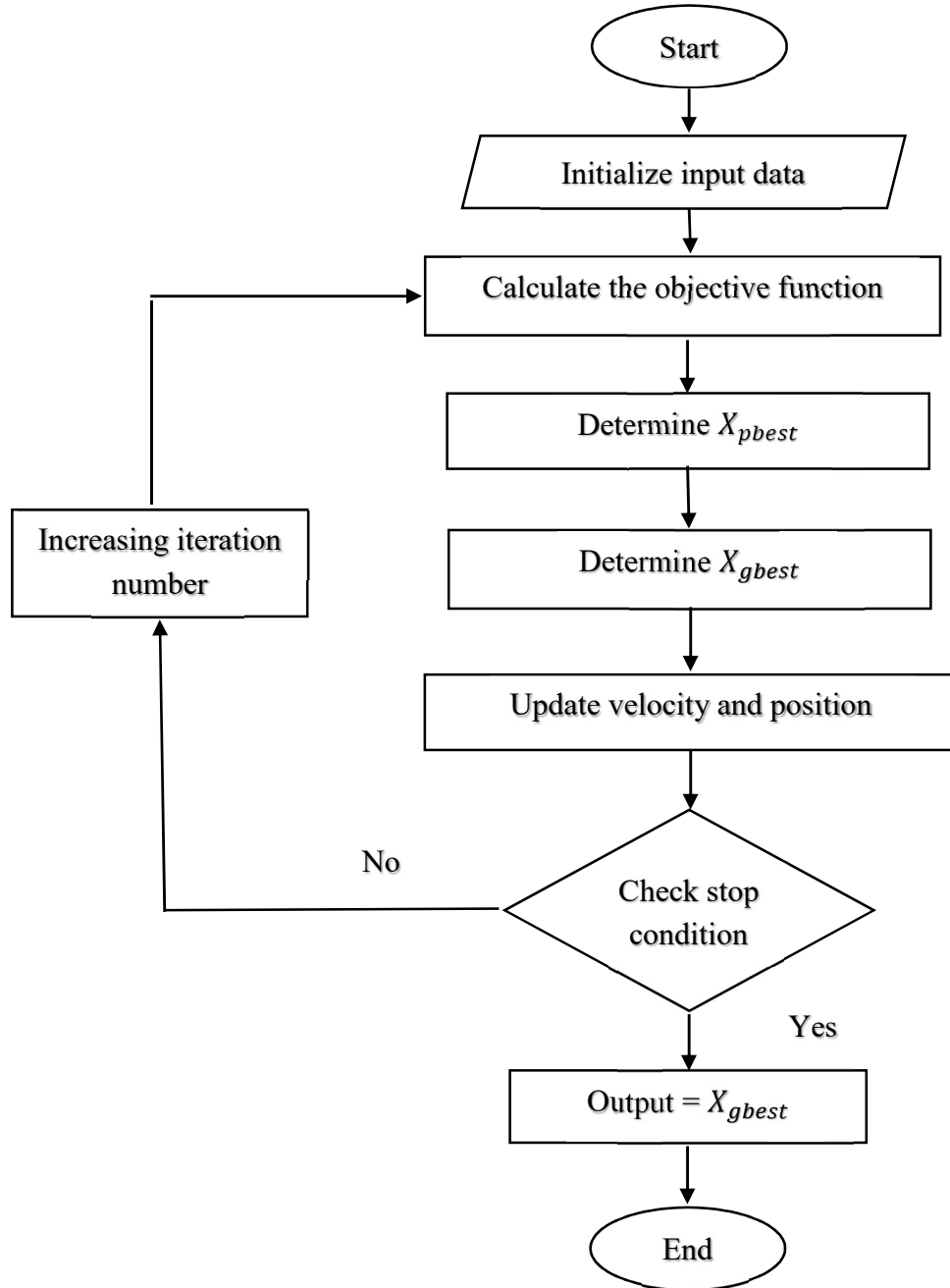


Figure (2.4): The flowchart of PSO algorithm [41].

2.4.2 Firefly (FF) Algorithm

The Firefly (FF) algorithm is an optimization algorithm that was invented by Dr. Xin – She Yang in 2009, which was based on an inspiration from the natural behavior of tropical fireflies. The FF algorithm tries to simulate the attraction behavior of fireflies and lighting pattern [17].

The sky is filled with the light of fireflies is a marvelous sight in summer in the moderately temperature regions. There are almost two thousands firefly species, and most of them produce short and rhythmic flashes. The pattern observed for these flashes is unique for most of times for a specific species. The rhythm of the flashes, rate of flashing and the amount of time for which the flashes are observed are together forming a kind of a pattern that attracts both the males and females to each other [17]. The primary purpose for a firefly’s flash is to act as a signal system to attract other fireflies. It has been successfully employed to find the optimal values of various test functions [42].

2.4.2.1 Firefly Mainframe

However, because an adaptation of the natural behavior of the fireflies in an algorithm is too complex, the following idealized rules are considering by firefly developing [18]:

- 1) The firefly is no gender-specific. Therefore, it will fly to more attractive and large brightness companion regardless of its gender.
- 2) Firefly attractive size is proportional to its brightness. Moreover, its brightness decreases with the distance between individuals. If there is no brighter or more attractive one, then it will fly randomly.
- 3) The brightness or attractiveness of a firefly is determined by the specified value of the objective function.

2.4.2.2 The Original FF Algorithm

There are two important points in the FF algorithm: the first point is the variation of the flash intensity and the second point is the attractiveness formulation. For simplicity, one can always assume that the attractiveness of a firefly is determined by its brightness that in turn is associated with the encoded objective function.

1. Flash Intensity or Brightness

For simplicity, the light intensity $F(rr)$ varies according to the inverse square law as in equation (2.25) [43].

$$F(rr) = \frac{F_s}{rr^2} \quad (2.25)$$

Where, F_s is the light intensity at the source. For stated medium with a fixed flash absorption factor (γ), the flash intensity (F) varies with the distance (rr) as in equation (2.26) [43].

$$F = F_o e^{\gamma rr^2} \quad (2.26)$$

Where, F_o is the original flash intensity.

2. Attractiveness towards Brightness

The form of the attractiveness function of a firefly since its proportional to the flash intensity seen by adjacent fireflies is as in equation (2.27) [43].

$$\beta = \beta_o e^{-\gamma rr^2} \quad (2.27)$$

Where, rr is the distance between any two fireflies, and β_o is the attractiveness at $rr = 0$.

3. Distance between Fireflies

The distance between firefly i and firefly j at (X_i, Y_i) and (X_j, Y_j) is Cartesian distance as in equation (2.28) [43].

$$rr = \sqrt{(X_i - X_j)^2 - (Y_i - Y_j)^2} \quad (2.28)$$

4. Movement of Fireflies

Finally, the movement of firefly (i) that attracted to more attractive firefly (j) is calculated by equation (2.29) [43].

$$X_i = X_i + \beta_o e^{-\gamma rr^2} (X_j - X_i) + \tilde{\alpha} \mathbf{\xi}_i \quad (2.29)$$

Where, the first part in equation above gives the current position of the firefly, whereas the second part is responsible for attractiveness, while ($\tilde{\alpha}$) is a randomization parameter and ($\mathbf{\xi}$) is the vector of random variables, which make the investigation of the search distance more effective. A firefly will be directed towards the brighter one, and if there is no brighter one surrounding to it, then it will move randomly as in equation (2.30) [43].

$$X_i = X_i + \tilde{\alpha} (rand - 0.5) \quad (2.30)$$

2.4.2.3 The FF Algorithm Flow

In summary, the FF optimization process is as follows [44]:

Step 1: Basic initialization of algorithm parameters.

Step 2: Initializing the position randomly for each firefly in the swarm. Then, the objective function value is calculated as the respective maximum fluorescence fireflies' brightness.

Step 3: By the equations (2.26) and (2.27), calculating the relative brightness of the firefly population F and attractiveness β , according to the relative brightness of the decision to move the direction of fireflies.

Step 4: According to equation (2.28) updating the spatial location of the firefly.

Step 5: Repeating to step two until the stopping criterion is satisfied.

Step 6: Taking the output of the global extreme point and the best individual values.

Finally, Figure (2.5) shows the procedure of the FF process [45].

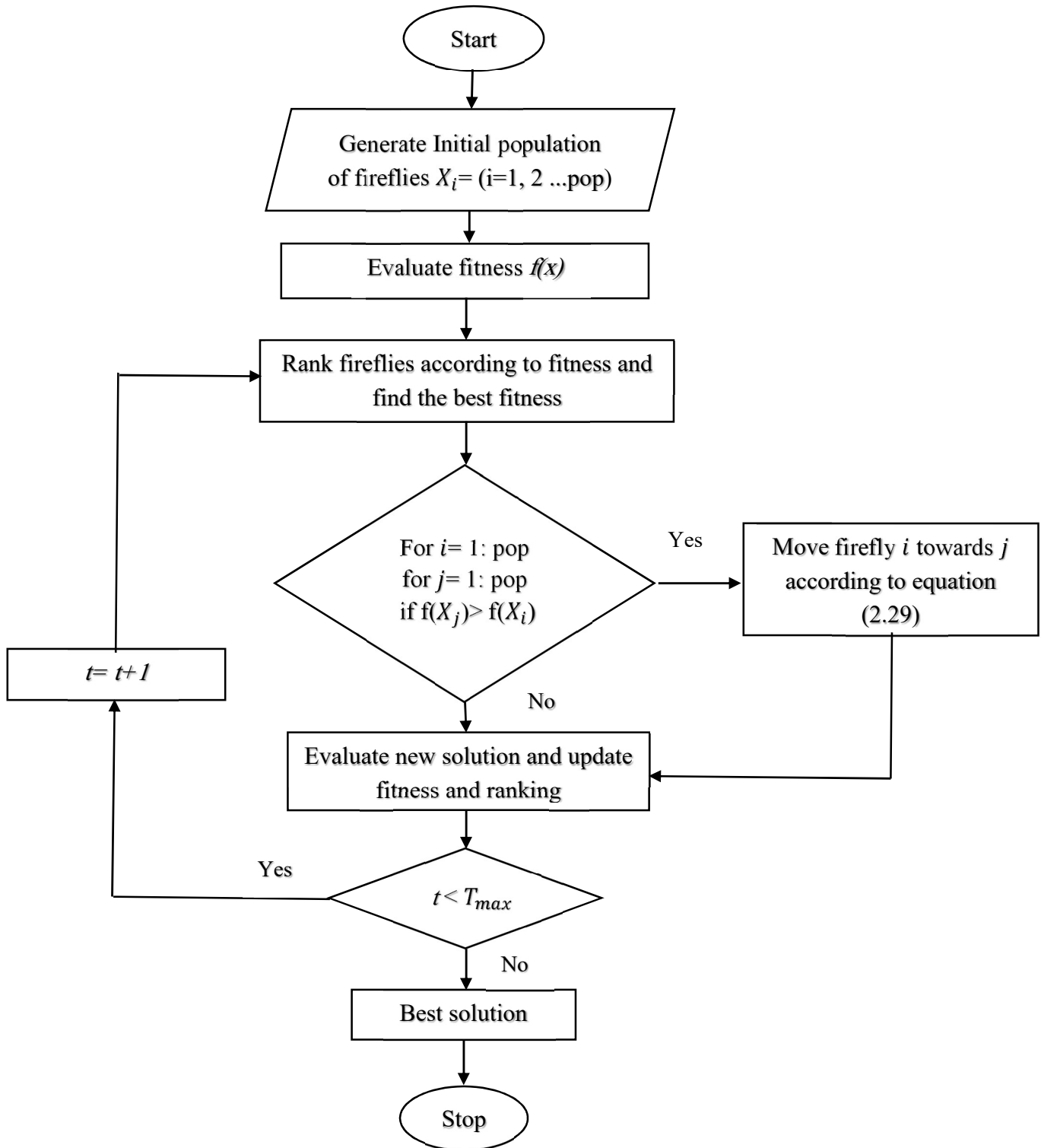


Figure (2.5): The flowchart of FF algorithm [45].

CHAPTER THREE

MODELING AND OPTIMIZATION BASED PATH PLANNING

CHAPTER THREE

MODELING AND OPTIMIZATION BASED PATH PLANNING

3.1 Introduction

In this chapter, path planning based decision making systematic is presented and three optimization algorithms are implemented for solving this problem. The first algorithm represents the enhanced Particle Swarm Optimization (PSO) algorithm, the second algorithm represents the original firefly (FF) algorithm and the third algorithm represents the proposed hybrid firefly with enhanced Particle Swarm Optimization algorithm for global path planning. This chapter also describes how the various optimization algorithms are used for solving multiple mobile robot problem.

3.2 Path Planning Based on Decision Making Systematic

Based on the systematic of decision making that was described in Chapter One (section (1.1)), path-planning problem can be applied, as shown in Figure (3.1). As stated before, there are four phases in DM process namely, intelligence, design, choice and implementation. In intelligence phase, the first step needs to identify a good understanding of path planning problem and environment boundaries, the second step requires to collect information about this problem, such as type of robot (arm or mobile), environment (static or dynamic) and planning type (offline or online) while the third step needs to analyze the requirements (constraints) that describe a set of the feasible solutions of path planning problem. The design phase (core of the systematic) includes three steps, formulating the kinematic model of differential drive wheeled mobile robot and its environment modeling, applying an optimization algorithm to obtain a set of feasible solutions, and then obtaining the

optimization criteria, such as energy, safety, distance and time. In choice phase, it includes the solution (path) evaluation based on optimization criteria and finally having a solution (path for the mobile robot). In implementation phase, the path-planning problem on a real environment is implemented.

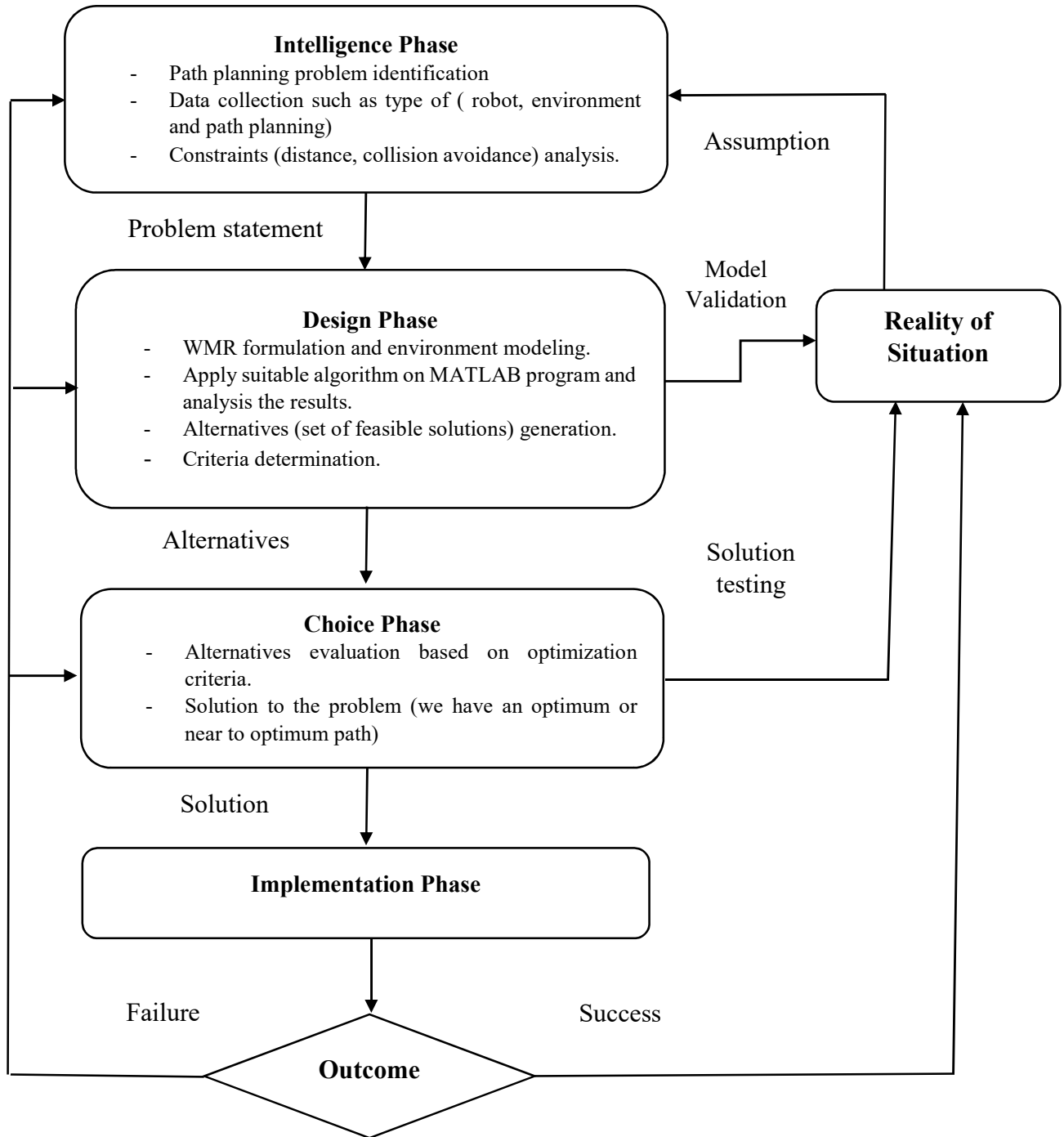


Figure (3.1): Path planning based on decision-making systematic.

3.3 Modeling of the Environment and Obstacles

In this thesis, the wheeled mobile robot environment, which is occupied by a number of static obstacles, is represented by a free-space model, consider a two-dimensional (X, Y) square map. This model makes the representation of obstacles and calculation of distance easier. In the real world, obstacles maintain all kinds of shape and size that make it is hard to model them, therefore, in order to simplify the task of modeling of the obstacles, only circles are used. Then, each obstacle must be inflated by the size of the mobile robot's radius (depending on the mobile robot type) in order to assure the safety of robot while trying in the environment, as shown in Figure (3.2).

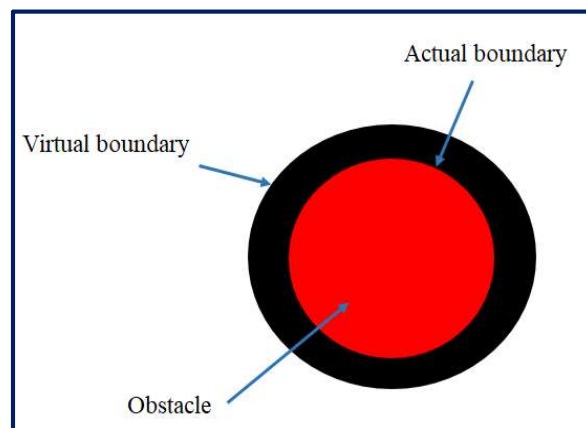


Figure (3.2): Original and virtual obstacle boundaries.

As displayed in the figure above, boundaries for obstacles area are formed by their actual boundaries plus a safety distance that is defined with consideration to the mobile robot size that is treated as a point in the workspace.

3.4 Path Construction

This process can be implemented as follows: The optimization algorithm will generate search waypoints randomly. Then, these points will be connected by polynomial interpolation. So, the cubic polynomial interpolation has been used in

this thesis to achieve this purpose. Therefore, the path has smoothness and the and as a result, velocity is not changed abruptly in this path when the mobile robot follows the smooth path. Thus, the mobile robot can be moved with a continuous velocity and acceleration without stop motion. This movement is efficient with energy and time because of the energy loss of a mobile robot is related to the smooth of the path—the smoother the path, the smaller the loss of energy. The basic form of cubic polynomial interpolation is as in equation (3.1) [46].

$$Y = a X^3 + b X^2 + cX + d \quad (3.1)$$

The primary and second-order differential functions are in equations (3.2) and (3.3), respectively [46].

$$\frac{dy}{dx} = 3aX^2 + 2bX + c \quad (3.2)$$

$$\frac{d^2Y}{dX^2} = 6aX + 2b \quad (3.3)$$

3.5 Objective Function

In this study, to obtain precise and effective solutions, two objectives are optimized: path length and path safety. The energy loss of a mobile robot is related to the length of the path—the longer the path, the greater the loss of energy. When the robot moves forward at a constant speed and only changes in velocity direction, the walking time of the robot is also related to the length. In addition, the most important factor that one needs to consider is the safety of the path. The safe path distance reflects the distance from the surrounding obstacles—the greater the safety distance, the safer the path. Therefore, this work describes the energy loss, walking time, and safety distance through the length and safety. The shortest and safest path is to be found. This is known as the_multi-objective optimization problem. The

mathematical definitions of these objectives are described in the following subsections.

3.5.1 Obstacles avoidance

The first goal is to find the obstacle-free path, which is essential to path planning to make the wheeled mobile robot travel in the workspace safely. This objective function must penalize trajectories with respect to their distance to obstacles considering the obstacles' density. To obtain a collision-free robot, the safe distance between the mobile robot and the obstacle should be larger than $(W/2)$, where W represents the distance between right and left wheel.

The most important property of metaheuristic (population-based) algorithms, including the PSO and FF algorithms, is that they are designed for the unconstrained optimization problems; they can also be adapted to the constrained optimization problems by using penalty. If a solution does not satisfy the constraints, this solution is not acceptable, even if the value of the objective function is minimum. So, a penalty function is added to the objective function.

The distance between a path point and the center of obstacle (D_{obs}) is calculated as in equations (3.4) and (3.5) in order to check the feasibility condition, where the path is touching or passing the obstacle.

$$dist [O (k), S_{ref}(i)] = \sqrt{(S_{ref}^x(i) - O^x(k))^2 + (S_{ref}^y(i) - O^y(k))^2} \quad (3.4)$$

$$D_{obs}(S_{ref}, O) = \sum_{k=1}^m \sum_{i=1}^{np} dist [O (k), S_{ref}(i)] \quad (3.5)$$

Where, k is the current obstacle, m is the total number of obstacles in the workspace, np is the total number of interpolation points that used to connect the waypoints, O is the obstacles positions and S_{ref} represents the reference path points.

Along the learning process for the actual (optimal) path, the approach of obstacle avoidance must be called contentiously in order to prevent the interpolation points from entering the obstacle regions. This can be done by defining the safety variable for each individual (path) as an indicator of whether there is a collision between path interpolation point (S_{ref}) and the obstacles (O) or not. The quantitative description of checking the collision is as explained in equation (3.6).

$$Safety(i) = Max \begin{cases} 1 - \frac{D_{obs}(S_{ref}, O)}{R_{obs} + 0.5 W} & collision \\ 0 & free \end{cases} \quad (3.6)$$

3.5.3 Minimum path length

The second and main goal is to minimize the distance between the starting point (X_s, Y_s) and the target point (X_t, Y_t). This makes the wheeled mobile robot travel in the workspace with minimum travelling time and distance. As mentioned before, the search waypoints that give the path lineaments is proposed by the optimization algorithms. The cubic polynomial will connect the start point (S) with end point (T) through the waypoints that are used to calculate the objective function that will be minimized by the optimization algorithms. Based on the Euclidean distance, the distance between two consecutive points can be calculated, as shown in equation (3.7).

$$dist(P_i, P_{i+1}) = \sqrt{(X_i - X_{i+1})^2 + (Y_i - Y_{i+1})^2} \quad (3.7)$$

Where $p_i = (X_i, Y_i)$ and $p_{i+1} = (X_{i+1}, Y_{i+1})$ denotes two consecutive points.

A path consists of some interpolation points (np), and the total distance (ML) of the path (i) can be calculated as in equation (3.8). By summing all the points of this path, the length of the entire path can be obtained.

$$ML(i) = \sum_{i=1}^{np} dist(P_i, P_{i+1}) \quad (3.8)$$

The metaheuristic algorithms are designed to find the minimum value of the objective function (Euclidian distance) within the bounds of the constraint (avoidance of obstacles). The total function is called fitness function. Namely, if constraints are in a feasible region, then the penalty function is equal to zero else, the fitness function is penalized as in equation (3.9), since the fitness value, measures the fitness of the solution to the objective function and because of path planning is the minimization problem, which is due it is looking for shortest path length.

$$Fitness(i) = \frac{1}{ML(i) + \mathcal{H} * Safety(i)} \tag{3.9}$$

Where, \mathcal{H} is the obstacle zone weight factor, which is used to balance the proportion of the path length and $Safety(i)$ is penalty added to reduce the fitness of $path i$ that passes through an obstacle and yields in preventing taking it.

Finally, the overall systematic provides the shortest, smooth and safe path on the workspace for static known environment that can be abstracted, as shown in Figure (3.3).

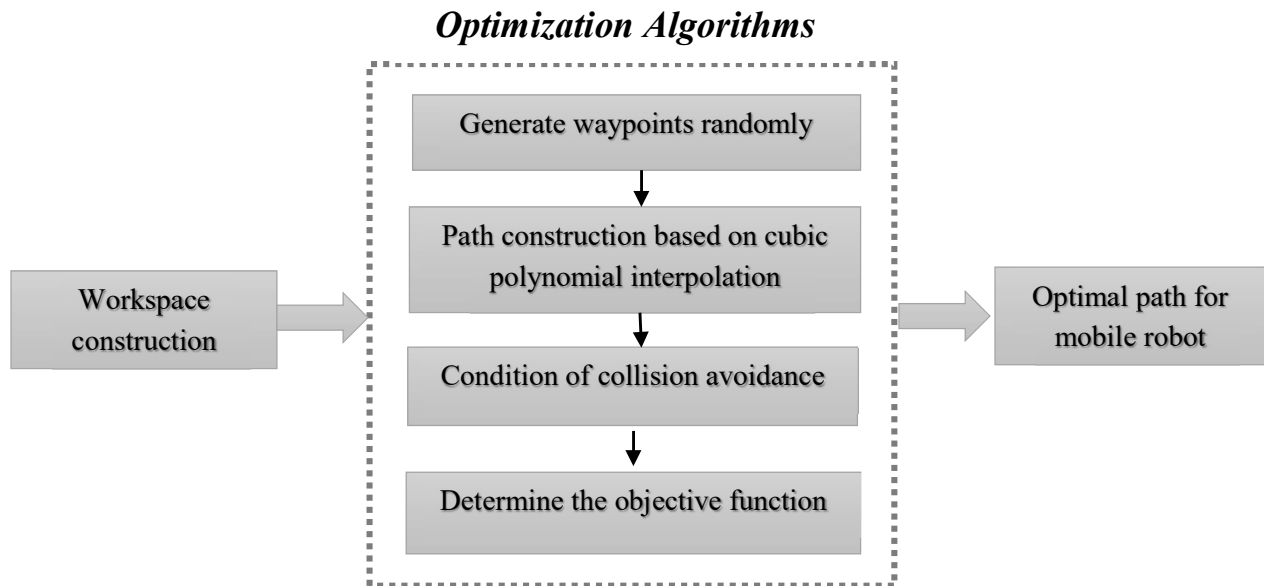


Figure (3.3): Overall scheme of applying optimization algorithms based path planning.

Following that, the Nature-Inspired algorithms are used in order to create paths for three mobile robots in the same map with the same destination point, but different source points are for each respective robot. With the implementation of multiple mobile robot, the effect on the various optimization techniques has to be considered. With more robots in play, it is decided to run the optimization methods for each robot at the same time because these methods need to be reconfigured in order to find all paths for each mobile robots that means three times of computational cost. That computational cost would scale even higher with the addition of more mobile robots.

The overall systematic of the multiple mobile robot implementation can be abstracted, as shown in Figure (3.4).

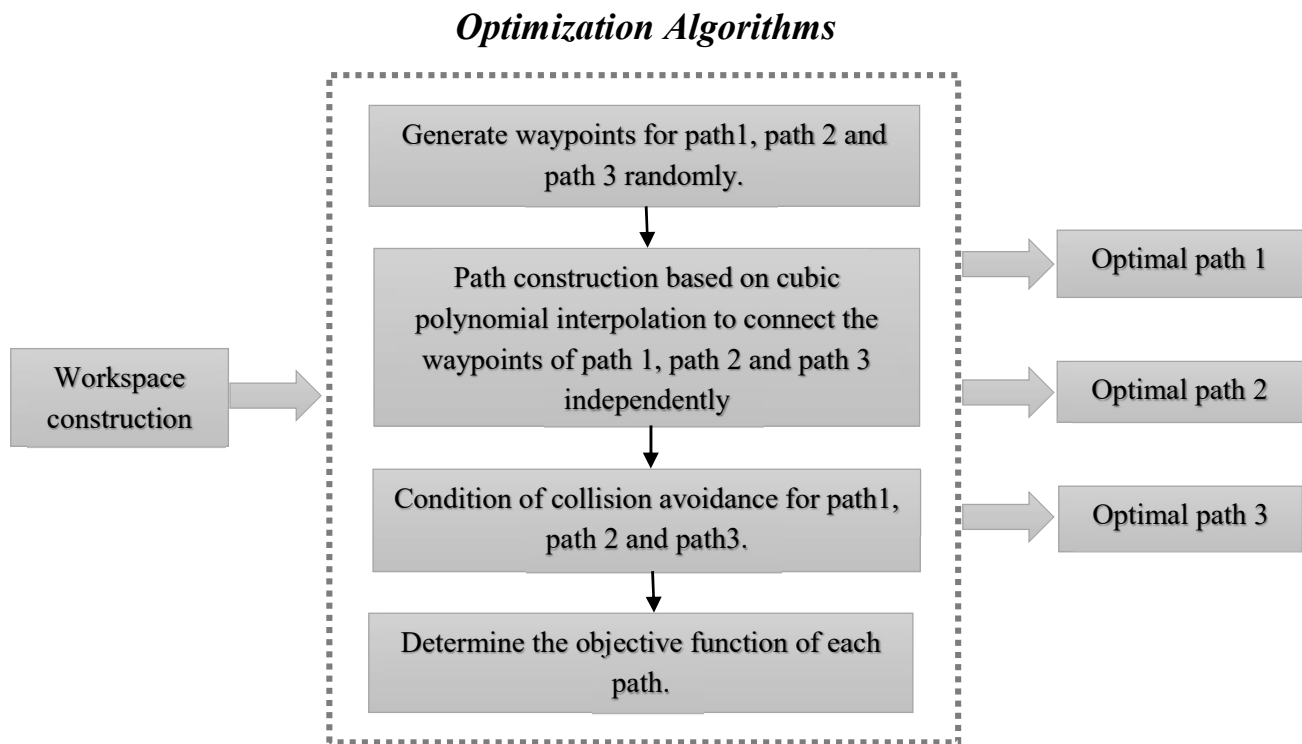


Figure (3.4): Overall scheme of applying optimization algorithms based path planning for mobile robots.

Based on previous idea, the problem of multi robot path planning can be described as follows:

1. Plan a shortest path for each robot from its start point to the end separately. This step ensures the lower complexity and completeness for each robot.
2. Use the kinematic equations to get the velocity arrangements so that the all robots can move on their paths designed before without collisions. This step ensures the optimality and will not influence the completeness for each mobile robot.

Moreover, equation (3.10) [28] investigates whether the new desired trajectory is optimal travelling time (T_v) for the wheeled mobile robot.

$$T_v = \varphi \times T_s \tag{3.10}$$

Where, T_s is the sampling time and φ is denoted as the number of samples.

3.6 Path Optimization Algorithms

There are three principles used to organize the mobile robot movement in order to reach the goal position without collision with the obstacles or other mobile robot in the workspace. These principles are:

- 1- Firstly, the mobile robot is the next position in order to align itself to a goal.
- 2- Check if there is a collision with the static obstacle. This may happen with an obstacle found in the next position. To avoid such collision, the mobile robot path changes its position with a penalty. In addition, it may be there is a collision with another mobile robot in the case of more than one mobile robot trying to take the same position. To avoid such a collision, time and velocity are needed to be calculated to know which mobile robot has a high velocity or minimum time and can reach the target firstly based on the desired path length.
- 3- Finally, if the mobile robot can align itself to the goal without any collision with the obstacle or other mobile robot, it will move to the next position safely.

In the next subsections, Chaotic PSO, original FF and the proposed hybrid FFCPSO algorithms are used to locate the optimal control points (waypoints) within the interpolation points to find the optimum path from start to target points.

3.6.1 Improved Basic PSO Algorithm

Although the PSO algorithm has the advantages of simple structure, easy to be described and implemented, uses a relatively small size of population, adjusts the few parameters, takes on fast convergence, good robustness and higher computational efficiency than the traditional methods, it is easy to fall into a local extreme value and cannot obtain the global optimal solution. There are two essential reasons for this problem namely, firstly the character of the optimization function and secondly the inappropriate parameters design and population size of algorithm in the operation process. These two reasons will rapidly vanish the diversity of particles in the calculation process and cause the premature problem. In order to improve the ability of global searching and prevent a slide into the premature convergence to local minima, PSO and chaotic map technique are combined to form a Chaotic Particle Swarm Optimization (CPSO) algorithm, which practically combines the behavior of the chaotic searching with the population-based evolutionary searching ability. Chaos is random and unpredictable, which can be described as a bounded non-linear system with deterministic dynamic behavior that has stochastic features. Logistic map as one of the simplest chaotic maps which has been paid much attention by the researchers over the last few decades is employed in this thesis for constructing the hybrid PSO as described in equation (3.11) [47].

$$Z^{t+1} = U Z^t(1 - Z^t) \quad 0 \leq Z^t \leq 1 \quad (3.11)$$

Where, U is a control parameter with a real number from [0 to 4]. Although (3.11) is deterministic, it exhibits the chaotic dynamic when $U = 4$ and $0 \leq Z^t \leq 1$. It

exhibits the sensitive dependence on the initial conditions, which is the basic feature of chaos. The new inertia weight factor (ω_{new}) is defined by multiplying the inertia weight in equation (2.24) by the logistic map in equation (3.11) as shown in equation (3.12) [47].

$$\omega_{new} = \omega \times Z^{t+1} \quad (3.12)$$

Finally, in order to improve the global search capability of basic PSO algorithm, one has to introduce a new velocity update as in equations (3.13) and (3.14) [47].

$$V_{ix}^{t+1} = \omega_{new} V_{ix}^t + c_1 r_1 (P_{besti} - X_i^t) + c_2 r_2 (G_{best} - X_i^t) \quad (3.13)$$

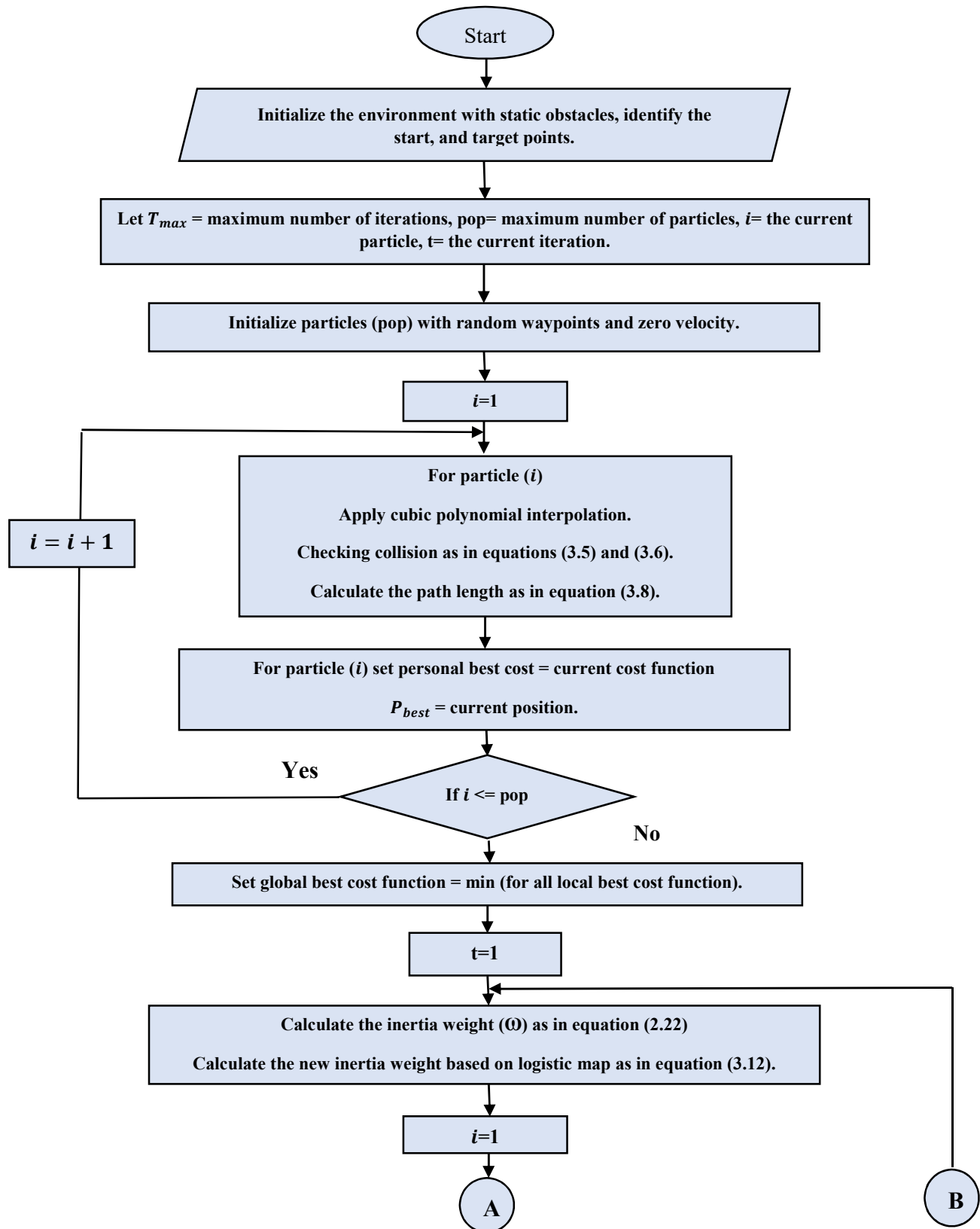
$$V_{iy}^{t+1} = \omega_{new} V_{iy}^t + c_1 r_1 (P_{besti} - Y_i^t) + c_2 r_2 (G_{best} - Y_i^t) \quad (3.14)$$

In the next iteration, these particles are then moved to the next position according to equations (3.15) and (3.16).

$$X_i^{t+1} = X_i^t + V_{ix}^{t+1} \quad (3.15)$$

$$Y_i^{t+1} = Y_i^t + V_{iy}^{t+1} \quad (3.16)$$

The flowchart of Chaotic PSO algorithm based on path planning application is illustrated in Figure (3.5).



