3-2002

Multi-Objective Mission Route Planning Using Particle Swarm Optimization

Kursat Yavuz

MULTI-OBJECTIVE MISSION ROUTE PLANNING USING PARTICLE SWARM OPTIMIZATION

THESIS

Kursat Yavuz, 1st LT, TUAF

AFIT/GCE/ENG/02M-04

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

Report Documentation Page

Report Date	Report Type	Dates Covered (from... to)
15 Mar 02	Final	Jun 2001 - Mar 2002

Title and Subtitle Multi-Objective Mission Route Planning Using Particle Swarm Optimization	**Contract Number**
	Grant Number
	Program Element Number
Author(s) 1st Lt Kursat Yavuz, TUAF	**Project Number**
	Task Number
	Work Unit Number
Performing Organization Name(s) and Address(es) Air Force Institute of Technology Graduate School of Engineering and Management, (AFIT/EN) 2950 P Street, Bldg 640 WPAFB, oH 45433-7765	**Performing Organization Report Number** AFIT/GCE/ENG/02M-04
Sponsoring/Monitoring Agency Name(s) and Address(es) Embedded Information Systems Engineering Branch, Information Technology Division, Information Directorate, Air Force Research Laboratory, AFRL/IFTA 2241 Avionics Circle WPAFB, OH 45433	**Sponsor/Monitor's Acronym(s)**
	Sponsor/Monitor's Report Number(s)
Distribution/Availability Statement Approved for public release, distribution unlimited	
Supplementary Notes The original document contains color images.	

Abstract
The Mission Routing Problem (MRP) is the selection of a vehicle path starting at a point, going through enemy terrain defended by radar sites to get to the target(s) and returning to a safe destination (usually the starting point). The MRP is a three-dimensional, multi-objective path search with constraints such as fuel expenditure, time limits, multi-targets, and radar sites with different levels of risks. It can severely task all the resources (people, hardware, software) of the system trying to compute the possible routes. The nature of the problem can cause operational planning systems to take longer to generate a solution than the time available. Since time is critical in MRP, it is important that a solution is reached within a relatively short time. It is not worth generating the solution if it takes days to calculate since the information may become invalid during that time. Particle Swarm Optimization (PSO) is an Evolutionary Algorithm (EA) technique that tries to find optimal solutions to complex problems using particles that interact with each other. Both Particle Swarm Optimization (PSO) and the Ant System (AS) have been shown to provide good solutions to Traveling Salesman Problem (TSP). PSO_AS is a synthesis of PSO and Ant System (AS). PSO_AS is a new approach for solving the MRP, and it produces good solutions. This thesis presents a new algorithm (PSO_AS) that functions to find the optimal solution by exploring the MRP search space stochastically.

Subject Terms
Mission Routing Problem, Particle Swarm Optimization, Evolutionary Algorithm, Ant System, Traveling Salesman Problem

Report Classification	**Classification of this page**
unclassified	unclassified
Classification of Abstract	**Limitation of Abstract**
unclassified	UU

Number of Pages
135

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, the U. S. Government, or the Government of the Turkish Republic.

AFIT/GCE/ENG/02M-04

MULTI-OBJECTIVE MISSION ROUTE PLANNING USING PARTICLE SWARM OPTIMIZATION

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Engineering

Kursat Yavuz, B.S.

1st LT, TUAF

March 2002

AFIT/GCE/ENG/02M-04

MULTIOBJECTIVE MISSION ROUTE PLANNING USING PARTICLE SWARM OPTIMIZATION

Kursat Yavuz, BS
1st LT, TUAF

Approved:

Gary B. Lamont (Chairman)	8 MAR 02 date
Richard A. Raines (Member)	8 Mar 02 date
James Moore (Member)	15 Mar 02 date

Acknowledgments

I would like to express my sincere appreciation to my faculty advisor, Dr. Gary B. Lamont, for his guidance and support throughout the course of this thesis effort. The insight and experience was certainly appreciated. I would, also, like to thank my wife for her endless support throughout my education at AFIT.

Special thanks goes to Hakan San, who helped me build a better design and was always available to answer my questions.

Kursat Yavuz

Table of Contents

Page

List of Figures

List of Tables

Abstract

The Mission Routing Problem (MRP) is the selection of a vehicle path starting at a point, going through enemy terrain defended by radar sites to get to the target(s) and returning to a safe destination (usually the starting point). The MRP is a three-dimensional, multi-objective path search with constraints such as fuel expenditure, time limits, multi-targets, and radar sites with different levels of risks. It can severely task all the resources (people, hardware, software) of the system trying to compute the possible routes. The nature of the problem can cause operational planning systems to take longer to generate a solution than the time available. Since time is critical in MRP, it is important that a solution is reached within a relatively short time. It is not worth generating the solution if it takes days to calculate since the information may become invalid during that time.

