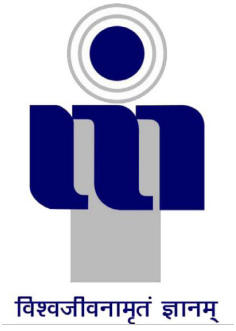
MVO-BASED PATH PLANNING SCHEME WITH COORDINATION OF UAVs IN 3-D ENVIRONMENT

A project report submitted in partial fulfillment of the requirements for B.Tech. Project

B.Tech.

by

Dhruv Prakash (2015IPG-024)
Gatij Jain (2015IPG-029)
Gaurav Yadav (2015IPG-031)



ABV INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT
GWALIOR-474 015

2018

CANDIDATES DECLARATION

We hereby certify that the work, which is being presented here in this report, entitled **MVO-Based path planning scheme with coordination of UAVs in 3-D Environment**, in partial fulfillment of the requirement for the award of the Degree of **Bachelor of Technology** and submitted to the institution is an authentic record of our own work carried out during the period *May 2018* to *September 2018* under the supervision of **Prof. Anupam Shukla** and **Dr. Ritu Tiwari**. We also cited the reference about the text(s)/figure(s)/table(s) from where they have been taken.

Date: Signatures of the Candidates

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date: Signatures of the Research Supervisors

ABSTRACT

The path planning of UAV deals with the process of figuring the most optimal path from the source to the destination and avoiding obstacles on course to avoid collisions in a given environment. This particular problem is classified as an optimisation problem in high dimension and comes under the category of NP-Hard. Due to the usage of UAVs in many domains like the army, communications, security, disaster management, survey mapping etc., the provisioning of time efficiency as well as accuracy is a critical and challenging task in a 3-D environment. Moreover, the ability of the UAVs can be enhanced if there is coordination between them to find the best cost-effective target in a dynamic manner. However, a majority of the solutions reported theoretically are not efficient concerning time efficiency,accuracy and coordination simultaneously for many applications. Drawing motivation from this, the paper gives a novel way of solving the problem of planning of path for UAVs in 3-dimension while maintaining the coordination for choosing the target. A new algorithm based on Multiverse optimizer (MVO) algorithm is applied to the problem. Munkres algorithm is applied for coordination of UAVs in case of multiple targets. By carrying out the simulations, on three different maps over 50 iterations the results shows that MVO algorithm perform better in the most of the cases for finding optimized path cost with minimum average execution time when compared to meta-heuristics GSO and BBO.

Keywords: Path Planning, Meta-heuristics, Multiverse Optimizer, Swarm intelligence, UAV, Robotics.

ACKNOWLEDGEMENTS

We are highly indebted to **Prof. Anupam Shukla** and **Dr. Ritu Tiwari** and are obliged for giving us the autonomy of functioning and experimenting with ideas. We would like to take this opportunity to express our profound gratitude to them not only for the academic guidance they provided but also for the keen interest they showed in our project. Their constant support coupled with confidence boosting and motivating sessions proved very fruitful and were instrumental in infusing self-assurance and trust within us. The nurturing and blossoming of the present work is mainly due to their valuable guidance, suggestions, astute judgment, constructive criticism and an eye for perfection. Our mentors always answered myriad of our doubts with smiling graciousness and prodigious patience, never letting us feel that we are novices by always lending an ear to our views, appreciating and improving them and by giving us a free hand in our project. It's only because of their overwhelming interest and helpful attitude, the present work has attained the stage it has.

Finally, we are grateful to our Director **Prof. S.G. Deshmukh**, the Institution and colleagues whose constant encouragement served to renew our spirit, refocus our attention and energy and helped us in carrying out this work.

(Dhruv Prakash) (Gatij Jain) (Gaurav Yadav)

TABLE OF CONTENTS

ABBREVIATIONS

UAV	Unmanned Aerial Vehicle
PSO	Particle Swarm Optimization
GA	Genetic Algorithm
PIO	Pigeon-Inspired Optimization
GSO	Glowworm Swarm Optimization
BBO	Biogeography-Based Optimization
MVO	Multiverse Optimizer
WEP	Wormhole Existence Probability
TDR	Travelling Distance Rate

CHAPTER 1

INTRODUCTION

One of the most significant advantages of UAVs is their ability to work in complex scenarios and in regions where human operations are difficult. The 3D path planning problem is categorized as an NP-hard problem with the primary purpose of creating an optimal trajectory from the source to the destination. One of the essential tasks for a UAV is to proceed optimally from an initial point to a goal while simultaneously avoiding other obstructions in the environment.

There are different methods for path and trajectory planning which consist of the expectation of the complete path before beginning the movement (offline) or the case where the vehicle generates the path while moving towards the destination point (online). The three-dimensional path is represented as an arrangement of coordinated points.

Many of the path planning algorithms make use of conventional methods such as the Voronoi diagram, cell decomposition method, potential field approach, visibility graph,and rapidly exploring random trees (RRTs). The use of these methods can be applied to the problem of three-dimensional path planning as well. Due to their proactive nature, these conventional approaches to path planning are not very efficient and face the problem of local minima trapping and have high time complexity Kimura and Premachandra (2016) Zeng et al. (2017).

Various meta-heuristic path planning algorithms in recent years of research are applied for finding a feasible path solution from the source to destination in a given environment. These algorithms give the most optimum paths in a complex domain with lesser execution times than those of the deterministic algorithms YongBo et al. (2017). These algorithms are very vigorous with a wide range of applications and the capability to obtain high-quality solutions for numerous issues and situations. Many techniques of optimisation like linear programming, A* search algorithm, evolutionary algorithm, PSO(particle swarm optimisation), GA(genetic algorithm), random trees etc, have been used for path planning of UAV and generation of trajectory Roberge et al. (2013).

Brief details about a few meta-heuristics which were proposed recently,and re-

viewed for understanding their probabilistic behaviour and results in various applications are provided in Table I.

For choosing algorithms to be comparision of the results the work Wolpert and Macready (1997) is viewed and after that the results of various cases of simulations are compared with meta-heuristics Glowworm Swarm Optimization (GSO) and Biogeography-based optimization (BBO).

Table 1.1: Various Meta-Heuristics Explored

Algo.	Inspiration	Parameters	Solution	Exploration	Exploitation
WOA Mirjalili and Lewis (2016)	Whales nature of hunting prey	X:Position vector r:Random Vector A,C:Coefficient vectors	Final shrunken circle indicating prey's position.	A random vector is initializes and, directed towards prey's location iteration by iteration.	Wolves encircle the prey, and attack after shrinking the circle.
MFO Mirjalili (2015)	Moth's navigation method	n,m:No of moths and flames D:distance between moth and flame d: No of variables	Reaching final destination by moving in the transverse orientation	Moth finds the next available position in its orientation space with the flame.	After finding the next position, moth navigates towards it.
GWO Mirjalili et al. (2014)	Hunting strategy and Living hierarchy of grey wolves.	X,Xp :Position vector of wolf and prey a,B,w,Y:Heirarchy of wolves A,C:Coefficient vectors Final shrunken circle indicating prey's position.	Final shrunken circle indicating prey's position.	The wolves diverge from each other to search for prey. And converge back on finding it.	Wolves encircle the prey, and attack after shrinking the circle.
GSO Krishnanand and Ghose (2009)	Luciferin updating and neighbor movement behavior of glowworms	nPop:No of glowworms İŞ,p:Luciferin enhancement and decay constant l0:Initial luciferin Range boundary, Range constant, Initial range	Best glowworm Having maximum fitness.	Finding neighbor in range of glowworm having high luciferin.	Move towards the selected best glowworm
DA Mirjalili (2016)	Static and dynamic swarming behavior of Dragon Fly N:Number of neighboring individuals	X:Position of current individual Xj :Position of jth neighbors Ai,Ci:Alignment,Cohesion	Finding Prey's location.	Hunting behavior of the dragon flies.	Navigation movements of the flies.
MVO Mirjalili et al. (2016)	Physics theory of multiple universes, white hole, black hole, and wormhole.	WEP:Wormhole existence probability TDR:Travelling Distance rate nPop:No of universes	The path with highest inflation rate By sorting the universes on basis of fitness value.	Exchanging objects through black holes and white holes.	By exchange of objects of the universes through wormholes.
BBO Simon (2008)	Geographical distribution of Biological organisms	HSI:Habitat Suitability Index SIV:Siutability Index variables λ : immigration rate μ : emigration rate N:No of solutions	The island with highest HSI	Immigration and emigration of species through island.	By probabilstic mutation of solution

This thesis is structured in the form of different sections : Section-2 presents the existing literature in the domain of UAV path planning and highlights the research gap.Section-3 describes the methodology which includes the formulation of the problem ,the description of the environment ,the associated parameters used and the procedure and implementation to get the solution and trajectory of the path planning problem. Section-4 contains the experimental results and the comparisions with other meta-heuristic algorithms such as BBO and GSO. It also concludes the thesis.