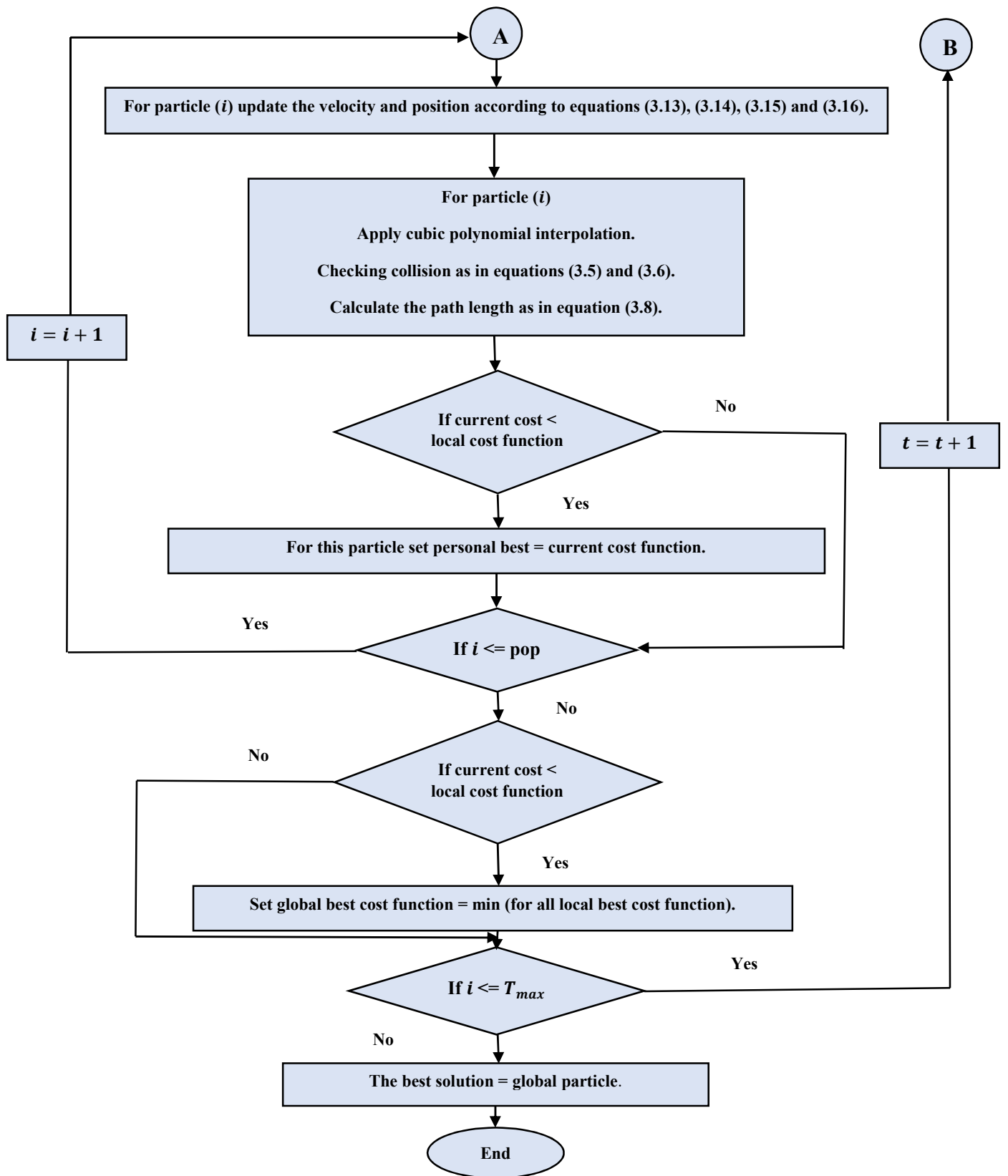


Figure (3.5): The Flowchart of CPSO Algorithm based path planning.

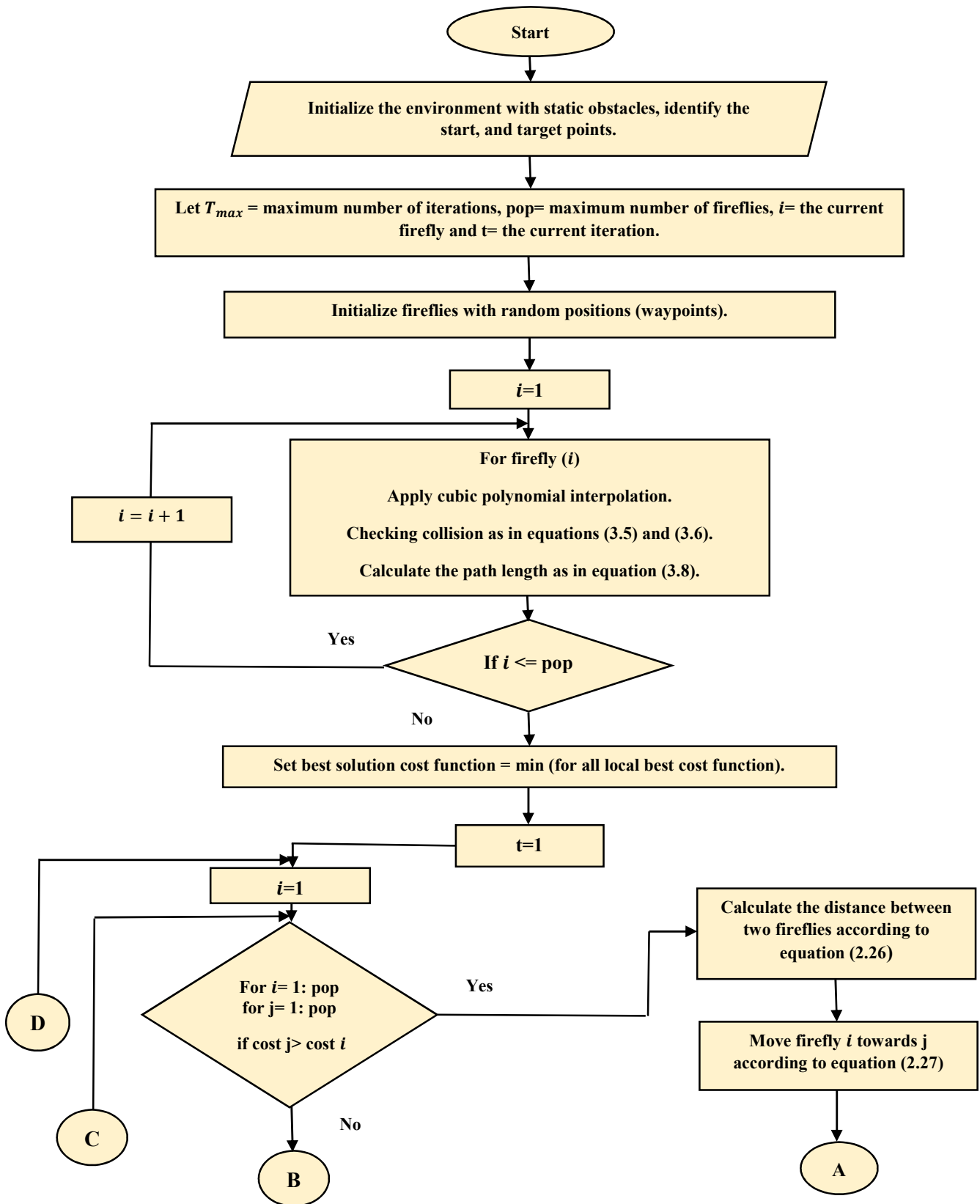
3.6.2 Firefly (FF) Algorithm

Firefly (FF) algorithm is vastly used for solving optimization and engineering problems, because of including features such as high error tolerance, automatic segmentation of the population into subgroups and non-sensitive to initial values, which can produce acceptable results. The flowchart of FF algorithm based path planning application is illustrated in Figure (3.6).

3.6.3 Proposed Hybrid FFCPSO Algorithm

Despite that the FF algorithm is widely used for solving optimization problems, the conventional FF algorithm also has some coming in term of trapping into local optima and the process of updating of the movement (position) of Fireflies is not faster. The FF algorithm does not have a velocity characteristic, and there are no parameters to use the previous best position of each firefly. Therefore, fireflies will move regardless of their previous best positions. As a result, it is advantageous for the fireflies to find a new optimum search space with a definite velocity to arrive at global optimum point very quickly. On the other hand, PSO has a faster convergence ability rather than some other population-based algorithms. Balance between exploration and exploitation in PSO can be efficiently controlled by using three control factors namely, (ω , c_1 and c_2). Thus, some modification and hybridization are proposed to overcome this problem.

In this thesis, an optimization algorithm that combines the search ability of firefly and chaotic PSO algorithms has been proposed. By using this combination, a balance between exploration and exploitation is aimed to establish and it benefits the strengths of both algorithms.



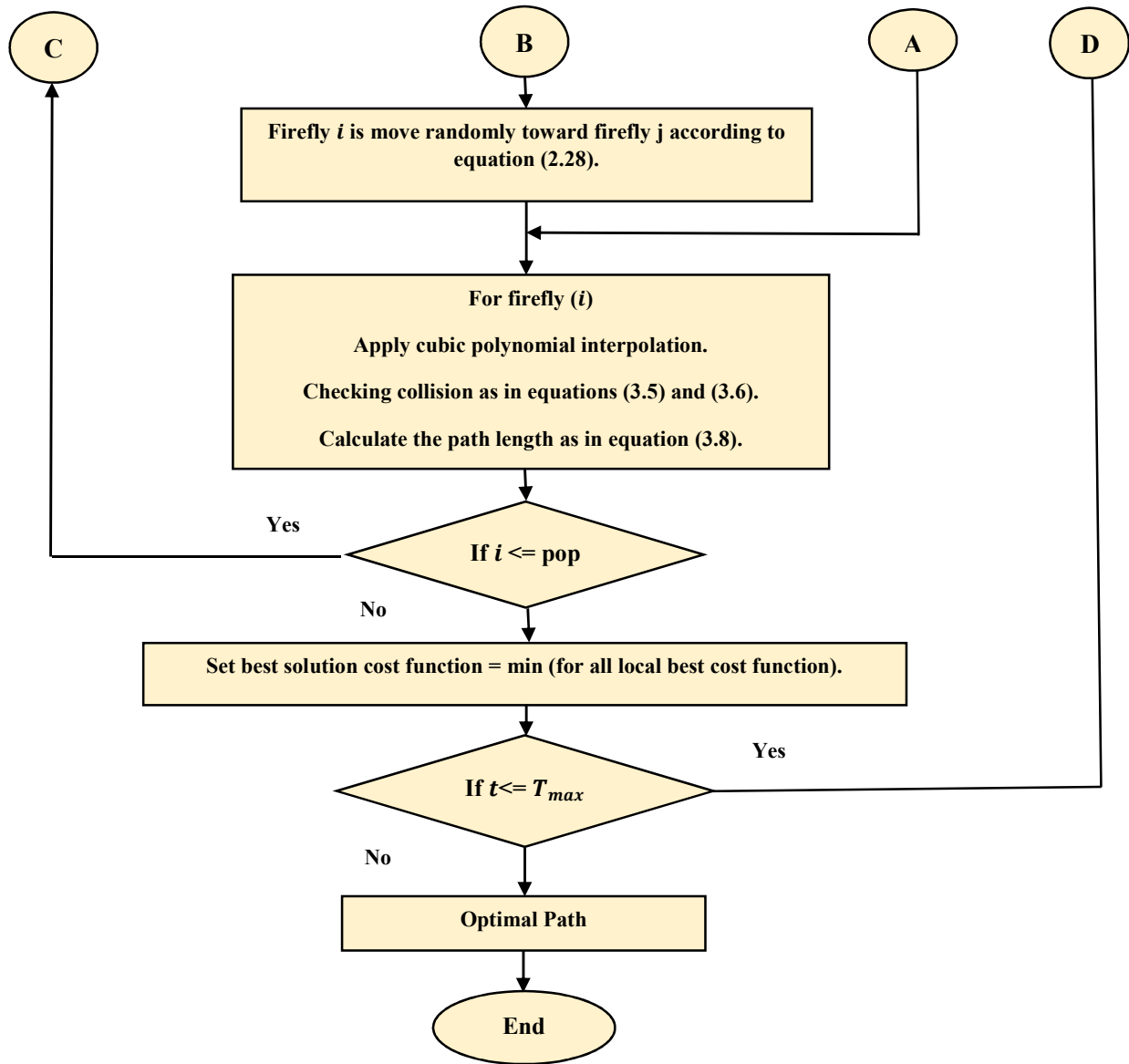


Figure (3.6): The Flowchart of FF Algorithm based path planning.

As mentioned before, fireflies have no velocity and personal best position (P_{best}) memories in comparison to particles. Previous works show that PSO is one of the popular methods for a global search. Therefore, the combination of FF and PSO will find a better solution to explore the search space by applying the capability of PSO in memorizing its previous best in determining the next possible solution.

In the suggested algorithm, the flash (light) intensity attraction step of each firefly is mutated by a PSO operator. At this step, each firefly is attracted randomly towards the global best position in the entire population. Hence, the velocity term with modification for faster convergence is appended in order to improve exploration and exploitation capability. So, the FFCPSO algorithm is modified as follows:

Firstly, the Cartesian distance between (X_i and P_{besti}) and (Y_i and P_{besti}) is in equations (3.17) and (3.18), respectively.

$$D_{px} = \sqrt{\sum_{k=1}^D (P_{besti,k} - X_{i,k})^2} \quad (3.17)$$

$$D_{py} = \sqrt{\sum_{k=1}^D (P_{besti,k} - Y_{i,k})^2} \quad (3.18)$$

Secondly, the Cartesian distance between (X_i and G_{best}) and (Y_i and G_{best}) is in equations (3.19) and (3.20), respectively.

$$D_{gx} = \sqrt{\sum_{k=1}^D (G_{best} - X_{i,k})^2} \quad (3.19)$$

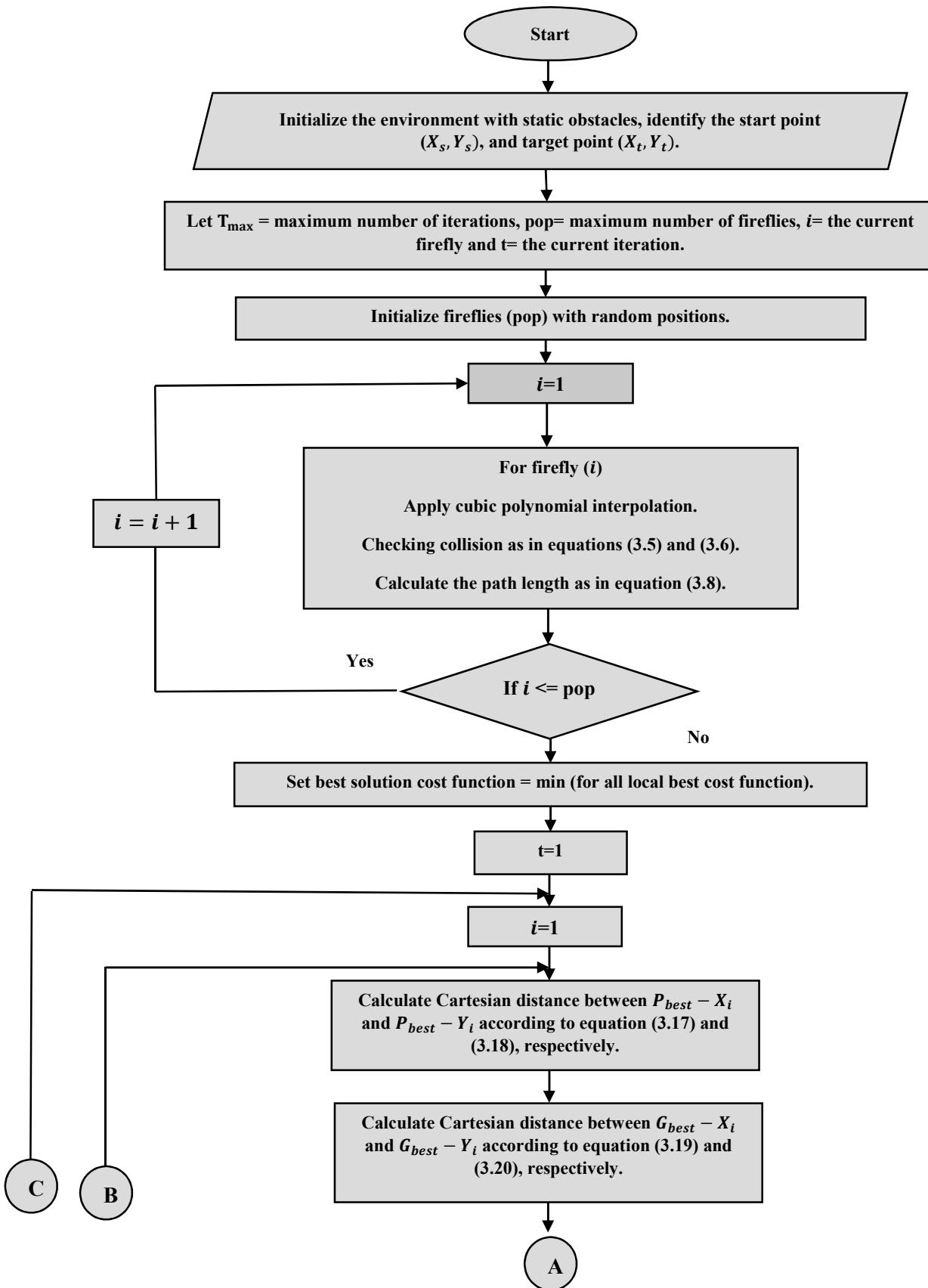
$$D_{gy} = \sqrt{\sum_{k=1}^D (G_{best} - Y_{i,k})^2} \quad (3.20)$$

Finally, the position vector is mutated by equations (3.21) and (3.22).

$$X_i^{t+1} = \omega_{new} X_i^t + c_1 * e^{-D_{px}^2} (P_{besti} - X_i^t) + c_2 * e^{-D_{gx}^2} (G_{best} - X_i^t) + \tilde{\alpha} \mathbf{E}_i \quad (3.21)$$

$$Y_i^{t+1} = \omega_{new} Y_i^t + c_1 * e^{-D_{py}^2} (P_{besti} - Y_i^t) + c_2 * e^{-D_{gy}^2} (G_{best} - Y_i^t) + \tilde{\alpha} \mathbf{E}_i \quad (3.22)$$

The flowchart of FFCPSO algorithm is illustrated in Figure (3.7).



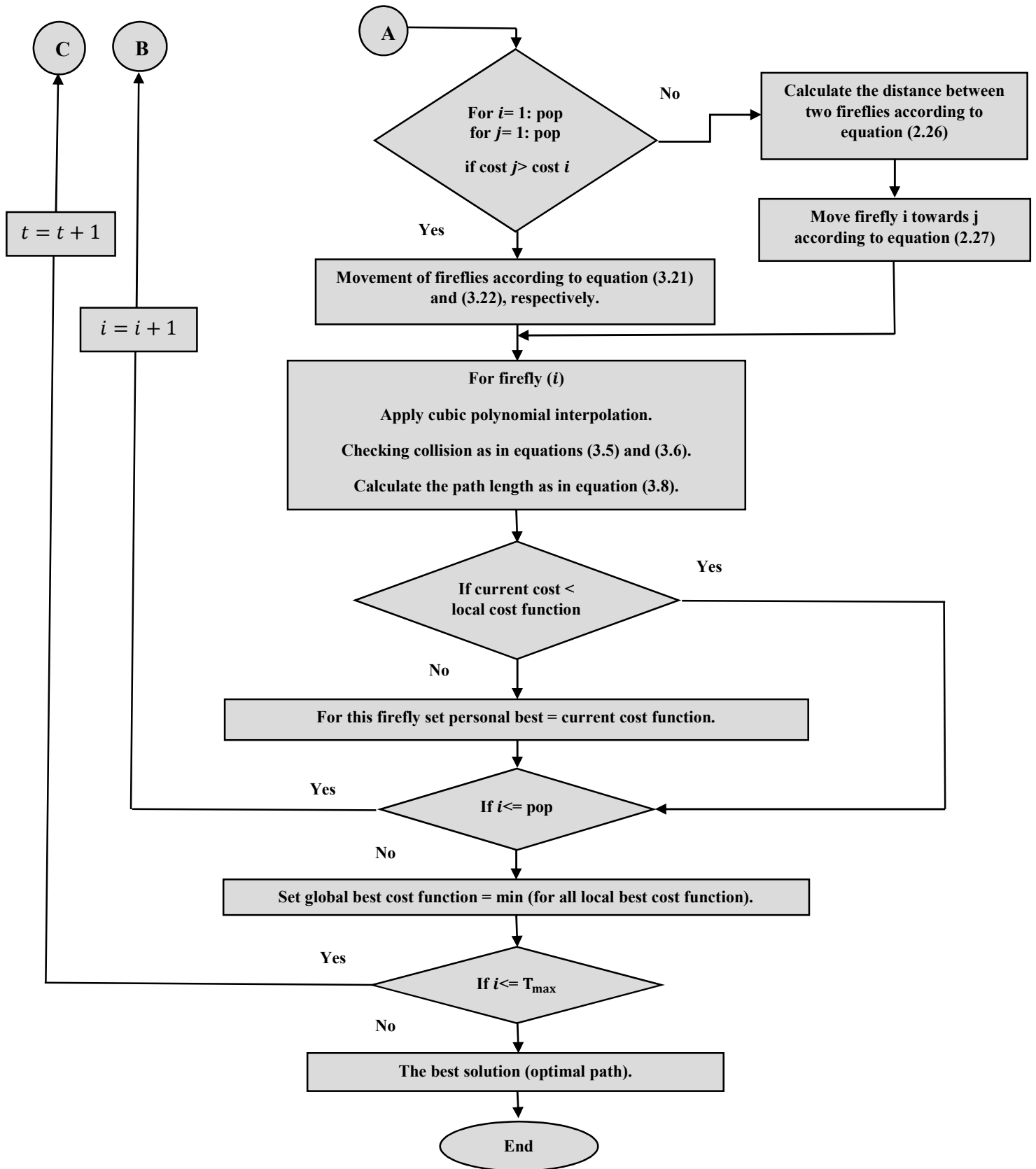


Figure (3.7): The Flowchart of FFCPSO Algorithm based path planning.

CHAPTER FOUR

SIMULATION RESULTS AND DISCUSSION

CHAPTER FOUR

SIMULATION RESULTS AND DISCUSSION

4.1 Introduction

In this chapter, the simulation results that evaluate the performance of various optimization algorithms, namely PSO, CPSO, FF and proposed hybrid FFCPSO for solving the path-planning problem of wheeled mobile robots are presented. The simulation was conducted with static known environment based on different cases. Furthermore, the collected results based on different optimization algorithms are verified through comparison with each other and also with other researcher's works. The simulation code was implemented by using the same personal computer (PC) with the Hardware and Software specifications shown in Table (4.1) in order to get an unbiased comparison of CPU times.

Table (4.1): The Hardware and Software specifications.

	Name	Setting
H. W	CPU	Core TM i7-7500U
	Frequency	2.90 GHz
	RAM	8.00 GHz
	Hard drive	953869 MB
S. W	Operating system	Windows 10
	Language	MATLAB R2014a

The kinematic model of the Non-holonomic Wheeled Mobile Robot (WMR), which has been presented in Chapter Two (section (2.2)), used in the simulation in order to calculate the robot velocities on the desired path. The simulation was executed (off line) by planning a feasible path from start position to the target position in static known environment. The National Instrument (NI) wheeled mobile robot was used in this simulation and its specifications taken from [48] are listed in Table (4.2) as follows:

Table (4.2): Wheeled NI- Mobile Robot parameters [48].

NI-Mobile Robot Parameter	Acronym	Value	Unit
Distance between two wheels	W	0.36	m
Wheel Radius	R	0.075	m
Period Time	T_s	0.1	sec
Mobile Robot Length	L	0.40	m
Max. Linear Velocity of Right and Left wheel	VR_{max} & VL_{max}	0.5	m /sec
Min. Linear Velocity of Right and Left wheel	VR_{min} & VL_{min}	-0.5	m /sec
Max. Angular Velocity of Right and Left wheel	WR_{max} & WL_{max}	6.67	rad /sec
Min. Angular Velocity of Right and Left wheel	WR_{min} & WL_{min}	-6.67	rad /sec
Max. Linear Velocity of Platform	V_{nmax}	0.5	m /sec
Min. Linear Velocity of Platform	V_{nmin}	0	m /sec
Max. Angular Velocity of platform	V_{amax}	2.77	rad /sec
Min. Angular Velocity of platform	V_{amin}	-2.77	rad /sec

4.2 Simulation Parameters' Setting

Different cases have been done with the various intelligent algorithms. In all cases, the map dimensions are (1000×1000) cm. The obstacles can be located at any place on the map except at starting and target points. The possible parameters' values of each optimization algorithm and the locations of obstacles are explained in the following subsections.

4.2.1 Parameters Setting for basic PSO and CPSO Algorithms

The flowchart of CPSO algorithm, which has been established in Chapter Three (Figure (3.5)), is used to locate the optimal waypoints (control points) to find the best path for wheeled NI- mobile robot. Table (4.3) shows the set of PSO and CPSO parameters that have been used in the simulation. For more details, see Appendix A.

Table (4.3): PSO and CPSO parameters.

PSO and CPSO parameter	Acronym	value
Max. Number of Iterations	T_{max}	80
Number of particles	pop_size	20
Acceleration Constant	c_1 & c_2	1.5
Min. Inertia Weight Factor	ω_{min}	0.4
Max. Inertia Weight Factor	ω_{max}	0.9
Random Values	r_1 & r_2	0-1
Control parameter	U	4
Chaotic Initial value	Z^1	0.3

4.2.2 Parameters' Setting for basic FF Algorithm

The flowchart of FF algorithm, which has been demonstrated in Chapter Three (Figure (3.6)), is used to optimize the waypoints (control points) to find the best path for wheeled NI- mobile robot. The wrong selection of ($\tilde{\alpha}$) can cause a small or big step increment and take away the solution in some other side far away from the global best. Table (4.4) shows the set of FF parameters that have been used in the simulation. For more details, see Appendix A.

Table (4.4): FF parameters.

FF parameter	Acronym	value
Max. Number of Iterations	T_{max}	80
Number of fireflies	pop_size	20
Flash absorption coefficient	γ	1
Initial attractiveness parameter	β_o	2
Randomness parameter	$\tilde{\alpha}$	0.2

4.2.3 Parameters' Setting for Hybrid FFCPSO Algorithm

The flowchart of FFCPSO algorithm, which has been demonstrated in Chapter Three (Figure (3.7)), is used to optimize the waypoints (control points) to find the best path for wheeled NI- mobile robot. Table (4.5) shows the set of FFCPSO parameters that have been used in the simulation.

Table (4.5): FFCPSO parameters.

FFCPSO parameter	Acronym	value
Max. Number of Iterations	T_{max}	80
Number of fireflies	pop_size	20
Acceleration Constant	c_1 & c_2	1.5
Flash absorption coefficient	γ	1
Initial attractiveness parameter	β_0	2
Randomness parameter	$\tilde{\alpha}$	0.2
Min. Inertia Weight Factor	ω_{min}	0.4
Max. Inertia Weight Factor	ω_{max}	0.9
Control parameter	U	4
Initial value	Z^1	0.3

4.2.4. Parameter Setting of Start and Target Position

In case A, single wheeled NI-mobile robot was used on a static environment. In case B, three wheeled NI- mobile robots were used, these robots have different start positions and same target position. While in case C, three follow up-wheeled NI-mobile robots were used, each robot has its start position and then follows up other robot to the same target. These locations are ordered in Table (4.6) as follows:

Table (4.6): Start and target definition for all cases.

Case No.	Robot No.	(X_s, Y_s)	(X_t, Y_t)
A	1	(100,100)	(900,900)
B	1	(500,100)	
	2	(100,200)	
	3	(700,0)	
C	1	(100,150)	
	2	(100,100)	
	3	(100,50)	

4.2.5. Parameter Setting of Obstacles Positions

The locations of all static obstacles in all cases are ordered in Table (4.7) as follows:

Table (4.7): Obstacles definition for all cases.

Obstacle No.	R_{obs} (cm)	Center (X_{obs}, Y_{obs})
1	34	(100,820)
2	64	(150,450)
3	40	(300,700)
4	34	(400,300)
5	54	(600,600)
6	40	(610,210)
7	34	(800,800)
8	60	(900,400)
9	64	(900,100)

4.3 Simulation Results

The four metaheuristic (population-based) optimization algorithms are implemented based on a path-planning application in order to find optimal or near to optimal actual paths for WMRs with three different cases. The paths are found in order to show the effectiveness of each algorithms and how can achieve three objectives (distance, smoothness and safety) as a first step, and then the distances of these paths between the mobile robot initial position and terminal position are determined based on equation (3.8). The second step can be achieved by obtaining the desired paths equations for the actual paths by using the basic fitting function. Therefore, in the third step the WMR velocities on the desired path are calculated.

4.3.1 Case A: Single Wheeled NI- Mobile Robot

In the case of empty map, the total path length from start to target point is equal to (1131.37) cm. In this case, the number of waypoints that needs to be optimized (D) is equal to 4.

4.3.1.1 Single Optimal Path Finding

Figure (4.1) reveals the simulation results of the optimum route for the wheeled NI- mobile robot based on PSO, CPSO, FF and hybrid FFCPSO algorithms. Based

on the number of samples between (X_s, Y_s) and (X_t, Y_t) , the WMR has travel time (T_v) equal to 80 sec on the desired path.

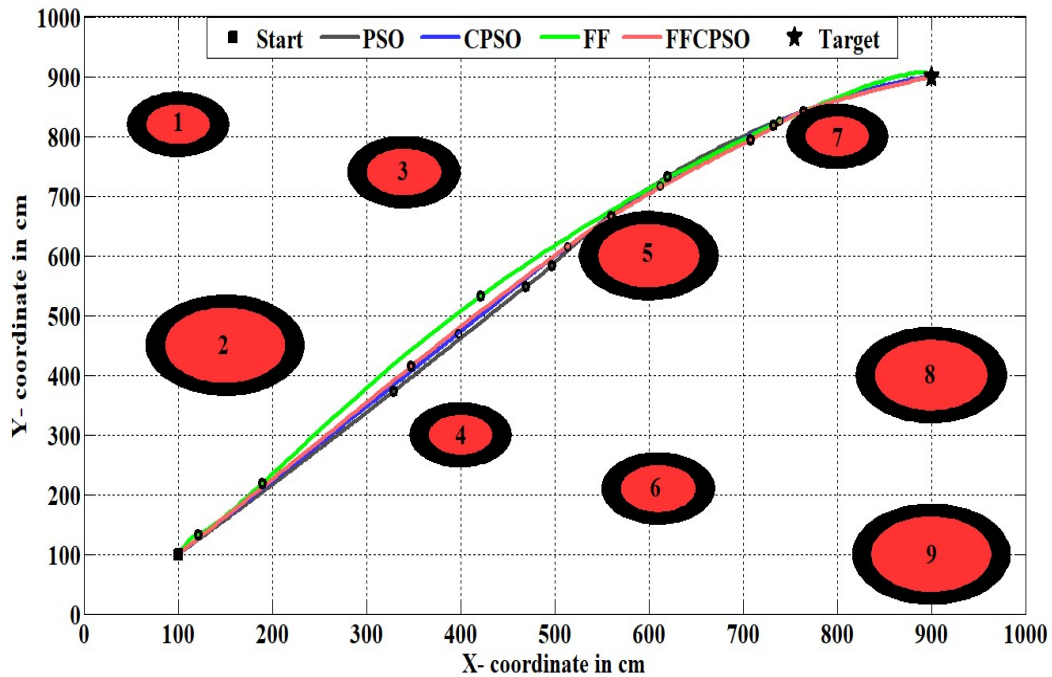


Figure (4.1): The shortest path for the case A based on optimization algorithms.

After ten runs, the outputted waypoints based on various algorithms are ordered in Table (4.8).

Table (4.8): Waypoints coordination for case A.

Waypoints (D)	Coordinate (X, Y)	Type of Intelligent Algorithm			
		PSO	CPSO	FF	FFCPSO
D ₁	X (cm)	328.729	397.724	121.796	347.563
	Y (cm)	372.737	469.780	132.757	414.511
D ₂	X (cm)	469.155	513.662	189.289	560.296
	Y (cm)	547.543	614.743	218.320	665.378
D ₃	X (cm)	496.680	612.248	421.841	708.256
	Y (cm)	582.702	717.285	532.636	794.053
D ₄	X (cm)	619.992	738.988	732.089	619.992
	Y (cm)	732.612	824.693	818.865	732.612

Figure (4.2) explains the variation of the objective function through the number of iterations based on presented methods until reaching the best values.

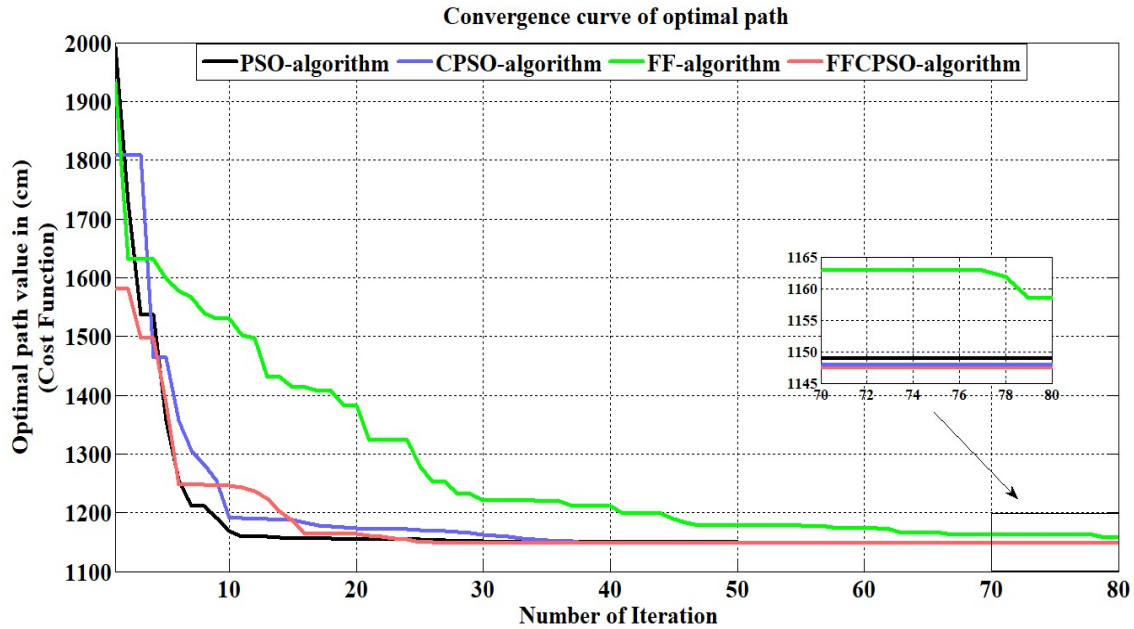


Figure (4.2): Variation of objective function through number of iterations / case A.

From the figure above, the optimal path with shortest distance using standard PSO is equal to (1148.81) with in iteration (53) and is equal to (1148.52) with in iteration (36). The optimal path with shortest distance using standard FF is equal to (1158.4) with in iteration (79) and is equal to (1148.01) with in iteration (39).

The simulation results for case A are summarized in Table (4.9). Standard deviation is a measure of how “spread out” a set of data is. If it is large, data has a large range of numbers. If it is small, most of data points are close to the average. While, the percentages of the objective function between techniques are explained in Table (4.10).

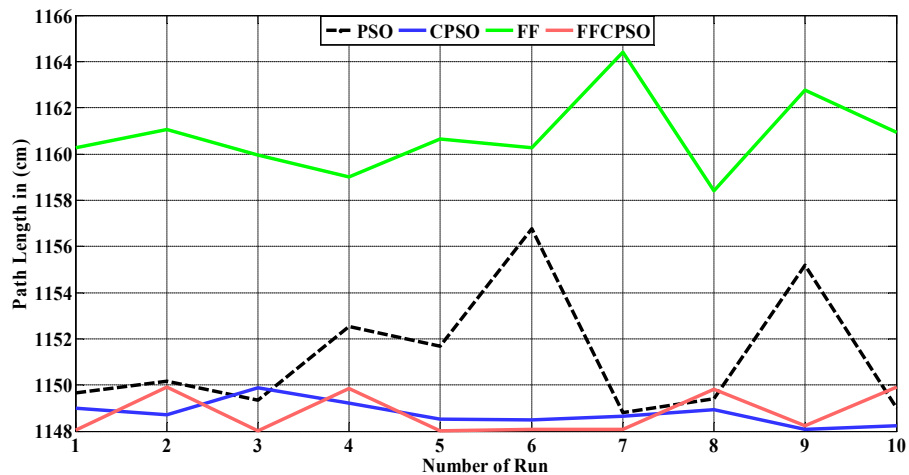
Table (4.9): Comparison results for case A.

Type of intelligent Algorithm	Min. Distance in (cm)	Fitness	Iteration of best value	Max. Distance in (cm)	Average in (cm)	Standard Deviation
PSO	1148.81	0.087	53	1156.77	1151.2	0.0254
CPSO	1148.52	0.087	36	1149.78	1148.77	5.638E-3
FF	1158.4	0.0863	79	1164.41	1160.73	0.0165
FFCPSO	1148.01	0.0871	39	1149.91	1148.75	8.067E-3

Table (4.10): Percentages of objective function between algorithms for case A.

Algorithm Type	PSO-CPSO	PSO-FF	PSO-FFCPSO	CPSO-FF	CPSO-FFCPSO	FF-FFCPSO
Percentage, %	0.0252	0.8278	0.0696	0.8529	0.0444	0.8969

One can conclude that the objective function values varied sharply within several runs since the particles or fireflies' best positions (candidate solutions) varied intensively, as shown in Figure (4.3).