Particle Swarm Optimization (PSO) is an Evolutionary Algorithm (EA) technique that tries to find optimal solutions to complex problems using particles that interact with each other. Both Particle Swarm Optimization (PSO) and the Ant System (AS) have been shown to provide good solutions to Traveling Salesman Problem (TSP). PSO_AS is a synthesis of PSO and Ant System (AS). PSO_AS is a new approach for solving the MRP, and it produces good solutions. This thesis presents a new algorithm (PSO_AS) that functions to find the optimal solution by exploring the MRP search space stochastically.

MULTI-OBJECTIVE MISSION ROUTE PLANNING USING PARTICLE SWARM OPTIMIZATION

1. INTRODUCTION

1.1. MRP Overview

Every flight needs some planning before it is flown. Pilots spend their available hours in determining the best flight path to meet the mission requirements. The mission and planning for it become very critical if it takes place on enemy territory. It is obvious that during wartime, every mission is critical. But even during peacetime, there are some missions such as surveillance missions that are critical. They are a must because the Unmanned Air Vehicle (UAV) and the reconnaissance aircraft are the eyes of a country keeping track of the areas and activities of national interest. More money and technology is invested in this area every day for more efficient and effective surveillance of other countries. Building computer programs for pilots and mission planners to determine optimal flight paths in a relative short time supports this activity.

In aircraft surveillance mission, the aircraft is expected to start at a point, go through defended terrain to the target(s) and return to a safe destination. Selection of such a flight path is known as the Mission Routing Problem (MRP) and it is a three-dimensional, multi-criteria path search. The criteria include multi-targets, radars with different levels of risk, shortest travel distance, and day or night factor [23]. The more criteria there are the more constraints there are and that reduces the number of feasible solutions to the problem. The programs, therefore, need more time to compute the solution route. In time critical situations, if the optimal route is found late then it is not a usable solution.

In an Unmanned Air Vehicle surveillance mission, the UAV takes off from the starting point (base), flies through defenses to get to each target, and returns back to the base. The UAV is expected to use the path that has the shortest distance and minimum risk. Usually the shortest path is not the one with the least threat. A weighted cost function model is used in determining the best route. The distance and the risk are both given a weight relative to their importance in the mission. The cost function is the addition of the distance and the risk each multiplied by their importance weight. The path that has the minimum value (fitness) for the cost function is the one that is chosen for the UAV.

In this research, the MRP problem domain consists of a single UAV and multi-targets. The problem can be divided into two sub-problems. The first sub-problem is finding the "optimal" visiting order of targets that provides the minimum (fitness) value for the cost function. The second sub-problem uses this order of targets as an input in creating an "optimal" path between successive targets.

1.2. Research Goals and Associated Objectives

The goal of this work is to design, implement, test, and analyze an effective and efficient MRP system that finds solutions to the multi-objective MRP. Ideas from previous studies have been used as a guide in extending previous approaches. This is the first Particle Swarm Optimization application to 3D Mission Routing Problem. A previous effort uses the Particle Swarm Optimization for the Traveling Salesman Problem (TSP) [22]. Another research uses Tabu Search for the Vehicle Routing Problem [13]. An objective of this research is to extend research [22] from 2D to 3D by applying PSO to the MRP. A previous research uses parallel A* algorithm for 3D MRP [23]. This

research looks at the similar problem but with a stochastic search algorithm instead of a deterministic algorithm.

MRP is computationally challenging. Designing an efficient and effective algorithm to find an acceptable solution for the MRP is the goal. Finding the shortest path, minimizing the flight time, minimizing the possibility of being detected or shot down by enemy fire, and consuming less fuel are the objectives. A route that is safe and effective using preplanning factors may be very dangerous and ineffective during the mission if the planning is not completed in a timely manner. During the time of calculation, the factors that the planning depends on may change. A short execution time, which is very critical for MRP, is one of the objectives.

Another objective of this research is to make radar sites more reflective of real world situations.

Designing a geocentric coordinate interface and integrating it with an existing visualization system is also another objective. This is already implemented in previous work [23]. By using a geocentric coordinate interface, it is possible to get real world elevation data and have the algorithm search on real map data.

The purpose of a surveillance mission is usually not one but many targets. One of the objectives of this research is to adapt the two-target A* MRP implementation [23] to a multi-target PSO_AS MRP implementation.

The last objective is to design a set of experiments that allow testing and evaluation of the performance of the designed algorithm. The performance metrics are selected in order to analyze the results that are gained from the conducted experiments. The attainment of the goal and the underlying objectives must be justified by the results.

This thesis is sponsored by the Embedded Information Systems Engineering Branch, Information Technology Division, Information Directorate, Air Force Research Laboratory (AFRL/IFTA).

1.3. Approach

As background, the problem domain is considered, and models to represent the problem are determined. Possible approaches and algorithms that are applicable to MRP are analyzed.