CHAPTER 2

Literature Review

There has been extensive research carried out in the use of meta-heuristic algorithms for UAV path planning. Concerning this, YongBo et al. YongBo et al. (2017) presented a modification of the wolf pack search (WPS) algorithm which was used for the path planning of UAVs. Similarly, Zhang et al. Zhang et al. (2016) applied the grey wolf optimizer (GWO) to the path planning problem in the battlefield. Phung et al. Phung et al. (2017) enhanced the discrete PSO technique and used it to formulate the path planning problem for surface inspection based on UAV vision. In a similar context, Chen et al. Chen et al. (2016) presented an optimisation based on modified central force related to the gravitational kinematics metaphor, for solving the problem of planning a 3-Dimensional path for UAVs. In this, the GA and PSO algorithms were applied to improve the original method for optimization of central force. The convergence analysis using the method of linear difference equation has shown that when we use their optimization technique, it produces an effective path planning result. In other work, YongBo et al. YongBo et al. (2017) modified the WPS algorithm to solve the problem of planning a path for a UAV. In this, to realize the WPS algorithm, mutation and crossover operators of the Genetic Algorithm were used. Also, to plan an efficient path, they used the cubic B-spline curve for the process of path smoothing.

In similar context, Duan and Zhang Zhang and Duan (2014) introduced Predator-Prey Pigeon Inspired Optimization for path planning of UAV in 3D and made comparisons with PSO and PIO algorithm. The performance of algorithm was also checked in dynamic environment by Duan and Zhang Zhang and Duan (2017) and good results were produced.

Although many meta-heuristics and deterministic algorithms exist, but their use for planning path of multiple UAVs in three-dimensional environments is not explored to the complete limit.

2.1 Research Gaps

2.1.1 MVO based path planning for multiple UAVs in 3-D

Many nature inspired algorithms like PSO(particle swarm optimization), Glowworm swarm optimization (GSO), etc. have been used in earlier works for optimization of the problem of path planning and trajectory generation. This inspired us to implement Multiverse optimization(MVO) algorithm for path planning of a UAV.
In one recent work Kumar et al. (2018) MVO algorithm was used for path planning in a 2D environment and the results obtained were good. This work inspired the selection of MVO for UAV path planning for a UAV in a 3-Dimensional environment .The area of implementing MVO for multiple UAVs has not been explored much this inspired us to extend this work for path planning of multiple UAVs by using MVO.
Also not a lot of work has been done on comparison of performances of different NIAs for multiple UAVs extensively this work includes doing comparison of its performance with that of other NIAs by varying different parameters which are common in various nature inspired algorithms.

2.1.2 Munkres coordination

In earlier works path planning for multiple UAVs has been done by assigning UAVs to target points with not much consideration for the total path cost of assigning multiple UAVs to different target points when all the target points are similar.
This thesis explores the use of the Munkres algorithm Kuhn (1955) for assigning different UAVs to multiple targets efficiently so that total path cost is reduced. Munkres algorithm runs in polynomial time so overall algo execution time changes only marginally but a significant reduction in overall path cost for different UAVs is observed.

CHAPTER 3

METHODOLOGY

3.1 Formulation of problem

3.1.1 Terrain construction

For solving the problem of the planning of path, we need such an environment where the path for the vehicles will be generated. There are several areas in the environment where there is a prohibition of movement. These are known as obstacles. The UAVs are required to stay away from these obstacles during its movement. The cuboidal obstacles have different sizes. Though in the real environment, the obstacles do not have exact geometrical shape, but because modeling in an environment with obstacles that do not have a perfect geometrical shape is difficult, it hinders the experimentation. However, the algorithm that we created will avoid irregular obstacles sufficiently. For this, the uneven obstacles are covered during the modeling of the environment and the testing process. The purpose of the algorithm for path planning is to find a collection of points that connect S(source) to T(destination). The design of the environment takes place in 3-Dimensional space. A point present in the 3D coordinate space is represented as P = (x, y, z). The source point has its coordinates as S = (S_x, S_y, S_z) and the point located at the destination is allocated the coordinates T = (T_x, T_y, T_z). In the coordinate space, the mapping of obstacles is done in such a way so as to make sure that every point inaccessible to the UAV is declared an obstacle. For path planning in 3D, a UAV can occupy any location in all the three dimensions. Its movement is not restricted in any dimension.

A 3-Dimensional environment is highly complicated, and many computations are needed for generating paths .2-Dimensional path planning is sufficient for the planning of motion for vehicles on land. Path planning of aerial and underwater vehicles is also done in 2-Dimensions while restricting one dimension. The path of UAVs and underwater vehicles becomes realistic when it is produced in 3-Dimensions because they take positions in the complete 3-Dimensional environment in reality. Pandey et al.

(n.d.)

Safe points are the points which do not risk collision with obstacles. Unsafe points are the points which exist at the location of the obstructions. During the planning of the path, the unsafe points are avoided while the vehicle is moving and only safe points are taken into account. The details of the boundaries and obstacles for the three maps used in this paper are given in Tables 3.1 and 3.2 respectively.

3.1.2 Cost function

Various factors are used to decide the cost of potential solutions. Since paths are generated randomly, a vehicle may get stuck in an obstacle and may not be able to continue its movement, and due to this, the destination may not be reached. So, besides the cost of path length and altitude cost, one more cost is added which signifies that there is a gap between the end of the path and the destination and that the path is not complete. The cost functions in this context are as follows :

1. C_{fuel}= It is the cost because of the path covered from the source to the destination. A smaller path implies that this cost is less which in turn implies lesser time and less fuel consumption.

2. $C_{divergence}$= It is the cost due to the sharp divergences occurring on course. Smooth path and avoidance of steep turns lead to reduced fuel consumption.

3. C_{end}= It is summed in the total cost in case the resultant path fails to reach the destination node because a principal obstacle lies in its path. It is of highest importance among all and is given the utmost priority. It needs to have zero value in the most optimum solution.

The solutions of the path planning problem using the MVO algorithm occur in the form of a collection of points representing the path of the UAV from source S to goal T. We represent the solution by $D_{x,y,z}$ in such a way:

$$D_{x,y,z} = \{S, p_1, p_2, p_3, p_4...., p_n/T\} \tag{3.1}$$

Where $p_1, p_2, p_3, p_4., p_n$ represent points present on the path. According to their properties, we set the cost functions respectively. The cost pertaining to the path traveled by the UAV on its course is indicated in terms of the fuel consumed on its journey. Now, we assume the UAV speed to be approximately constant during the journey and evaluate the fuel cost as the total distance covered by the UAV. Mathematically, the fuel cost is described as given:

$$C_{fuel} = \sum \left(D_{x,y,z}\right) \tag{3.2}$$

$D_{x,y,z}$ represents a collection of points lying in succession which describe the UAV path.

Table 3.1: Start and End Boundaries for Maps

Map	Start boundary	End boundary
Map 1	(0,-5,0)	(10,20,6)
Map 2	(0,0,0)	(20,5,6)
Map 3	(0,-5,0)	(10,20,6)

Another cost is $C_{divergence}$ and is defined as the additional cost which occurs when the change in direction from p_{i-1} to p_i does not match the change in direction from p_i to p_{i+1}. $C_{divergence}$ is evaluated by searching the cases in the obtained path with a significant amount of turn in the angle between the two. Let us assume a case with 3 points with coordinates P (x, y, z), Q ($x^{'}$, $y^{'}$,$z^{'}$) and R ($x^{''}$, $y^{''}$, $z^{''}$). Vector PQ and vector QR are obtained. In case the vector product of the two vectors is not zero, it means there is a divergence in relation to the movement of the vehicle. Otherwise, the vehicle moves on the same path. We evaluate the total number of cases where the vector product does not equal zero and consider it the divergence cost. C_{end} signifies the additional cost if the end point obtained in solution is not coinciding with goal point. C_{end} is the gap from the target point to the destination point. If the target point and end point coincide, the C_{end} equals 0, otherwise we set it to a higher value because of its relevance. C_{end} is expressed in the following way:

$$C_{end} = \sqrt{(X_{end} - T_x)^2 + (Y_{end} - T_y)^2 + (Z_{end} - T_z)^2} \tag{3.3}$$

$(X_{end}, Y_{end}, Z_{end})$ represent the end points which are present in the path generated through the solution and (T_x, T_y, T_z) represents the destination node. The total cost function is the sum total of all the cost functions. It helps in determining the most optimum path among the path set. The total cost of the solution of path planning is:

$$C_{Total} = \alpha_1 * C_{fuel} + \alpha_2 * C_{divergence} + \alpha_3 * C_{end} \tag{3.4}$$

Here α_1, α_2, α_3 denote experimentally determined parameters and their value is decided with respect to the parameters of the problem statement. In case a requirement arises such that path length is relatively preferred more, then α_1 is given a greater value. Similarly, α_2 provides preference to path smoothness. Similarly, α_3 gives preference to the fact that the path reaches the destination. The vehicles only tend to have details about their locality and for the next iteration, they may change their location among one of the cells in the neighborhood. So, the UAV does not have details on when and where it will encounter an obstacle in the beginning and so the UAV decides and modifies its course when it finds an obstacle.