**Figure (4.3): Variation of objective function through number of runs for case A.**

From the above figure, PSO algorithm can achieve the best path with the shortest distance in run no. 7. While, the best path with the shortest distance by using CPSO algorithm is achieved in run no. 9. FF algorithm can achieve it in run no. 8 and FFCPSO algorithm in run no. 5.

The presented algorithms generate an optimal obstacle free trajectory for single robot in static environment that can contain known multiple obstacles. From the case A results, it is plainly to say that both PSO and CPSO algorithms always lead to an optimum or near to optimum path, but there is a difference in the number of iteration between the two algorithms, where the basic PSO algorithm requires more number of iterations than CPSO algorithm. Hybrid FFCPSO algorithm can enhance the performance of FF algorithm by providing more smoothness and optimal path value

with less number of iterations and minimum execution time. Hybrid FFCPSO technique is perfect in providing excellent path tracking equals to (1148.01) cm at 39 iteration. On the other hand, CPSO is better than FFCPSO in iteration number equal to 36 and time consuming. Additionally, the small percentage between both methods equals (0.0444 %).

4.3.1.2 Single Mobile Robot Velocities

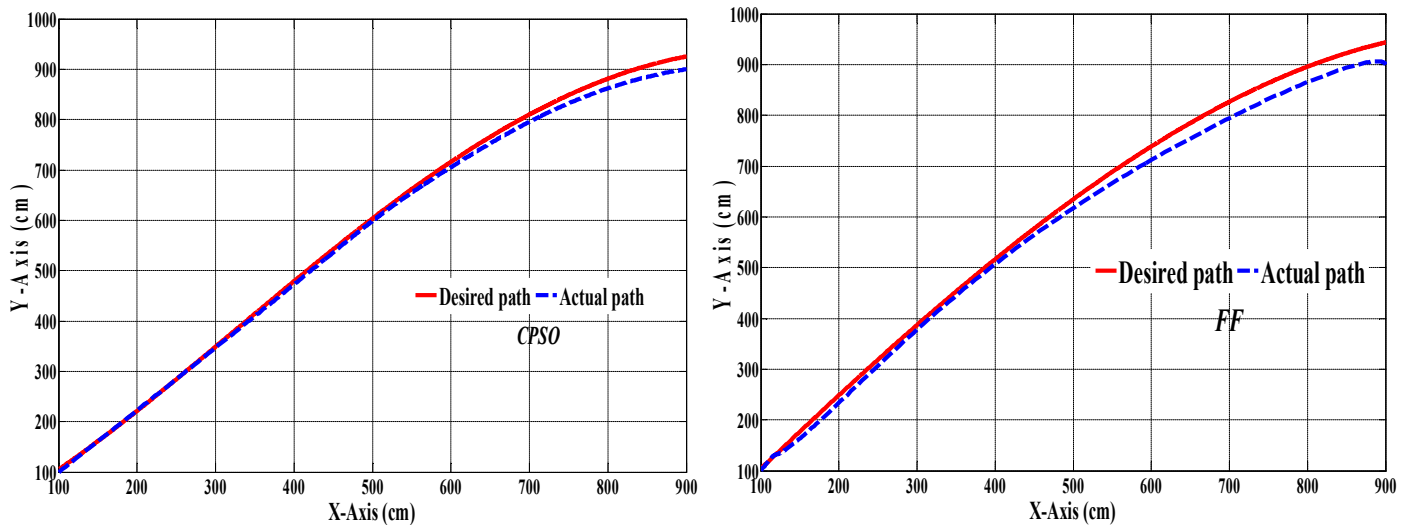
After finding the actual robot's path for each optimization algorithm, the desired path fitting function for the CPSO, FF and FFCPSO algorithms are given in equations (4.1), (4.2) and (4.3), respectively.

$$y(x) = 4.2 \times 10^{-10}x^4 - 1.9 \times 10^{-6}x^3 + 0.0016x^2 + 0.81x + 9.9 \quad (4.1)$$

$$y(x) = -4.1 \times 10^{-7}x^3 - 7.7 \times 10^{-5}x^2 + 1.5x - 45 \quad (4.2)$$

$$y(x) = -7.6 \times 10^{-10}x^3 + 0.00053x^2 + 1.2x - 23 \quad (4.3)$$

Figure (4.4) shows the both actual and desired path based on CPSO, FF and FFCPSO algorithms, respectively.



Continued

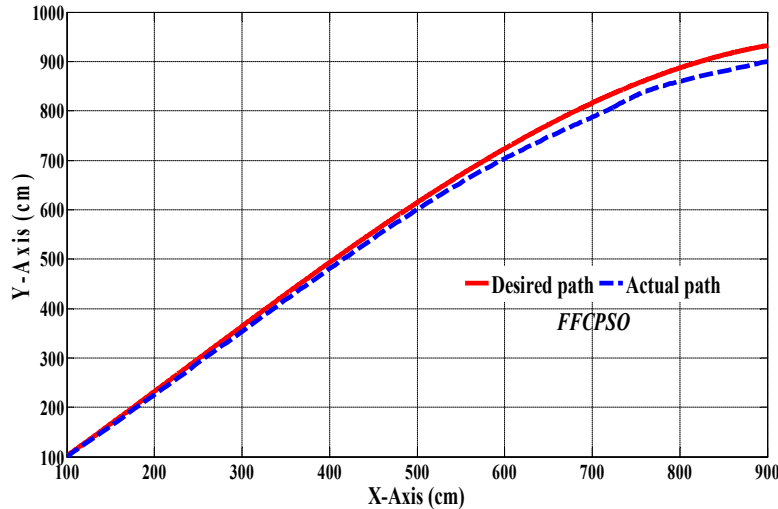


Figure (4.4): Desired and actual path for case A.

From the above figure, the distance error between actual and desired paths is equal to (0.95%) for CPSO, (1.67%) for FF and (1.44%) for FFCPSO algorithms. Then, based on kinematic equations of the differential drive mobile robot, one can calculate the robot velocity on the desired path as follows:

1. Wheels Angular Velocity

The right and left wheels angular velocity of the robot system is described in Figures (4.5), (4.6) and (4.7) based on the CPSO, FF and FFCPSO algorithms, respectively.

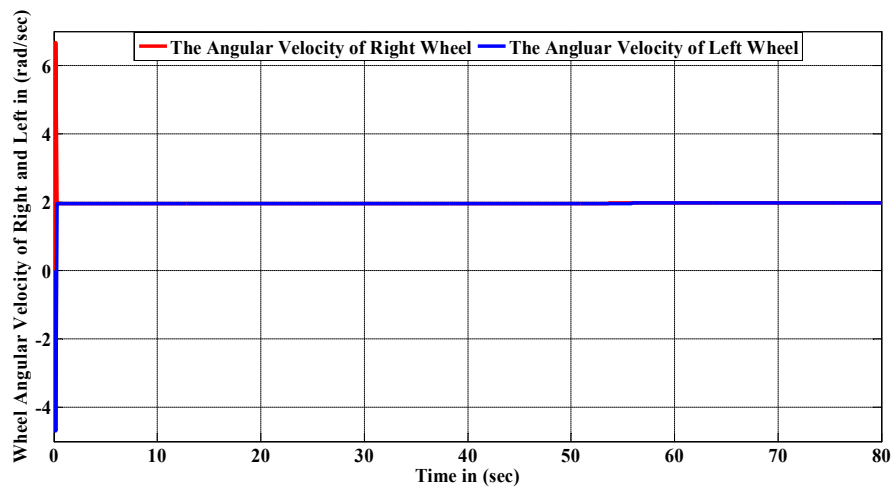


Figure (4.5): The angular velocity of the right and left wheels / case A based on CPSO algorithm.

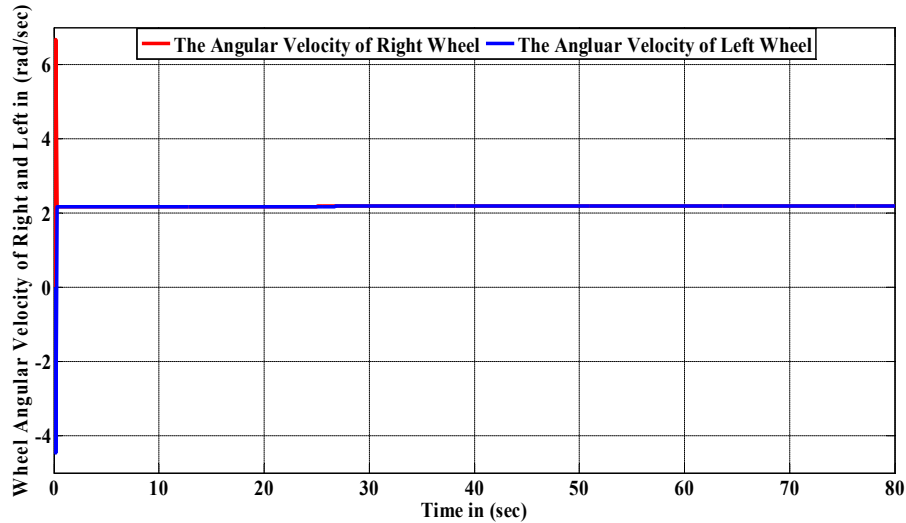


Figure (4.6): The angular velocity of right and left wheels / case A based on FF algorithm.

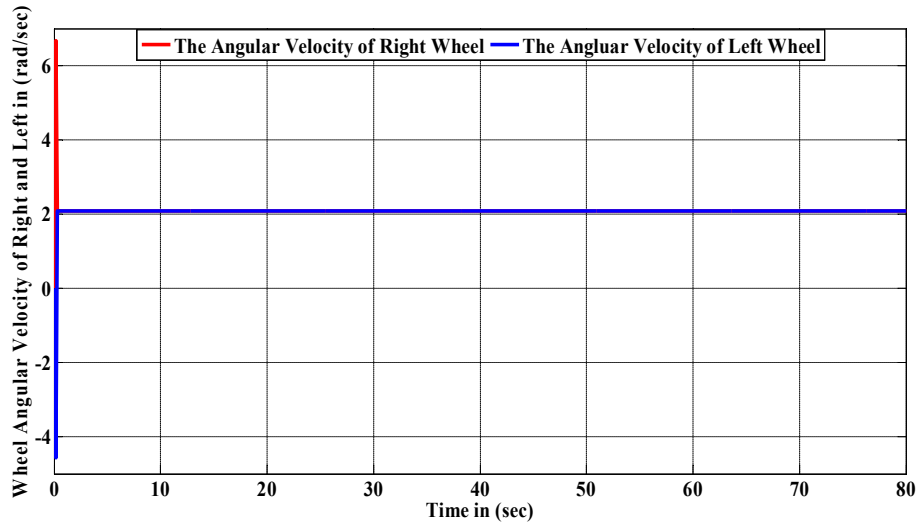


Figure (4.7): The angular velocity of right and left wheels / case A based on FFCPSO algorithm.

From Figure (4.5), (4.6) and (4.7), the angular velocity of the left and right wheels (WL & WR) based on the CPSO, FF and FFCPSO approaches are equal to (1.9) rad/sec, (2.18) rad/sec and (2.09) rad/sec, respectively and should range between $(-6.67, +6.67)$ rad/sec.

2. Wheels Linear Velocities

The linear velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms is illustrated in Figures (4.8), (4.9) and (4.10), respectively.

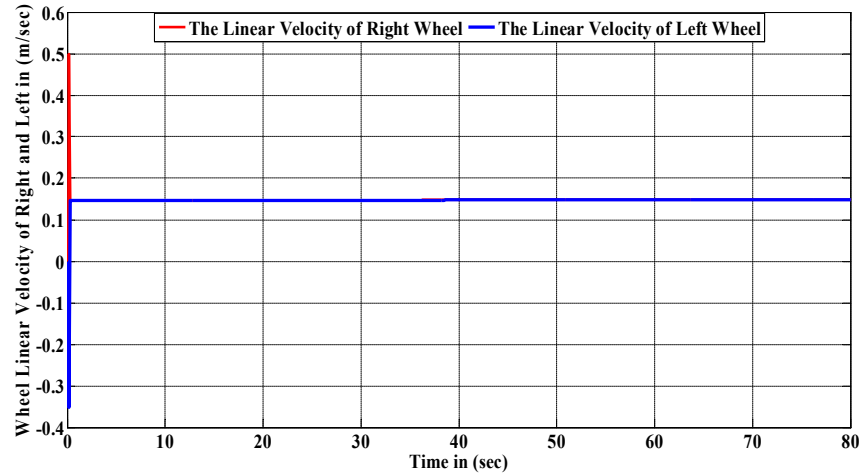


Figure (4.8): The linear velocity of right and left wheels / case A based on CPSO algorithm.

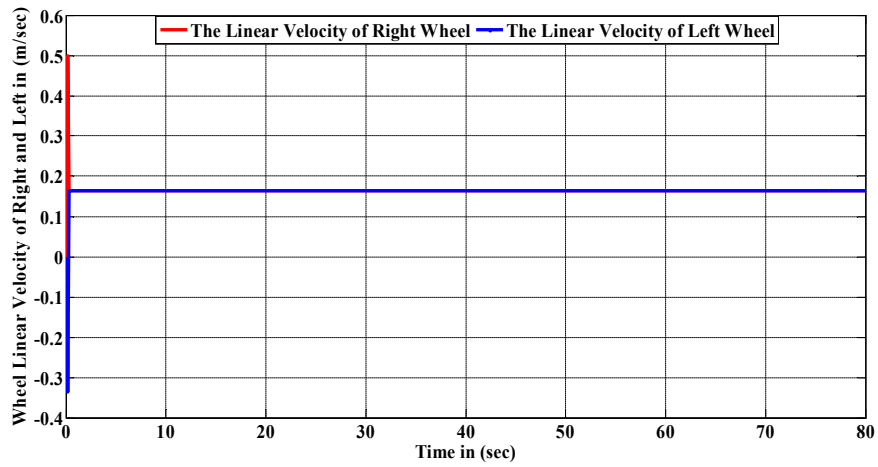


Figure (4.9): The linear velocity of right and left wheels / case A based on FF algorithm.

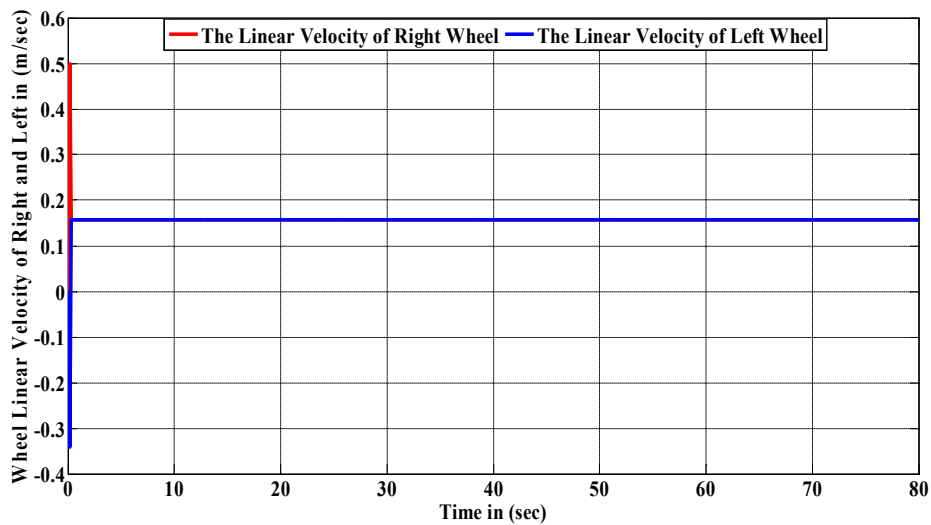


Figure (4.10): The linear velocity of the right and left wheels / case A based on FFCPSO algorithm.

The linear velocity of the left and right wheels (V_L & V_R) based on the CPSO, FF and FFCPSO approaches are equal to (0.14) m/sec, (0.16) m/sec and (0.15) m/sec, respectively and should range between (-0.5, +0.5) m/sec.

3. Platform Linear and Angular Velocities

The angular and linear velocities of the platform (V_a & V_n) based on the CPSO, FF and FFCPSO algorithms are manifested in Figures (4.11), (4.12) and (4.13), respectively.

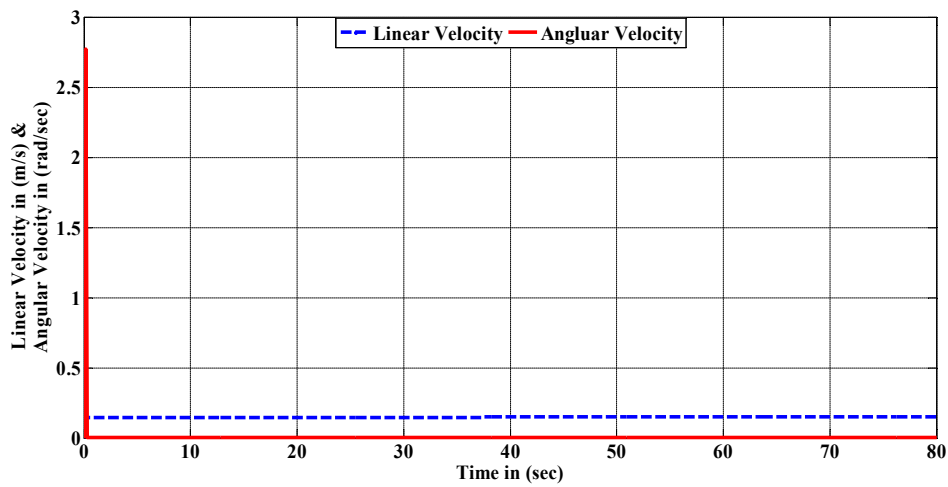


Figure (4.11): The platform angular and linear velocities/ case A based on CPSO algorithm.

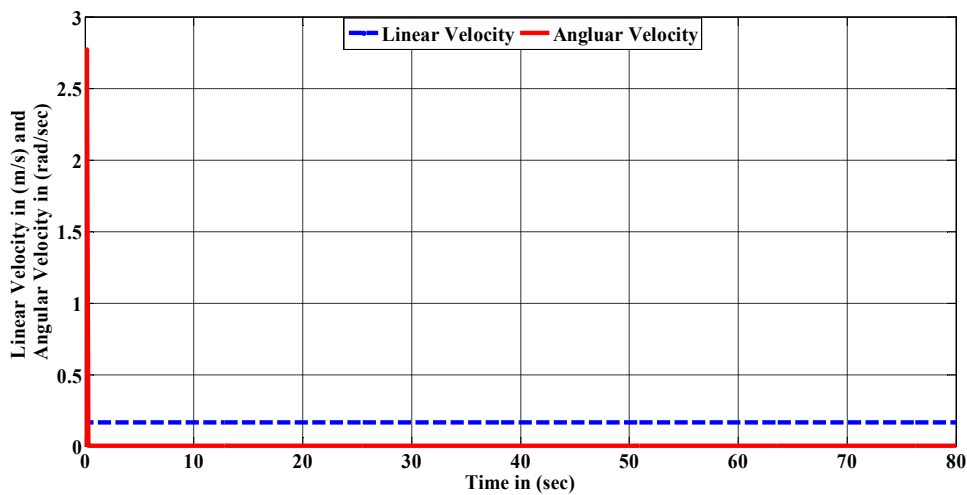


Figure (4.12): The platform angular and linear velocities/ case A based on FF algorithm.

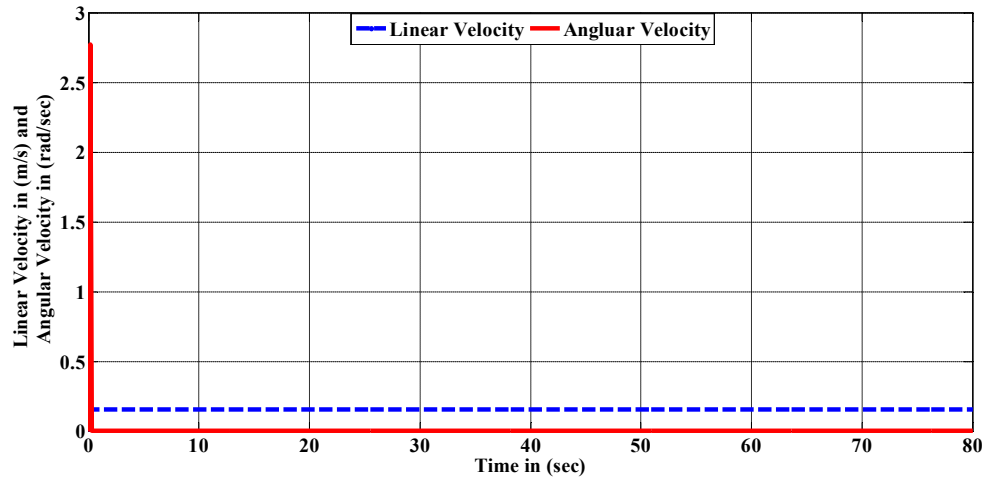


Figure (4.13): The platform angular and linear velocities/ case A based on FFCPSO algorithm.

4.3.2 Case B: Three Independent Wheeled NI- Mobile Robots

In the case of empty map, the total path length for the first, second and third wheeled NI- mobile robot from start to target point is equal to (894.427) cm, (1063.014) cm and (921.954) cm, respectively. In this case, the number of waypoints that needs to be optimized (D) is equal to 3 for each path. In tacit way, the optimization algorithms work as a dynamic path planning because every mobile robot must avoid collision with another mobile robot during its movement to the same target. So, each mobile robot is considered as a dynamic obstacle to other mobile robots.

4.3.2.1 Multi Optimal Path Finding

Figure (4.14) displays the resulted paths for the three wheeled NI- mobile robots based on PSO, CPSO, FF and hybrid FFCPSO algorithms.

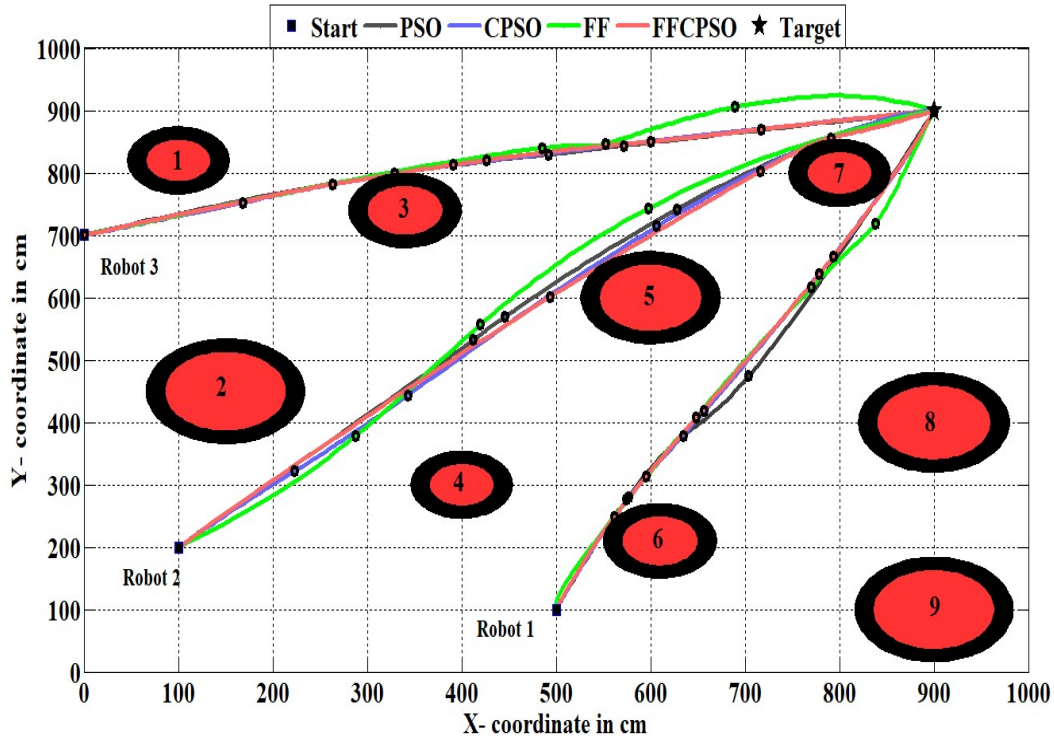


Figure (4.14): The shortest path for each robot / case B.

After ten runs, the outputted waypoints based on various algorithm are abstracted in Table (4.11).

Since the candidate solutions varied intensively, the PSO algorithm can achieve best path with shortest distance in run no.7. While the best path with the shortest distance by using CPSO algorithm is achieved in run no. 5. FF algorithm can achieve it in run no. 7 and FFCPSO algorithm in run no. 6. Depends on the number of samples between (X_s, Y_s) and (X_t, Y_t) , the WMR_1 has travel time (T_v) equal to 40 sec, the WMR_2 has travel time (T_v) equal to 80 sec and the WMR_3 has travel time (T_v) equal to 90 sec. Thereafter, based on the kinematic equations of differential drive mobile robot, one can calculate the robot velocity on its specific path. In this case, all mobile robots start to move to the target at ($T_v=0$) sec.

Table (4.11): Waypoints coordination for case B.

Path No.	Waypoints (D)	Coordinate (X, Y)	Types of Intelligent Algorithm			
			PSO	CPSO	FF	FFCPSO
Path 1	D ₁	X (cm)	574.649	577.006	595.198	562.162
		Y (cm)	276.490	279.072	312.357	248.874
	D ₂	X (cm)	634.478	656.897	770.467	648.581
		Y (cm)	378.611	417.631	616.499	407.635
	D ₃	X (cm)	703.852	793.283	838.292	778.190
		Y (cm)	474.164	665.875	718.212	636.921
Path 2	D ₁	X (cm)	412.551	223.359	287.493	493.487
		Y (cm)	532.630	321.973	377.432	600.880
	D ₂	X (cm)	445.648	342.890	419.634	716.700
		Y (cm)	569.036	443.390	556.245	802.813
	D ₃	X (cm)	627.777	606.640	598.260	790.809
		Y (cm)	741.597	714.316	743.441	855.571
Path 3	D ₁	X (cm)	329.119	168.254	485.423	391.060
		Y (cm)	798.859	752.526	839.609	813.245
	D ₂	X (cm)	491.968	263.544	552.923	600.653
		Y (cm)	828.929	781.950	846.302	850.045
	D ₃	X (cm)	571.733	426.158	689.632	717.287
		Y (cm)	843.544	819.375	905.631	869.566

Figures (4.15), (4.16) and (4.17) explain the changing of the objective function through the number of iterations until reaching the optimal value for the first, second and third paths, respectively.

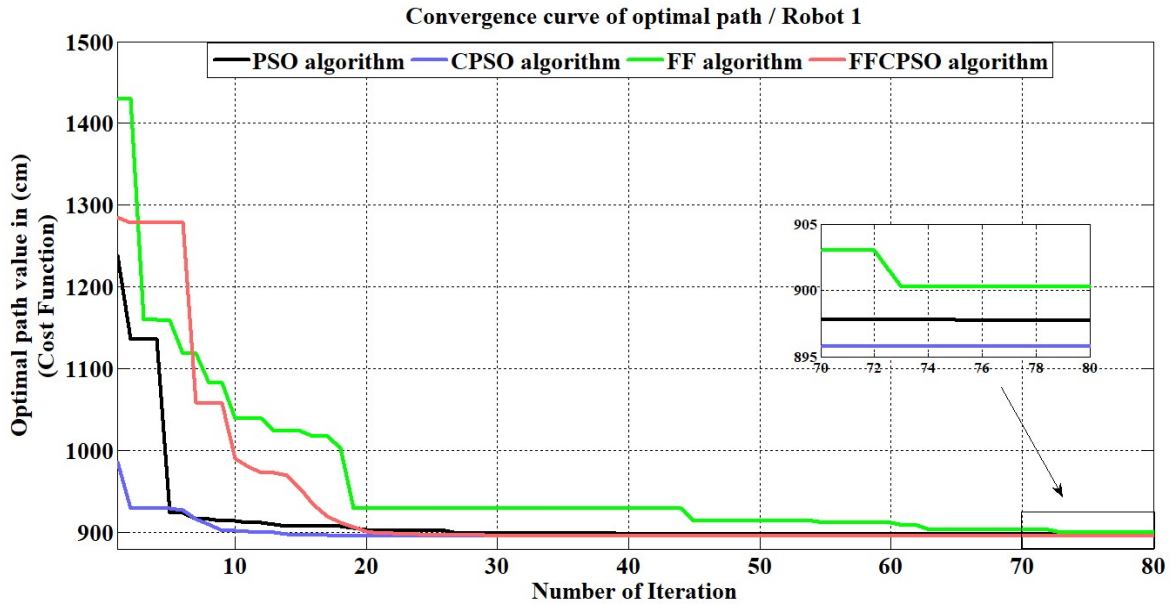


Figure (4.15): Variation of objective function through the number of iterations for first path / case B.

From the figure above, the optimal path with shortest distance using standard PSO is equal to (897.75) with in iteration (75) and is equal to (895.82) with in iteration (29) in CPSO algorithm. The optimal path with shortest distance using standard FF is equal to (900.25) with in iteration (73) and is equal to (895.78) with in iteration (33) in the hybrid FFCPSO algorithm.

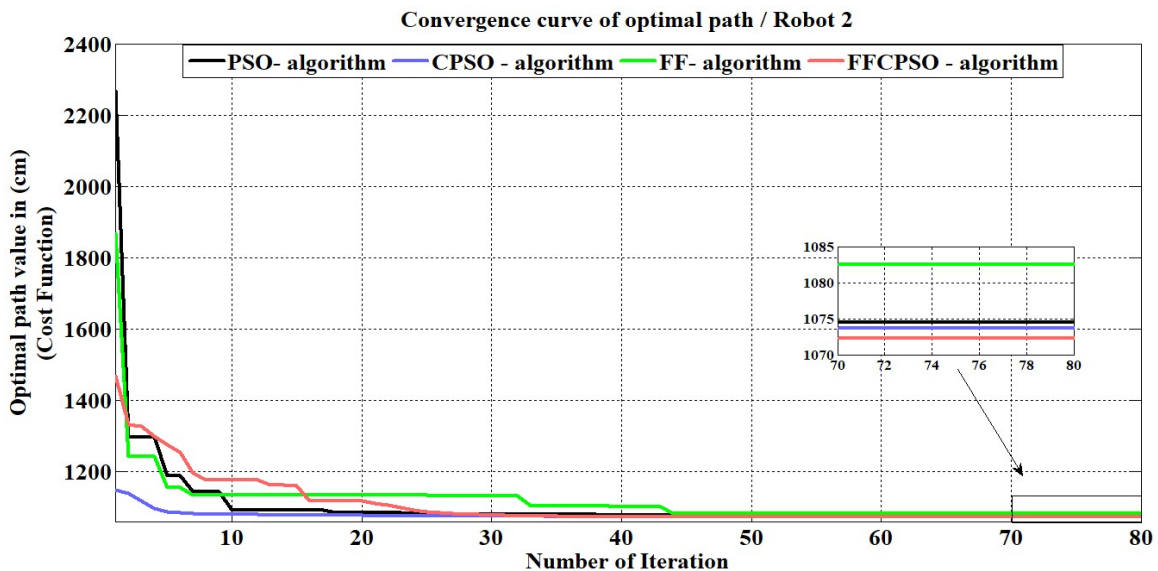


Figure (4.16): Variation of objective function through the number of iterations for second path / case B.

From the figure (4.16), the optimal path with shortest distance using standard PSO is equal to (1074.43) with in iteration (66) and is equal to (1073.74) with in iteration (47) in the CPSO algorithm. The optimal path with shortest distance using standard FF is equal to (1082.54) with in iteration (44) and is equal to (1072.34) with in iteration (59) in the proposed FFCPSO algorithm.

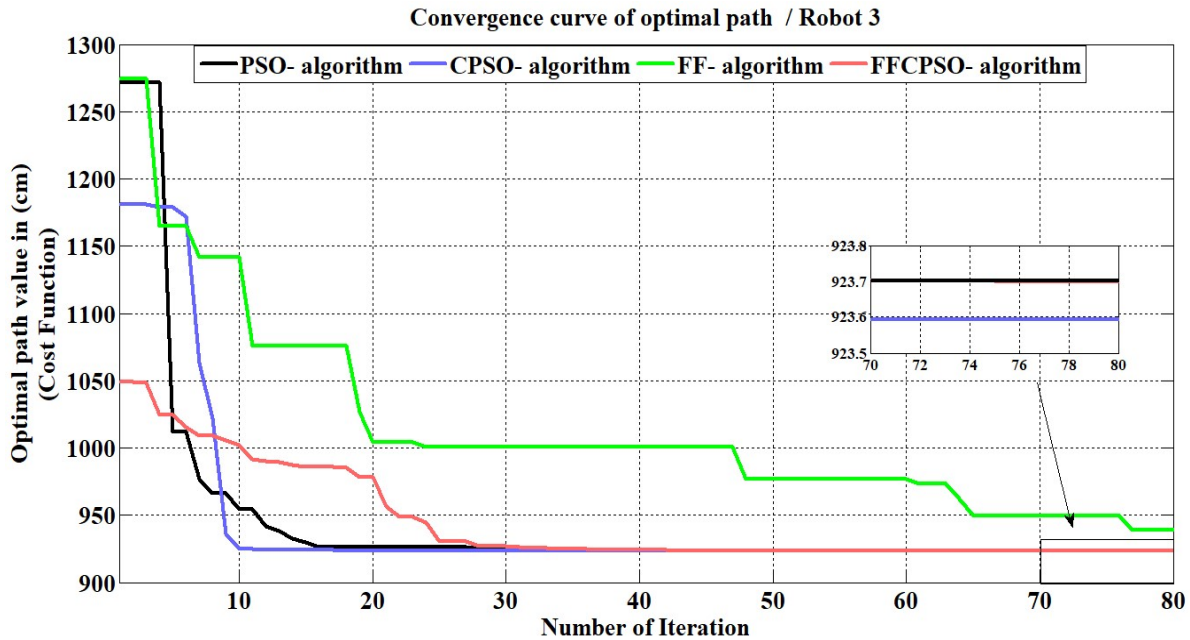


Figure (4.17): Variation of objective function through the number of iterations for third path/ case B.

While from the figure (4.17), the optimal path with shortest distance using standard PSO is equal to (923.7) with in iteration (58) and is equal to (923.59) with in iteration (38) in the CPSO algorithm. The optimal path with shortest distance using standard FF is equal to (938.66) with in iteration (77) and is equal to (923.77) with in iteration (44) in the proposed hybrid FFCPSO algorithm.

Table (4.12) summarizes the results of case B, which can be achieved by the various intelligent algorithms after ten runs. Posteriorly, the percentages of the objective function between algorithms for each path are explained in Table (4.13).

Table (4.12): Comparison results for case B.

Robot No.	Type of Intelligent Algorithm	Min. Distance in (cm)	Fitness	Iteration of best value	Max. Distance in (cm)	Average Distance in (cm)	Standard Deviation
1	PSO	897.75	0.1113	73	899.7	897.85	7.473E-3
	CPSO	895.82	0.1116	29	897.3	896.42	5.546E-3
	FF	900.25	0.1110	73	907.44	903.4	0.0253
	FFCPSO	895.78	0.1116	33	898.19	896.24	4.509E-3
2	PSO	1074.43	0.0930	66	1079.9	1075.1	7.313E-3
	CPSO	1073.74	0.0931	47	1076.1	1074.43	9.154E-3
	FF	1082.54	0.0923	44	1107.39	1096.64	0.1429
	FFCPSO	1072.34	0.0932	59	1076.6	1073.99	0.0111
3	PSO	923.7	0.1082	58	924.5	923.82	3.096E-3
	CPSO	923.59	0.1082	38	923.78	923.63	1.476E-3
	FF	938.66	0.1065	77	949.4	944.4	3.1644
	FFCPSO	923.77	0.1082	44	924.5	923.73	2.915E-3

Table (4.13): Percentages of the objective function between algorithms for case B.

Robot No.	Algorithm Type	PSO-CPSO	PSO-FF	PSO-FFCPSO	CPSO-FF	CPSO-FFCPSO	FF-FFCPSO
1	Percentage,%	0.2149	0.2777	0.2194	0.492	4.465E-3	0.4965
2		0.0642	0.7491	0.1945	0.8129	0.1303	0.9422
3		0.0119	1.5937	7.577E-3	1.6054	0.0194	1.5863

The discussed algorithms generate short, safe and smooth paths for three independent two-wheeled mobile robots. Firstly, the first mobile robot arrived to the target. Then, the second and third mobile robots depend on their velocities and travel time. From the case B results, it is clearly to say as follows:

1. For path1, it is clearly from Figures (4.14) and (4.15) that CPSO and FFCPSO can get the same path length difference in few cm so there is a small percentage between them equal to (4.465E-3%), but CPSO can get it just with (29) iterations and has a good convergence curve than FFCPSO.

2. For path 2, Figures (4.14) and (4.16) show that FFCPSO is better than other algorithms in the minimum path length equal to (1072.34) cm with a high fitness value equal to (0.0932) however, FF became stable at (44) epoch.
3. For path 3, Figures (4.14) and (4.17) depict that FFPSO is better than PSO to get the same third path length with (44) epoch, but CPSO is better in the iteration number and the shortest distance.