This research investigation consists of the following tasks:

- Building a program that finds optimal solutions to the single target MRP using the PSO_AS algorithm.

- Integrating the program with a previous effort that is currently used for TSP to extend the problem domain from a single target to multi-targets.

- Using the coordinate system in a geocentric format to be able to import and test the program on real world data.

- Investigating the efficient and effective performance metrics.

- Generating visualization models using Mat lab to determine if the results are effective and efficient.

1.4. Assumptions

It is hard to fully simulate a real world situation. Therefore, some assumptions about modeling the MRP have to be made to lessen the complexity. These assumptions are:

- Maneuver capability of the aircraft. There is only one type of aircraft which is able to make all types of moves including the 90-degree sharp moves.

- RCS (Radar Cross Section) of the aircraft. It is known that the orientation of an aircraft affects its possibility of being detected by radar. Aircraft RCS depends on the radar operating frequency, the material of the aircraft, and the orientation of the aircraft. Calculating aircraft orientation as part of the mission planning greatly increases the

complexity. Some have accepted the airplane as a sphere in their research [12,23] to overcome this complexity. In this research, this assumption is made to reduce complexity.

- Radar range. Radar capability is implemented using a spherical range. An obstacle between the radar and an aircraft within the radar range is assumed not to prevent the radar from detecting the airplane.

- Map range. The aircraft is not able to go out of the map boundaries. It has to find the solution path within the map.

1.5. Thesis Overview

The thesis consists of seven chapters and four appendices. Chapter II is a literature review of the MRP problem and presents an overview of the search strategies used in this field. Chapter III provides information about the PSO_AS algorithm and a high-level design of MRP solution. Chapter IV is about low-level design and implementation issues. Chapter V explains the experiment and testing design. Chapter VI presents the results and the analysis. Chapter VII includes concluding remarks and recommendations for future effort.

2. BACKGROUND

2.1. Introduction

This chapter discusses the historical perspective of the Mission Routing Problem, provides a description of the MRP, and presents the statistical and software engineering techniques used in this research. The historical perspective explains the past efforts on MRP and gives some information about Unmanned Air Vehicles. The MRP models give foundational information about the MRP. The rest of the chapter is about the statistical and the software engineering techniques used in this research.

2.2. Historical Perspective

MRP has always been an interesting topic due to its importance in real world situations. Mission failure can be reduced by the use of real world constraints. The more constraints used, the more effective the solution is. The most used algorithm for MRP has been A* because of its applicability to MRP. There have been many A* implementations in solving MRP. Each successive algorithm has been improved and made more effective [7, 9, 12, 23]. Other approaches that have been proposed for solving the TSP have also been applied to the MRP.

Variations of MRP research have been done at the Air Force Institute of Technology (AFIT) by reporting solutions to a multi-criteria aircraft routing problem using A* and Genetic Algorithm (GA) search techniques [7, 12, 20, 23]. The MRP model was improved as more research added more constraints and increased the efficiency of the model. But the PSO_AS algorithm has not been directly applied to MRP.

2.2.1. Past Efforts

Within the last decade, especially, there has been a considerable amount of work on MRP at AFIT. Table 1 presents a list of past AFIT research efforts that influence this research.

Table 1. Past AFIT Theses on MRP

Date	Research
Dec-85	A Fighter Pilot's Intelligent Aide for Tactical Mission Planning [3]
Dec-86	A Pilot's Planning Aid for Route Selection and Threat Analysis in a Tactical Environment [5]
Mar-91	Multi-criteria Network Routing of Tactical Aircraft in a Threat Radar Environment [29]
Jun-92	Low-Level Aerial Route Planning for Detection Avoidance [30]
Dec-92	Solution to a Multicriteria Aircraft Routing Problem Utilizing Parallel Search Techniques [9]
Dec-93	Genetic Algorithms Applied to a Mission Routing Problem [20]
Dec-93	Multicriteria Mission Route Planning Using Parallelized A* Search [7]
Dec-94	Multicriteria Mission Route Planning Using a Parallel A* Search [12]
Mar-00	Mission Route Planning with Multiple Aircraft&Targets Using Parallel A* Algorithm [23]
Mar-00	A Hybrid Jump Search and Tabu Search Metaheuristic for The Unmanned Air Vehicle Routing (UAV) Problem [15]
Mar-00	A Java Universal Vehicle Router In Support Routing of Unmanned Aerial Vehicles [13]
Mar-01	Traveling Salesman Problem for Surveillance Mission Using Particle Swarm Optimization [22]

2.2.2. UAVs

Unmanned Air Vehicles (UAVs) play an important role in reconnaissance. In the United States Air Force, the 11th Reconnaissance Squadron, consists of UAVs called Predators. In July 1995, the Air Force Air Combat Command commissioned the 11th Reconnaissance Squadron, the Air Force's first operational Predator squadron. The second Predator squadron, the 15th Reconnaissance squadron, was commissioned in August 1997 [25]. Following is a discussion of some of the unique engineering design issues that the UAVs address. These design issues are the constraints that can be incorporated into the UAV routing model.