Table 3.2: Position of Obstacles in different maps

Obstacle number	Map 1	Map 2	Map 3
1	(0,2,0)-(10,2.5,1.5)	(3.1,0,2.1)-(3.9,5,6)	(0,2,0)-(10,2,5,1.5)
2	(0,2,4)-(10,2.5.6)	(9,1,0,2.1)-(9.9,5,6)	(0,2,4)-(10,2.5,6)
3	(0,2.1,5)-(3,2.5,4.5)	(15.1,0,2.1)-(15.9,5,6)	(0,2,1.5)-(3,2.5,4.5)
4	(7,2,1.5)-(10,2.5,4.5)	(0.1,0,0)-(0.9,5,3.9)	(7,2,1.5)-(10,2.5,4.5)
5	(3,0,2.4)-(7,0.5,4.5)	(6.1,0,0)-(6.9,5,3.9)	(3,0,2.4)-(7,0.5,4.5)
6	(0,15,0)-(10,20,1)	(12.1,0,0)-(12.9,5,3.9)	(0,15,0)-(10,20,1)
7	(0,15,1)-(10,16,3.5)	(18.1,0,0)-(18.9,5,3.9)	(0,15,1)-(10,16,3.5)
8	(0,18,4.5)-(10,19,6)	NA	(0,18,4.5)-(10,19,6)
9	NA	NA	(0,-2,3)-(10,-1.5,1.5)
10	NA	NA	(0,-2,3)-(10,-1.5,1.5)
11	NA	NA	(0,7,0)-(10,7.5,0.5)
12	NA	NA	(0,7,2)-(10,7.5,5.5)
13	NA	NA	(0,11,0)-(10,11.5,2.5)
14	NA	NA	(0,11,4)-(10,11.5,5.5)

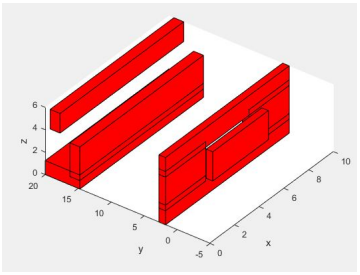


Figure 3.1: Map 1

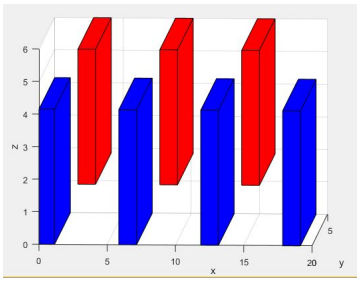


Figure 3.2: Map 2

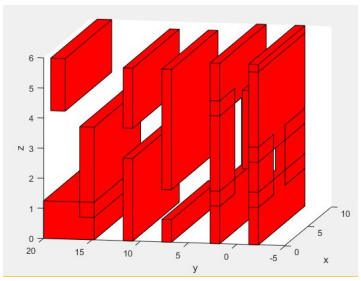


Figure 3.3: Map 3

3.2 Multi-Verse Optimizer

S Mirjalili Mirjalili et al. (2016) introduced the MVO algorithm. It takes inspiration from the theory of multi-verse, a popular theory in astrophysics Barrow et al. (2004). This theory says that not just one, but many big bangs occurred and that every big bang caused the creation of a universe. In this theory, Multi-verse refers to the existence of more than one universe and not just the universe that we live in Barrow et al. (2004). In the multi-verse theory, these multiple universes interact with each other and might even collide. The important multi-verse theory concepts that are selected for the Multiverse optimizer(MVO) algorithm are black holes, wormholes, and white holes. Although in our universe a white hole has never been seen, physicists share the opinion that the big bang can be considered a white hole and this may be one of the principal components for the creation of a universe [28]. The Multi-verse theory [29] cyclic model states that white holes are born when there is a collision of parallel universes. The more observed holes are the black holes, and their behaviour is in contrast to that of white holes. They have a very high gravitational force [30] that pulls everything including light rays. The third type of holes is the Wormholes. They are for connecting one part of a universe

to the other. In the theory of multi-verse, the wormholes function as the tunnels for time/space travel through which objects can travel instantly from one part of a universe to the other and also from one universe to another [31]. This theory says that every universe has a rate of inflation(eternal inflation) that is responsible for the expansion of the universe through space [32]. The rate of inflation for a universe plays a vital role in the creation of stars, asteroids, planets, wormholes, white holes, and black holes and it determines the physical laws and suitability of life. Multi-verse cyclic models [33] states that these universes aspires to reach a stable condition by interacting through black , white and wormholes with the other universes. This is our inspiration for the Multi-Verse Optimizer algorithm.In MVO algorithm, the process of optimization begins by first creating a set of universes randomly and then with increasing iteration, the objects of the universes that have a higher rate of inflation travel to the universes that have a low rate of inflation via black/white holes. Moreover, objects of each universe can be teleported randomly towards the best universe obtained so far. This process iterates till an end criterion is satisfied like highest count of iterations.

3.2.1 Rules applied on Universes of MVO

1. If a universe has a greater rate of inflation , then there is a greater probability of the presence of a white hole.
2. If a universe has a greater rate of inflation , then there is a lesser probability of black holes .
3. If a universe has a greater rate of inflation then there is a tendency to send objects via white holes.
4. A lower rate of inflation implies that there is a greater likeness of receiving objects via black holes.
5. The presence of wormholes can cause random movement of objects towards the best universe obtained so far whatever maybe the inflation rate.

3.2.2 Random Universes Initialization

In this stage, the randomly created universes, are spread out in the solution space. Here n denotes the number of universes (candidate solutions), and d represents the number of parameters (variables).

$$U = \begin{bmatrix} x_1^1 & x_1^2 & \dots & x_1^d \\ x_1^1 & x_2^2 & \dots & x_2^d \\ \vdots & \vdots & \dots & \vdots \\ x_n^1 & x_n^2 & \dots & x_n^d \end{bmatrix} \tag{3.5}$$

3.2.3 Objects Exchange through White/Black Hole Tunnel

In each iteration, there is a tendency for objects to move from the universes with a greater rate of inflation(probably having a white Hole) to the universes with a lower rate of inflation (probably having a black Hole) through black/white holes.

$$x_i^j = \begin{cases} x_k^j & r1 < NI(U_i) \\ x_i^j & r1 \geq NI(U_i) \end{cases} \tag{3.6}$$

where x_i^j denotes the j^{th} parameter of i^{th} universe, U_i represents the i^{th} universe, NI(U_i) is normalized inflation rate of the i^{th} universe, r1 is a randomly generated number in [0, 1], and x_k^j represents the j^{th} parameter of the k^{th} universe chosen with the help of a roulette wheel mechanism.

3.2.4 Objects Teleportation through Worm Hole passage

Wormhole passages are established between a universe and the best universe formed so far. This mechanism is expressed in the following way:

$$x_i^j = \begin{cases} \begin{cases} X_j + TDR \times ((ub_j - lb_j) \times r4 + lb_j) & r3 < 0.5 \\ X_j - TDR \times ((ub_j - lb_j) \times r4 + lb_j) & r3 \geq 0.5 \end{cases} & r2 < WEP \\ x_i^j & r2 \geq WEP \end{cases} \tag{3.7}$$

where X_j denotes the j^{th} parameter of Best universe formed so far, TDR and WEP are coefficients, lb_j represents the lower bound of the j^{th} variable, ub_j represents the upper bound of the j^{th} variable, x_i^j represents the j^{th} parameter of the i^{th} universe, and r2, r3, r4 are randomly generated numbers in the range [0, 1].

3.2.5 WEP and TDR Updation phase

There are two main coefficients here: The traveling distance rate (TDR) and The wormhole existence probability (WEP). TDR is a factor for defining the distance rate (variation) at which the teleportation of an object can take place towards the best universe obtained so far through a wormhole. WEP defines the probability of wormhole existence in the universes. The increase should be in a linear fashion with the iterations so that the focus is on exploitation in the optimization process.The traveling distance rate (TDR) is increased as the iteration number increases so that a more accurate local search/exploitation can be done around the best universe obtained so far. The adaptive

formula for both the coefficients are :

$$WEP = min + l \times \left(\frac{max - min}{L}\right) \tag{3.8}$$

where min represents the minimum (0.2 in this thesis), max represents the maximum (1 in this thesis), l denotes the current iteration, and L denotes the maximum iterations.

$$TDR = 1 - \frac{l^{\frac{1}{p}}}{L^{\frac{1}{p}}} \tag{3.9}$$

where p (in this thesis equals 6) defines the accuracy of exploitation over the iterations. The higher the value of p, the sooner and more accurate the exploitation/local search.