4.3.2.2 Multi Mobile Robot Velocities

After finding the actual paths of mobile robots based on each presented approach, the desired path equations for WMR₁ based on CPSO, FF and FFCPSO by using basic fitting function are given in equations (4.4), (4.5) and (4.5), respectively.

$$y(x) = -1.6 \times 10^{-8}x^4 + 5.2 \times 10^{-5}x^3 - 0.016x^2 + 33x - 6.5 \times 10^{+3} \quad (4.4)$$

$$y(x) = 9.7 \times 10^{-6}x^3 - 0.02x^2 + 16x - 3.9 \times 10^{+3} \quad (4.5)$$

$$y(x) = -9.8 \times 10^{-9}x^4 + 3.4 \times 10^{-5}x^3 - 0.043x^2 + 25x - 5.2 \times 10^{+3} \quad (4.6)$$

The desired path equations for WMR₂ based on CPSO, FF and FFCPSO are given in equations (4.7), (4.8) and (4.9), respectively.

$$y(x) = -4.8 \times 10^{-10}x^4 - 1.6 \times 10^{-8}x^3 + 0.00047x^2 + 0.80x + 1.2 \times 10^{+2} \quad (4.7)$$

$$y(x) = -1.6 \times 10^{-6}x^3 + 0.0017x^2 + 0.62x + 1.1 \times 10^{+2} \quad (4.8)$$

$$y(x) = -1.9 \times 10^{-9}x^4 + 3.1 \times 10^{-6}x^3 - 0.0019x^2 + 1.5x + 62 \quad (4.9)$$

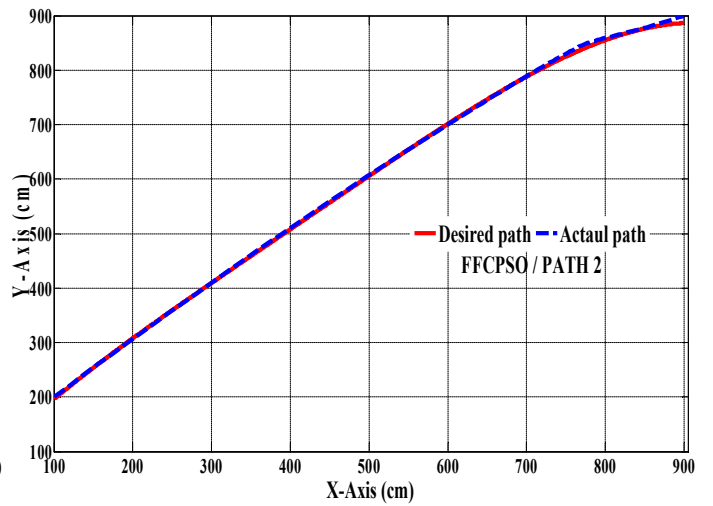
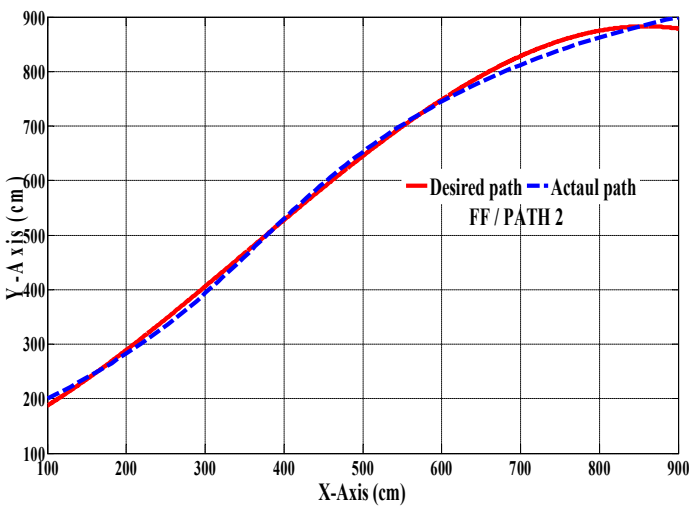
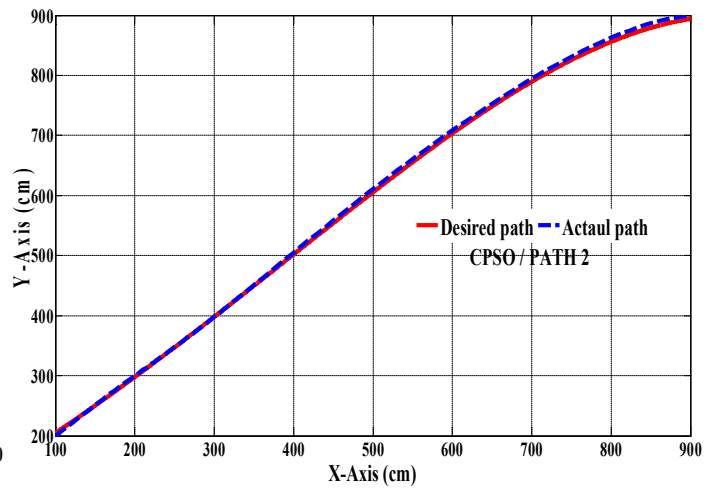
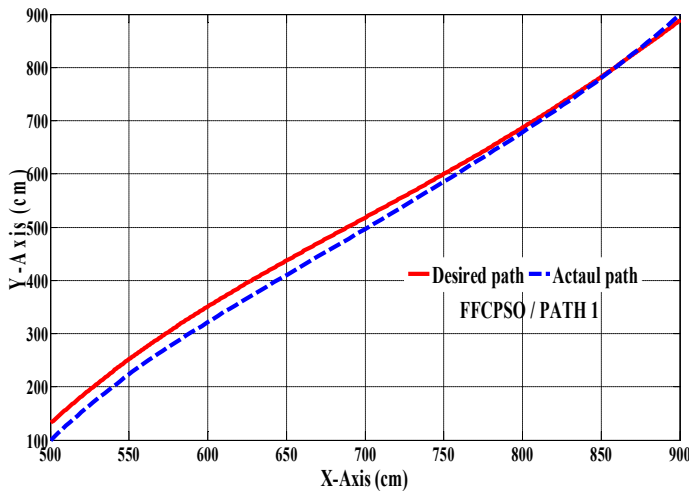
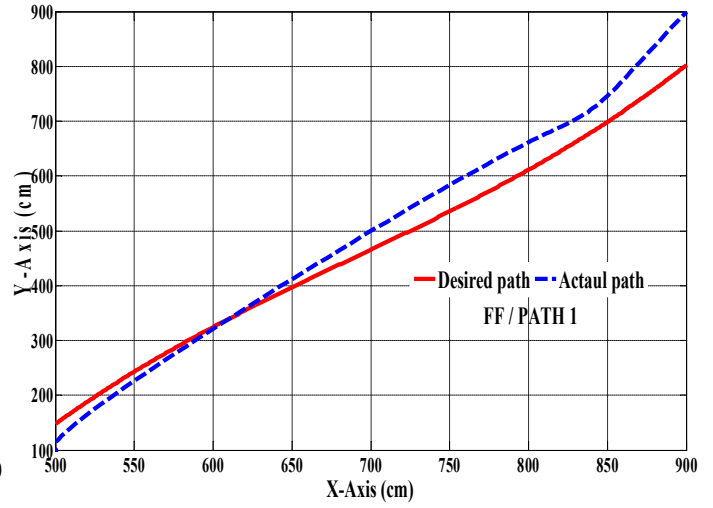
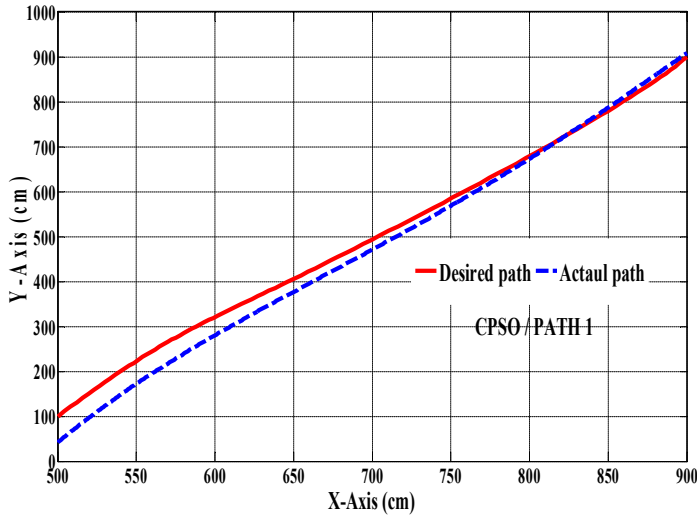
While, the desired path equations for WMR₃ based on CPSO, FF and FFCPSO are obtained in equations (4.10), (4.11) and (4.12), respectively.

$$y(x) = -5.6 \times 10^{-11}x^4 + 2.9 \times 10^{-7}x^3 - 0.00043x^2 + 0.41x + 7 \times 10^{+2} \quad (4.10)$$

$$y(x) = -2.6 \times 10^{-9}x^4 + 4.5 \times 10^{-6}x^3 - 0.0025x^2 + 0.77x + 6.9 \times 10^{+2} \quad (4.11)$$

$$y(x) = 2 \times 10^{-10}x^4 - 2.1 \times 10^{-7}x^3 - 0.00012x^2 + 0.36x + 7 \times 10^{+2} \quad (4.12)$$

Figure (4.18) shows the both actual and desired paths based on CPSO, FF and FFCPSO algorithms, respectively.



Continued

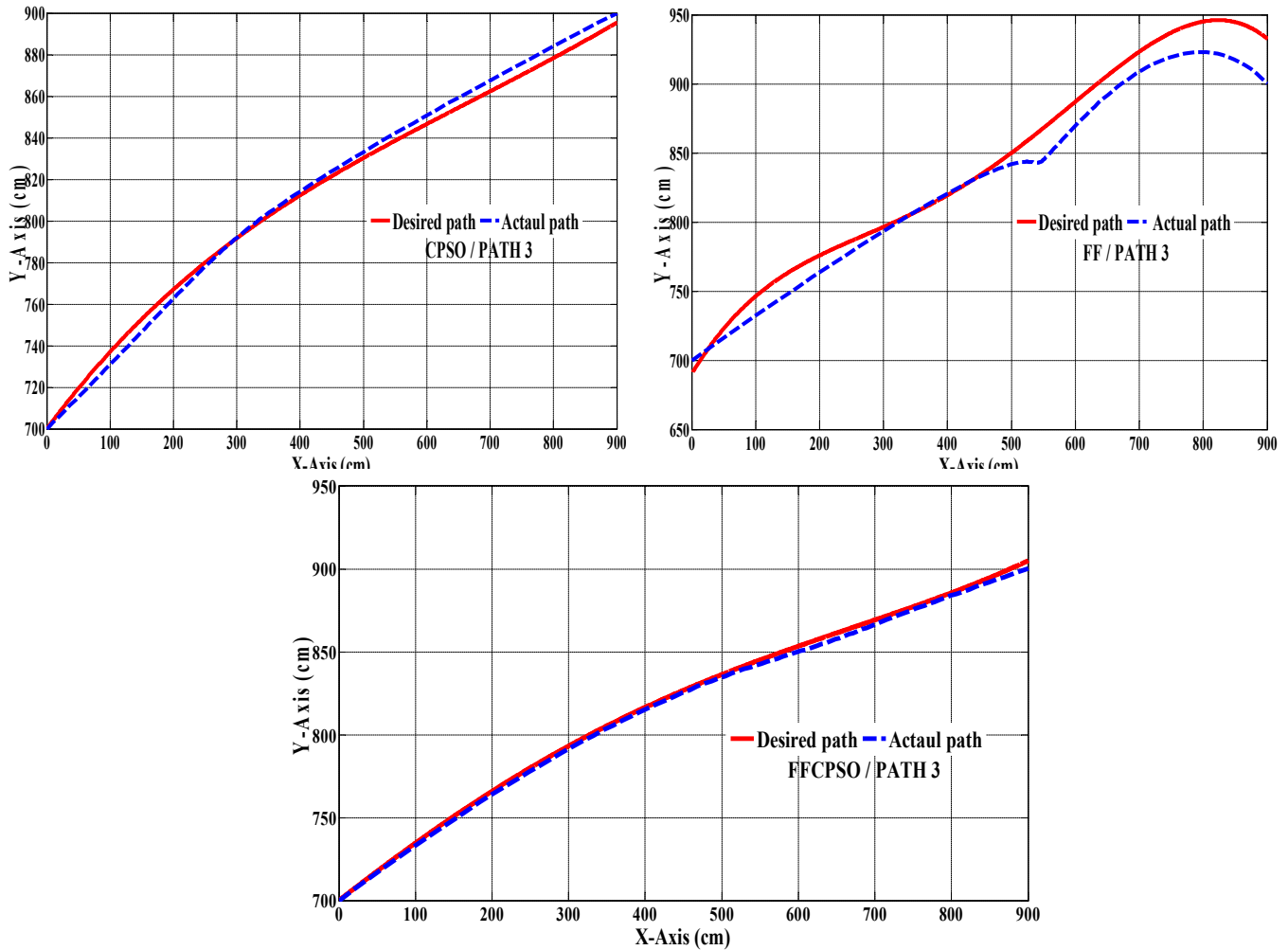


Figure (4.18): Desired and actual path for case B.

The distance error between actual and desired path for each NI-mobile robot based on three approaches are explained in Table (4.14).

Table (4.14): Distance error for case B.

Path No.		Type of Intelligent Algorithm		
		CPSO	FF	FFCPSO
1	Distance Error 100%	6.25	14.55	4.27
2		0.72	0.86	0.24
3		0.17	0.62	5.41E-3

Then, based on kinematic equations of the differential drive mobile robot, one can calculate the robot velocity on the desired path as follows:

1. Wheels Angular Velocity

The angular velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR₁ is revealed in Figures (4.19), (4.20) and (4.21), respectively.

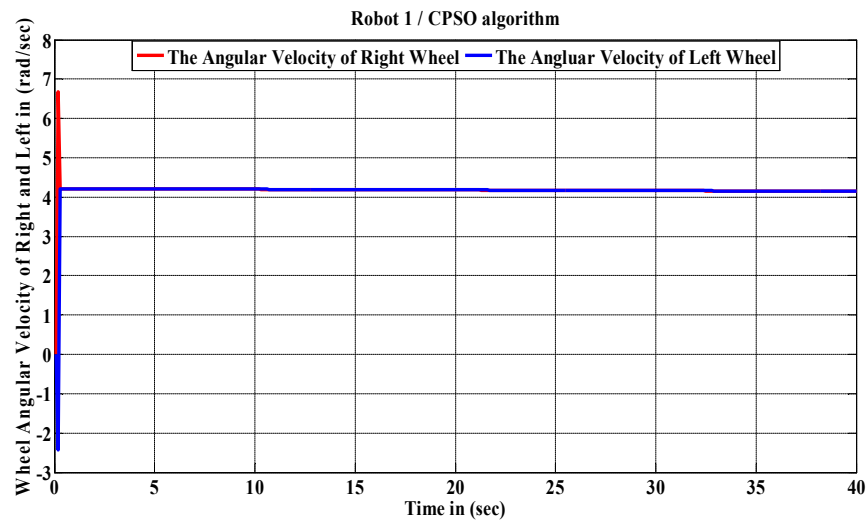


Figure (4.19): The angular velocity of right and left wheels for 1st robot / case B based on CPSO algorithm.

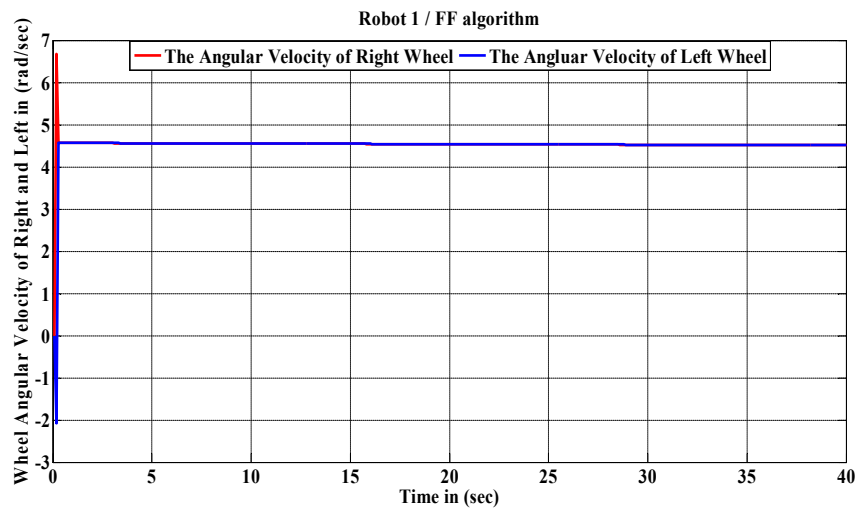


Figure (4.20): The angular velocity of right and left wheels for 1st robot / case B based on FF algorithm.

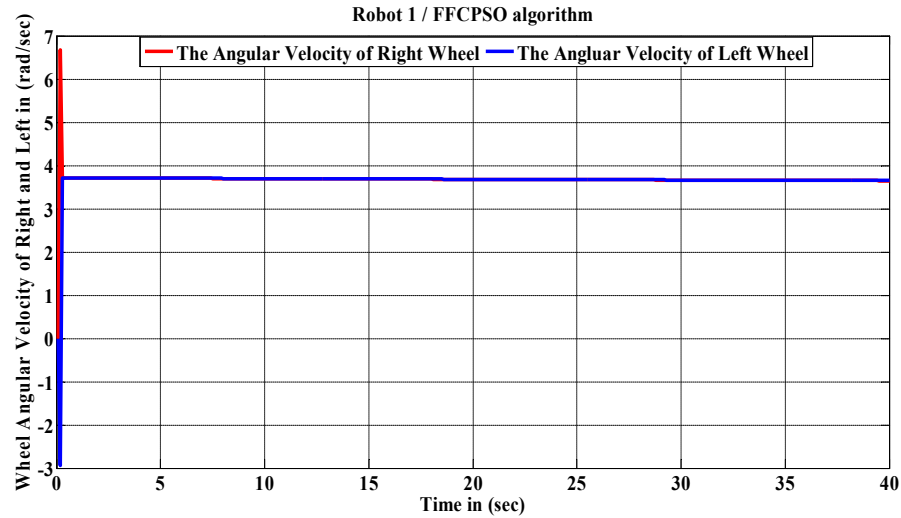


Figure (4.21): The angular velocity of right and left wheels for 1st robot / case B based on FFCPSO algorithm.

From the figure (4.19), (4.20) and (4.21), the angular velocity of the right and left wheels for WMR₁ based on the CPSO, FF and FFCPSO approaches are equal to (4.2) rad/sec, (4.8) rad/sec and (3.7) rad/sec, respectively.

The angular velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the specific path of WMR₂ is illustrated in Figures (4.22), (4.23) and (4.24), respectively.

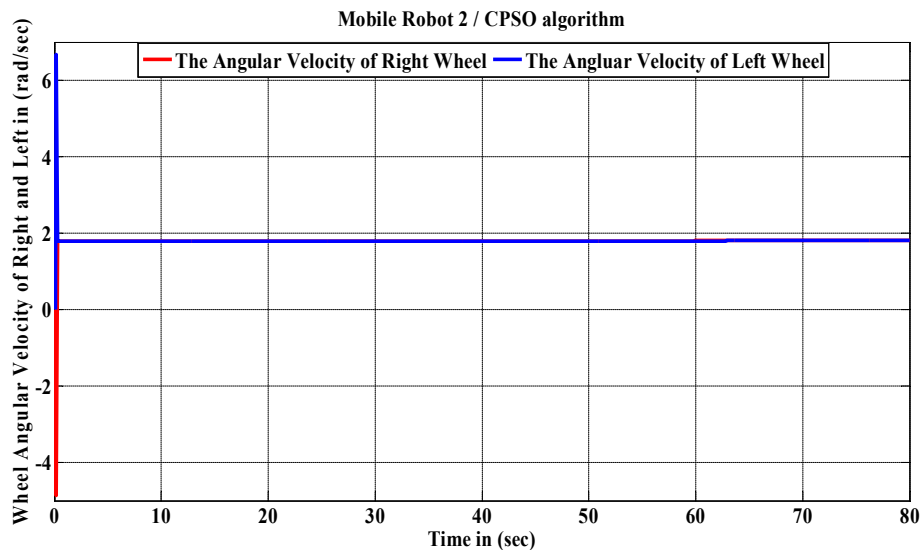


Figure (4.22): The wheel angular velocity of right and left for 2nd robot / case B based on CPSO algorithm.

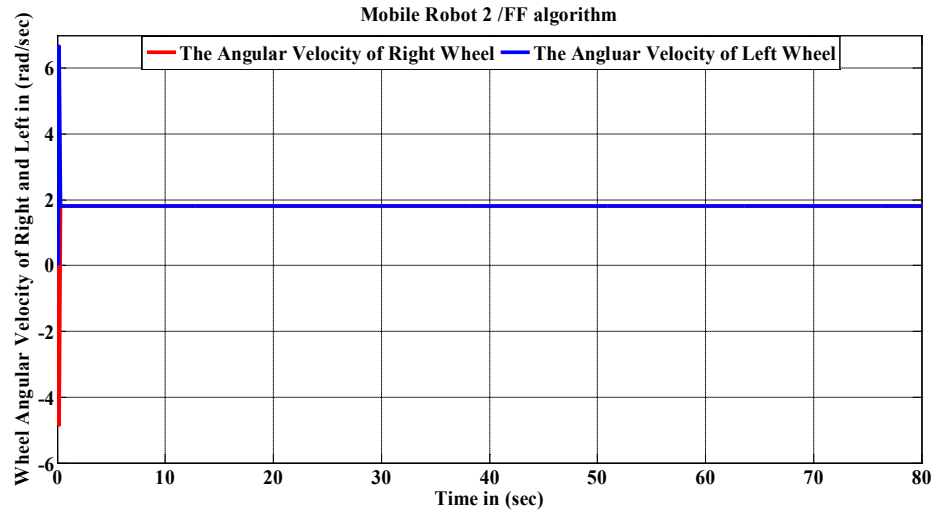


Figure (4.23): The angular velocity of right and left wheels for 2nd robot / case B based on FF algorithm.

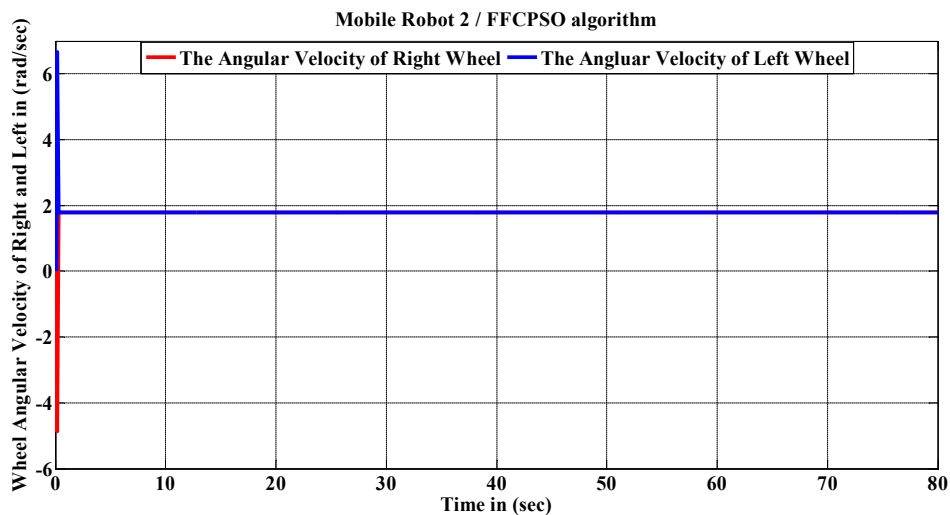


Figure (4.24): The angular velocity of right and left wheels for 2nd robot / case B based on FFCPSO algorithm.

From figure (2.22), (2.23) and (2.24), the angular velocity of the right and left wheels for WMR₂ based on the CPSO, FF and FFCPSO approaches are equal to (1.8) rad/sec, (1.81) rad/sec and (1.79) rad/sec, respectively.

The angular velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR₃ is manifested in Figures (4.25), (4.26) and (4.27), respectively.

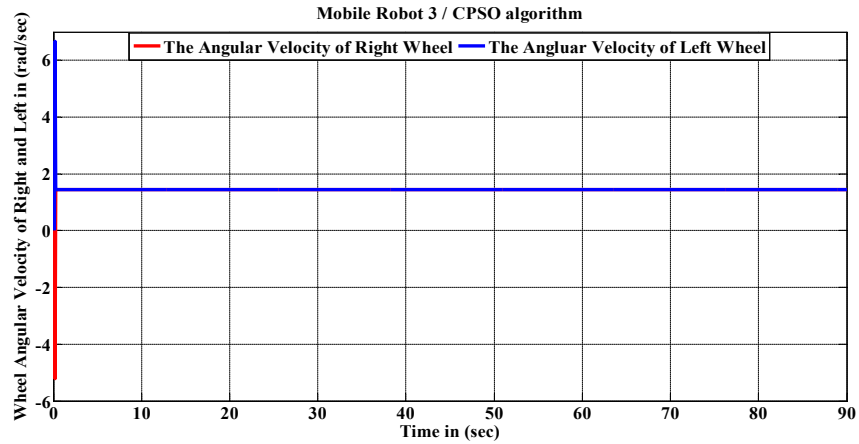


Figure (4.25): The angular velocity of right and left wheels for 3rd robot / case B based on CPSO algorithm.

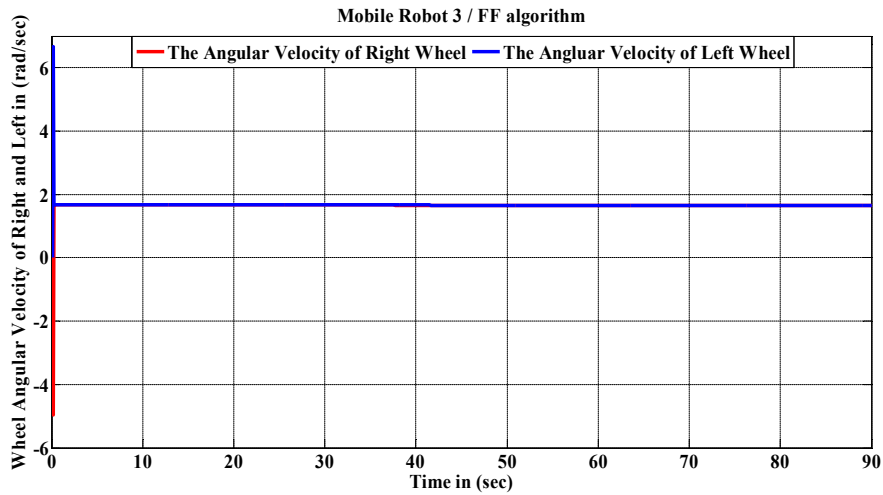


Figure (4.26): The angular velocity of right and left wheels for 3rd robot / case B based on FF algorithm.

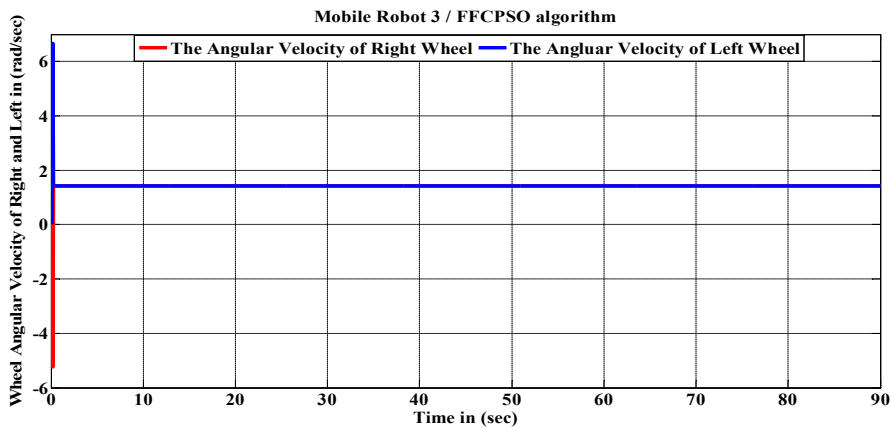


Figure (4.27): The angular velocity of right and left wheels for 3rd robot / case B based on FFCPSO algorithm.

From the figure (4.25), (4.26) and (4.27), the angular velocity of the right and left wheels for WMR₃ based on the CPSO, FF and FFCPSO approaches are equal to (1.44) rad/sec, (1.66) rad/sec and (1.41) rad/sec, respectively.

2. Wheels Linear Velocity

The linear velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR₁ system is displayed in Figures (4.28), (4.29) and (4.30), respectively.

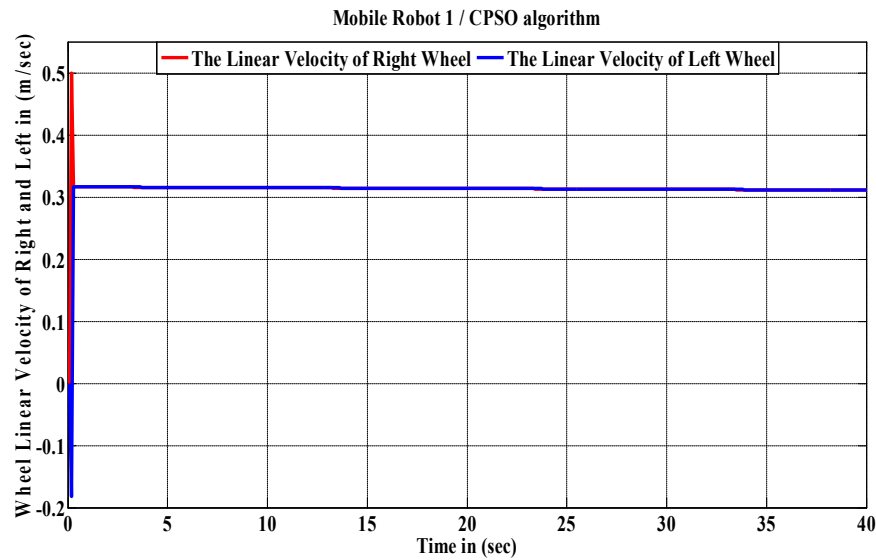


Figure (4.28): The linear velocity of right and left wheels for 1st robot / case B based on CPSO algorithm.

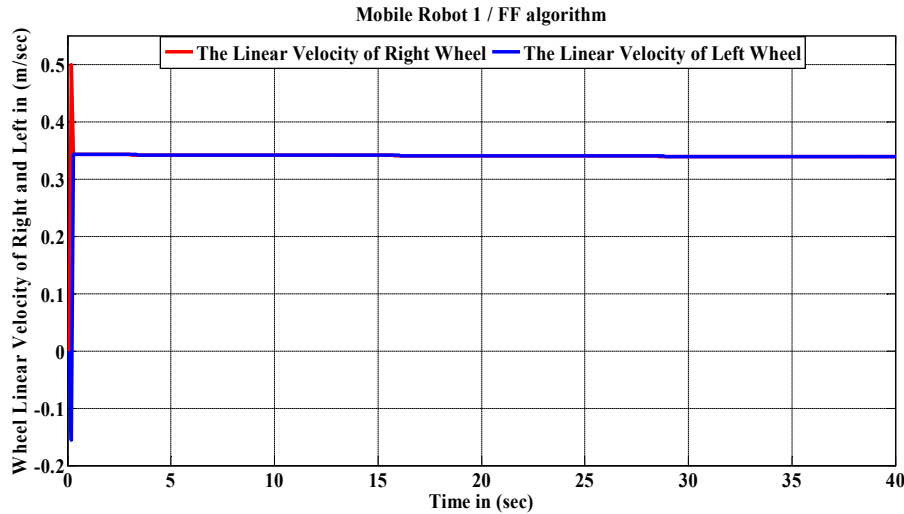


Figure (4.29): The linear velocity of right and left wheels for 1st robot / case B based on FF algorithm.

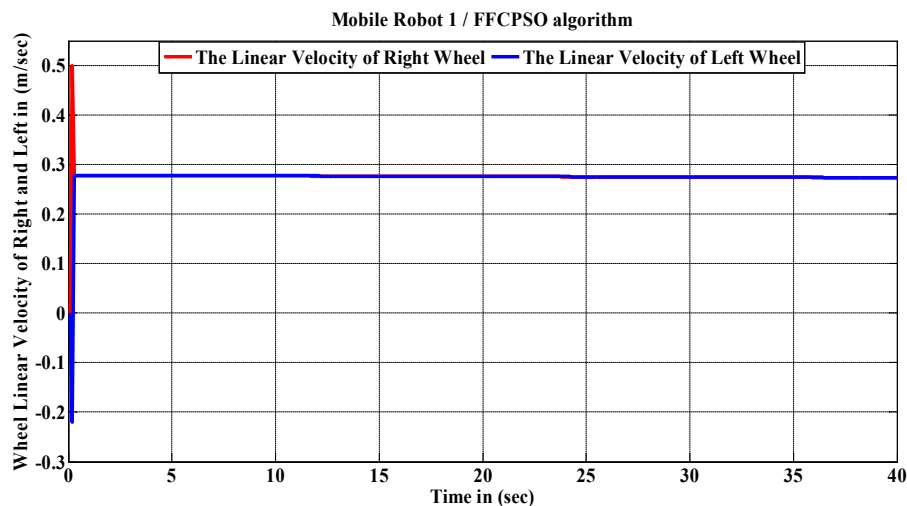


Figure (4.30): The wheel linear velocity of right and left for 1st robot / case B based on FFCPSO algorithm.

From figure (4.28), (4.29) and (4.30), the linear velocity of the right and left wheels for WMR_1 based on the CPSO, FF and FFCPSO approaches are equal to (0.31) m/sec, (0.34) m/sec and (0.28) m/sec, respectively.

The linear velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR_2 is clarified in Figures (4.31), (4.32) and (4.33), respectively.

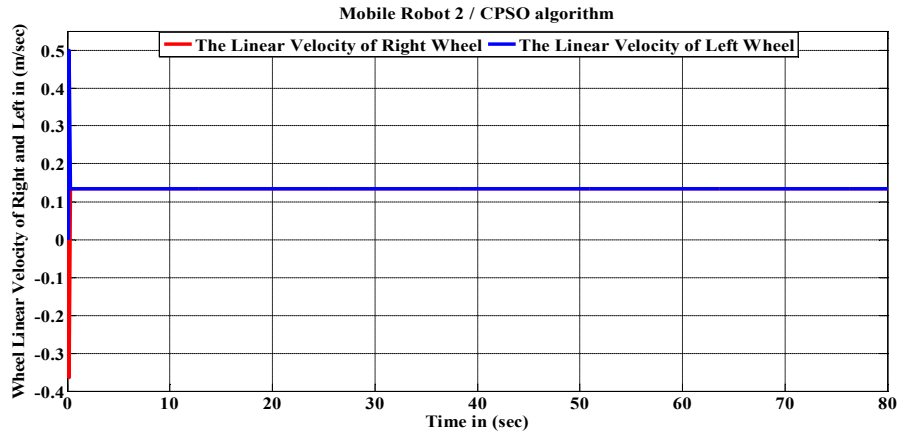


Figure (4.31): The linear velocity of right and left wheels for 2nd robot / case B based on CPSO algorithm.

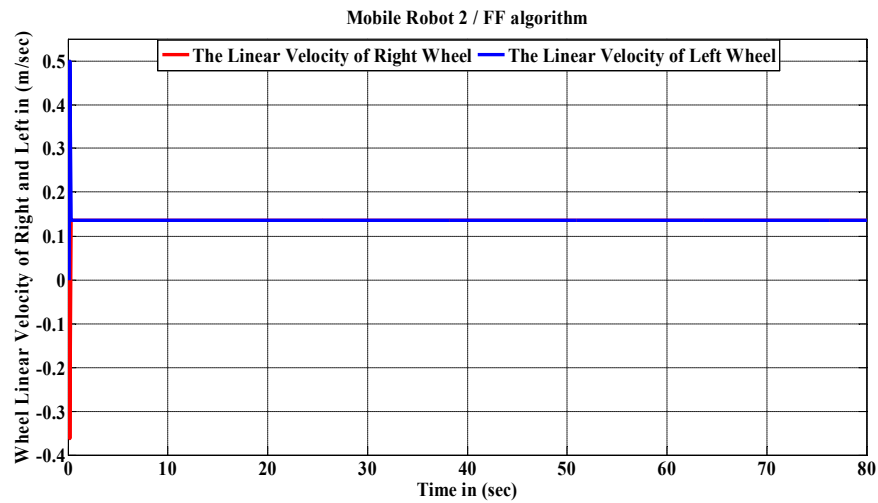


Figure (4.32): The linear velocity of right and left wheels for 2nd robot / case B based on FF algorithm.

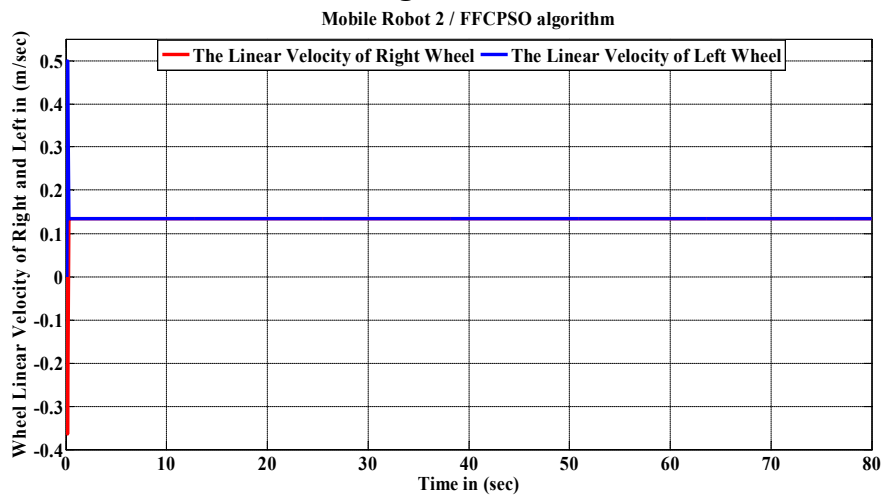


Figure (4.33): The linear velocity of right and left wheels for 2nd robot / case B based on FFCPSO algorithm.

From the figures (4.31), (4.32) and (4.33), the linear velocity of the right and left wheels for WMR₂ based on the CPSO, FF and FFCPSO approaches are equal to (0.134) m/sec, (0.136) m/sec and (0.135) m/sec, respectively.

The linear velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the specific path of WMR₃ is exhibited in Figures (4.34), (4.35) and (4.36), respectively.