2.2.2.1. Predator

The RQ-1 Predator uses common avionics and mechanical systems. It has demonstrated the ability to remain airborne for over 40 hours. Predators are currently in production for the U.S. Air Force [25]. It is the only reconnaissance system available in the U.S. inventory that provides near real time video imagery day or night in all-weather conditions via satellite worldwide - without exposing pilots to combat fire. As the first successful unmanned aircraft surveillance program to be fielded in decades, Predator provides tactical and strategic intelligence to operational commanders worldwide [25].

Figure 1. Picture of Predator

2.2.2.2. GNAT Tactical Endurance Aircraft

The GNAT-750 and the IGNAT systems offer the combination of long endurance (over 40 hours), large payload capacity, ease of use and low maintenance while providing a very low cost per flight hour. GNAT aircraft systems are in operation in the U.S. and two overseas countries [25].

Figure 2. Picture of GNAT

2.2.2.3. Prowler II Tactical Aircraft

One of GA-ASI' s newest aircraft, the Prowler II is a scaled-down version of the GNAT System. The Prowler II has an endurance of over 12 hours, payload capacity of 100 lbs providing customers with the capability to meet diverse tactical surveillance requirements [25].

Figure 3. Picture of Prowler II

2.2.2.4. The Altus High Altitude Aircraft

Operational with NASA and the Department of Energy (DOE), the ALTUS has been deployed in support of atmospheric research for the DOE with future plans to use the high altitude capabilities to further atmospheric research, understand the genesis of and predict hurricane paths and damage potential, as well as many other advanced scientific applications [25].

Figure 4. Picture of Altus

2.3. MRP Models

Solving the MRP requires the selection of an optimal path from a friendly base to a target through a defended terrain and return to the base. There are many factors that affect the mission, such as the type of the aircraft, radar sites, threat conditions, weather conditions, refueling capability, terrain masking issues, and distance. A planner has to consider all these factors and plan the flight with minimum risk for the mission. Planning of a mission puts a lot burden on the planner who has to consider the many factors affecting the mission. Automated tools built to help in planning missions must be designed to give consistent, effective, and efficient answers in a short time. There are

some research efforts in this field or similar ones such as Traveling Salesman Problem (TSP)[16, 22], Orienteering [11], and Aircraft Routing [3, 5, 7, 9, 12, 18, 20].

Vehicle Routing Problem (VRP) and TSP can be considered as 2D Mission Routing Problems. In the TSP, the salesman must visit all cities from 1 to n exactly once and then return to the starting city in such a way to minimize the distance traveled. It is a famous NP-complete (not polynomial-time solvable) problem [26]. The distances between cities are provided in an nxn distance matrix. Using this matrix the objective is to find the best order of the cities that minimizes the total distance traveled. Table 1 summarizes the problem domain for TSP [22].

Table 2 - Formalized Table of the TSP Problem Domain [22]

Solution Space Size:	O(N!) where N is the number of coordinates
Domains:	Di: (C) – a list of coordinates (x,y) Do: (C') – an ordered permutation of C such that the total distance of the Hamiltonian cycle is minimized
Objective Function:	Min c= $\sum d_{ij}$ where d_{ij} is the physical distance between adjacent elements of C'
Candidates:	C' is an ordered permutation of C
Selection Function:	Always True. Normally this would be concerned with if C' is in fact a Hamiltonian cycle, but since UAVs are a flying vehicle we thus have a totally connected graph, and all ordered permutations of C are Hamiltonian cycles.
Feasibility Function:	Always True. Normally this would be concerned with if a Hamiltonian cycle exists, but since UAVs are a flying vehicle we thus have a totally connected graph, and all ordered permutations of C are Hamiltonian cycles.
Solution Function:	C' is an ordered permutation of C and Min c= $\sum d_{ij}$ where d_{ij} is the physical distance between adjacent elements of C'

The multi-target MRP is the combination of a TSP and a single target MRP. The first problem is finding the best order of targets to visit. The second problem is building paths in three-dimensional space between each consecutive target.

When the routing takes place in a 3D environment rather than 2D, the problem domain changes. The MRP is more complex. In MRP, the traveler can use any grid points of the map to build the route between two cities (starting point and the target). Starting at a point in a three dimensional grid by using only the adjacent grid points around an aircraft can get to the desired target point. A flight path is formed when the grid points visited are connected.