3.3 Path generation for the Universes

With regard to the statement of the problem, the solution is a set of continuous points beginning from the start node and going all the way to the destination node as depicted in Eq. 3.1. While generating the path, for the movement of the point in the 3-Dimensional domain six potential directions are possible. Take, for instance, a point P (x, y, z) in a 3-Dimensional domain, All of the six potential neighbor points are given by :

$$A(x+\delta, y, z), B(x-\delta, y, z), C(x, y+\delta, z), D(x, y-\delta, z), E(x, y, z+\delta), F(x, y, z-\delta) \tag{3.10}$$

Here, δ indicates the least increment value for the specified coordinate point in the objective space to reach the next point. Technically, it can be defined as the shortest distance between any two points present on the map. This converts the space to a grid-based structure. The points which are diagonal can be reached by using these points and thus there is a need for only the basic additions to be specified.

Path generation is the process of finding a path beginning from start point to the destination point by producing a set of points that are connected by use of the neighboring point that lies outside an obstacle. The constant factor ′N′ here is defined for the maximum number of nodes that are possible in the path matrix. If the path matrix exceeds the constant factor N's value, the solution which is returned is the current generated path. The algorithm for generating random paths is : The following algorithm creates the universes as a path from one point to other in the design space. The solutions have a cost value characterized by the cost function defined in the previous section. The value of cost decides how fit a solution is. The solution with a lesser cost is better for the current problem. Anyway because of haphazardness of the neighbor taken from the neighbor set, the arrangement may get into circles or go in the opposite direction to

Algorithm 1 for initial path generation

1) A solution matrix 'P' is initialized .The starting point is added to it .
   This starting point is then set as the current node
2) All the neighbors of the current node are created and they are filtered
   so that only those that don't collide with any obstacle remain.
3) From the set of remaining neighbor nodes,
   one node is randomly chosen and added to the solution matrix
   The present node is set to this node
4) If the number of nodes in the solution matrix exceeds a threshold 'N'
   or if the present node is the ending point, then proceed to step 5 with 'P',
   else, proceed to step 2
5) Any loops in the path are removed by eliminating all nodes that
   occur between two nodes A and B that are the same.
   This reduces path length. 'P' is returned.

what is required to reach the goal point. This causes wastefulness and various cycles are required for acquiring a feasible solution not to mention the optimal solution. The suggested way is creating the initial paths using a pseudo-random strategy by picking neighbors from the neighbor set course towards the goal point which is viable in view of a heuristic to the goal point. In this procedure, rather than picking the 6 neighbors of a point on the map, we choose three reasonable neighbor that shortens the separation towards the goal point. In the process, if one viable neighbor is not present because of obstacle a neighbor is picked from the non-Viable Neighbors set. In the process, if non-Viable selection of neighbors surpass a specific number, it suggests that there is no improvement in the path amid its run and further path generation is terminated. This path can, however, can be enhanced by utilizing further techniques in order to acquire the best path. So, a set of nodes named as Viable neighbors is produced and it stores the neighbors which reduce the separation between the current node and the goal node. Viable Neighbors are the nodes which are current Neighbour or the current neighbour in operable direction towards the goal.

The new algorithm to generate the solutions is given in Algorithm 2

The algorithm leads to the generation of pseudo-random paths from the source point to destination point. This algorithm defines initial universes as well as universes during the wormhole teleportation phase of the MVO algorithm.

3.4 Multi-Verse Optimization algorithm for path optimization

The first MVO algorithm is appropriate to be applied to optimization of continuous functions. The equations for objects exchange updates the objects of the universe as indicated by the Best universe and the objects from white/Blackhole passage of chose

Algorithm 2: First initial solutions are generated by the use of semi-random selection of neighbors.

1) The solution matrix 'P' is initialized. The starting point is inserted. The present node is defined as the starting point 'S'. Viable directions towards the ending point are defined.
2) Create neighbors of the present node. Out of these, those that do not collide with any obstacle are selected. Selection of the neighbors is in the viable directions and then they are added to the ViableNeighbors set.
3) Add a node to the solution matrix 'P' which is randomly chosen from the non-collision Viable-Neighbors. A node is chosen from the CurrentNeighbour set and is added to the solution matrix 'P' when the ViableNeighbors set becomes empty, and the Non-Viable Node count is incremented by one.The node that was selected is now set as the the present Node.
4) If the Nonviable node count exceeds the threshold or if the present Node is the ending node, choose step 5.Else, choose step 2.
5) If there are any nodes in between two repeating nodes,they constitute a loop and are removed. This shortens the path. Path set 'P' is returned .

universes.

In any case, for optimization of path problem, a universe is a path corresponding to path beginning from the start node progressing towards the goal node. Fitness value of the solution path is cost value related to it. Thus, the Multi-Verse optimization algorithm needs to be changed for path planning. So, a few adjustments are introduced in the Multi-Verse algorithm for optimization and are depicted later in this segment. The Multi-Verse algorithm deals with object exchange via White/Blackhole passage and Wormhole tunnel.

The universe which has the higher rate of inflation is considered to contain white holes, and the universe that has less rate of inflation is accepted to possess black holes. The path solution of white hole universe is exchanged with black hole universe. This solution permits the universes simple exchange of path solution as an object. At each iteration, the algorithm sorts the universes by their rate of inflation (fitness) and picks a universe to have the white hole from sorted universe list by using roulette wheel mechanism.

The exploration of the solution space can be ensured by utilizing this component since it is are required that the universes exchange objects and face unexpected changes and thus explore the search space. With the help of the above technique, the universes continue exchanging objects with no disturbance. To maintain a diversity of the universes and to perform exploitation, It is considered that every universe is having wormholes to transport objects of it universe through space in the random fashion. Wormholes cause the haphazard change of objects of the universes without bothering about their rate of inflation. To provide changes that are local to each universe and

that have the high likelihood of enhancing the rate of their inflation, we assume that a wormhole pass is continuously established between each universe and the best universe shaped up until now. There are two important parameters in this: Traveling distance rate (TDR) for finding through exploring new solutions from the current solution and wormhole existence probability(WEP) for enhancing weak solutions.

The universe in the current problem depicts a set of points representing a path on the map. It is needed that the path is changed to ensure that it keeps up its property that it will reach the goal while improving from the solutions which are better. The ideas utilized for the reason are the WEP and TDR for the universe solution for reaching its goal by exploring new paths. The algorithm for path update is as follows:

Case 1:- For the present universe there is a wormhole pass that connect the universe to the Best Universe shaped so far.In this case, the current universe will be changed by objects trading from Best universe. Now the objective is to utilize the Best universe path matrix solution to change the present universe.This would be accomplished by first finding a point that is common in the path solution of the present universe and the Best universe. when a common point is found there it suggests that path solution of present universe can be enhanced utilizing the path solution of Best universe.This can be achieved by picking an arbitrary point after the common point discovered earlier. After the arbitrary point, the path can be generated by the algorithm that produces path as mentioned in segment 3.3

In the event that a common point isn't discovered then present universe way is enhanced concurring the Best universes completely.This is achieved by picking two arbitrary points in the path on the two universes ,after that another path is attempted to be obtained between the two picked points.If a path that is also complete be produced inside n number of takes using algorithm for path generation the then we connect both the points and replace it in the present universe path and afterward whatever path remains between the point and the goal is duplicated from Best universe or produced concurring the algorithm for path generation contingent upon whether the Best universe path effectively achieves the goal point.

Case 2:- In this situation there is no wormhole passage exists between the present universe and Best universe. At that point, we find a different solution from the present solution which could or couldn't be better than the present one. This is done by picking an arbitrary point from the set of path and after that replace it with some other point on the map which isn't present within the boundaries of an obstacle and after that attempt to produce a path that lies between the picked point in the universe and the new point that was picked.

On the off chance that a path isn't generated, at that point on the map pick another point until it is possible to generate a path that is connected between the two points. In case that the path can be possibly generated between the two points, replace the

path that was generated recently generated in the original universe path after the picked point and afterward produces the path from the end point of the path that was recently produced to the goal point. This would be our universe for the present iteration.

Algorithm 3: MVO algorithm for UAV path planning

```
1-Define Parameters: no of universes: n, maximum iteration: MaxIt, max=1, min=0.2,
   P=6, K_1, K_2, K_3, NVNodes_{max},lb=map_lower_bound,ub=map_upper_bound,time=1
2-U= Generated Initial universe Solutions using steps described in section 3.3 .
   Initialize WEP=min, TDR=1, Best_universe=Universe having minimum cost function value.
3-Sort all universes based on the value of cost function of their path solutions.
   SU= Sorted Universes
   NI= Normalize the cost function value (fitnesses) of the sorted universes.
   U=SU
4-While time <MaxIt
      for i=2 to n
            1) Update WEP and TDR according to eqn 3.8 and eqn 3.9 respectively
            2) Black_hole_index=i
            3) r1= random([0,1])
                  if r1 <Normalize cost function value of Black_hole_index
                  universe(NI(U_i))
                        Select White_hole_index by roulette wheel mechanism
                        update path solution of Black_hole_index universe with path solution of
                        White_hole_index universe path solution as described in section 3.2.3
            4) r2=random([0,1])
                  if r2<WEP
                        update path solution of Black_hole_index universe with path solution of
                        Best universe so far as described in section 3.4 Case I
                  else
                        Generate new universes as described in section 3.4 Case II
      end for
   time= time+1
   end while
5-Output the Best universe path solution as the most suitable path.
```

3.5 Munkres algorithm for Coordination of UAVs

Munkres algorithm Kuhn (1955) is a combinatorial optimization algorithm having polynomial time complexity and used to solve assignment problem.