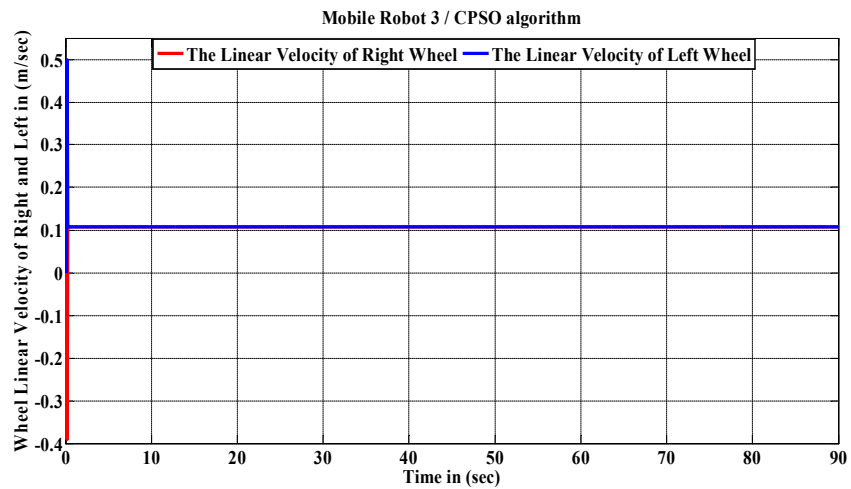


Figure (4.34): The linear velocity of right and left wheels for 3rd robot / case B based on CPSO algorithm.

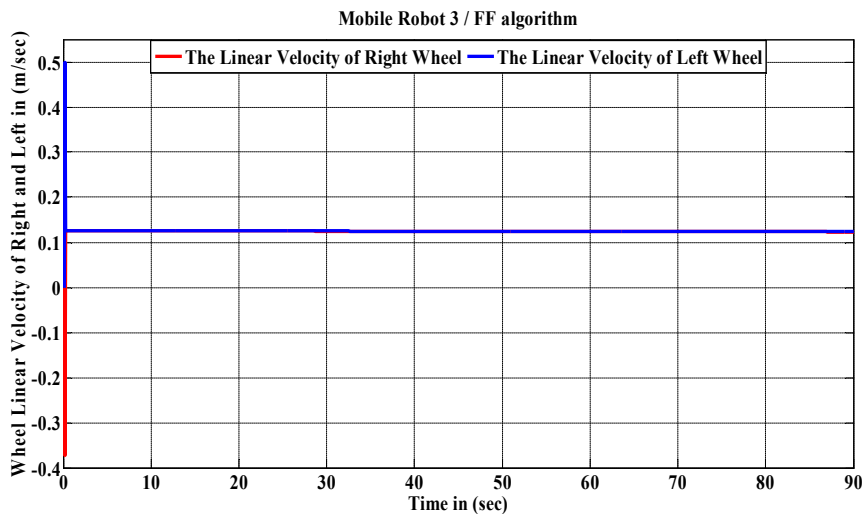


Figure (4.35): The linear velocity of right and left wheels for 3rd robot / case B based on FF algorithm.

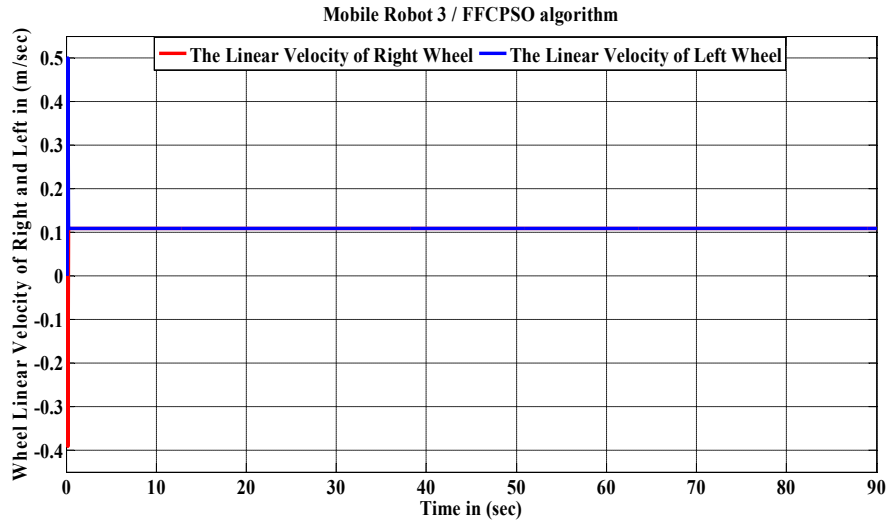


Figure (4.36): The linear velocity of right and left wheels for 3rd robot / case B based on FFCPSO algorithm.

From the figures (4.34), (4.35) and (4.36), the linear velocity of the right and left wheels for WMR₃ based on the CPSO, FF and FFCPSO approaches are equal to (0.108) m/sec, (0.125) m/sec and (0.108) m/sec, respectively.

3. Platform Linear and Angular Velocities

The linear and angular velocities of the platform based on CPSO, FF and FFCPSO algorithms for the path of WMR₁ is demonstrated in Figures (4.37), (4.38) and (4.39), respectively.

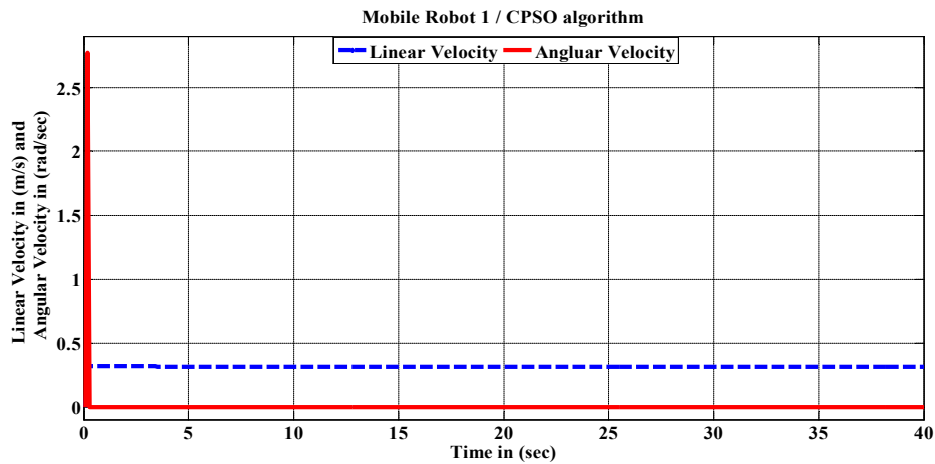


Figure (4.37): The platform angular and linear velocities for 1st robot / case B based on CPSO algorithm.

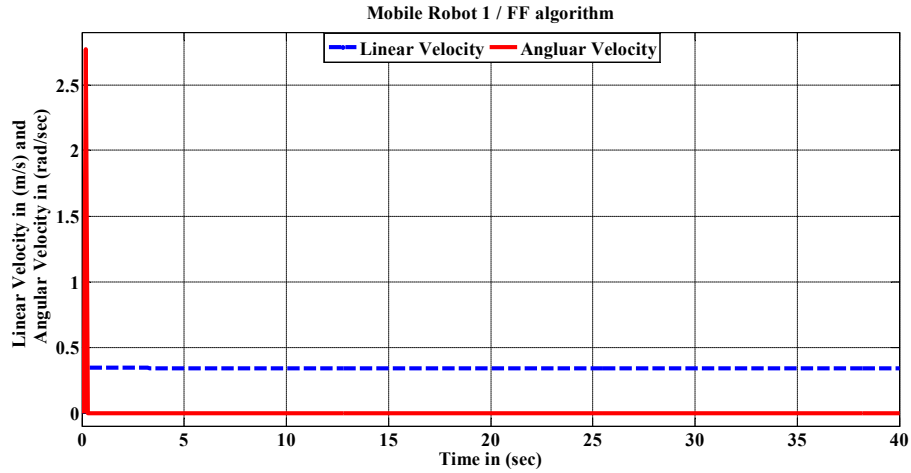


Figure (4.38): The platform angular and linear velocities for 1st robot / case B based on FF algorithm.

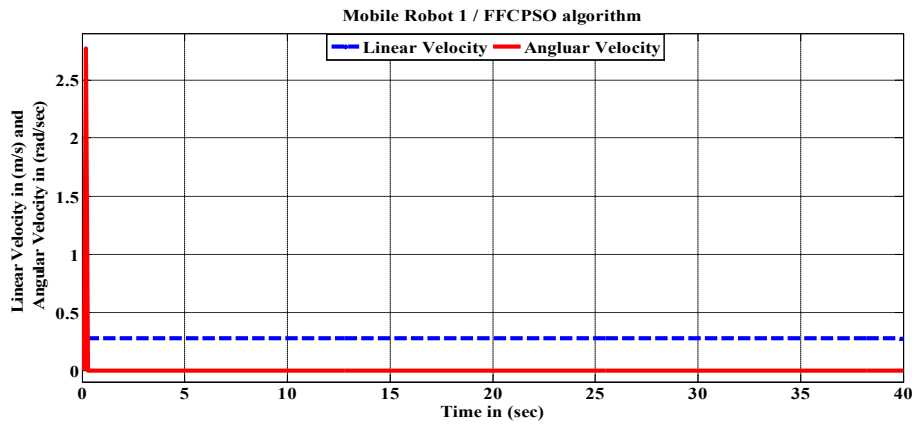


Figure (4.39): The platform angular and linear velocities for 1st robot / case B based on FFCPSO algorithm.

The linear and angular velocities of the platform based on the CPSO, FF and FFCPSO algorithms for the path of WMR₂ is exhibited in Figures (4.40), (4.41) and (4.42), respectively.

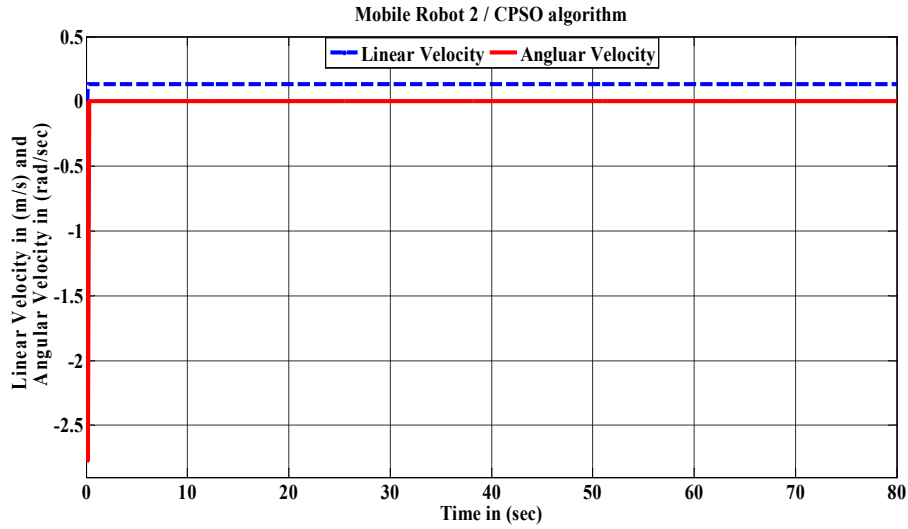


Figure (4.40): The platform angular and linear velocities for 2nd robot / case B based on CPSO algorithm.

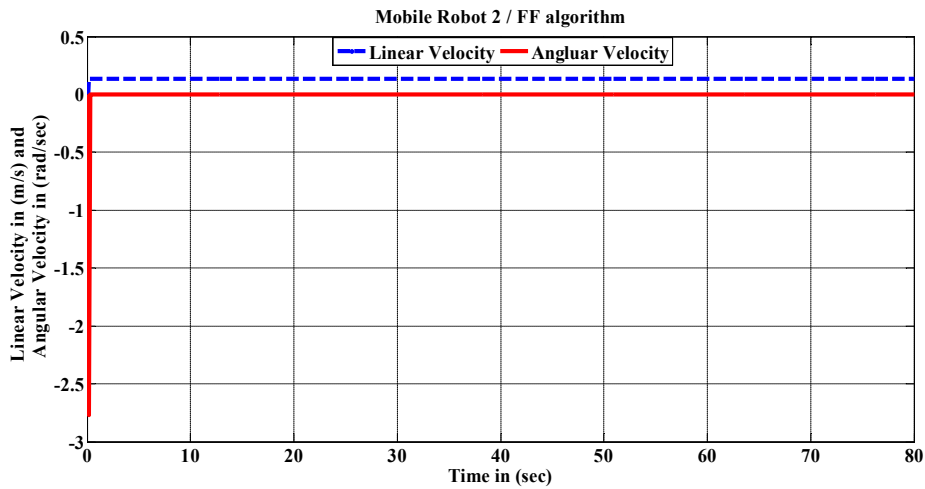


Figure (4.41): The platform angular and linear velocities for 2nd robot / case B based on FF algorithm.

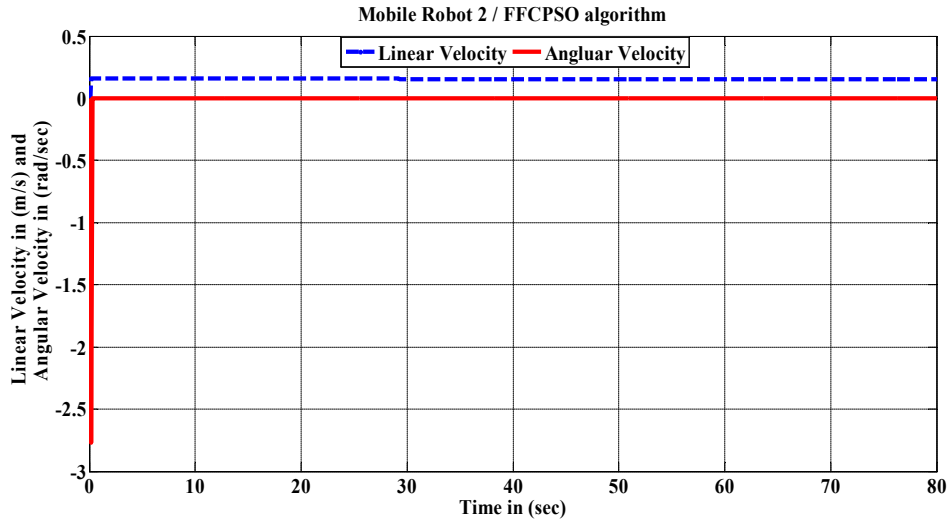


Figure (4.42): The platform angular and linear velocities for 2nd robot / case B based on FFCPSO algorithm.

The linear and angular velocities of the platform based on the CPSO, FF and FFCPSO algorithms for the path of WMR₃ is clarified in Figures (4.43), (4.44) and (4.45), respectively.

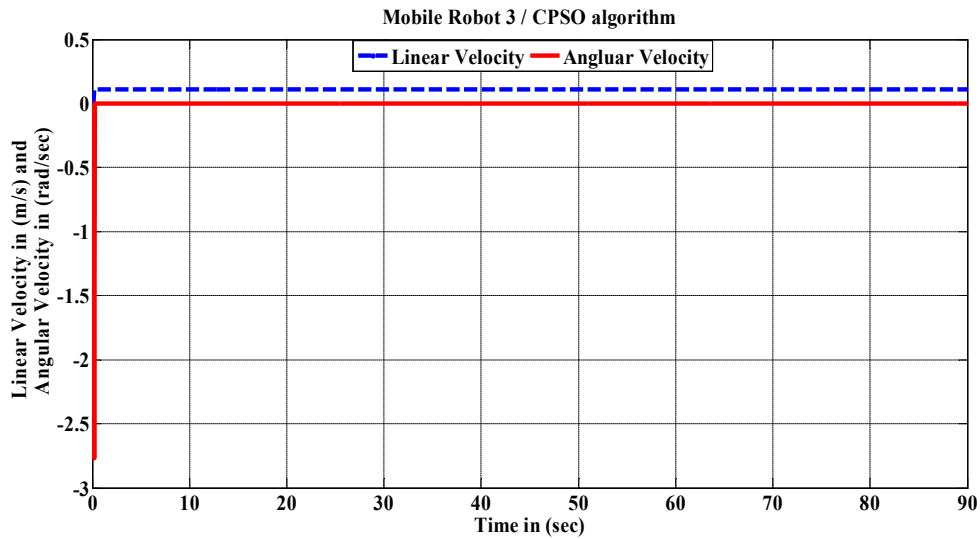


Figure (4.43): The platform angular and linear velocities for 3rd robot / case B based on CPSO algorithm.

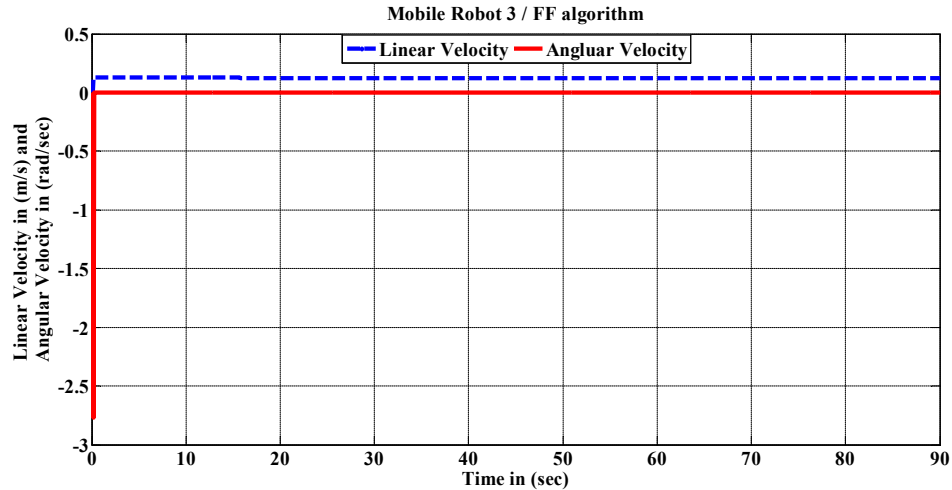


Figure (4.44): The platform angular and linear velocities for 3rd robot / case B based on FF algorithm.

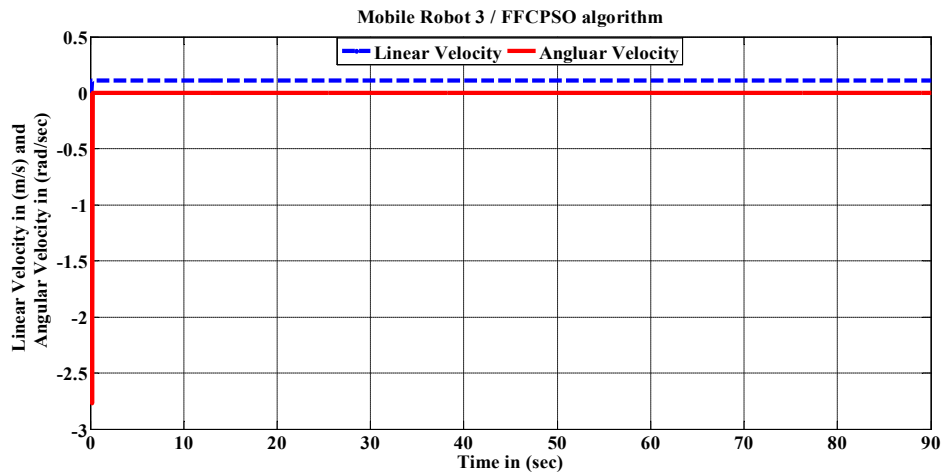


Figure (4.45): The platform angular and linear velocities for 3rd robot / case B based on FFCPSO algorithm.

4.3.3 Case C: Three Follow Up Wheeled NI- Mobile Robots

In the case of empty map, the total path length for the first, second and third wheeled NI- mobile robot from start to target point is equal to (1096.585) cm, (1131.37) cm and (1167.261) cm, respectively.

4.3.3.1 Follower Optimal Path Finding

Figures (4.46), (4.47), (4.48) and (4.49) reveal the simulation results of the optimum route for each wheeled NI- mobile robot based on the PSO, CPSO, FF and FFCPSO algorithms, respectively.

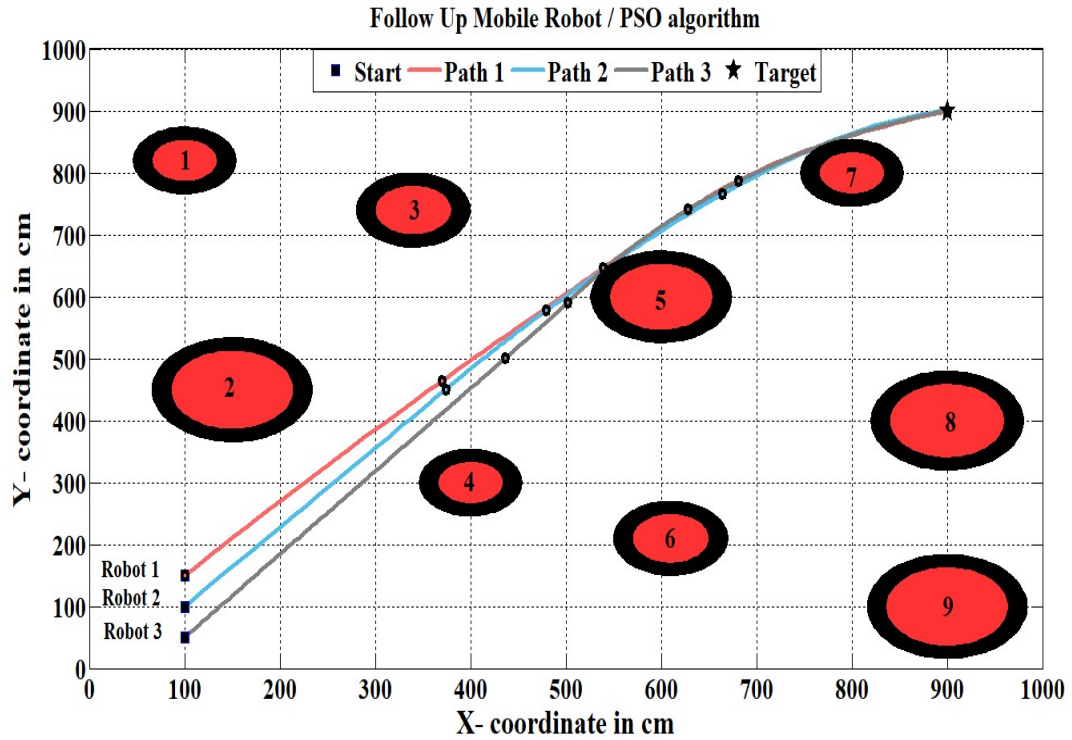


Figure (4.46): Follower Mobile Robots / case C Based on basic PSO algorithm.

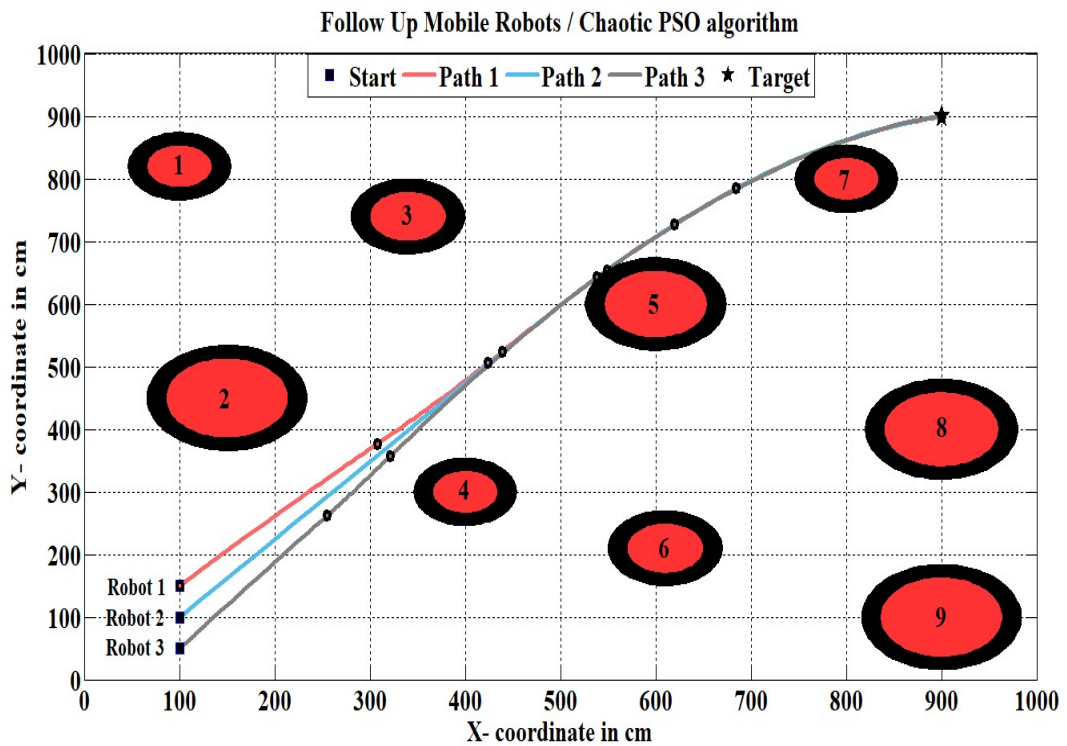


Figure (4.47): Follower Mobile Robots / case C Based on CPSO algorithm.

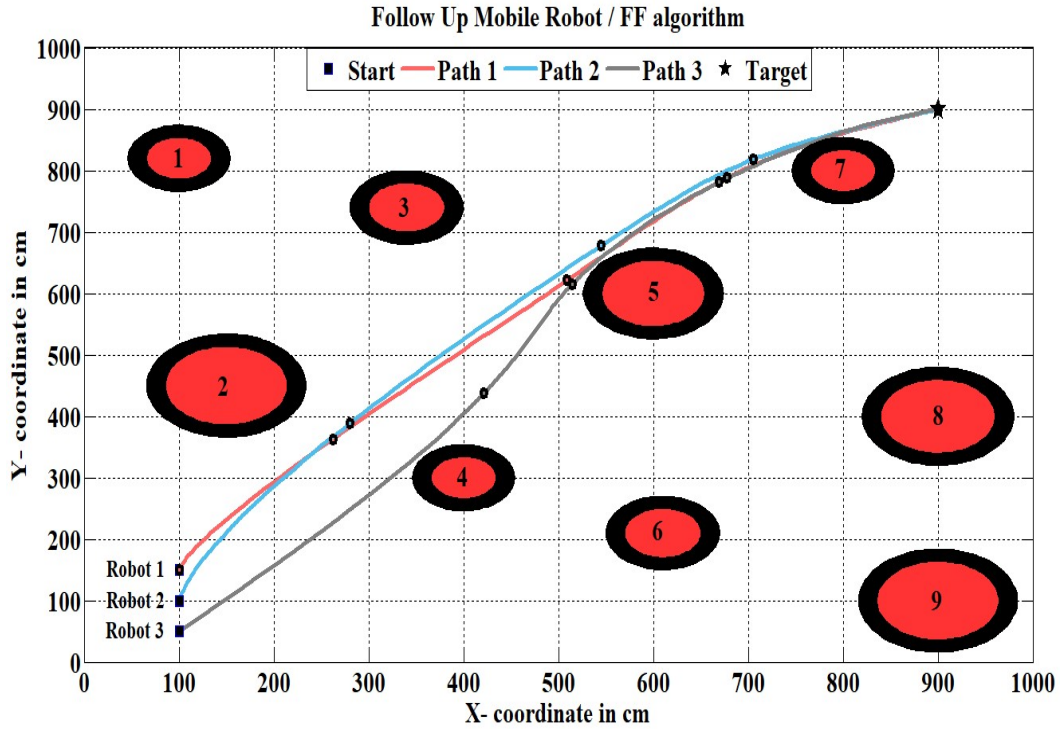


Figure (4.48): Follower Mobile Robots / case C Based on basic FF algorithm.

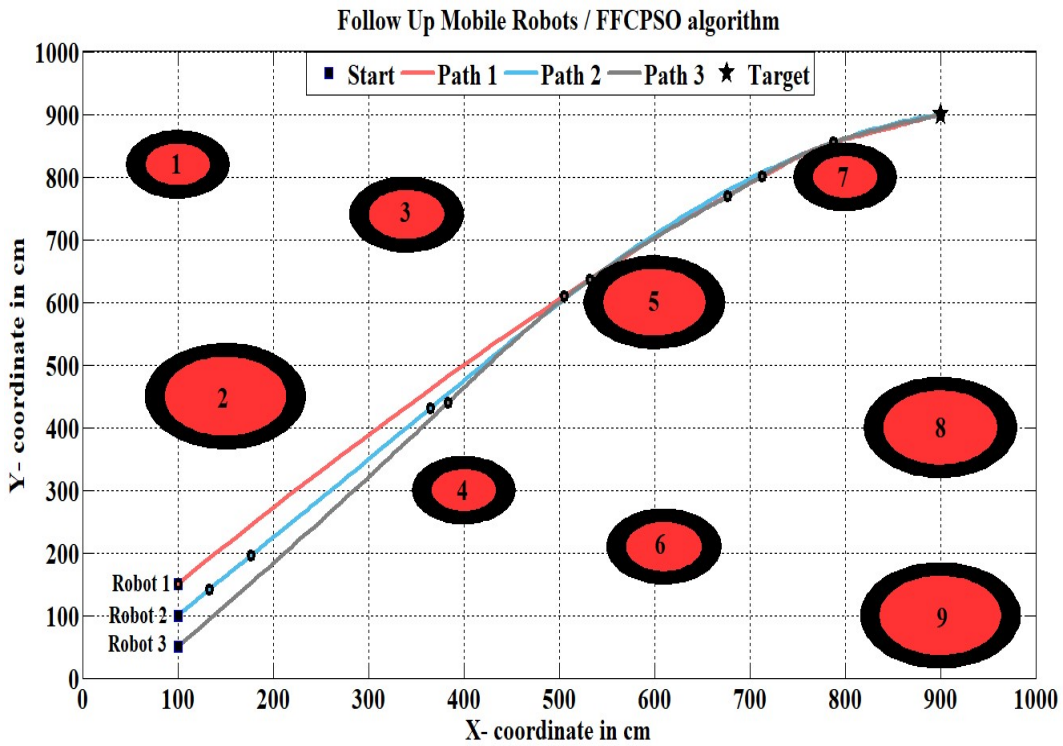


Figure (4.49): Follower Mobile Robots / case C Based on hybrid FFCPSO algorithm.

Based on the number of samples between start locations (X_s, Y_s) and target location (X_t, Y_t) , all the wheeled NI- mobile robots have the same travel time ($T_v = 80$) sec. Therefore, to avoid the collision that may occur between them, the WMR_1 has less velocity than the WMR_2 and the WMR_2 has less velocity than the WMR_3 . After ten runs, the outputted waypoints based on various algorithm are abstracted in Table (4.15).

Table (4.15): Waypoints coordination for case C.

Path No.	Waypoints (D)	Coordinate (X, Y)	Type of Intelligent Algorithm			
			PSO	CPSO	FF	FFCPSO
Path 1	D ₁	X (cm)	370.366	376.315	262.277	505.389
		Y (cm)	467.166	112.841	362.895	609.776
	D ₂	X (cm)	539.111	506.144	508.853	713.274
		Y (cm)	646.807	125.203	621.166	801.019
	D ₃	X (cm)	608.755	619.456	677.588	787.644
		Y (cm)	787.195	726.561	788.128	854.462
Path 2	D ₁	X (cm)	373.953	439.438	280.369	132.986
		Y (cm)	450.269	522.830	388.581	141.205
	D ₂	X (cm)	479.229	538.403	544.960	176.969
		Y (cm)	577.363	642.184	677.696	195.978
	D ₃	X (cm)	664.174	684.755	705.875	365.075
		Y (cm)	765.556	784.528	818.550	430.423
Path 3	D ₁	X (cm)	298.845	254.816	421.596	383.230
		Y (cm)	331.062	261.703	437.264	439.346
	D ₂	X (cm)	410.864	321.636	514.929	532.083
		Y (cm)	481.889	357.383	615.450	636.074
	D ₃	X (cm)	607.865	549.014	669.483	677.026
		Y (cm)	718.221	653.913	780.972	769.369

Figures (4.50), (4.51) and (4.52) explain the changing of the objective function through the number of iteration until reaching the optimal value for the first, second and third paths, respectively.

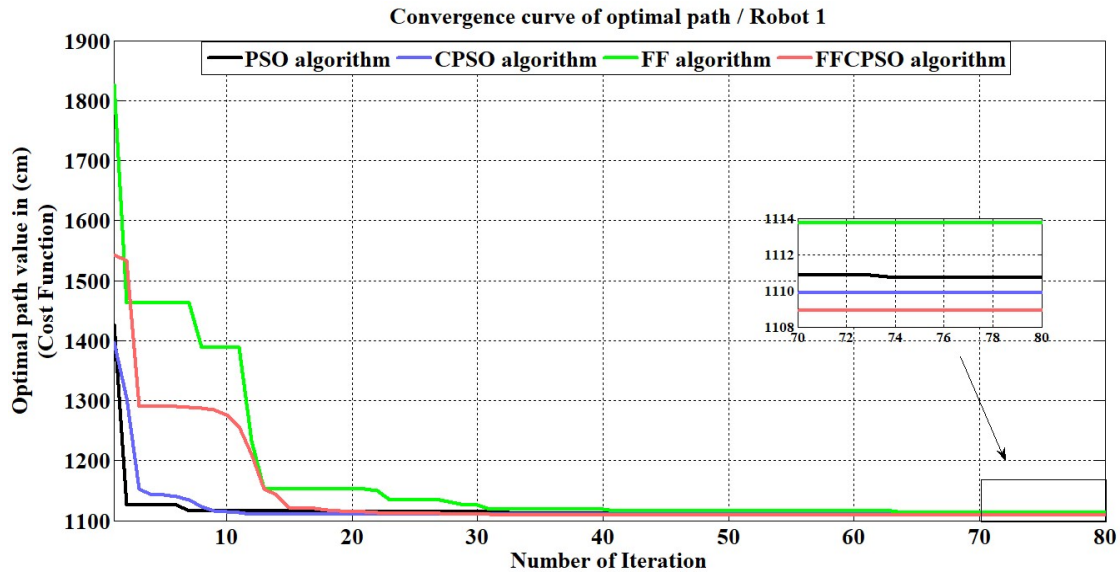


Figure (4.50): Variation of objective function for first path through the number of iterations / case C.

From the figure (4.50), the optimal path with shortest distance using standard PSO is equal to (1110.71) with in iteration (74) and is equal to (1109.89) with in iteration (24) in the CPSO algorithm. The optimal path with shortest distance using standard FF is equal to (1113.74) with in iteration (64) and is equal to (1108.98) with in iteration (48) in the proposed hybrid FFCPSO algorithm.

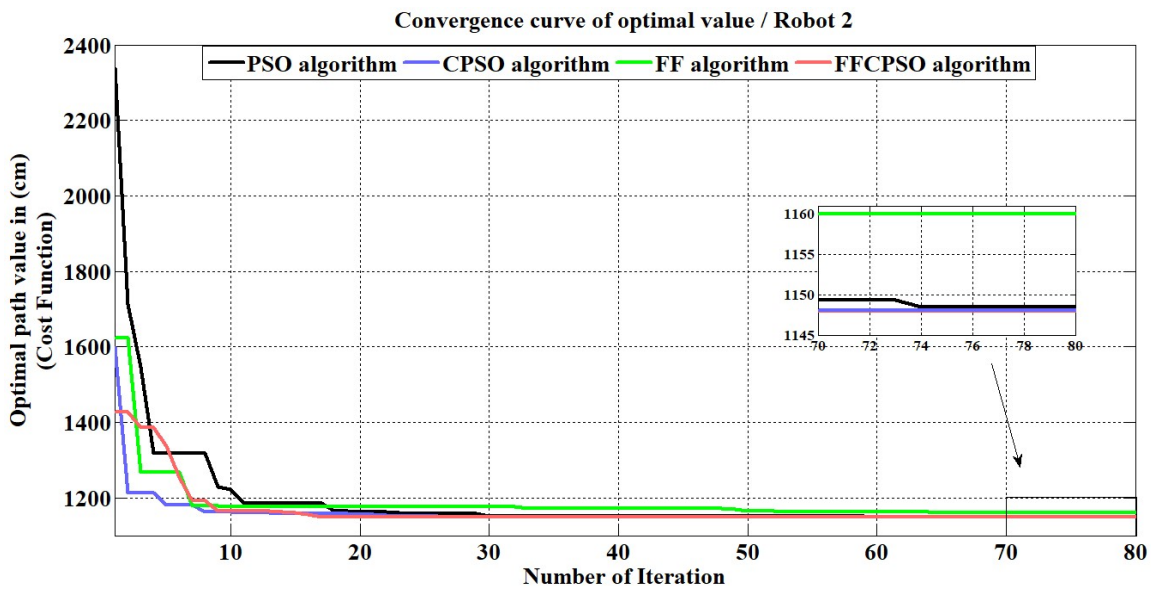


Figure (4.51): Variation of objective function for second path through the number of iterations / case C.

From the figure (4.51), the optimal path with shortest distance using standard PSO is equal to (1148.48) with in iteration (74) and is equal to (1148.04) with in iteration (45) in the CPSO algorithm. The optimal path with shortest distance using standard FF is equal to (1159.98) with in iteration (65) and is equal to (1147.96) with in iteration (46) in the proposed hybrid FFCPSO algorithm.