The complexity of the problem depends on the aircraft model and the length of the possible routes between the starting point (base) and the target. To calculate the time complexity of the problem, it is assumed that the multiple criteria for evaluating the path are combined into a single additive cost function that is bounded above by a constant, Cmax. It is also assumed that there are no negative values for the cost function. Only two criteria, distance and radar are considered. For each pair of points, the distance is the length l(e) of the edge connecting adjacent points in the path. The radar threat is the probability of detection $0=P_d=1$ times a radar multiplication factor k times the distance l(e). The maximum cost per edge is therefore $l(e) + P_d\, kl(e) = (k+1)l(e)$. Consider a function $c(i, j) = (k+1)l(e_{ij})$ which determines the arc cost of moving from one point i to point j. The most direct route is defined as the shortest distance obtained by summing the lengths of line segments connecting consecutive points in the path. Cmax is obtained by setting all criteria to their upper bound for one unit length between two adjacent grid points and multiplying by the total length of the route represented by L. The cost of the optimal route from i to j is therefore bounded above by the maximum cost of a direct path between those points. In other words, the time complexity of finding an optimal solution does not depend on the total number of points in the search space. Instead, the search

time depends on the maximum length for an optimum path. To calculate the maximum length we need to determine a priori the following [23]:

- A constant $k = \frac{C_{\max}}{C_{\min}}$ representing the ratio of the cost of worst case travel (fully exposed to radar) to the cost of optimum travel (with no radar exposure), and
- L representing the length of a "direct" path. L may not be unique due to digitization bias.

The longest path Lmax with cost = Cmax is one with no radar exposure. The longest path with no radar exposure must be equal to kL. Consequently, the length of the optimal path is bounded above by kL = Lmax. The paths of length = Lmax that have endpoints at start S and goal G are confined to the area of an ellipse drawn such that the sum of the radii is = Lmax. Then since only the points within the ellipse should be checked, the total number of points N that must be considered in a search for an optimal solution is the area of an ellipse $O((kL)^2)$ for two dimensions, and the volume of an ellipsoid $O((kL)^3)$ for three dimensions. The maximum number of paths that must be generated is determined by the branching factor b, and the maximum number of points in the optimal path as follows [23]:

$$\sum_{i=L}^{kL} b^i = (b^{kL} - b^L)b/(b-1)$$

The lower bound for the number of paths generated is readily determined if we assume that at each point, the successor is chosen such that it lies on the optimal path. Thus, MRP has lower bound $\Omega(bL)$ =MRP=$O(b^{kL})$ [23].

Typical values for b and k are 17 and 4 respectively, with L ≅ 100 or larger. The branching factor b is determined by the aircraft model, and represents the number of next points reachable from the current point. The variable k is the radar multiplication factor

that is set according to the level of threat assigned to the radar. The higher the value of k, the more the route planner tries to avoid that radar in the route selection. With these values for b and k, even the largest, fastest computers in the world would not be much help in the worst case. Luckily, time complexity is closer to the lower bound in practice [23].

The mission routing problem requires models included to represent the real world. The models needed are as follows:

- Terrain model (maps, grids, etc.).
- Radar model.
- UAV model.
- Cost model.

2.3.1. Model of Terrain

Representing the terrain and flight path of an aircraft requires a three-dimensional discretized space. A three dimensional (x, y, z) grid that shows the latitude, longitude, and elevation respectively is a simple representation that serves this purpose. Arrays are suitable as the data structures for representing three-dimensional grids. Each dimension of a 3-D array can represent a physical dimension of the space. Discrete points are used in representing the coordinates of the flight path and dynamically calculating the radar exposure.

Doubling the resolution of the grid means multiplying all dimensions of the grid by two. Multiplying all 3 dimensions of the grid by two increases the number of cells by $2^3=8$ [23]. The advantage in grid representation is that it is simple to express distance or elevation. Also, since the 3D grid points represent the space, it is easy to make calculations of radar detection and path distance.

2.3.2. Radar Model

There are two different radar models (one for Bistatic and one for Monostatic radars) that are used to calculate the radar exposure. The standard radar equations are used as input functions of the radar system characteristics, the position of the aircraft with respect to the radar, and the radar-cross section (RCS) of the aircraft [17]. The monostatic radar receivers are located at the same place as their transmitters while the bistatic radar receiver locations are separated from their transmitter's location. The radar cross section of the aircraft affects the amount of energy reflected. The signal-to-noise ratio S/N which is used in calculating radar exposure for the two radar types are defined by the following equations [17, 24].