For our problem we applied the same,the algorithm 4 explain the same.

Table 3.3: Algorithm 4

Munkres algorithm for goal assignment and coordination of UAVs	
1	We first calculate euclidean distance between each pair of UAV,U_i (X_i, Y_i, Z_i) and goal point, G_j (X_j, Y_j, Z_j) using eqn. $D_{ij} = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2 + (Z_i - Z_j)^2}$.
2	Make N*N matrix A and fill each entry A_{ij} by euclidean distance of U_i and G_j pair of ith UAV and jth Goal. 1<= i,j <=N, where N is number of UAVs present.
3	For each row of the matrix, find the minimum element and subtract it from every element in its row.
4	For all columns,do the same (as step 3).
5	Using minimum number of horizontal and vertical lines cover all zeros in the matrix.
6	Optimal assignment test: If the minimum possible number of covering lines is N, an optimal assignment can be done and we are finished. Else if lines are lesser than N, we havenâĂŹt found{0.3cm}the optimal assignment, and must proceed to step 7.
7	Determine the smallest element not covered by any line. Subtract this element from each uncovered row, and then add it to each covered column. Return to step 5.
8	After finding the matrix A in which optimal assignment is possible then choose exactly one zero entry from each row such that no two zeroes lie in same column. if $A_{ij} = 0$ is selected entry in optimal solution then we assign UAV(U_i) to Goal(G_j).

CHAPTER 4

RESULTS AND CONCLUSION

4.1 Experimental simulation Results

The MVO algorithm is implemented for the path planning in the 3D environment. It is compared with the Glowworm Swarm Optimization(GSO) and Bio-geography Based Optimizer(BBO) meta-heuristic algorithms. The experimentation is done in MATLAB R2018a in a personal computer with a Central Processing Unit of 3.18 GHz and a RAM of 8 GB. The algorithms are run on three different maps, each of which differs in the position and arrangement of obstacles. Map 1,Map 2 and Map3 are shown in Figure 3.1,Figure 3.2 and Figure 3.3 respectively on page xiv.

4.1.1 3-D path planning for single UAV

In this case, there is no overhead of coordination as only a single UAV and single target is present in each map. Details of coordinates of the initial position of UAV and target for each map is given in Table 4.1. Paths obtained by executing MVO algorithm on each map for given coordinates in Table 4.1 is shown in fig 4.1,fig 4.2 and fig 4.3. Similarly generated trajectories for each map is in fig 4.4,fig 4.5 and fig 4.6 for path obtained above for each map. Various instances of path planning simulation process on Map2 is in Fig 4.7,4.8,4.9.

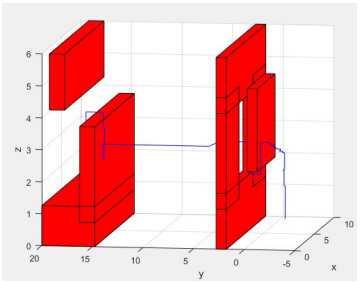


Figure 4.1: Path obtained on Map 1

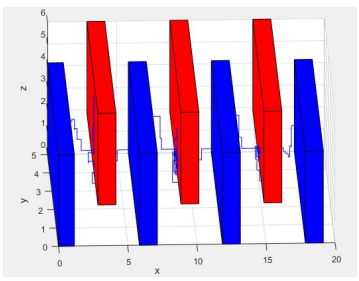


Figure 4.2: Path obtained on Map 2

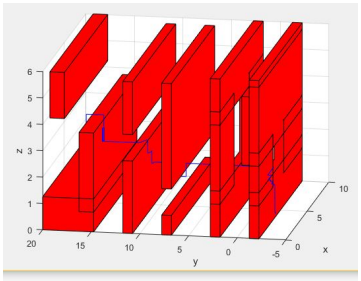


Figure 4.3: Path obtained on Map 3

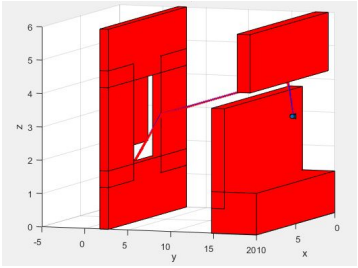


Figure 4.4: Trajectory generated on Map 1

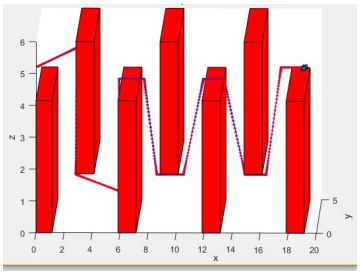


Figure 4.5: Trajectory generated on Map 2

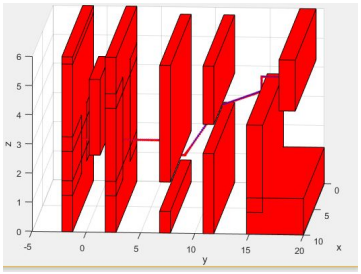


Figure 4.6: Trajectory generated on Map 3

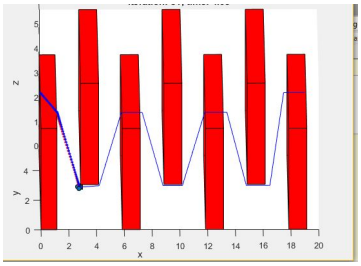


Figure 4.7: Initial instance of simulation on Map 2

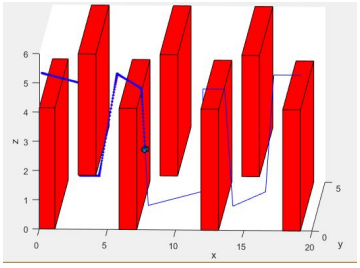


Figure 4.8: Middle instance of simulation on Map 2

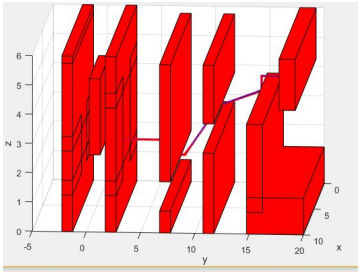


Figure 4.9: Final instance of simulation on Map 2

Table 4.1: Details of coordinates of UAV and Target

3D Environment	Initial coordinates of UAV	Target Coordinates
MAP 1	(0,-4,1)	(2,17,3)
MAP 2	(0,1,5)	(19,1,5)
MAP 3	(0,-4,1)	(2,17,3)

4.1.1.1 Analysis of Average Best Cost and Execution time

The cost function and execution time for different alorithms on different environment is recorded over 20 simulations for two different cases. In Case 1 Maxit=25 and nPop=20 whereas in Case 2 the value of maxIt=40 and nPop=25 are chosen. Each algorithm run on the same set of coordinate points for a given map.The results have been tabulated in Table 4.2, 4.3, 4.4.Graphs plotted for these results are in fig 4.10,4.11,4.12,4.13

It can be observed that the MVO algorithm performs better at finding an optimal cost path followed by the GSO and BBO algorithms for Map 1 and Map 3, whereas GSO outperformed MVO and BBO for Map 2 over the experiment of 50 simulations.MVO and GSO algorithm explores the search space much better than BBO. BBO is the fastest algorithm, but traps in local minima and hence optimal cost path is not obtained by BBO. Also, MVO is the best option out of these three algorithms for path planning as it produces the best optimal path in approximately half the time taken by GSO, which gives the advantage to MVO for path planning in real-time applications.