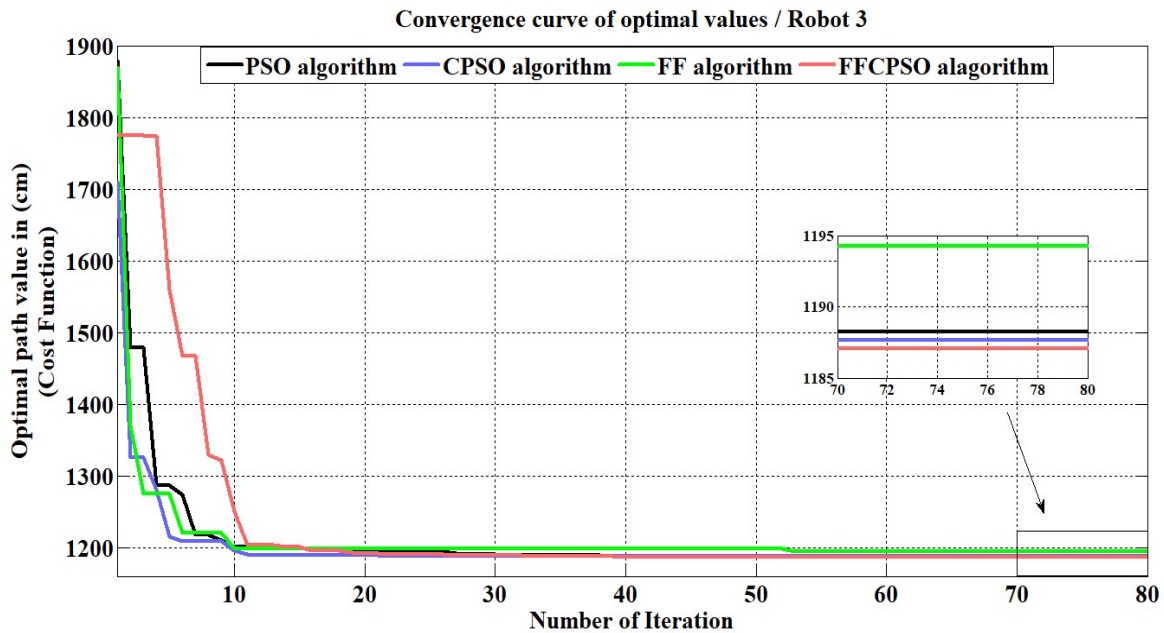


Figure (4.52): Variation of objective function for third path through iterations / case C.

From the figure (4.52), the optimal path with shortest distance using standard PSO is equal to (1188.24) with in iteration (57) and is equal to (1187.68) with in iteration (43) in the CPSO algorithm. The optimal path with shortest distance using standard FF is equal to (1194.27) with in iteration (53) and is equal to (1187.17) with in iteration (45) in the FFCPSO algorithm.

Since the candidate solutions varied intensively, the PSO algorithm can achieve best path with shortest distance in run no.7. While the best path with the shortest distance by using CPSO algorithm is achieved in run no. 6. FF algorithm can achieve it in run no. 10 and FFCPSO algorithm in run no. 8.

After executing the programs of various intelligent optimization algorithms ten times, the results for the case C are summarized in Table (4.16). Moreover, the percentages of the objective function between algorithms are given in Table (4.17).

Table (4.16): Comparison Results for case C.

Robot No.	Type of Intelligent Algorithm	Min. Distance in (cm)	Fitness	Iteration of best value	Max. Distance in (cm)	Average in (cm)	Standard Deviation
1	PSO	1110.71	0.09	74	1114.2	1111.49	0.0323
	CPSO	1109.89	0.09	24	1112.28	1110.15	7.023E-3
	FF	1113.74	0.0897	64	1119.04	1115.34	0.0227
	FFCPSO	1108.98	0.0901	48	1110.2	1109.58	4.481E-3
2	PSO	1148.48	0.087	74	1148.83	1148.6	1.506E-3
	CPSO	1148.04	0.0871	45	1148.79	1148.18	2.191E-3
	FF	1159.98	0.0862	65	1162.77	1160.83	8.737E-3
	FFCPSO	1147.96	0.0871	46	1148.35	1148.05	1.468E-3
3	PSO	1188.24	0.0841	57	1190.45	1188.81	5.637E-3
	CPSO	1187.68	0.0841	43	1188.92	1188.13	4.883E-3
	FF	1194.27	0.0837	53	1199.39	1197.49	0.015
	FFCPSO	1187.17	0.0842	45	1188.15	1187.77	2.610E-3

Table (4.17): Percentages of objective function between algorithms for case C.

Robot No.	Algorithm Type	PSO-CPSO	PSO-FF	PSO-FFCPSO	CPSO-FF	CPSO-FFCPSO	FF-FFCPSO
1	Percentage, %	0.0738	0.272	0.1557	0.3456	0.0819	0.4273
2		0.0383	0.9913	0.0452	1.0293	6.968E-3	1.0362
3		0.0681	0.4839	0.111	0.5518	0.0429	0.5945

The presented algorithms tries to achieve the best path for the three follower mobile robots. These mobile robots have the same travel time on the desired path because they have the same number of samples between starting and target positions but difference in robot's start time in order to avoid the collision that may occur between these robots and based on national instrument robot length, there are difference in robot's start time. Where, the WMR_1 is start to move at ($T_v = 0$) sec,

the WMR₂ is start to move at ($T_v = 5$) sec and the WMR₃ is start to move at ($T_v = 10$) sec. From the case C results, it is clearly to say that CPSO approach can provide a wonderful following for three wheeled NI- mobile robots.

4.3.3.2 Follower Mobile Robots Velocities

After finding the shortest paths of WMRs based on each swarm-based optimization technique, the desired path equations for WMR₁ based on CPSO, FF and FFCPSO are given in fitting equations (4.13), (4.14) and (4.15), respectively.

$$y(x) = -8.9 \times 10^{-10}x^4 + 4.9 \times 10^{-7}x^3 + 0.00028x^2 + 0.91x + 61 \quad (4.13)$$

$$y(x) = -2.5 \times 10^{-9}x^4 + 4.2 \times 10^{-6}x^3 - 0.0028x^2 + 1.9x - 5.3 \quad (4.14)$$

$$y(x) = -1.4 \times 10^{-9}x^4 + 2.2 \times 10^{-6}x^3 - 0.0015x^2 + 1.6x + 3.2 \quad (4.15)$$

The desired path equations for WMR₂ based on CPSO, FF and FFCPSO are given in equations (4.16), (4.17) and (4.18), respectively.

$$y(x) = 2 \times 10^{-10}x^4 - 1.5 \times 10^{-6}x^3 + 0.0013x^2 + 0.89x + 3.9 \quad (4.16)$$

$$y(x) = -2.7 \times 10^{-9}x^4 + 5.1 \times 10^{-6}x^3 - 0.004x^2 + 2.5x - 1 \times 10^{+2} \quad (4.17)$$

$$y(x) = -1 \times 10^{-6}x^3 + 0.00095x^2 + 0.99x - 4.8 \quad (4.18)$$

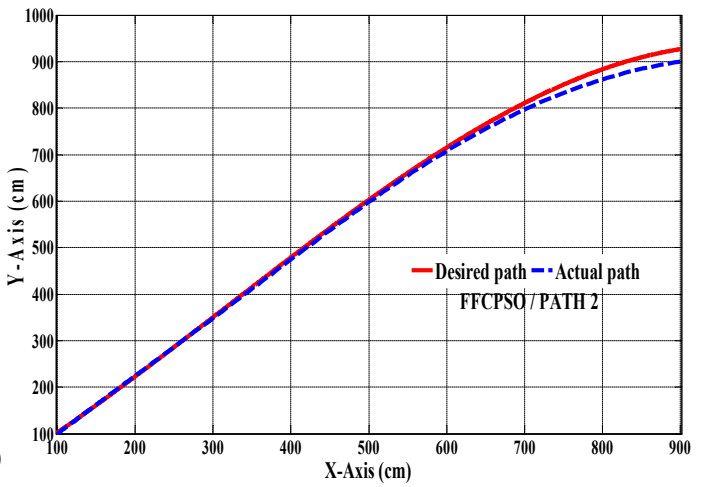
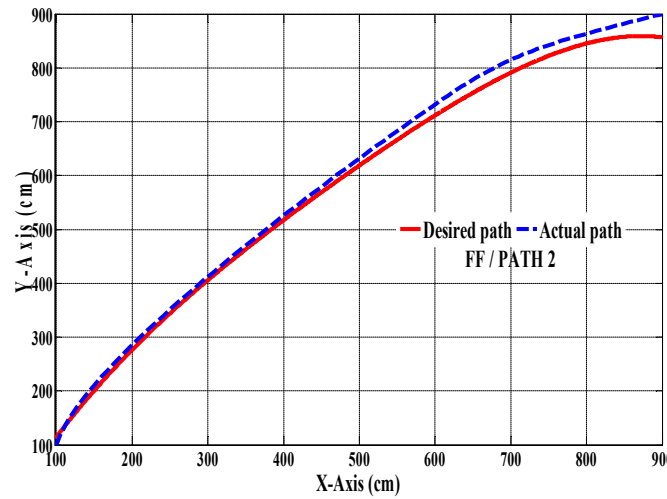
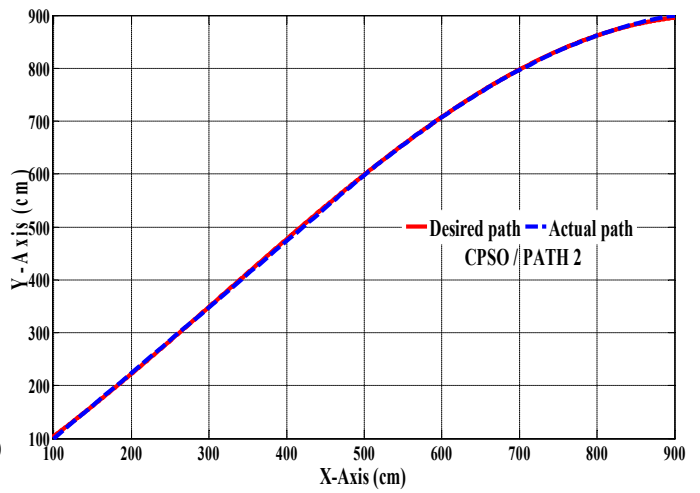
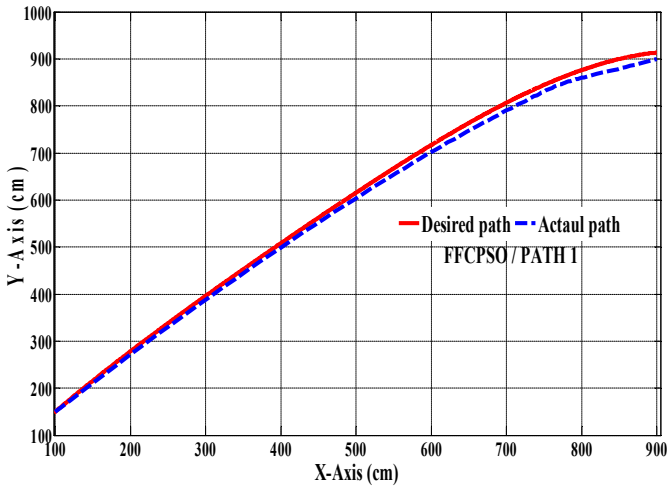
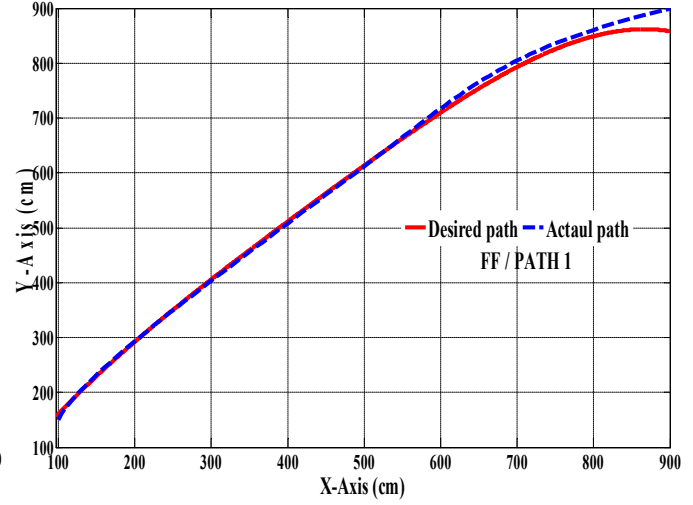
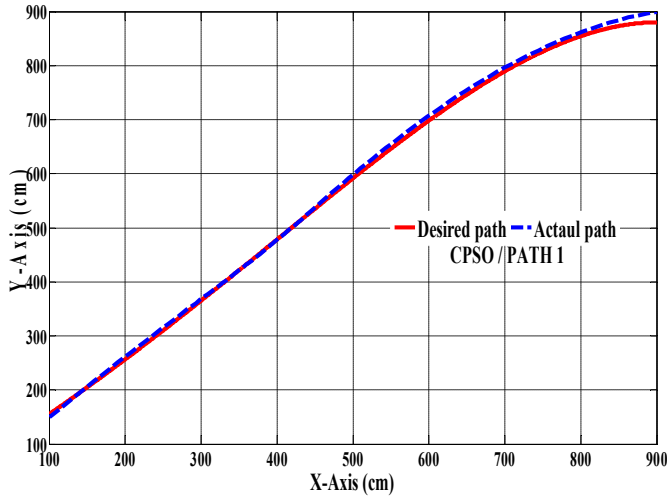
While, the desired path equations for WMR₃ based on CPSO, FF and FFCPSO are given in equations (4.19), (4.20) and (4.21), respectively.

$$y(x) = 1.2 \times 10^{-9}x^4 - 3.3 \times 10^{-6}x^3 + 0.0023x^2 + 0.84x - 49 \quad (4.19)$$

$$y(x) = -2.8 \times 10^{-6}x^3 + 0.0035x^2 + 0.061x + 16 \quad (4.20)$$

$$y(x) = 1.5 \times 10^{-9}x^4 - 3.9 \times 10^{-6}x^3 + 0.0026x^2 + 0.74x - 45 \quad (4.21)$$

Figure (4.53) shows the both actual and desired paths for WMR₁, WMR₂ and WMR₃ based on CPSO, FF and FFCPSO algorithms, respectively.



Continued

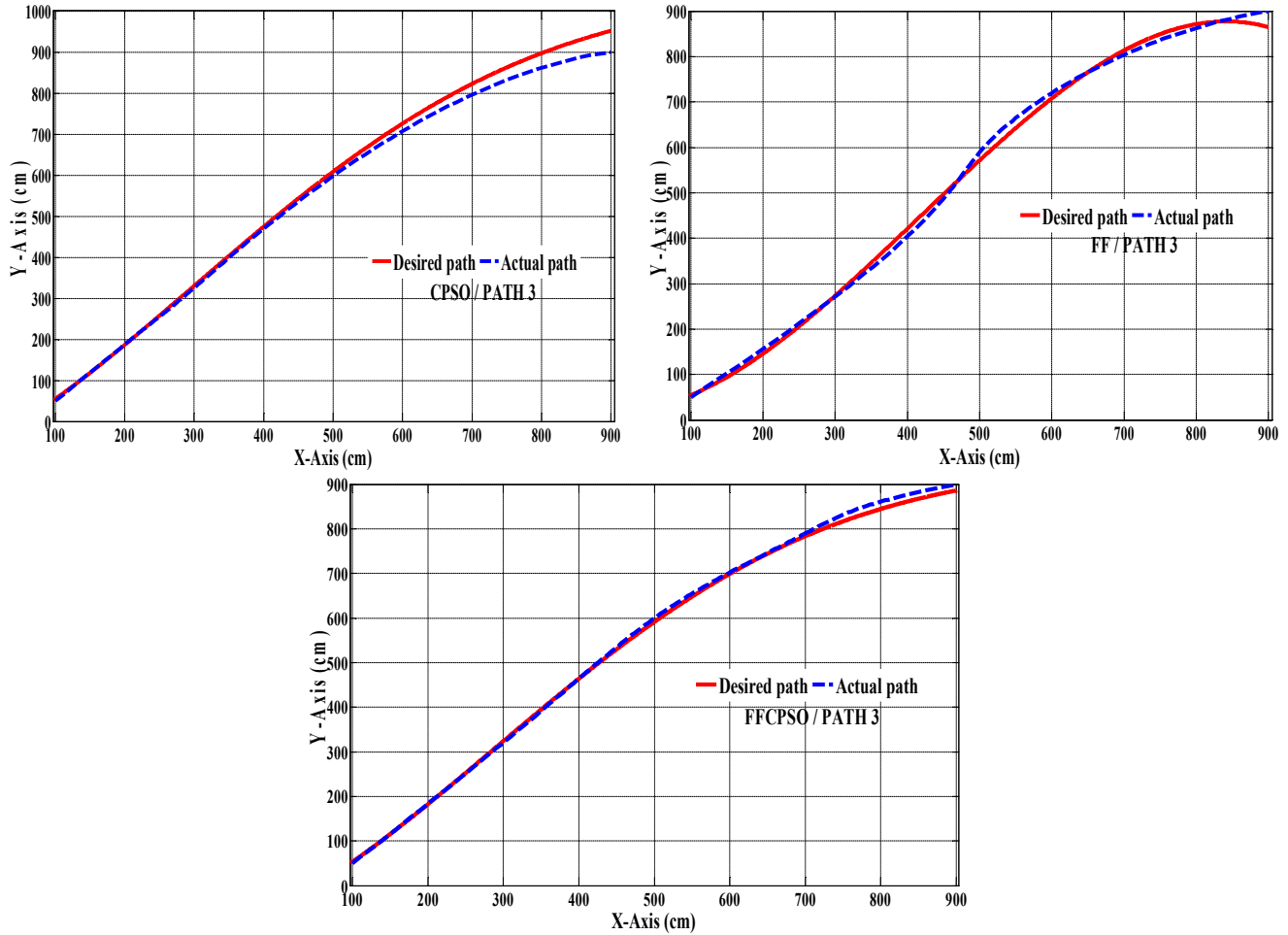


Figure (4.53): Desired and actual path for case C.

The distance error between actual and desired path for each NI-mobile robot based on three approaches are explained in Table (4.18).

Table (4.18): Distance error for case C.

Path No.		Type of Intelligent Algorithm		
		CPSO	FF	FFCPSO
1	Distance Error 100%	1.12	2.05	0.98
2		0.39	2.92	1.19
3		2.28	8.43	2.03

Subsequently, based on the kinematic equations of differential drive mobile robot, one can calculate the robot velocity on its special path as follows:

1. Wheels Angular Velocity

The angular velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR_1 is shown in Figures (4.54), (4.55) and (4.56), respectively.

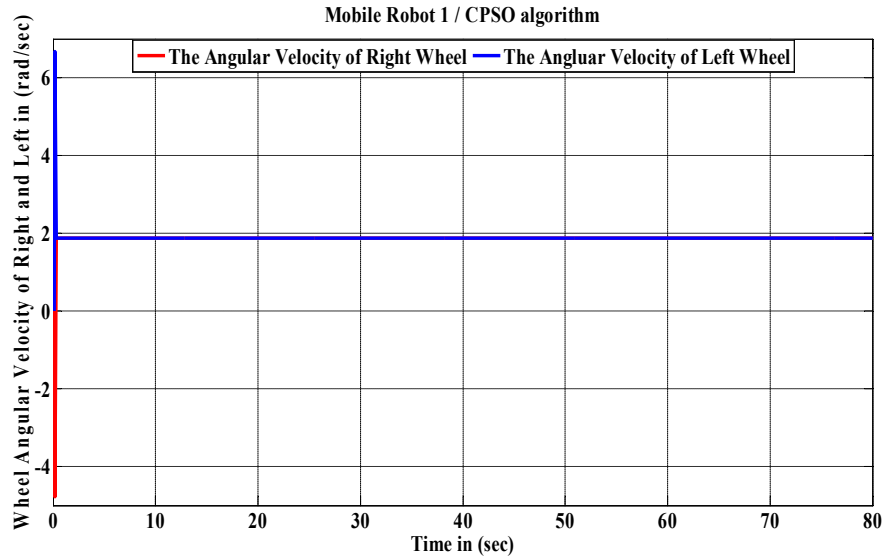


Figure (4.54): The angular velocity of right and left wheels for 1st robot / case C based on CPSO algorithm.

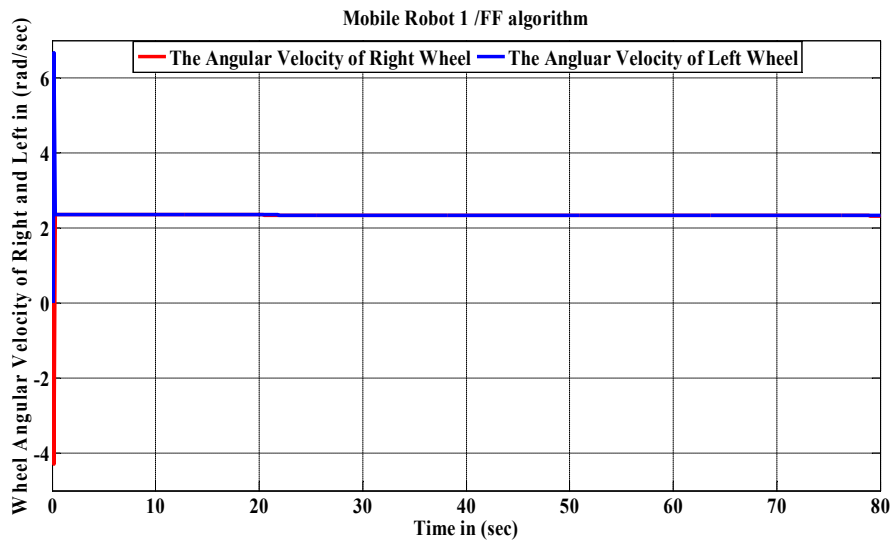


Figure (4.55): The angular velocity of right and left wheels for 1st robot / case C based on FF algorithm.

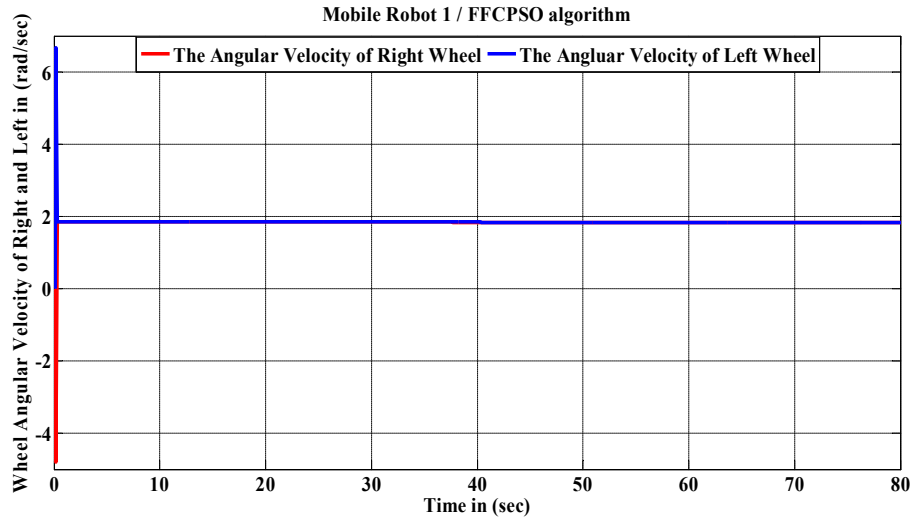


Figure (4.56): The angular velocity of right and left wheels for 1st robot / case C based on FFCPSO algorithm.

From the figure (4.54), (4.55) and (4.56), the angular velocity of right and left wheels for WMR_1 based on CPSO, FF and FFCPSO approaches are equal to (1.86) rad/sec, (2.34) rad/sec and (1.85) rad/sec, respectively.

The angular velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR_2 is illustrated in Figures (4.57), (4.58) and (4.59), respectively.

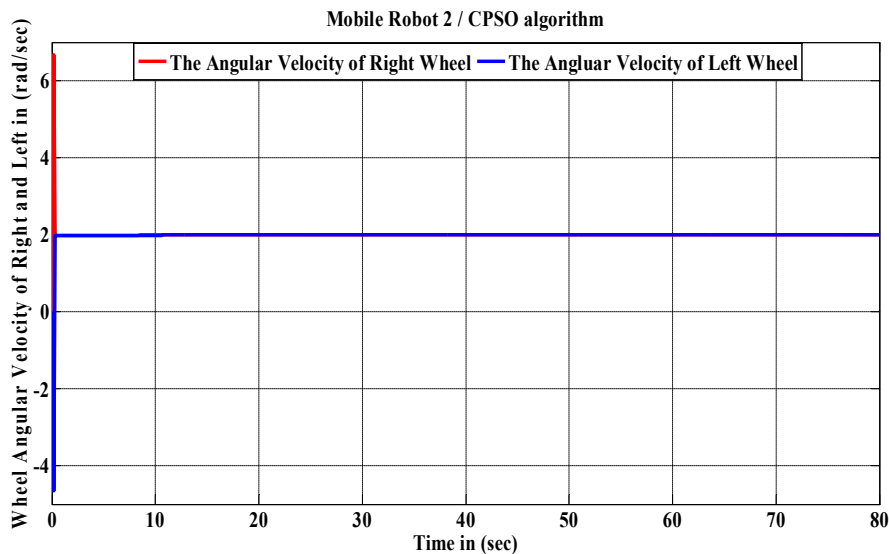


Figure (4.57): The angular velocity of right and left wheels for 2nd robot/ case C based on CPSO algorithm.

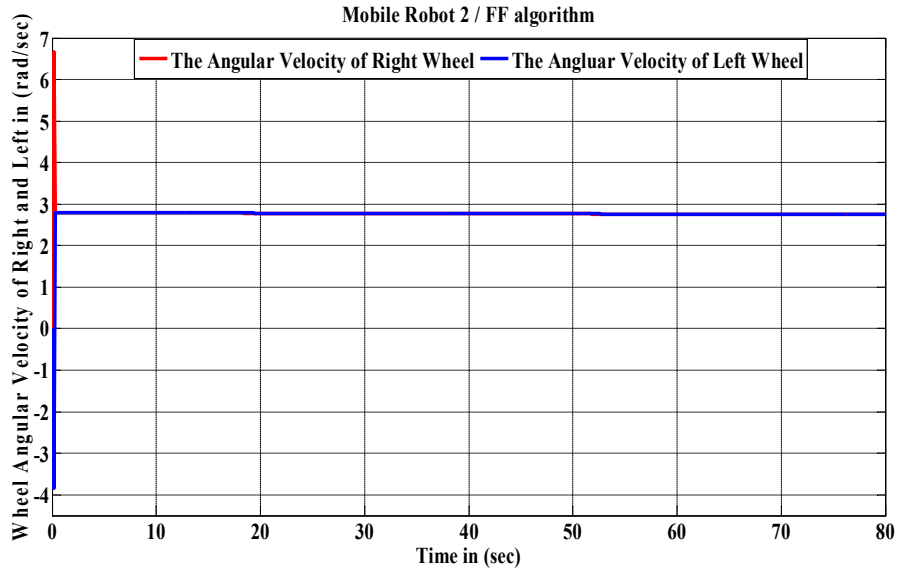


Figure (4.58): The angular velocity of right and left wheels for 2nd robot / case C based on FF algorithm.

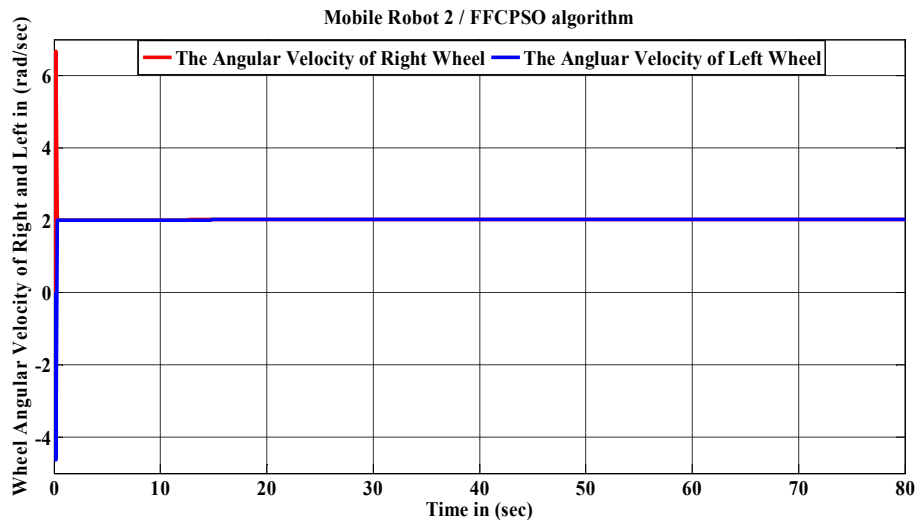


Figure (4.59): The angular velocity of right and left wheels for 2nd robot / case C based on FFCPSO algorithm.

From the figure (4.57), (4.58) and (4.59), the angular velocity of right and left wheels For WMR₂ based on CPSO, FF and FFCPSO approaches are equal to (1.99) rad/sec, (2.75) rad/sec and (2.01) rad/sec, respectively.

The angular velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR₃ is displayed in Figures (4.60), (4.61) and (4.62), respectively.

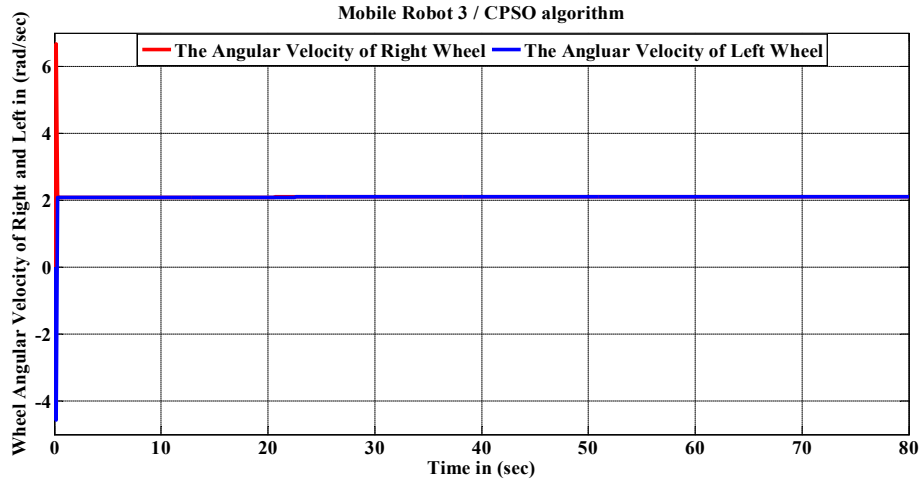


Figure (4.60): The angular velocity of right and left wheels for 3rd robot/ case C based on CPSO algorithm.

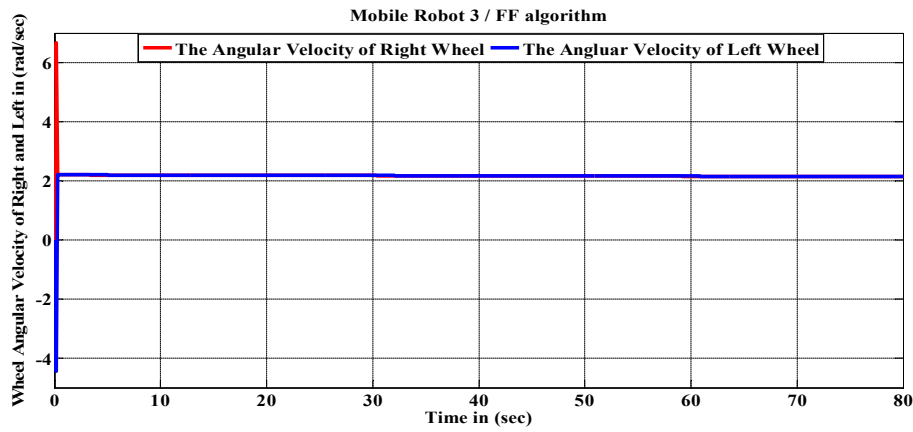


Figure (4.61): The angular velocity of right and left wheels for 3rd robot/ case C based on FF algorithm.

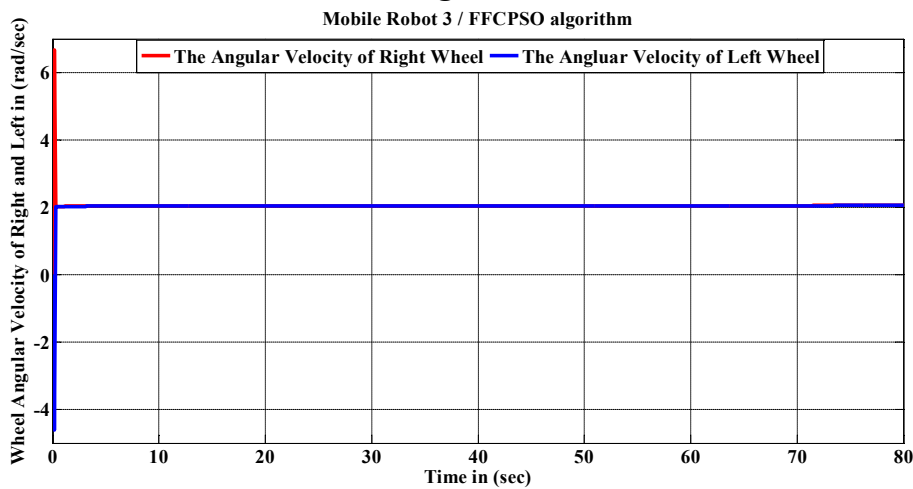


Figure (4.62): The angular velocity of right and left wheels for 3rd robot / case C based on FFCPSO algorithm.

While from the figure (4.60), (4.61) and (4.62), the angular velocity of right and left wheels for WMR₃ based on CPSO, FF and FFCPSO approaches are equal to (2.08) rad/sec, (2.15) rad/sec and (2.04) rad/sec, respectively.

2. Wheels Linear Velocity

Figures (4.63), (4.64) and (4.65) clarify the left and right wheels linear velocity for the path of WMR₁ based on the CPSO, FF and FFCPSO algorithms, respectively

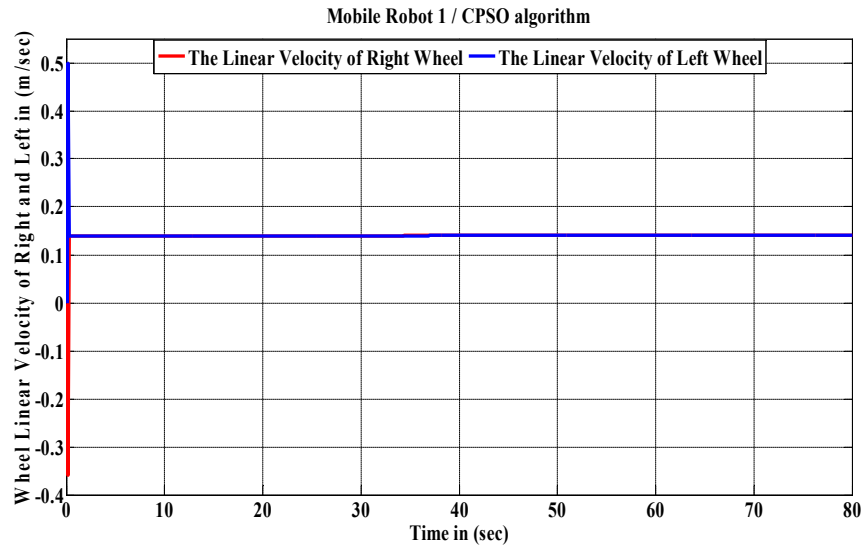


Figure (4.63): The linear velocity of right and left wheels for 1st robot / case C based on CPSO algorithm.

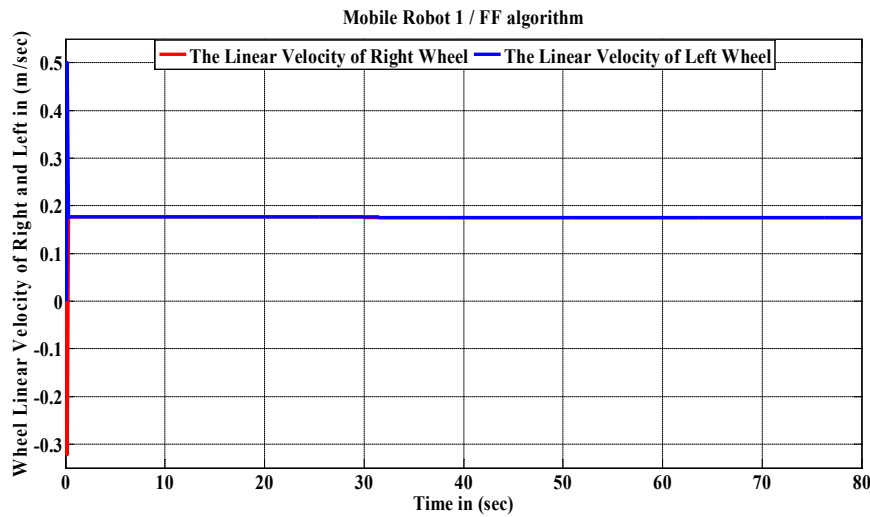


Figure (4.64): The linear velocity of right and left wheels for 1st robot / case C based on FF algorithm.

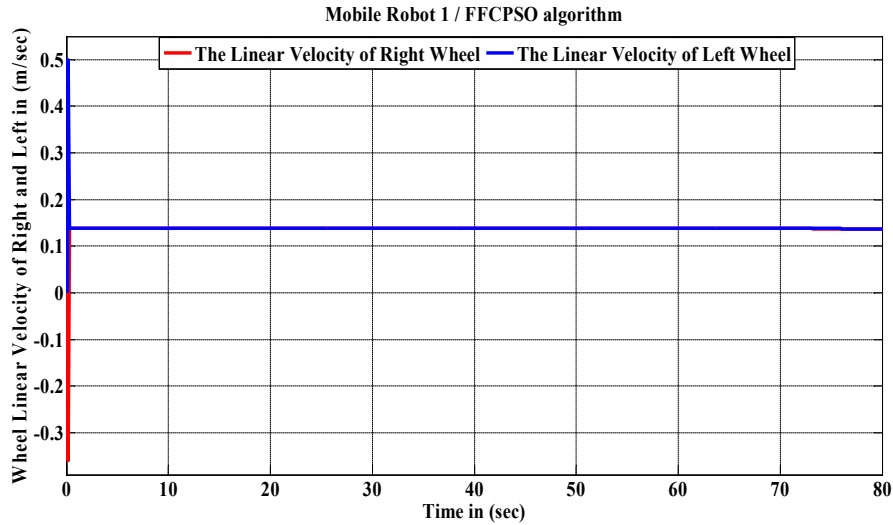


Figure (4.65): The linear velocity of right and left wheels 1st robot/ case C based on FFCPSO algorithm.