S/N for Bistatic Radar Equation:

$$\frac{S}{N} = \frac{P_t G_t G_r l^2 s F_t^2 F_r^2}{(4p)^3 K T_s B_n L_t L_r R_t^2 R_r^2}$$

S/N for Monostatic Radar Equation:

$$\frac{S}{N} = \frac{P_t G_t G_r l^2 s}{(4p)^3 K T_s B_n L_m R^4}$$

S = received power (in watts)
Pt = power transmitted by the radar (in watts)
Gt = power gain of the transmitting antenna
Gr = power gain of the receiving antenna
λ = wavelength of the signal frequency (in meters)
σ = aircraft RCS (in square meters)
F_t = Transmitter to target pattern propagation factor
F_r = Receiver to target pattern propagation factor
Rt = distance from transmitter to the aircraft (in meters)
Rr = distance from the receiver to the aircraft (in meters)
R = distance from the radar to the aircraft (in meters)

N= noise power at the input to the receiver (in watts)
K= Boltzman's constant (1.38 x 10-23 joules Kelvin)
Ts= received noise temperature (in degrees Kelvin)
Bn= noise bandwidth of the receiver (in Hertz)
L_t = Transmitting system loss
L_r = Receiving system loss

The shape and the material of the aircraft, the position of the aircraft, and the radar operating frequency are the characteristics that form the RCS of the aircraft. The aircraft is accepted as a moving sphere with the same radar cross section from all angles. This assumption is made to avoid the complexity of the RCS calculation for different views of the aircraft. A weighting factor w is used in the cost function and indicates the importance of avoiding radar exposure. The higher the radar coefficient the more risk the aircraft is exposed to at a certain point.

In calculating radar exposure either a dynamic or a static model can be used. In the static model the radar threat is pre-calculated for every grid point and the orientation of the aircraft is not considered. A previous research uses the static method [12]. In the dynamic model, the radar exposure is calculated at every grid point as the aircraft moves.

The probability of detection affects route selection. If the signal is over a threshold value, the radar is able to detect the aircraft. This threshold value depends on the characteristics of the radar. The exposure to radar is calculated at every grid point along the path by using the radar characteristics, aircraft RCS, and the distance of the aircraft from the transmitter and the receiver of the radar.

2.3.3. UAV Model

In real world situations there are many characteristics of the vehicles that limit the possible solutions. Such characteristics used in modeling the aircraft are the combat radius, maximum rate of climb and dive, minimum turn radius, and aircraft RCS. While

planning the path, the capabilities of the UAVs must be taken into consideration. The movements beyond the aerodynamic capabilities of a vehicle must be eliminated. Also the UAV capabilities such as the ability to remain airborne, payload capacity, and fuel capacity mentioned in Section 2.2.2 can be taken into consideration.

Limiting the capability of the UAV to remain airborne due to limited fuel capacity can be incorporated into the UAV model by referencing the real UAV capabilities mentioned in Section 2.2.2. In a 3D grid the number of possible moves increases from the 2D situation of 8 to 26. Figures 5 and 6 show the possible moves.

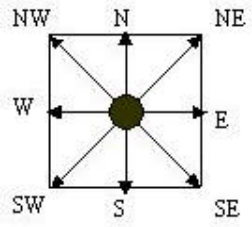

Figure 5. Possible Moves on a 2D Grid

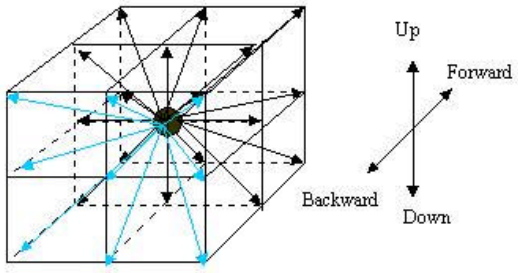

Figure 6. Possible Moves on a 3D Grid

2.3.4. MRP Cost Model for Fitness Evaluation

Distance is calculated knowing the coordinates of each location. The coordinates of the cities (targets including the base) are represented by ($C_{i,x}$, $C_{i,y}$, $C_{i,z}$) where i = 1,2,…n. The coordinates of the radars are represented by ($R_{i,x}$, $R_{i,y}$, $R_{i,z.}$). Fitness of the route is the result of the cost function.

If we accept the distance between cities as Euclidean, then the cost function is the sum of the total distance and the total risk of the route starting from and ending at the starting city. Every C_{ij} represents a point in the fitness landscape. R_{ij} represents the risk going from point i to point j. The grid points that the UAV follows are successive then we know that i=j+1.

$$C_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} + R_{ij}$$

The total cost for n targets would be:

$$C = \sum_{i=1}^{i=n} C_{ij} + C_{n1}$$

The formulas above are for the 3D TSP problem. But in 3D MRP the Euclidean distance cannot be taken as the real distance due to the real world constraints such as the terrain and the radar.