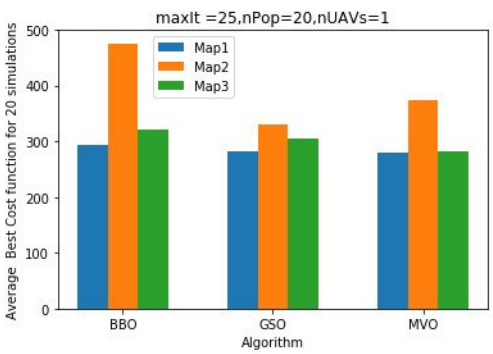


Figure 4.10: Avg. Best Cost in different Maps for different algorithms

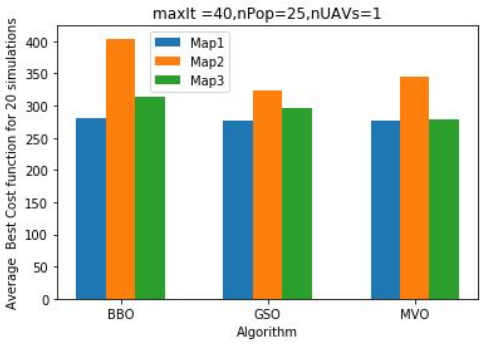


Figure 4.11: Avg. Best Cost in different Maps for different algorithms

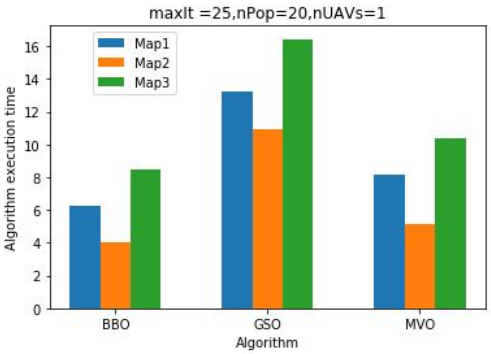


Figure 4.12: Avg. Execution time for different algorithms in different Maps

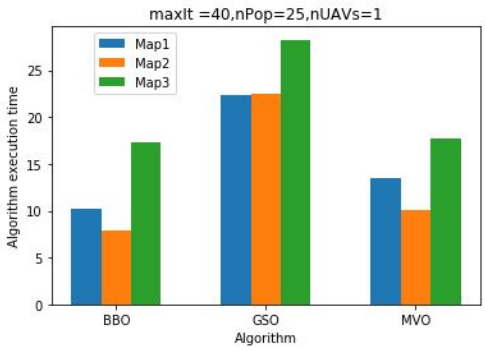


Figure 4.13: Avg. Execution time for different algorithms in different Maps

One more interesting result observed is that as the number of iteration and population size increases from case 1 to case 2 the spread of values of Best Cost of simulation decreases.Also here MVO performs best for case 1 as std.Dev is minimum for Map2 and Map3.Also for GSO improvement in std.Dev is better than other two algorithms in case 2 for all Maps.BBO runs fastest but here is we pay for that and spread in value of Best cost of path is maximum for it. Overall MVO can be best suitable choice for path planning as it is good in finding the best optimal path as GSO and that is also in less time.

4.1.1.2 Convergence Analysis of Average Best Cost with variation in Iterations

Readings of average Best cost of optimal path obtained is taken over 50 simulations by varying number of iterations and keeping population size constant.Results obtained are plotted in fig 4.14,fig 4.15 and fig 4.16 for map1,map2 and map3 respectively.Reading are given in Table 4.5.

MVO outperforms over GSO and BBO in map1 and for map2 but GSO performs best.But Map2 is most complex map and here again MVO outperforms GSO and BBO

Table 4.2: Map 1 Source(0,-4,1) to Destination(2,17,3)

Algorithm	**Iterations**	**Population Size**	**Avg. Best Cost**	**Std.Dev**	**Time(Sec)**
BBO	25	20	293.7	18.04	**6.26**
GSO	25	20	280.9	9.65	13.20
MVO	**25**	**20**	**280.4**	**6.65**	8.13
BBO	40	25	279.9	8.32	**10.17**
GSO	40	25	277.2	**5.73**	22.4
MVO	**40**	**25**	**276.1**	7.32	13.5

Table 4.3: Map 2 Source(0,1,5) to Destination(19,1,5)

Algorithm	**Iterations**	**Population Size**	**Avg. Best Cost**	**Std.Dev**	**Time(Sec)**
BBO	25	20	475.4	124.5	**4.06**
GSO	25	20	**331.2**	**22.2**	10.9
MVO	**25**	**20**	373.6	27.8	5.12
BBO	40	25	404.1	82.2	**7.93**
GSO	40	25	**324.4**	**21.4**	22.5
MVO	**40**	**25**	344.8	29.6	10.04

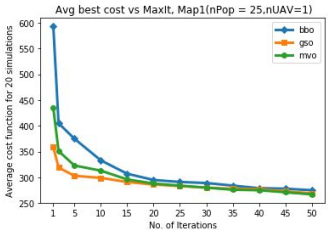


Figure 4.14: Avg. Best Cost vs Iteration for Map 1

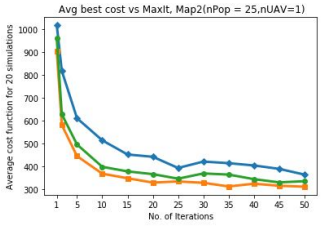


Figure 4.15: Avg. Best Cost vs Iteration for Map 2

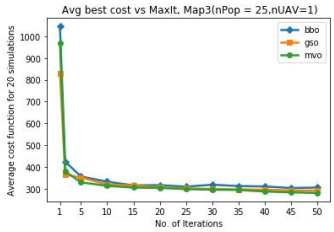


Figure 4.16: 2Avg. Best Cost vs Iteration for Map 3

even at less number of iterations.On average MVO converge better at 30 iterations than GSO and BBO.So MVO is faster at convergence. Spread in values of Best Cost decreases with increases in iterations ,however rate of decreasing in spread is faster for GSO than MVO.

4.1.1.3 Distribution Analysis of Minimum Best Cost

By keeping population size = 25 and iterations = 50 the readings of minimum Best cost is taken over 20 simulations.Here is the Box plot obtained for the distribution of Best Cost of optimal path of all three algorithms on each map in fig 4.17,fig 4.18 and fig 4.19

Table 4.4: Map 3 Source(0,-4,1) to Destination(2,17,3)

Algorithm	**Iterations**	**Population Size**	**Avg. Best Cost**	**Std.Dev**	**Time(Sec)**
BBO	25	20	320	23.89	**8.48**
GSO	25	20	304.6	16.29	16.43
MVO	**25**	**20**	**281**	**14.27**	10.35
BBO	40	25	313	11.77	**17.37**
GSO	40	25	296.2	**8.06**	28.26
MVO	**40**	**25**	**279**	10.77	17.76

Table 4.5: Convergence analysis keeping Population size = 25 (Constant)

Iterations	**Map 1**			**Map 2**			**Map3**		
	BBO	**GSO**	**MVO**	**BBO**	**GSO**	**MVO**	**BBO**	**GSO**	**MVO**
1	593	359	435	1018	904	960	1045	830	970
2	405	319	351	818	581	628	424	365	379
5	375	303	323	611	445	496	356	351	**329**
10	333	299	313	514	368	398	332	319	313
15	307	291	296	452	348	378	315	314	305
20	295	286	288	442	329	366	316	306	303
25	291	283	284	393	334	346	309	301	298
30	289	280	**280**	421	328	369	318	299	295
35	284	278	276	414	312	364	312	297	294
40	279	277	275	404	324	344	310	295	287
45	278	273	271	389	315	330	303	292	283
50	275	269	**267**	364	**311**	335	305	291	**280**

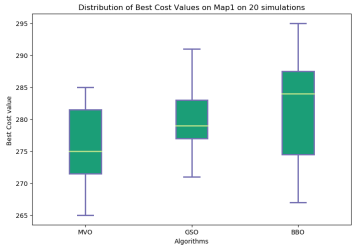


Figure 4.17: Distribution of Best Cost for each Algorithm on Map 1

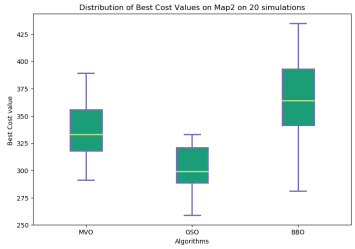


Figure 4.18: Distribution of Best Cost for each Algorithm on Map 2

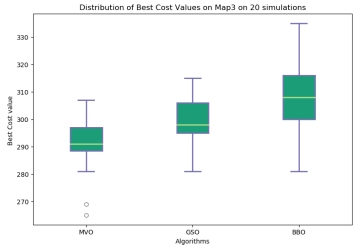


Figure 4.19: Distribution of Best Cost for each Algorithm on Map 3

4.1.1.4 Analysis of Algorithm execution time with variation in Population size

Two cases for variation of algorithm execution time and population size were considered.In case 1 iterations = 25 and in case 2 iterations = 40 were chosen and kept constant throughout for each environment of simulation.