From the figures (4. 63), (4.64) and (4.65), the linear velocity of the right and left wheels for WMR₁ based on the CPSO, FF and FFCPSO approaches are equal to (0.139) m/sec, (0.165) m/sec and (0.138) m/sec, respectively.

The linear velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR₂ is revealed in Figures (4.66), (4.67) and (4.68), respectively.

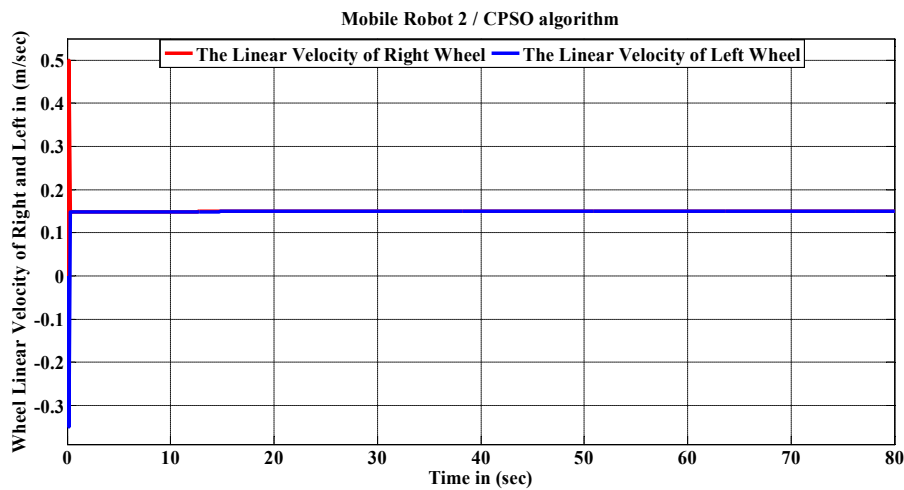


Figure (4.66): The linear velocity of right and left wheels for 2nd robot / case C based on CPSO algorithm.

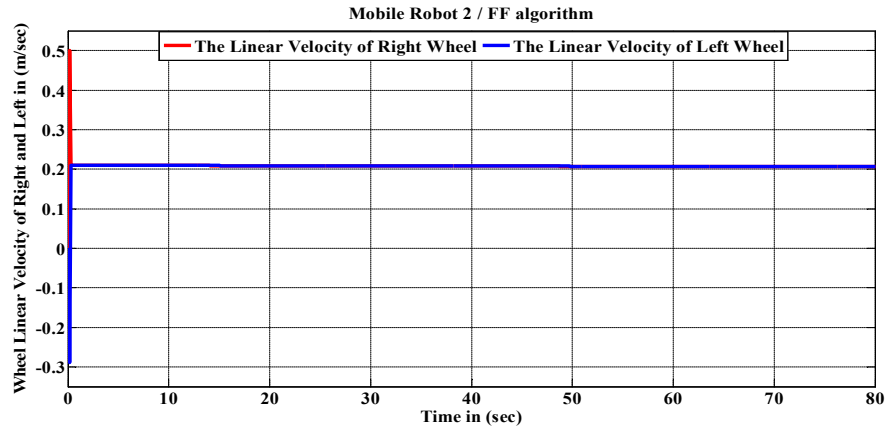


Figure (4.67): The linear velocity of right and left wheels for 2nd robot / case C based on FF algorithm.

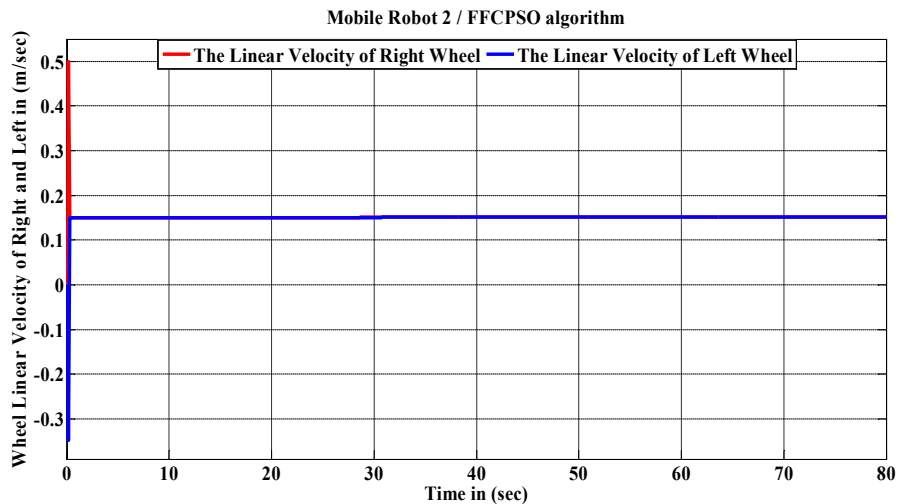


Figure (4.68): The linear velocity of right and left wheels for 2nd robot/ case C based on FFCPSO algorithm.

For WMR₂, the linear velocity of the right and left wheels based on the CPSO, FF and FFCPSO approaches are equal to (0.149) m/sec, (0.177) m/sec and (0.15) m/sec, respectively.

The linear velocity of the left and right wheels based on the CPSO, FF and FFCPSO algorithms for the path of WMR₃ is manifested in Figures (4.69), (4.70) and (4.71), respectively.

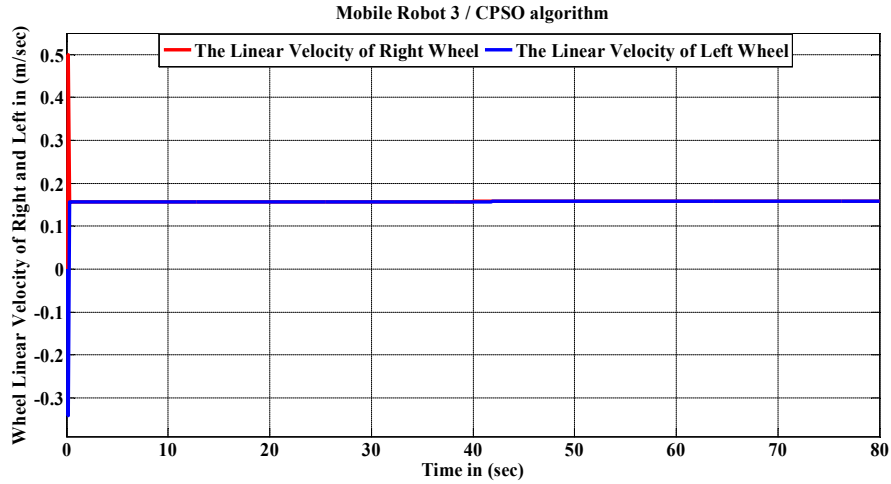


Figure (4.69): The linear velocity of right and left wheels for 3rd robot / case C based on CPSO algorithm.

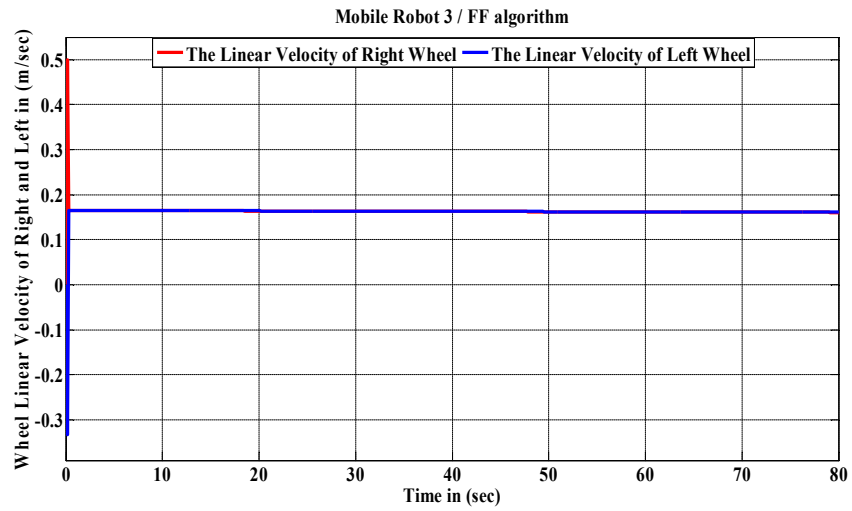


Figure (4.70): The linear velocity of right and left wheels for 3rd robot / case C based on FF algorithm.

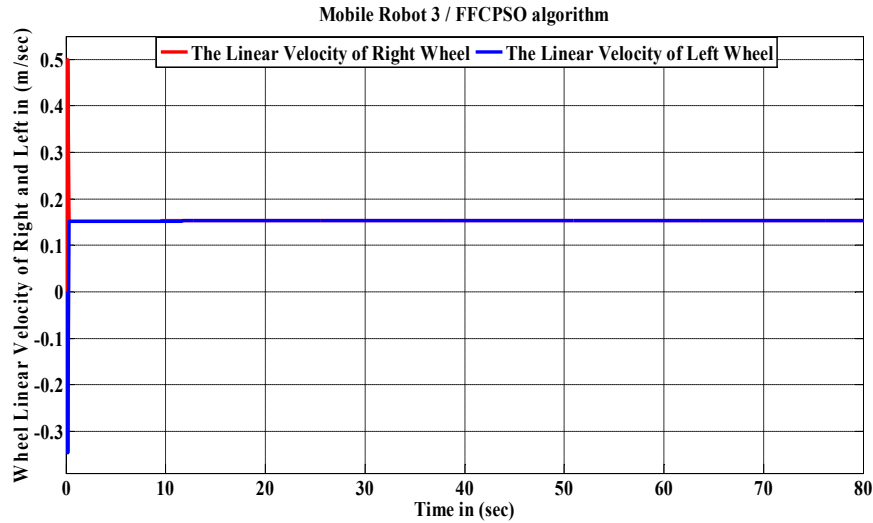


Figure (4.71): The linear velocity of right and left wheels for 3rd robot / case C based on FFCPSO algorithm.

While for WMR₃, the linear velocity of the right and left wheels based on the CPSO, FF and FFCPSO approaches are equal to (0.157) m/sec, (0.21) m/sec and (0.154) m/sec, respectively.

3. Platform Linear and Angular Velocity

The angular and linear velocities of the platform based on the CPSO, FF and FFCPSO algorithms for WMR₁ is demonstrated in Figures (4.72), (4.73) and (4.74), respectively.

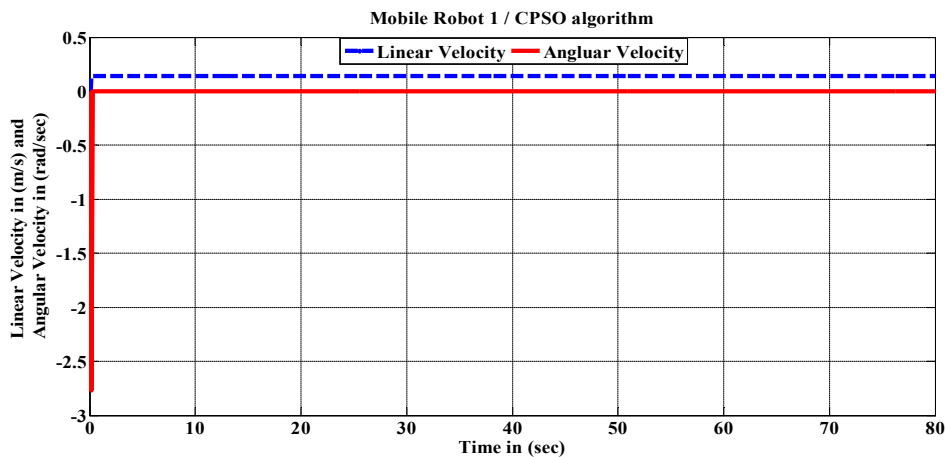


Figure (4.72): The platform angular and linear velocities for 1st robot / case C based on CPSO algorithm.

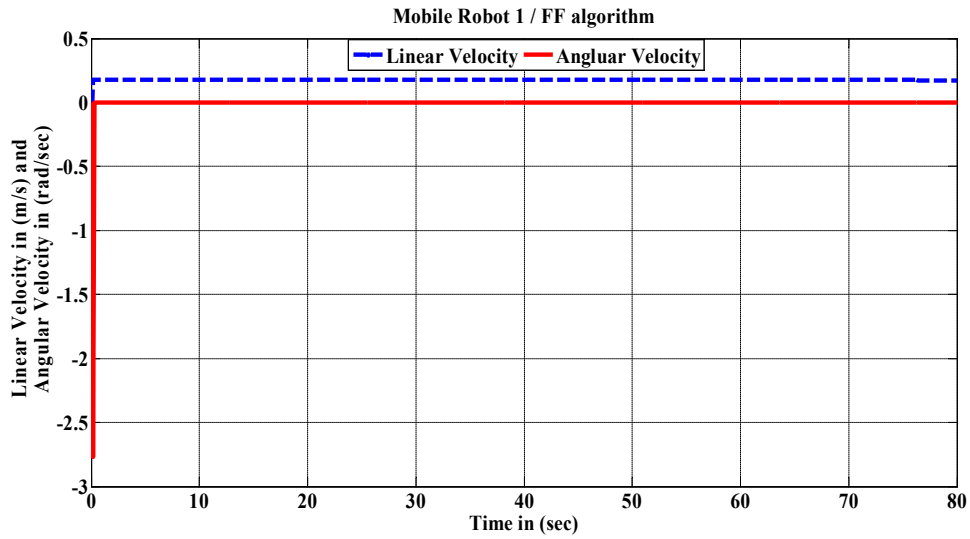


Figure (4.73): The platform angular and linear velocities for 1st robot / case C based on FF algorithm.

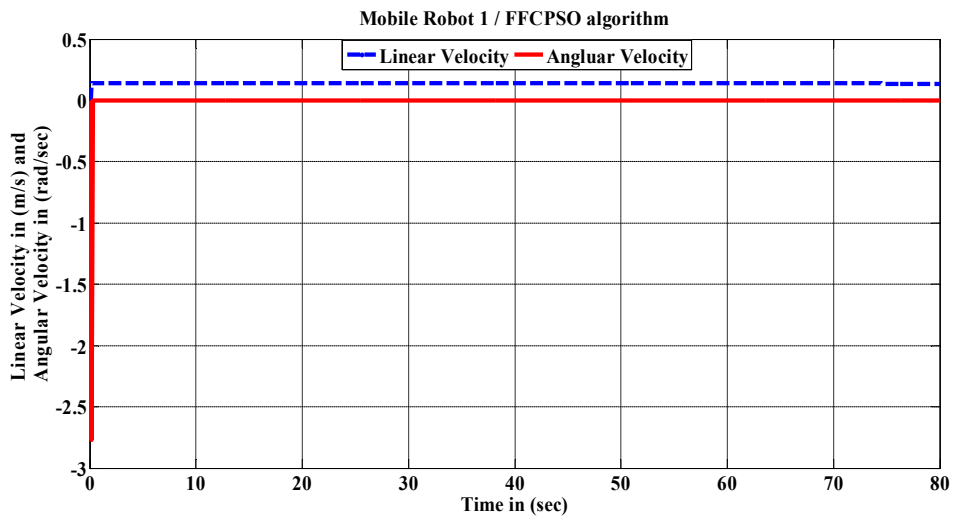


Figure (4.74): The platform angular and linear velocities for 1st robot / case C based on FFCPSO algorithm.

The angular and linear velocities of the platform based on the CPSO, FF and FFCPSO algorithms for WMR₂ is viewed in Figures (4.75), (4.76) and (4.77), respectively.

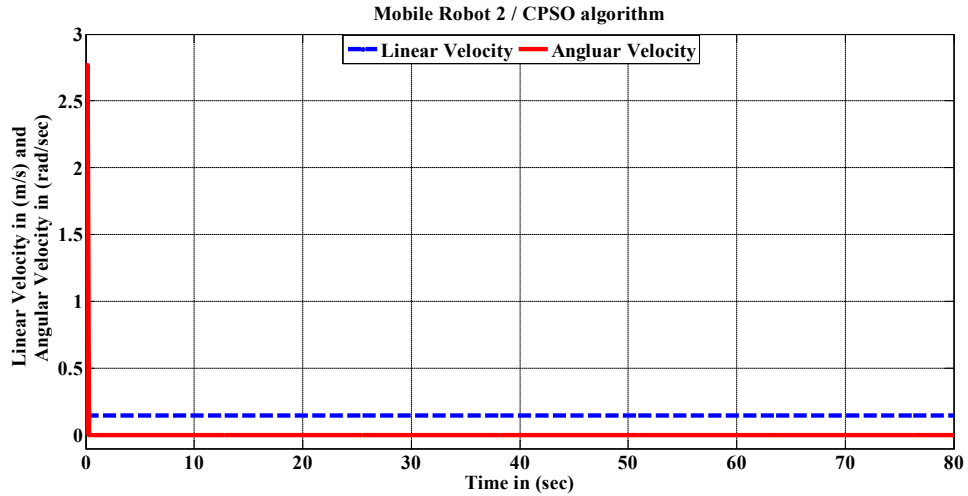


Figure (4.75): The platform angular and linear velocities for 2nd robot/ case C based on CPSO algorithm.

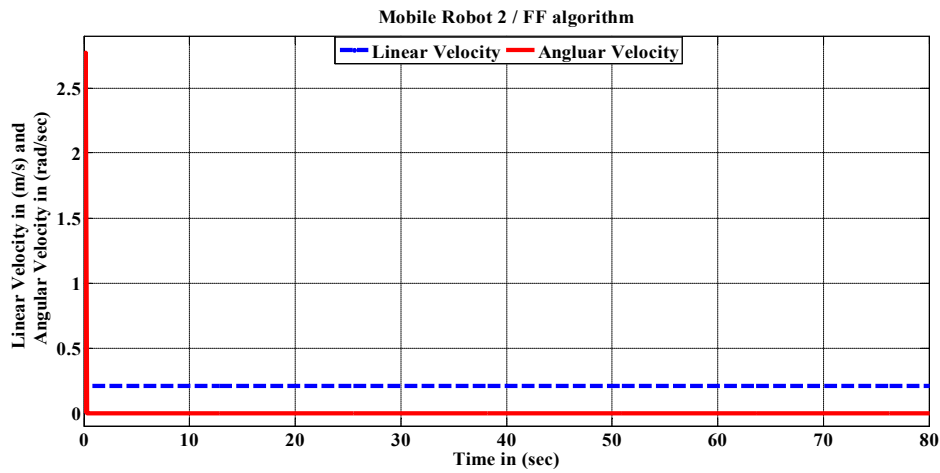


Figure (4.76): The platform angular and linear velocities for 2nd robot/ case C based on FF algorithm.

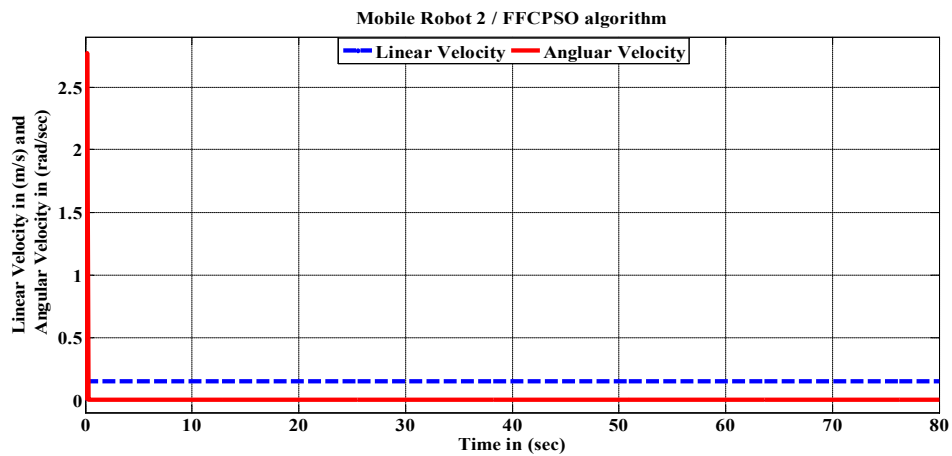


Figure (4.77): The platform angular and linear velocities for 2nd robot/ case C based on FFCPSO algorithm.

The angular and linear velocities of the platform based on the CPSO, FF and FFCPSO algorithms for WMR₃ is explained in Figures (4.78), (4.79) and (4.80), respectively.

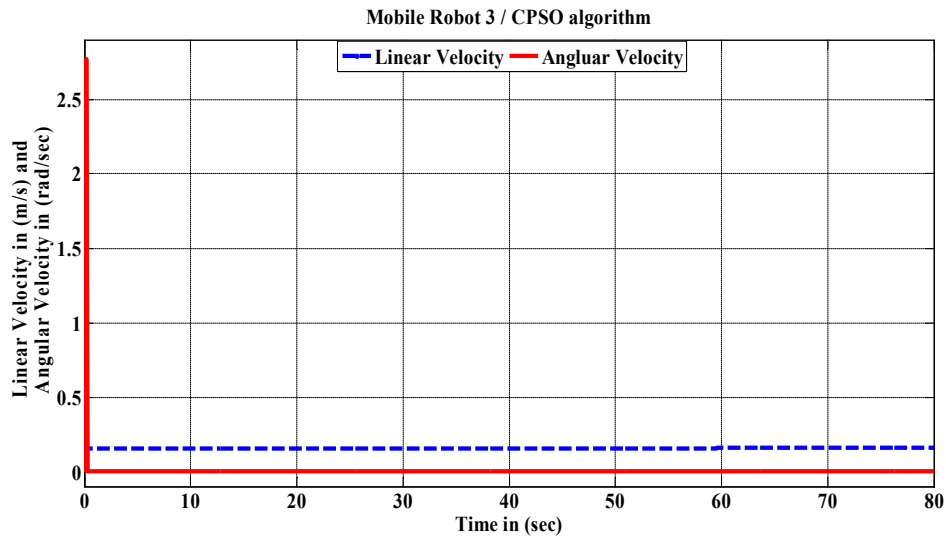


Figure (4.78): The platform angular and linear velocities for 3rd robot / case C based on CPSO algorithm.

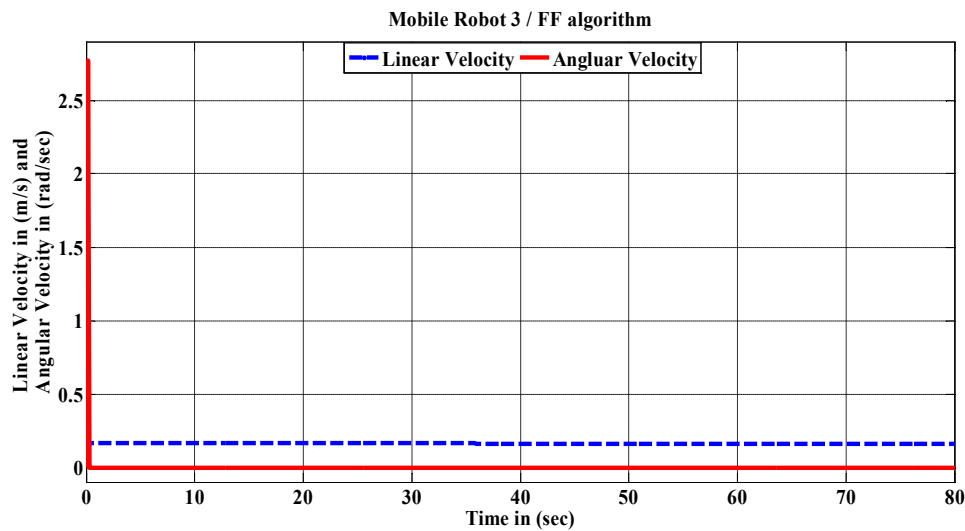


Figure (4.79): The platform angular and linear velocities for 3rd robot/ case C based on FF algorithm.

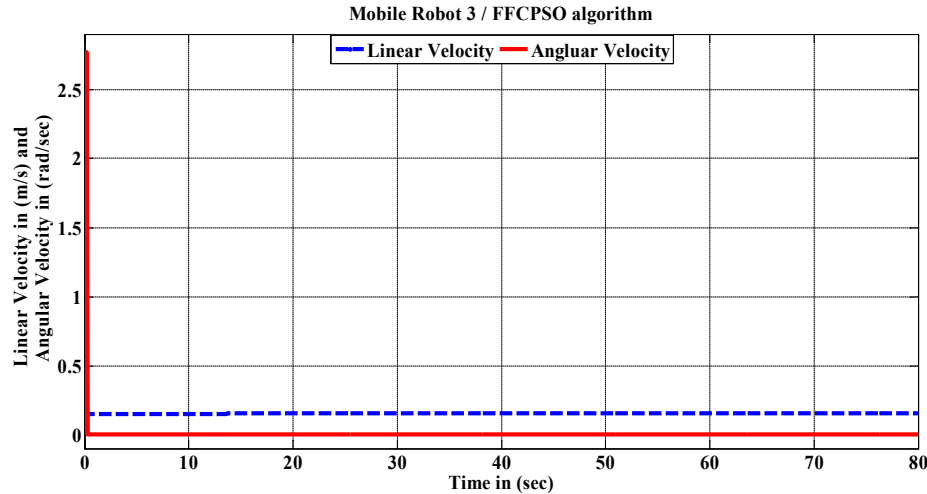


Figure (4.80): The platform angular and linear velocities for 3rd robot / case B based on FFCPSO algorithm.

From the all velocities figures, the angular velocity of the left and right wheels (WL & WR) should range between $(-6.67, +6.67)$ rad/sec. The linear velocity constrain of the left and right wheel (VL & VR) should range between $(-0.5, +0.5)$ m/sec. While, the angular velocity of platform (V_a) should range between $(-2.77, +2.77)$ rad/sec, and the linear velocity constrain of platform (V_n) should not exceed (0.5) m/sec. Clearly, these figures demonstrate the effectiveness of the optimization algorithms by showing its ability to produce smooth and small values of the angular and linear velocities of left and right wheels, this leads to a small power that is wanted by the mobile robot to move on its path.

4.4 Performance Evaluation

In this section, three research papers are utilized in order to evaluate the performance of the metaheuristic (population-based) algorithms and cubic splines interpolation with the same research's parameters setting.

In the first evaluation phase, a comparison is done between four optimization algorithms with the basic Artificial Bee Colony (ABC) and Directed Artificial Bee Colony (DABC), which were already presented in [7]. The four optimization

algorithms are applied on (10×10) m two-dimensional workspace with start point (0, 0) and target point (10, 10). Figures (4.81) offers the results of ABC and DABC algorithms. While, Figure (4.82) shows the results of optimization algorithms.

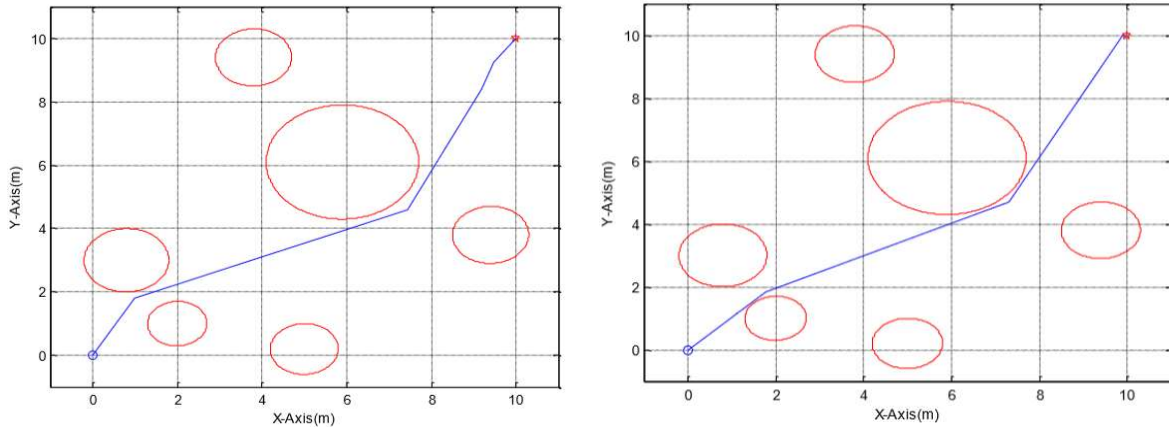


Figure (4.81 a-b): The best results achieved by [7] case study 2.

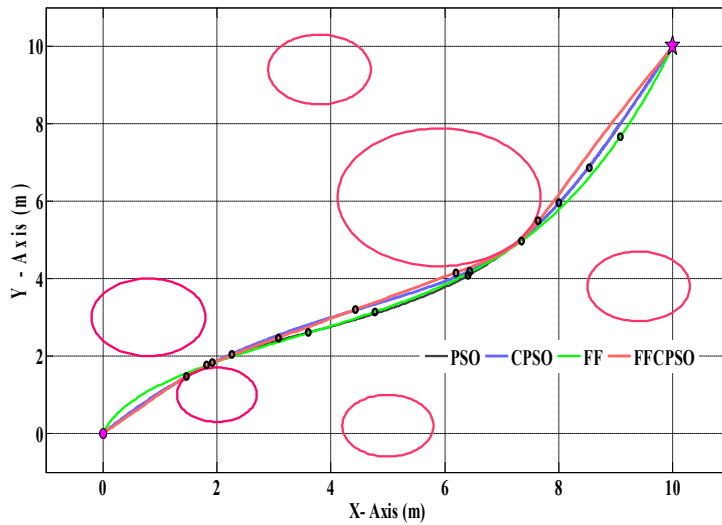


Figure (4.82): The best results of first comparison.

Table (4.19) summarizes the best path length, which can be achieved by various intelligent algorithms.

Table (4.19): Results Comparison with [7].

Algorithm	[7]		The proposed Algorithms			
	ABC	DABC	PSO	CPSO	FF	FFCPSO
Distance (m)	15.073	14.798	14.684	14.653	14.758	14.618

In the second evaluation phase, a comparison is done between the cubic polynomial interpolation and Bezier curve based on PSO-w, GA and FA algorithms, which were already presented in [21]. The algorithms are applied on (15×15) m 2D arena with start point $(5, 5)$ and target point $(15, 15)$. Figure (4.83) reveals the simulation results of [21], and Figure (4.84) show the results of the cubic polynomial interpolation for the two maps.

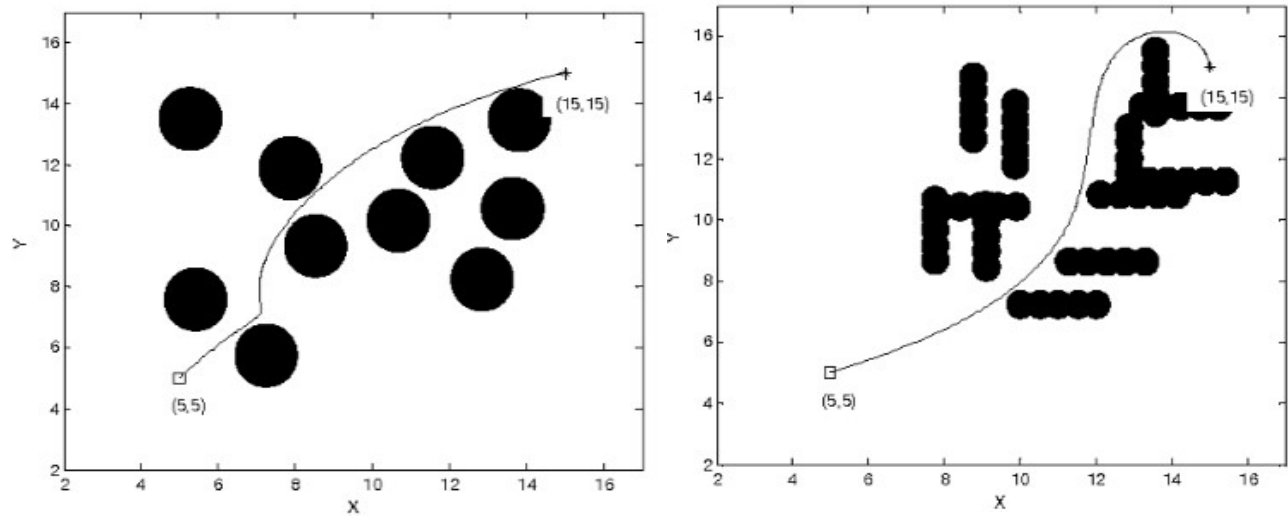


Figure (4.83 a-b): The best results achieved by [21].

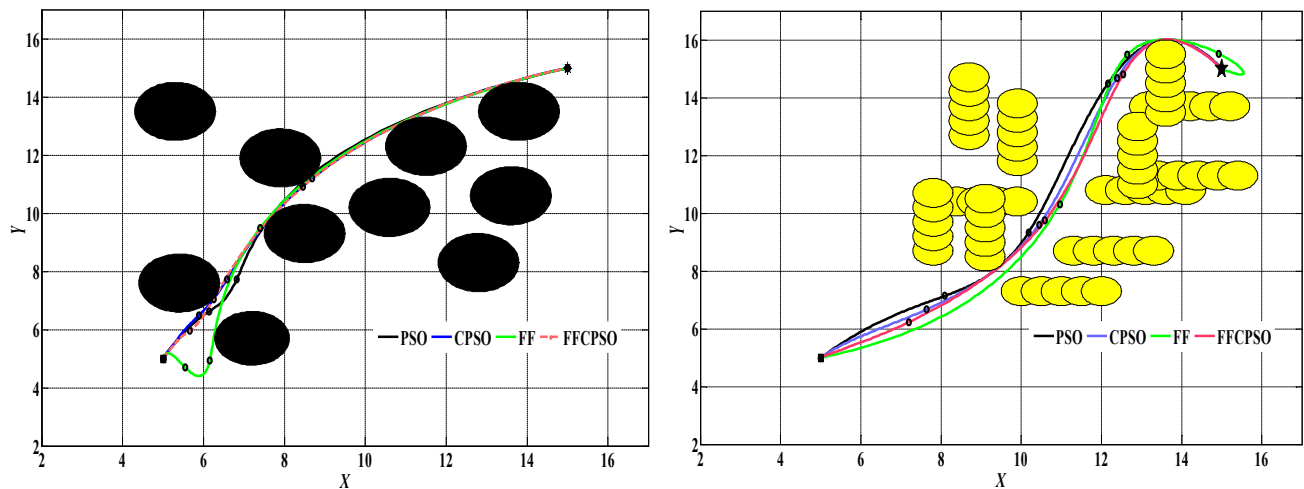


Figure (4.84 a-b): The best results of second comparison / Map a and Map c.

Table (4.20) summarizes the best path length for two maps, which can be achieved by the various intelligent algorithms.

Table (4.20): Results Comparison with [21].

Algorithm	[21]			The Proposed Algorithms			
	GA	PSO-w	FA	PSO	CPSO	FF	FFCPSO
Map a	17.3	17.13	17.44	14.795	14.698	16.377	14.714
Map c	18.23	17.76	18.31	16.385	16.353	16.744	16.349

While, in the third evaluation phase, a comparison is done between the cubic polynomial interpolation and Bezier curve based on AFA and CFA-OAS algorithms, which were already presented in [24]. The algorithms are applied on (10×10) m two-dimensional arena with the start point (1, 1) and target point (11, 17). Figure (4.85) shows the results of [25], and Figure (4.86) shows the results of cubic polynomial interpolation based on PSO and FF algorithms.

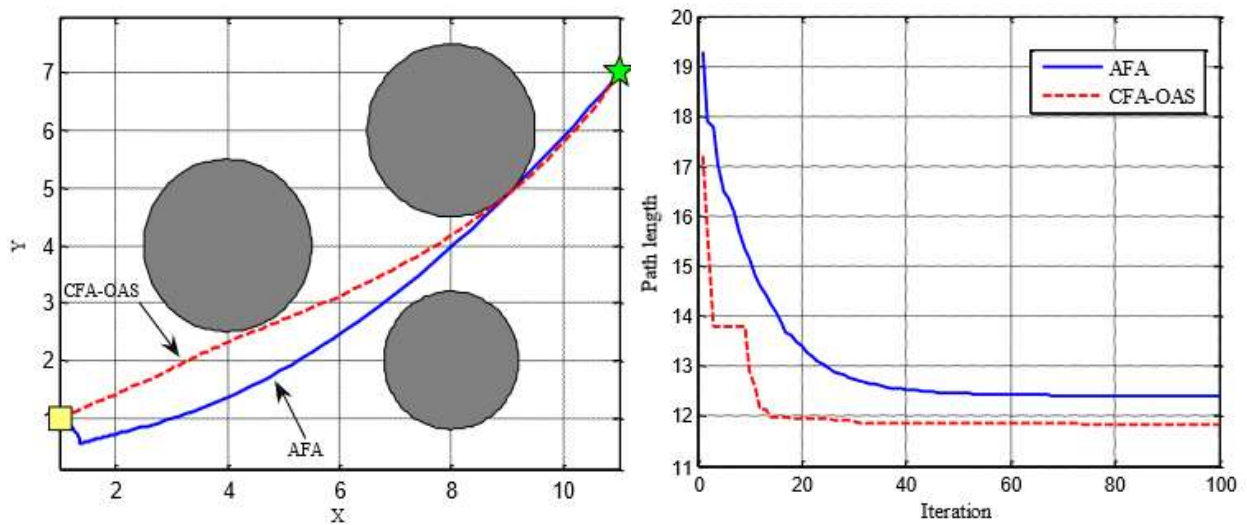


Figure (4.85a-b): The best results achieved by [24] case one.

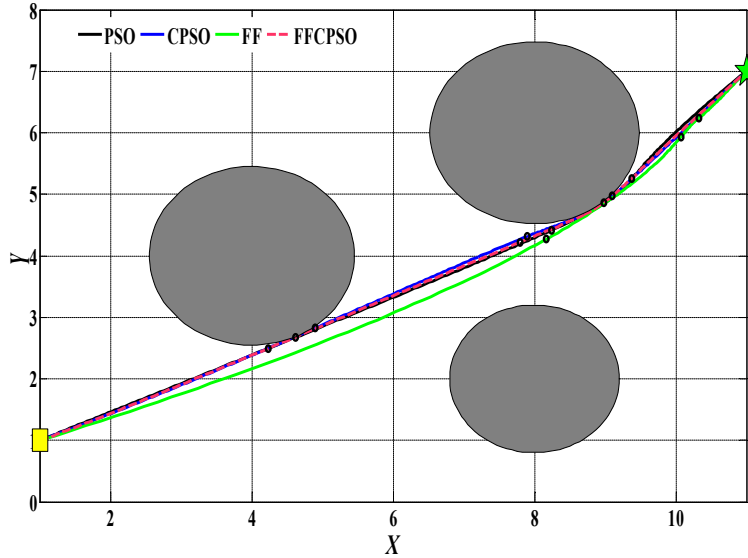


Figure (4.86): The best results of third comparison.