Given a set V of vertices, set E of edges. And a graph G=(V,E), let the length d_i of an edge e_i between two adjacent vertices u, v $\in$ V be defined by the function d(u,v) = d(e_i). Also, let a radar cost r_i for the same edge be defined by the function r(u,v) = r(e_i) = $w_i r_i$, where w_i is the weighing factor of the radar cost, and r_i is the cost of radar exposure for e_i. Total cost of traveling along edge e_i is $c_i = d_i + w_i r_i$. Then the total cost C(s,g) of traveling along a path of N vertices from start s to goal g is simply the sum of the individual edge costs in the path as given by the following equation:

$$C(s,g) = \sum_{i=1}^{N} c_i = D(s,g) + R(s,g)$$

The length d(u,v) = d_i is the distance between u and v. The total length $D(s,g) = \sum d_i$, and the total radar cost $R(s,g) = \sum w_i r_i$.

2.4. Statistical Techniques

In sample data, the mean represents the average of all the observations. The median is obtained by sorting the observations in an increasing order and taking the observation that is in the middle of the series and the mode is obtained by plotting a histogram and specifying the midpoint bucket where the histogram peaks [14].

The disadvantage of mean is that the mean is more affected by the outliers than the median or mode. A single outlier can make a big change in the mean. The median and the mode are resistant to outlying observations. The mean makes full use of the sample since it gives equal weight to each observation [14].

Choosing the proper index of central tendency is important. If the variable is categorical, the mode is more proper to use. If the total number of observations is of interest, then the mean is a more proper index of central tendency. If the total is of no interest, then either the mean or the median can be used depending on the type of distribution.

If the data is skewed, the median is more representative of a typical observation than the mean. A simple way to determine the skewness is to examine the ratio of the maximum and the minimum, y_{max}/ y_{min}, of the observations. If the ratio is large, the data is skewed [14].

Variability is specified using one of the following measures, which are called indices of dispersion. One of the ways to show variability to use the minimum and maximum of the values observed.

2.5. Software Engineering Approach

A good software engineering method requires the software to be built after careful planning. The software must be upgradeable to future work. A code that has to be rewritten for future work can be considered poor in terms of software engineering. An object-oriented approach in software makes upgrading for future work a lot easier.

The development of a software system includes certain steps. These steps are required in order to have a scientific development. By approaching the problem scientifically we can definitely get a better quality product. It might seem, at the beginning, that the scientific development takes unnecessary time, but after going through the steps it is easier to backtrack, or change, or add to the code. Without scientific development it would take days to modify any part of the code. Documentation is another tool that helps. It is to our advantage to go through the scientific steps one by one.

Generally these steps can be divided into four phases; requirements, design, implementation, and testing [18, 19, 23]. The requirements phase is the highest level of abstraction, and consists of analysis and specification. At each level we should apply engineering principles and go into more detail until we develop the implementation.

The output of a software engineering application should be a deliverable software product that is author-independent, useful, and maintainable by third parties who are not part of the development team. The code should be written in ANSI Standard C if execution time is one of the metrics. C or C++ language provides lower execution time

compared to Java, which makes it preferable for high performance computations. A maximum line width should be used to avoid wraparound. Each module should include a header block that includes title, functional description, identification of input and output variables, author's name and contact information, software module versions, and the revision history of functionality for the module including dates. The code segments or lines of code with functionality that is not obvious from the code should have explanatory comments placed before the code in question. Meaningful names that are easy to read and pronounce should be used, together with a consistent capitalization policy. Individual modules, functions or subroutines not exceeding approximately 200 lines of code are preferred. The use of Go To should be prohibited.

2.6. Presentation Techniques

Data visualization is one of the most used techniques for data, especially complex data, presentation. It is the art and science of turning complicated sets of data into visual sight. It enables people to easily make sense out of what otherwise would just be a set of meaningless numbers by using the one third of their brain devoted to visual processing and uses the power of the human eye and brain to discern relationships by presenting complex data as multi-dimensional color images and animations [8, 23].

Mission route planners use the optimal flight path information before planning their mission. For both scientific purposes and daily use, a visual display of both the input and the output data is necessary. The input data is the elevation of the terrain, location of the targets and radars, radar ranges, aircraft data, and other threats specified within the problem domain. The output data is the flight path and more information related with the objectives of the research. A 3D visualization is necessary to show the elevation changes in the flight path. Being able to zoom into the visual display and make rotations to see the

different views of the path is an optional but strongly recommended feature that the visualization program should support. This gives more perspective to the user about the effectiveness of the solution. With the 3D visualization, the decision maker is able to decide if the path meets the required constraints. Otherwise an output of data in numbers is not of any help in making decisions.

It is important that we know the art of data representation. It is the responsibility of the analyst to ensure that the results of the analysis are conveyed to the decision makers as clearly and simply as possible. Graphic charts such as line charts, bar charts, pie charts, and histograms are commonly used in presenting performance results [14]. One of the main reasons for graphical representation of the data is the saying "A picture is worth a thousand words". Graphic charts save time and present the information more concisely. It can also be used to interest the reader. Also graphic charts are a good way to emphasize or clarify a point or to reinforce a conclusion, and to summarize the results of a study.