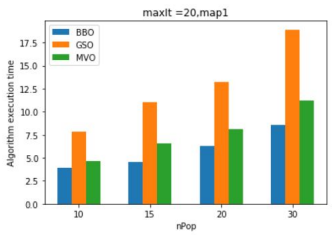


Figure 4.20: Pop. Size vs Algo. Execution Time for Map 1

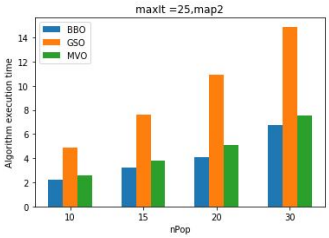


Figure 4.21: Pop. Size vs Algo. Execution Time for Map 2

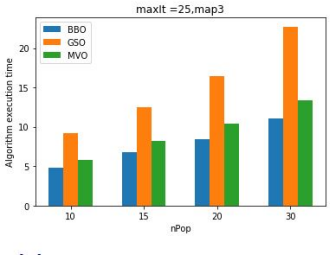


Figure 4.22: Pop. Size vs Algo. Execution Time for Map 3

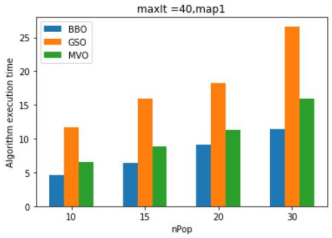


Figure 4.23: Avg. Best Cost vs Iteration for Map 1

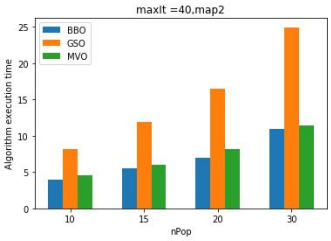


Figure 4.24: Avg. Best Cost vs Iteration for Map 2

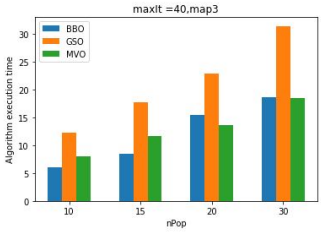


Figure 4.25: 2Avg. Best Cost vs Iteration for Map 3

4.1.2 Multiple UAV Path Planning

In this case, multiple UAVs are present in the environment, so coordination between the UAVs become an important factor for path planning. Results obtained for multiple UAV paths planning on map 3. Results are described below in this section. Trajectory generated with coordination and without coordination is shown in fig 4.26 and fig 4.27.

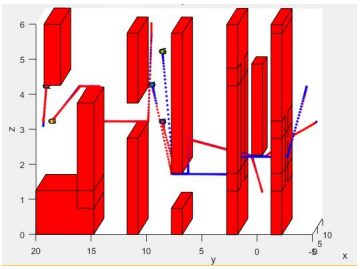


Figure 4.26: Trajectory generated for 5 UAVS on MAP 3 with Munkres

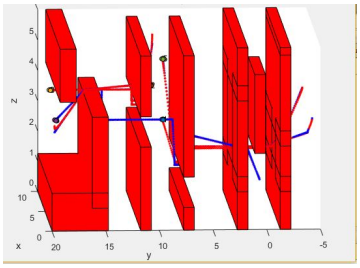


Figure 4.27: Trajectory generated for 5 UAVS on MAP 3 without Munkres

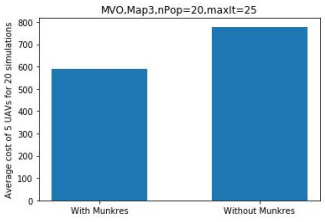


Figure 4.28: Comparision of Avg. Total Best cost of 5 UAVS on MAP 3

4.1.2.1 Effect of the Coordination of UAVs on Average Best Cost

Munkres algorithm is applied for purpose of coordination of UAVs and total average Best cost obtained is compared with the case in which no coordination is present between UAVs and every target is assigned to particular UAV without deciding which

Table 4.6: Algorithm execution time Analysis keeping iterations = 25 (constant)

Population Size	**Map 1**			**Map 2**			**Map 3**		
	BBO	**GSO**	**MVO**	**BBO**	**GSO**	**MVO**	**BBO**	**GSO**	**MVO**
10	3.95	7.84	4.67	2.24	4.90	2.54	4.79	9.21	5.78
15	4.56	11.08	6.54	3.25	7.59	3.81	6.75	12.46	8.23
20	6.26	13.20	8.13	4.10	10.9	5.12	8.48	16.43	10.35
30	8.54	18.92	11.20	6.76	14.9	7.56	11.10	22.72	13.4

Table 4.7: Algorithm execution time Analysis keeping iterations = 40 (constant)

Population Size	**Map 1**			**Map 2**			**Map 3**		
	BBO	**GSO**	**MVO**	**BBO**	**GSO**	**MVO**	**BBO**	**GSO**	**MVO**
10	4.62	11.65	6.60	3.91	8.12	4.60	6.07	12.21	8.02
15	6.39	15.95	8.79	5.53	11.91	5.95	8.54	17.70	11.71
20	9.09	18.21	11.27	6.95	16.49	8.13	15.43	22.92	13.60
30	11.37	26.65	15.89	10.98	24.95	11.43	18.71	31.46	18.53

can be the best UAV for that target. Results for 20 simulations are plotted in fig 4.28. Coordinates chosen on Map 3 is given in Table 4.8. The path is planned using MVO algorithm only as an effect of coordination is analyzed specifically, which will have same nature for other meta-heuristics also. After coordination, the average of the total best cost is reduced considerably as compared to without coordination case. However as the complexity of munkres algorithm is polynomial so we need to consider that effect but for our case execution time of munkres is of the order of 0.1 seconds, therefore, it is ignored. But for a large number of UAVs it should be considered and it will increase execution time.

Table 4.8: Coordinates chosen on Map 3

	Map 3
Start	[5,-4.9,3],[2,19.5,3.2],[5,10,5.8],[5,0,1],[5.1,-4,4]
End	[5,9,3],[6,10,4],[5,19.5,4],[5,19,3],[5,9,5]

4.1.2.2 Effect of number of UAVs on Overall Run Time

In this section, the Overall execution time of path planning is compared with the size of the problem in terms of the number of UAVs that are used for path planning. As using Munkres algorithm shows good results, therefore, readings are taken with coordination case only, but for all three different algorithms on Map, 2.Readings are in Table 4.10. Coordinates chosen for simulation are given in Table 4.9. From results obtained the observed relation between the overall execution time and number of UAV is close to linear.

Table 4.9: Intial coordiantes of UAVs and coordinates of Targets on Map 2

nUAVs	Start
4	[11.6,2.8,5.8],[14.1,2,5],[14,4,5],[11.2,4.5,5]
3	[11.6,2.8,5.8],[14.1,2,5],[11.2,4.5,5]
2	[11.6,2.8,5.8],[11.2,4.5,5]
nUAVs	**End**
4	[19.8,2,5],[19.8,4.5,5.6],[2,4,5.6],[19.8,3,5.9]
3	[19.8,2,5],[19.8,4.5,5.6],[1.8,4,4]
2	[19.8,4.5,5.6],[1.8,4,4]

Table 4.10: Overall time variation with Number of UAVs (Map 2)

Overall time variation with Number of UAVs						
2***No. of UAVs**	**iterations= 25 Pop size =20**			**iterations= 40 Pop size =25**		
	MVO	**GSO**	**BBO**	**MVO**	**GSO**	**BBO**
2	28.18	43.95	20.88	42.02	72.11	27.88
3	36.12	57.93	25.81	57.85	91.16	34.73
4	45.50	70.96	30.24	63.95	116.33	40.06
5	49.5	80.85	34.06	73.03	133.20	44.80

4.1.2.3 Convergence Analysis of Average of Total Best Costs with variation in Iterations

Variation of total best cost with number of iterations is observed on map 1, map 2 and map 3 with all three MVO,GSO and BBO when five UAVs were present in the environment.Coordinates of all five UAVs and five targets for each map is given in Table 4.11,4.12,4.13. Graphs plotted are in fig 4.29,4.30,4.31.

Table 4.11: Coordinates chosen on Map 1

	Map 1
Start	[5,-4,3],[6,-4,3],[5,2.5,3],[5,3,3],[5,-4,5.5]
End	[5,19,2],[6,19.5,3],[6,18,3.2],[5,15,5.5],[5,14,5.2]

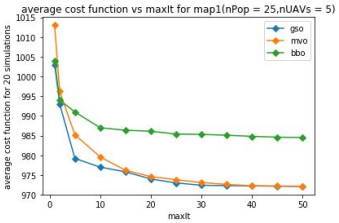


Figure 4.29: Iterations Vs Total Best Cost for Map 1

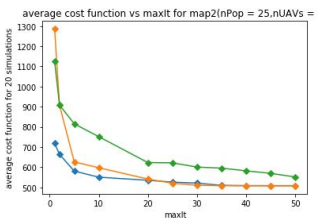


Figure 4.30: Iterations Vs Total Best Cost for Map 2

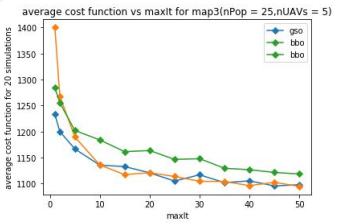


Figure 4.31: Iterations Vs Total Best Cost for Map 3

Table 4.12: Coordinates chosen on Map 2

	Map 2
Start	[14.1 ,2.8, 5.8],[11.6 ,2, 5],[8.1,3,5],[14,4,5],[11.2,4.5,5]
End	[19.8 ,2, 5],[19.8, 4.5 ,5.6],[1.8,4,5.6],[2,3,5.9],[5,4,5.2]

Table 4.13: Coordinates chosen on Map 3

	Map 3
Start	[5,-4.9,3],[6,-4,4],[5,0,1],[5,0,5.8],[5.1,-4,4]
End	[5,19,3],[2,19.5,3.2],[5,14,3],[5,14,4],[5,15,5]

Table 4.14: Caption

4.1.3 Convergence and Complexity analysis of MVO

The presence of heuristic operators makes the algorithms more convergent in the case on less clutter in the environment. Over here, we define a term best cost as the total optimal cost for all the UAV(s) present in the environment after a specified number of iterations(maxIt). This value is noted down for 20 simulations after which the average of all the values is taken. The behavior of this average value with an increase in the value of MaxIt is observed. As the solution is not obtained on initial 1-2 iterations in a complex environment where a line of sight path is not present between source and destination.