Table (4.21) summarizes the best path length that can be achieved by various intelligent algorithms.

Table (4.21): Results Comparison with [24].

Algorithm	[24]		The proposed Algorithms			
	AFA	CFA-OAS	PSO	CPSO	FF	FFCPSO
Distance (m)	12.41	11.85	11.82	11.814	11.85	11.813
Iteration No.	68	72	42	28	32	30

The above Comparisons analysis with previous papers between various intelligent algorithms are concluded as follows:

1. Compared with other previous techniques with the same research’s parameter setting, the features of presented algorithms stands out on path planning problems where cubic polynomial interpolation is used to generate the smooth path.
2. The results also tells that cubic polynomial interpolation is suitable on path planning problems for its stable feature when we employed population-based techniques in order to optimize its waypoints.

CHAPTER FIVE

CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

CHAPTER FIVE

CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

5.1 Conclusions

This thesis presents in details, a comparative study of various types of optimization algorithms including Chaotic Particle Swarm Optimization (CPSO) and proposed Hybrid Firefly Chaotic Particle Swarm Optimization (FFCPSO) algorithms with standard version of Particle Swarm Optimization (PSO) and Firefly (FF) algorithms that applied to single and multi-mobile robot path planning problem. The effectiveness of these algorithms was tested on different cases; the results achieved were verified through comparison of adaptive algorithms with each other and with the previous works. From the collected results and the comparison, the following can be concluded:

1. The simulation results showed the validity of the kinematic model of the non-holonomic wheeled mobile robot (NWMR).
2. Both PSO and FF algorithms are powerful methods for their efficiency. PSO is simple to implement as compared to FF due to its less number of variables. FF has more variables and most of them are random. In addition, these parameters have been changed to obtain an optimal, safe, smooth and guaranteed path.
3. The effective minimization capability of specific paths for NI-mobile robots model before (80) iterations for presented techniques is based on three cases according to three objectives: shortest path to target, safety and smoothness to follow a desired continuous path.

4. Development of the standard version of PSO with Chaotic optimization in order to form Chaotic PSO algorithm has a better ability in order to get away from local minima with shortest path and less number of iteration than basic PSO algorithm.
5. PSO keeps a memory of its earlier iteration by storing the values of personal and global best, so it can be concluded that there is a balance between the local and global minima. FF does not memorize or remember any history of better situation for each firefly and this makes them to move regardless of its previous better situation, and they may end up missing their situations. Therefore, the hybrid FFCPSO is proposed firstly for this purpose and then to reduce the FF randomization.
6. The four intelligent algorithms have an effectiveness and good performance by finding feasible, shortest and smoothness without colliding any obstacles in the environment in order to solve the problem of robot path planning. Therefore, these algorithms could be suitable for multi robot systems to find the shorted path length and avoid the collision between them.
7. From Tables (4.9), (4.12) and (4.16) show, Chaotic PSO algorithm is a perfect optimization algorithm due to its effectiveness to provide best path with minimum time and less number of iteration.
8. Figure (4.47) offers that CPSO algorithm is a very good method in the case of follow up mobile robot than other presented methods.
9. The simulation results explained the effectiveness of the cubic polynomial interpolation based on suggested techniques by displaying its ability to produce very good smooth values of the velocities for left and right wheels and mobile robots velocities without exceeding the limited values (less than 0.5 m/sec for linear velocity).

5.2 Suggestions for Future Work

Many issues interested in solving the mobile robot path planning problems. Some of them, which might entice a researcher to work with and can be recommended as future works, are indicated as follows:

1. Using other types of intelligent algorithms to find the optimal desired path, such as Cat Swarm algorithm (CSA), Fruit fly algorithm (FA), etc.
2. Applying the enhanced algorithms to real wheeled NI- mobile robot can be a good challenge because of nonlinear factors such as noise, which the optimization algorithm has to be able to deal with these factors. Then, making a comparison between theoretical part and practical part.
3. Future research can investigate the performance of the suggested algorithms in dynamic unknown environments and use fuzzy rules as a decision maker in order to prevent a collision with dynamic obstacles or other mobile robot.
4. Workout other types of environment like maze-type, and enhancements in terms of obstacles like including the different shapes of obstacles. The different shapes, like square, upward U, inverted U, upward V, and inverted V can be included in the environment.
5. Another future direction is to examine the effectiveness of the suggested approaches by solving the obstacle collisions limited to three dimensions (3D) environment.

REFERENCES

REFERENCES

1. J. Lu, G. Zhang, D. Ruan and F. Wu, “**Multi-Objective Group Decision Making,**” World Scientific Publishing Co. Pte. Ltd., 2007.
2. F. C. Lunenburg, “**Decision Making Process,**” National Forum of Educational Administration and Supervision Journal, Vol. 27, No. 4, 2010
3. K. M. Passino and S. Yourkovich, “**Fuzzy Control,**” Addison Wesley Longman, 1998.
4. M. Xie, “**Fundamentals of Robotics,**” Linking Perception to Action, Singapore: World Scientific Publishing Co. Pte. Ltd., 2003.
5. S. G. Tzafestas, “**Introduction to Mobile Robot Control,**” 1st Edition, Elsevier, 2014.
6. N. H. Abbas and J. A. Abdulsahab, “**An Adaptive Multi- Objective Particle Swarm Optimization Algorithm for Multi – Robot Path Planning ,**” Journal of Engineering, Vol. 22, No. 7, pp.164-180, 2016.
7. N. H. Abbas and F. M. Ali, “**Path Planning of an Autonomous Mobile Robot using Directed Artificial Bee Colony Algorithm,**” International Journal of Computer Applications, Vol. 96, No. 11, pp.11-16, 2014.
8. M. S. Alam, M. U. Rafique and M. U. Khan, “**Mobile Robot Path Planning in Static Environments using Particle Swarm Optimization,**” International Journal of Computer Science and Electronics Engineering, Vol. 3, pp. 253-257, 2015.
9. E. Masehian, and D. Sedighzadeh, “**Classic and Heuristic Approaches in Robot Motion Planning – A Chronological Review,**” World Academy of Science, Engineering and Technology, Vol. 5, No. 29, pp. 101-106, 2007.
10. S. Sedhumadhavan and E. Niranjana, “**An Analysis of Path Planning for Autonomous Motorized Robots,**” International Journal of Advance Research, Ideas and Innovations in Technology, Vol. 3, pp. 1234-1257, 2017.
11. P. Benavidez and M. Jamshidi, “**Mobile Robot Navigation and Target Tracking System,**” International Conference on system of systems Engineering, New Mexico, USA, 2011.
12. M. W. Abbas, “**Path Planning of Mobile Robots using Genetic Algorithms and Modified Artificial Potential Field,**” Master Thesis, Control and Systems Engineering Dept., University of Technology, Baghdad, Iraq, 2012.

REFERENCES

13. H. Ahmed and J. Glasgow, “**Swarm Intelligence: Concepts, Models and Applications,**” Technical Report, School of Computing, University of Queen, Ontario, Canada, 2012.
14. G. Beni and J. Wang, “**Swarm intelligence in cellular robotic systems,**” In NATO Advanced Workshop on Robots and Biological Systems, IL Ciocco, Tuscany, Italy, 1989.
15. J. Kennedy and R. C. Eberhart, “**Particle Swarm Optimization,**” In Proceedings of IEEE International Conference on Neural Networks, NJ, USA, pp. 1942–1948, 1995.
16. R. C. Eberhart and J. Kennedy, ” **A new optimizer using particle swarm theory,**” In Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43, 1995.
17. X. Yang, “**Firefly algorithms for multimodal optimization,**” in: Stochastic Algorithms: Foundations and Applications, Lecture Notes in Computer Sciences, Vol. 5792, pp. 169-178, 2009.
18. C. Liu, Z. Gao and W. Zhao, “**A New Path Planning Method Based on Firefly Algorithm,**” In Proceedings of the IEEE International Conference on Computational Science and Optimization, pp. 775-778, 2012.
19. C. Purcaru, R. -E. Precup, D. Iercan, L.O. Fedorovici, and R.C. David ,“**Hybrid PSO-GSA Robot Path Planning Algorithm in Static Environments with Danger Zones**”, In Proceedings of the IEEE International Conference on System Theory, Control and Computing, Sinaia, Romania, pp. 434-439, 2013.
20. E. Masehian and D. Sedighzadeh, “**An Improved Particle Swarm Optimization Method for Motion Planning of Multiple Robots,**” Springer Tracts in Advanced Robotics, Vol. 83, pp. 175-188, 2013.
21. B. Li, L. Liu, Q. Zhang, D. Lv, Y. Zhang, J. Zhang and X. Shi, “ **Path Planning Based on Firefly algorithm and Bezier curve,**” In Proceedings of the IEEE International Conference on Information and Automation, pp. 630-633, 2014.
22. M. R. Panda, R. Priyadarshini and S. K. Pradhan, “**Autonomous Mobile Robot Path Planning Using Hybridization of Particle Swarm Optimization and Tabu search,**” In Proceedings of the IEEE International Conference on Computational Intelligence and Computing Research, 2016.

REFERENCES

23. E. Cholodowicz and D. Figuirowski, “**Mobile Robot Path Planning with Obstacle Avoidance using Particle Swarm Optimization,**” *Pomiary Automatyka Robotyka*, Vol. 21, No. 3, pp. 59-68, 2017.
24. D. Pang, G. Guan and J. Li, “**Chaotic Firefly Algorithm with the Optimization Adjustment Strategy for Mobile Robot Path Planning,**” *International Journal of Science*, Vol. 4, No. 3, 2017.
25. A. Tharwart, M. Elhoseny, A. E. Hassanien, T. Gabel and A. Kumar, “**Intelligent Bezier curve-based Path Planning model using Chaotic Particle Swarm Optimization Algorithm,**” Springer, *Journal of Cluster Computing*, pp. 1-22, 2018.
26. O. Mohareri, “**Mobile Robot Trajectory Tracking using Neural Networks,**” Master Thesis, Dep. of Electrical Engineering ,University of Sharjah, Sharjah, American, 2009.
27. A. S. Al-Araji, “**Development of Kinematic Path-Tracking Controller Design for Real Mobile Robot via Back-Stepping Slice Genetic Robust Algorithm Technique,**” *Arabian Journal for Science and Engineering*, Vol. 39, No.12, pp. 8825–8835, 2014.
28. D. Hanafi, Y. M. Abueejela and M. F. Zakaria, “**Wall Follower Autonomous Robot Development Applying Fuzzy Incremental Controller,**” *Intelligent Control and Automation*, Vol. 4, pp. 18-25, 2013.
29. K. M. Han, “**Collision Free Path Planning Algorithms for Robot Navigation Problem,**” Master Thesis, University of Missouri, Columbia, 2007.
30. A. M. Eliwa, “**Mobile Robot Path Planning Using Genetic Algorithm Global Path Planning And Potential Field Path Adjusting,**” Master Thesis, University of Dalhousie, Halifax, Nova Scotia, 2017.
31. H. Miao, “**Robot Path Planning in Dynamic Environments using Simulated Annealing Based Approach,**” Master Thesis, University of Technology, Queensland, Australia, 2009.
32. S. A. Mnubi, “**Motion Planning and Trajectory for Wheeled Mobile Robot,**” *International Journal of Science and Research*, Vol. 5, pp. 1064-1068, 2016.
33. F. M. Ali, “**Improvement of Path Planning for Autonomous Mobile Robots Using Population-Based Optimization Algorithms,**” Master Thesis, University of Baghdad, Baghdad, Iraq, 2014.

REFERENCES

34. L. Sudraba, A. Nikitenko, “**Application of Mapping methods for Solving Navigation Tasks of Autonomous Intelligent System,**” International Standard Serial Number of Computer Science, pp. 67-79, 2008.
35. A. Astolfi, “**Optimization: An Introduction,**” 2006.
36. A. Pandey, “**Mobile Robot Navigation and Obstacle Avoidance Techniques: A Review,**” International Robotics and Automation Journal, Vol. 2, pp. 1-12, 2017.
37. R. Islam, Tajmiruzzaman, M. H. Muftee and S. Hossain, “**Autonomous Robot Path Planning Using Particle Swarm Optimization in Dynamic Environment with Mobile Obstacles & Multiple Target,**” International Conference on Mechanical, Industrial and Energy Engineering, 2014.
38. A. Cosic, M. Susic and D. Katic, “**Advanced Algorithms for Mobile Robot Motion Planning and Tracking in structured Static Environments Using Particle Swarm Optimization,**” Serbian Journal of Electrical Engineering, Vol. 9, No. 1, 2012.
39. D. Wang, D. Tan and L. Liu, “**Particle Swarm Optimization Algorithm: an overview,**” Springer-Verlag Berlin Heidelberg, Vol. 22, pp. 387–408, 2018.
40. Y. Shi and R. C. Eberhart, “**A Modified Particle Swarm Optimizer,**” Proceedings of the IEEE International Conference on Evolutionary Computation, Piscataway, NJ, IEEE Press, pp. 69-73, 1998.
41. M. Shamshiri, C. K. Gan, K. Jusoff, I. J. Hasan, M. R. Abghani and M. Yusoff, “**Using Particle Swarm Optimization Algorithm in the Distribution Systems Planning,**” Australian Journal of Basic and Applied Sciences, Vol. 7, pp. 85-92, 2013.
42. A. Kundur, “**Evolution of Firefly Algorithm using Benchmark Functions,**” Master Thesis, North Dakota State University of Agriculture and Applied Science, Fargo, North Dakota, 2013.
43. X. S. Yang, “**Nature-Inspired Metaheuristic Algorithms,**” 2nd Edition, Luniver Press, 2010.
44. T. Hu, “**The Comparative Analysis and Prospect of Two Heuristic Algorithms: The Firefly Algorithm and the Basic Ant Colony Algorithm,**” International Journal of Business and Social Science, Vol. 5, No. 7, pp. 257-260, 2014.
45. K. R. Subhashini and A. T. PraveenKumar, “**Comparative Analysis of Linear and Nonlinear Pattern Synthesis of Hemispherical Antenna Array Using Adaptive Evolutionary Techniques,**” Hindawi Publishing Corporation, International Journal of Antennas and Propagation, Article ID 987140, pp. 1-10, 2014.

REFERENCES

46. S. R. Chang and U. Y. Huh, “**Continuity Smooth Path Planning Using Cubic Polynomial Interpolation with Membership Function,**” International Journal of Electrical Engineering, Vol. 10, No. 2, pp. 742-753, 2015.
47. A. N. Hussain , F. Melk , M. A. Rashid , L. Moamed , and N. A. Mohd Affendi, ” **Optimal Coordinated Design of Multiple Damping Controllers Based on PSS and UPFC Device to Improve Dynamic Stability in the Power System,**” Hindawi Publishing Corporation Mathematical Problem in Engineering, Article ID 96528, pp1-15, 2013.
48. Text manual from NI Company. <http://sine.ni.com/nips/cds/view/p/lang/en/> (2015). Accessed March 2015.

APPENDIX

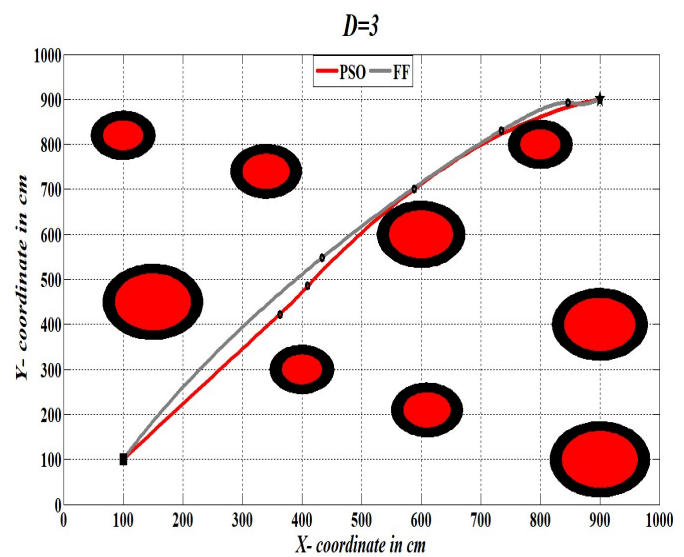
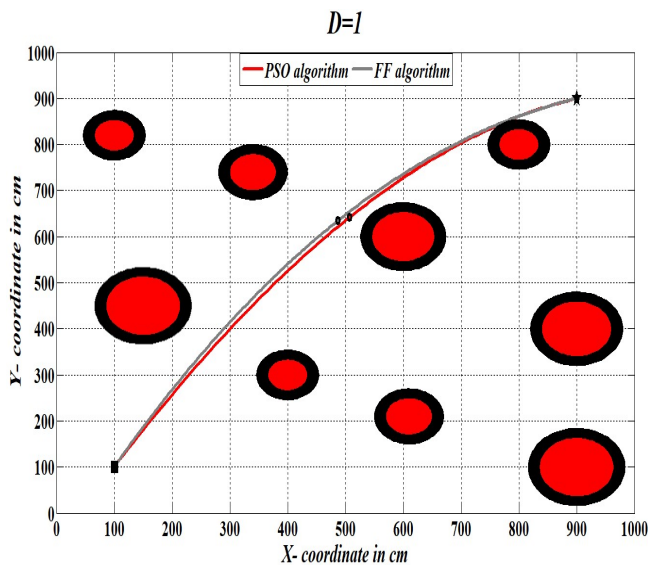
Appendix A

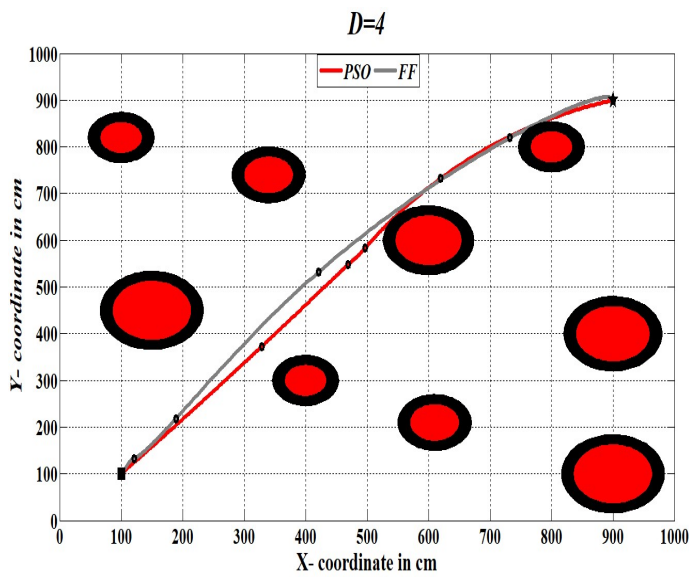
Parameters Setting

Parameters setting for any evolutionary algorithm is important as designing the algorithm itself. In the following sections, the effect of the control points (waypoints), number of individuals and the number of iterations on the results were investigated as illustrated below.

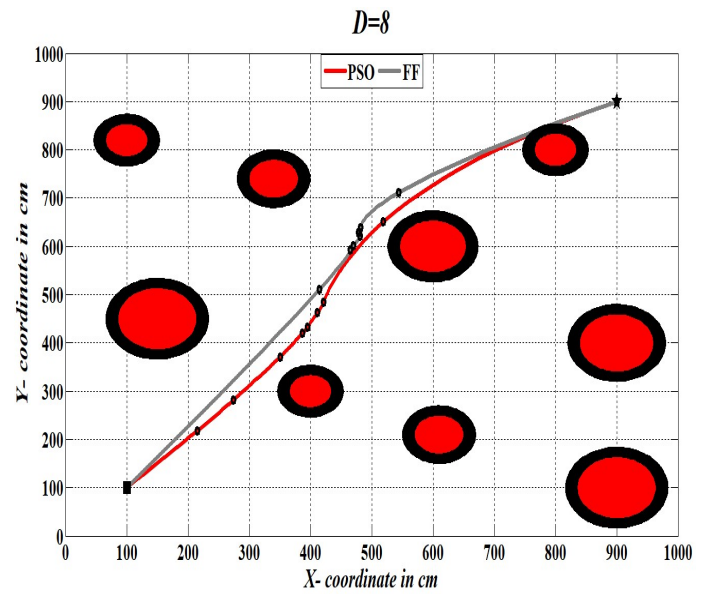
- **Control Points Setting**

In this simulation, the number of control points (waypoints) are randomly selected from 1 to 16 control points based on basic PSO and basic FF algorithms. The size of population is 20, and the maximum number of iteration is 80. Figures (A.1), (A.2) and (A.3) show the results of this testing.

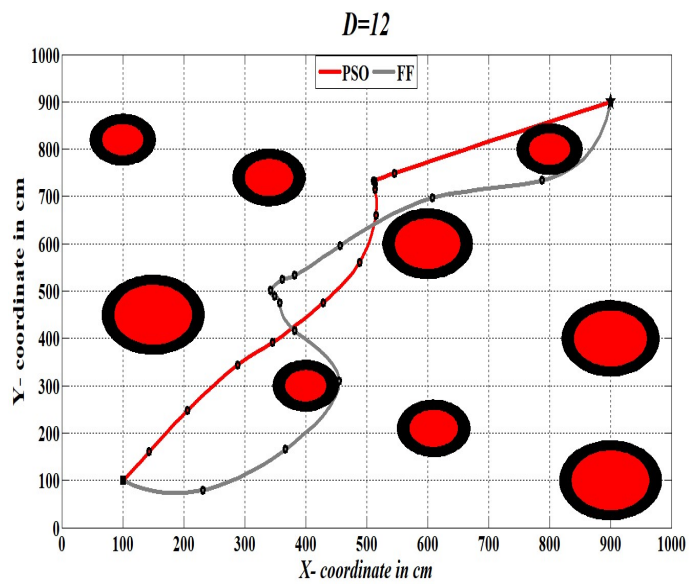




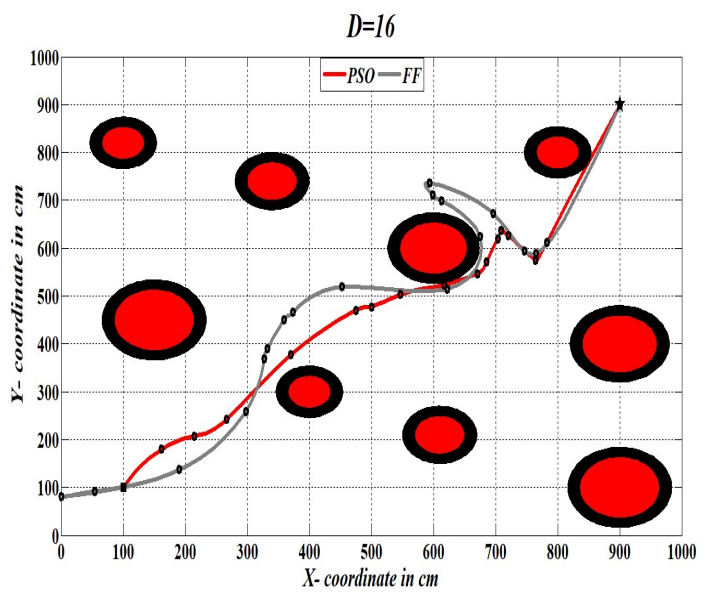
(C)



(D)



(E)



(F)

Figure (A.1): Simulation of one run for the basic PSO algorithm with different numbers of waypoints.

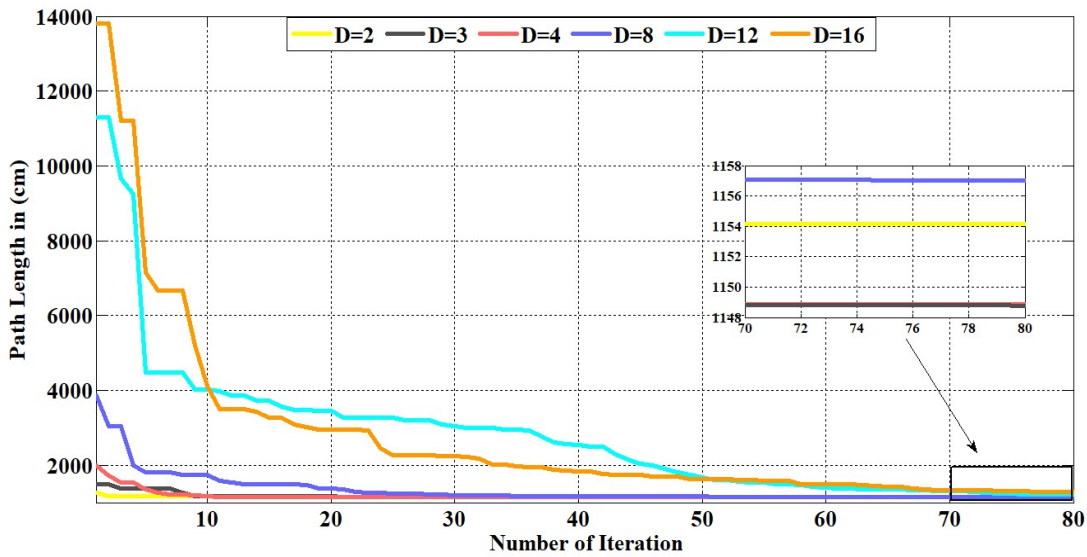


Figure (A.2): The performance of the basic PSO algorithm with different numbers of waypoints.

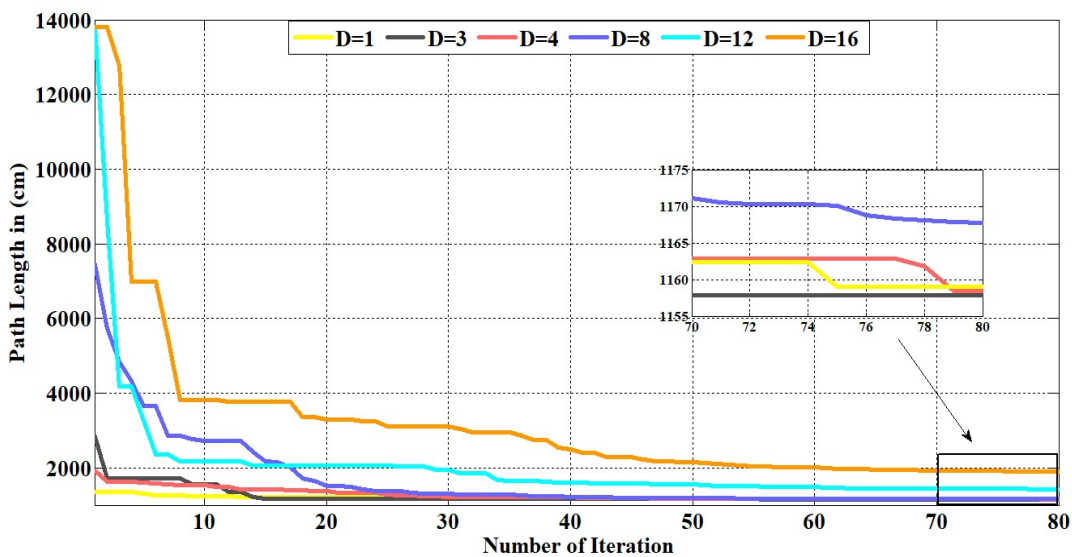


Figure (A.3): The performance of the basic FF algorithm with different numbers of waypoints.

From the above figures, the best results achieved when the number of control points (waypoints) was small such as (3 or 4). Obviously, the distance increased when the value of (D) was more than eight, and the worst results achieved when (D=16) because of making the path lose its smoothness.

- **Population Size Setting**

Population size is an important parameter for the quality of solution and the convergence of the population-based algorithms. In this test, the effect of the population size on the results was investigated when the number of particles ranged from 2 to 30 particles. The cost value of PSO and FF techniques after ten runs are shown in Figure (A.4).

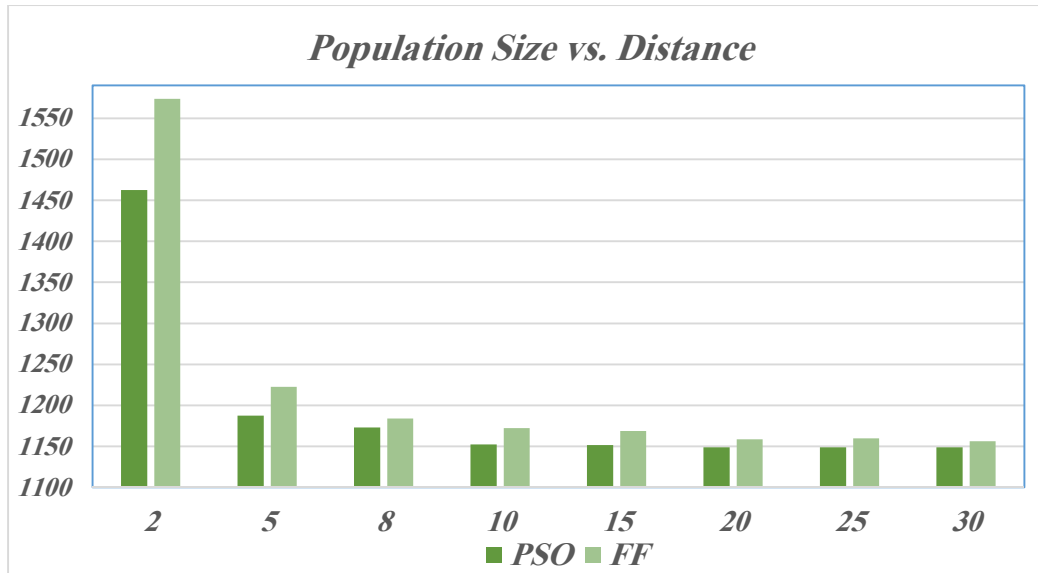


Figure (A.4): Comparison in the distance through population numbers based on PSO & FF algorithms.

From this figure, the population size makes the path go toward the optimal value if it increases. That is because the PSO and FF algorithms will take better chance to find more odds of feasible path. On the other hand, increasing the number of population leads to increase the execution time.

- **Number of Iteration Setting**

The number of iterations also has a great impact on the performance of two Nature-Inspired algorithms based path planning problem. In this test, the effect of the number of iterations on the cost function was tested when the number of iterations was ranged from 10 to 100. The cost value of the PSO and FF algorithms shown in Figure (A.5).

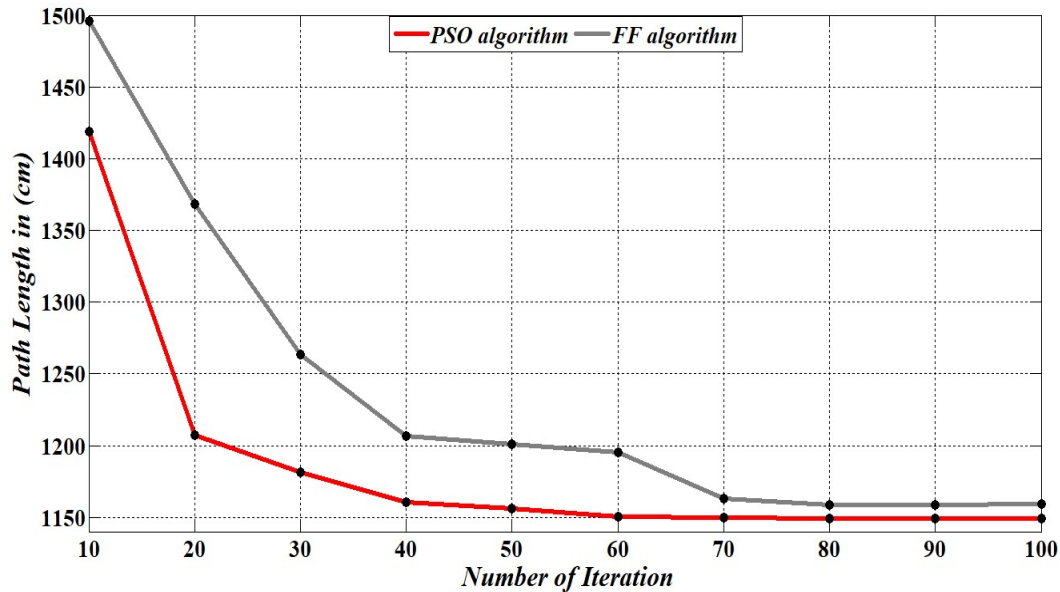


Figure (A.5): Effect of the number of iterations on the performance of PSO and FF algorithms.

The above trials and comparison analysis between the PSO and FF techniques are concluded as follows:

1. The control points significantly influence the length and smoothness of the path.
2. In PSO and FF, a population and variables is generated randomly. However, FF is very much random in nature, while PSO is not random in nature.
3. In PSO and FF, the large size of population is not considered. If the large size is considered, it does not improve the quality of solution but increases the computational time. Therefore, the pop size is kept small around 15-20. Therefore, it can be concluded that both algorithms do not require a large size of population.
4. In FF, the wrong selection of ($\tilde{\alpha}$) can cause a small or big step increment and take away the solution in some other side far away from the global best. In addition, FF parameters are set fixed and they do not change with the time. In PSO, most of parameters are updated during the iteration process.
5. Regardless of the number of obstacles, increasing the number of iterations does not lead to increase the chance to find the optimal path that has the minimum distance. On the contrary, the maximum number of iterations that range from 60 to 80 was enough to reach the optimal path.

LIST OF PUBLICATIONS

[1] Muna M. Jawad, Khulood E. Dagher and Esraa A. Hadi, “**A Comparative Study for Wheeled Mobile Robot Path Planning Based on Modified Intelligent Algorithms,**” Accepted in Iraqi Journal for Mechanical and Material Engineering, University of Babylon, Iraq - Babylon.

[2] Muna M. Jawad and Esraa A. Hadi, “**A Comparative Study of Various Intelligent Algorithms based Path Planning for Mobile Robots,**” Accepted in Journal of Engineering, University of Baghdad, Iraq - Baghdad.

الملخص

اتخاذ القرار يشمل جمع المعلومات واستخراج البيانات والتصميم والتحليل. وقد تؤدي القرارات الغير الصحيحة الى فشل المنظمة لذلك فإن العملية المتبعة يجب أن تكون صحيحة. مشكلة تخطيط المسار الأمثل هي واحدة من تطبيقات اتخاذ القرار في مجال الروبوتات، والغرض منه هو العثور على أقصر مسار من نقطة البدء ولغاية نقطة الهدف.

يقدم هذا البحث أربع خوارزميات لايجاد مسار متعدد الاهداف لروبوت واحد و عدد الروبوتات المتنقلة. تعتمد الخوارزميتان الاولى والثانية على خوارزمية اسراب الطيور الاساسية وعلى هي خوارزمية اسراب الطيور المطورة مع خريطة الفوضى لتشكيل خوارزمية اسراب الطيور المشوشة من اجل منع الانزلاق الى الحدود الدنيا المحلية. تعتمد الخوارزميتين الاخرتان على النسخة الأساسية من خوارزميات اليراعات المضيفة وهي تقينه حديثة وقوية ولكنها تعاني من المشكلة المثلى المحلية والخوارزمية المقترحة التي تعتمد على النسخة الأساسية لخوارزمية اليراعات المضيفة مع خوارزمية اسراب الطيور المحسنة لتشكيل تقنية هجينة تسمى خوارزمية اليراعات المضيفة – اسراب الطيور المشوشة لتخطيط المسار العالمي. يستخدم المنحى المكعب من اجل انشاء مسار سلس عن طريق الاستيفاء من خوارزميات التحسين ويتم تقييم دالة الهدف من خلال قيدين وهما طول المسار وتجنب العوائق. علاوة على ذلك، على أساس نموذج حركي لروبوتات متنقلة ذات عجلات تفاضلية يتم حساب السرعة الخطية والدورانية للعربة والسرعة الخطية والدورانية للعجلتين اليمنى واليسرى لتوجيه عجلات اليه متنقلة لمتابعة المسار المطلوب للوصول الى الهدف المحدد مسبقا.

الخوارزميات تم محاكاتها باستخدام لغة البرمجة (الماتلاب) وان نتائج المحاكاة أظهرت ان خوارزمية اسراب الطيور المشوشة هي أفضل من خوارزمية اسراب الطيور الاصلية من حيث حصولها على أقصر مسافة بعدد اقل من التكرارات وان خوارزمية اليراعات المضيفة الهجينة مع اسراب الطيور المشوشة هي أفضل من خوارزمية اليراعات المضيفة الاصلية وهذا أدى في النهاية الى حصول الموبايل روبوت ذات العجلات المتنقلة على سرعة خطية سلسلة للعجلة اليمنى والعجلة اليسرى بحيث لا تتجاوز ال 0.5 متر/ثانية. بالإضافة الى ذلك تم مقارنة هذه الخوارزميات مع أوراق بحثية أخرى لتقييم أدائها. من خلال مقارنة النتائج التي حققتها الخوارزميات المقترحة مع النتائج التي حققتها الأعمال السابقة تحت نفس الظروف نستنتج ان الاستيفاء متعدد الحدود التكعيبي هو ميزة جيدة عن طريق توليد مسار سلس دون حواف حادة خلال عملية التعلم لذلك يمكن للروبوت المحمول التحرك بسلاسة و سلامة.



جمهورية العراق
وزارة التعليم العالي والبحث العلمي
الجامعة التكنولوجية
قسم هندسة السيطرة والنظم

أخذ قرار متعدد الأهداف لتخطيط المسار لروبوت واحد و روبوتات متعددة

رسالة مقدمة الى قسم هندسة السيطرة والنظم الجامعة التكنولوجية كأحد المتطلبات لنيل
شهادة الماجستير في علوم هندسة الحاسبات.

من قبل

أسراء عدنان هادي

بإشراف

الدكتورة منى محمد جواد