For small values of maxIt, the reduction in average best cost is pretty drastic. With further increase in maxIt , the average best cost begins to converge, and for very high iterations the change is pretty negligible. The BBO algorithm performs the worst among the three algorithms and is unable to converge to as good a value as the GSO and MVO algorithms. The GSO and MVO algorithms perform pretty similarly in this respect for higher values of maxIt. For lower values of maxIt, the MVO algorithm starts out worse but converges very quickly as maxIt is increased, eventually catching up to the GSO algorithm. But in most of the cases MVO converge faster than GSO, but in some cases of map 2 GSO perform better than MVO. Results obtained in previous sections shows that MVO algorithm shows better convergence speed as well as ability to find minimum cost path.

One of the fact observed is that MVO also performs better as the complexity of the environment increases in most of the cases. Execution time for small maps are comparable but for a more complex map the execution time difference becomes significant and MVO shows significant improvement there. It is observed that MVO algorithm shows polynomial complexity for path planning problem in a 3D environment and finding optimal path for UAVs.

This performance of MVO makes it suitable for application in the real world for uses like rescue operation and military operation because there both accuracy and time

to obtain to reach the target are important.

4.1.4 Variation in Algo performance with change in parameters

4.1.4.1 Influence of size of population

For the creation of optimal solutions, the number of iterations and the size of the population help each other for the creation of the most optimal solutions. Keeping a very large value of population size is not required and only increases the algorithm execution time. If the population size is kept too low, there would be difficulty in finding the smooth and viable optimal solutions.

4.1.4.2 Effect of the number and arrangement of obstacles

The number and arrangement of the obstacles in the map play a key role in deciding the number of iterations it takes to obtain an optimal solution. If the there is a lot of obstacle clutter between the UAVs starting points and their targets, then more iterations will be required. However, in the presence of a line of sight path between the source and the destination, the number of iterations needed will be lesser.

4.1.4.3 Impact of the cost function parameters and constants

The constant values appended to the cost functions depend on the importance assigned to its differing parts. The role of the path incompletion cost is to ensure that the pseudo-randomly created paths arrive at the target point in an effective manner. In many scenarios, turns are inevitable and hence the turn cost is assigned a lower weight than the other costs . Paths are generated randomly which can often create a lot of unwanted turns which are then removed by the trajectory generator.

4.2 Conclusion

In this thesis, modifications have been proposed to the Multiverse Optimizer(MVO) algorithm for single UAV and Multi-UAV path planning along with munkres algorithm for coordination in a three-dimensional environment. The algorithm successfully generates optimal paths and viable trajectories from source to destination for each UAV while avoiding the obstacles present in the environments provided. The performance of the MVO algorithm was compared with the Glowworm Swarm Optimization(GSO) and Biogeography-Based Optimization(BBO) algorithms. On comparing the performance for a single UAV and target,it was noticed that MVO performed better than BBO based on the average best cost it converged to after 50 iterations.MVO and GSO performed almost similarly in terms of convergence but in that case, MVO is faster than GSO.

On comparing the algorithms for multiple UAVs,it was noticed that for any given value of population size, MVO had a significantly smaller algorithm execution time compared to GSO.BBO was faster than both of these.

MVO algorithm is not best for all applications but its convergence in 3D path planning and faster execution time makes it suitable for various rescue and military application where both accurate and faster technology matters. According to No Free Lunch theorem Wolpert and Macready (1997) nothing can get without paying. However, on performing convergence analysis of these algorithms, it was noticed that BBO failed to converge to as good an average best cost value as MVO or GSO. Both MVO and GSO began performing similarly in this respect as the number of iterations increased. Since convergence to an optimal average best cost value is more important than speed, BBO is clearly the worst algorithm among the three. In terms of convergence, both MVO and GSO perform pretty similarly. However, MVO is faster than GSO in terms of algorithm execution time which makes it the most suitable among all the three algorithms for path planning. The value of average best cost also depends on the starting and ending coordinates and position of obstacles on the map. The Munkres algorithm was also used to optimally assign UAVs to targets in polynomial time. It was observed that the MVO algorithm converged to a lower average best cost value when the Munkres algorithm was used as compared to when it was not used. Also, the change in the overall run time of the algorithms with an increase in the number of UAVs was analyzed.

Future research possibilities can include an extension of the above work to an environment consisting of dynamic obstacles to better simulate a real-world scenario. Modifications can also be suggested by the proposed MVO algorithm to further improve its performance. Apart from this, hybrid algorithms based on MVO can also be explored for applications to this problem.

REFERENCES

[1] Barrow, J. D., Davies, P. C. and Harper Jr, C. L.: 2004, *Science and ultimate reality: Quantum theory, cosmology, and complexity*, Cambridge University Press.

[2] Chen, Y., Yu, J., Mei, Y., Wang, Y. and Su, X.: 2016, Modified central force optimization (mcfo) algorithm for 3d uav path planning, *Neurocomputing* **171**, 878–888.

[3] Kimura, T. and Premachandra, C.: 2016, Optimal relay node selection in two-hop routing for intermittently connected manets., *JoWUA* **7**(1), 23–38.

[4] Krishnanand, K. and Ghose, D.: 2009, Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions, *Swarm intelligence* **3**(2), 87–124.

[5] Kuhn, H. W.: 1955, The hungarian method for the assignment problem, *Naval research logistics quarterly* **2**(1-2), 83–97.

[6] Kumar, P., Garg, S., Singh, A., Batra, S., Kumar, N. and You, I.: 2018, Mvo-based two-dimensional path planning scheme for providing quality of service in uav environment, *IEEE Internet of Things Journal* .

[7] Mirjalili, S.: 2015, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, *Knowledge-Based Systems* **89**, 228–249.

[8] Mirjalili, S.: 2016, Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, *Neural Computing and Applications* **27**(4), 1053–1073.

[9] Mirjalili, S. and Lewis, A.: 2016, The whale optimization algorithm, *Advances in Engineering Software* **95**, 51–67.

[10] Mirjalili, S., Mirjalili, S. M. and Hatamlou, A.: 2016, Multi-verse optimizer: a nature-inspired algorithm for global optimization, *Neural Computing and Applications* **27**(2), 495–513.

[11] Mirjalili, S., Mirjalili, S. M. and Lewis, A.: 2014, Grey wolf optimizer, *Advances in engineering software* **69**, 46–61.

[12] Pandey, P., Shukla, A. and Tiwari, R.: n.d., Three-dimensional path planning for unmanned aerial vehicles using glowworm swarm optimization algorithm, *International Journal of System Assurance Engineering and Management* pp. 1–17.

[13] Phung, M. D., Quach, C. H., Dinh, T. H. and Ha, Q.: 2017, Enhanced discrete particle swarm optimization path planning for uav vision-based surface inspection, *Automation in Construction* **81**, 25–33.

[14] Roberge, V., Tarbouchi, M. and Labonté, G.: 2013, Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning, *IEEE Transactions on Industrial Informatics* **9**(1), 132–141.

[15] Simon, D.: 2008, Biogeography-based optimization, *IEEE transactions on evolutionary computation* **12**(6), 702–713.

[16] Wolpert, D. H. and Macready, W. G.: 1997, No free lunch theorems for optimization, *IEEE transactions on evolutionary computation* **1**(1), 67–82.

[17] YongBo, C., YueSong, M., JianQiao, Y., XiaoLong, S. and Nuo, X.: 2017, Three-dimensional unmanned aerial vehicle path planning using modified wolf pack search algorithm, *Neurocomputing* **266**, 445–457.

[18] Zeng, J., Ke, F., Zuo, Y., Liu, Q., Huang, M. and Cao, Y.: 2017, Multi-attribute aware path selection approach for efficient mptcp-based data delivery., *J. Internet Serv. Inf. Secur.* **7**(1), 28–39.

[19] Zhang, B. and Duan, H.: 2014, Predator-prey pigeon-inspired optimization for uav three-dimensional path planning, *International Conference in Swarm Intelligence*, Springer, pp. 96–105.

[20] Zhang, B. and Duan, H.: 2017, Three-dimensional path planning for uninhabited combat aerial vehicle based on predator-prey pigeon-inspired optimization in dynamic environment, *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* **14**(1), 97–107.

[21] Zhang, S., Zhou, Y., Li, Z. and Pan, W.: 2016, Grey wolf optimizer for unmanned combat aerial vehicle path planning, *Advances in Engineering Software* **99**, 121–136.