The aim of the visualization of massive scientific data sets is to devise algorithms and methods that transform numerical data into pictures and other graphic representations, thereby facilitating comprehension and interpretation. By coloring and connecting different segments of data (based upon a user-defined color map and threshold), the algorithms highlight regions of activity.

Terrain visualization is one of the many application areas of scientific visualization. The objective of the Terrain Visualization is to integrate and demonstrate capabilities to collect and process high resolution digital terrain elevation data needed to accurately represent the 3-D image. By doing so it can solve complex geographical problems through computer analyses, design and implement graphic workstation networks, conduct research in graphic technologies and spatial algorithms, and provide

services such as remote sensing, geo-database management, computer graphics, mapping, image processing, digitizing, and spatial modeling [23].

2.7. Summary

This chapter first gives some information about the historical perspective of MRP. Second, the MRP models that are integrated are discussed. Then the statistical techniques, software engineering approach, and the presentation techniques are discussed. The models discussed in Section 2.3 are the foundation for the models used in this research, and they are discussed in more detail in the following chapters.

3. HIGH LEVEL DESIGN OF MULTI TARGET MRP WITH PSO_AS

3.1. Introduction

This chapter presents a high level description of the search technique used in this research for MRP, a modified Particle Swarm Optimization algorithm, PSO_AS, and the high level design issues associated with it. First the Mission Route Planning is described and then the high level methodology used in this research effort is discussed. The high level design explains the general approach taken, some models created for manipulation with computers and the mapping of problem domain to selected algorithm domain.

3.2. MRP Model Representation

This thesis focuses on the issues of designing and implementing a multi-objective MRP for sending a vehicle against multiple targets. The number of targets can be as many as 200. The main purpose of this research is to design an approach that can find an *optimal* solution in a reasonably short amount of time. The optimal solution in this research is the least cost flight path for an aircraft from a starting base to many different destination targets and back to the starting base. The notations that symbolize the models used at this level are as follows:

V = {Vertices}
E = {Edges}
Earth = Graph(V, E)
Terrain $\subset$ Earth
Sky = Earth - Terrain
Path p_i = [S, v_1, v_2,…, v_k, G], $v_k \in$ V, k = ($|p_i|$ - 2)

The models are kept general on purpose at this stage so that the formal high-level design is not dependent on the type of models selected. The refinements are made later at lower level design steps.

3.2.1. Terrain Model

The format of the terrain model is designed to be expressed in both geocentric format and Cartesian format. That allows the planner to import real world elevation data. A mapping from Cartesian to geocentric format is necessary for location representation in real world elevation data. Three different scale factors for the resolution of terrain space are used. The first scale factor is 100m where one increment in Cartesian format corresponds to 3.247 seconds in latitude and 3.769 seconds in longitude of geocentric format. The second scale factor is 1 km in which one increment in Cartesian format corresponds to 32.47 seconds in latitude and 37.69 seconds in longitude of geocentric format. The third scale factor is 100 feet and is easy to convert since one increment in Cartesian format is assumed to correspond to nearly 1 second in both longitude and the latitude. A previous research only uses 100 feet as the scale factor due to its simplicity [23]. Since real world elevation data provided from current web sites [27] are in both meters and kilometers, the other two scale factors are also used. Table 3 shows the necessary information for conversion.

Table 3. Conversion of Geocentric Units to Metric System

Unit	Latitude	Longitude
1 degree	110.874 km	95.506 km
1 minute	1.84798 km	1.59176 km
1 second	30.7984 m	26.5294 m

The first terrain is a 400x400 grid terrain of central Turkey. The scale factor used in the first terrain is 100m and the terrain corresponds to 40km x 40km. The second terrain is a 252x255 grid terrain of Southern Europe. The scale factor used in the second terrain is 1km so the terrain is actually 252kmx250km. The third terrain grid is 200x100 artificially generated terrain and it corresponds to 20,000 ft x 10,000 ft. The third terrain is artificially generated for ease of evaluation and testing. The resolution and the shape of this terrain can easily be changed to the scenario desired by just changing the terrain file.

The base and target locations are chosen from the points that are within the map. The algorithm uses the discrete points in the map. Each change in X or Y direction causes 1 unit change of 100m, 1km, or 100ft depending on which terrain and corresponding scale factors are used. For example, on the artificially generated terrain one minute is one Nautical Mile on a longitude and is approximately one NM on latitude. It is further assumed that one second is 1/60 of a NM, i.e., 1/60 of 6000 feet. That corresponds to 100 feet. It can then be concluded that a change of one unit in X or Y direction is equal to a 1 second change. So, it is easy to make the translation from Cartesian to geocentric format by using simple mathematical calculations.

3.2.2. Radar Model

For the radar to be able to make a decision about whether or not the reflected signal is a vehicle, a certain threshold value must be specified. The threshold value of signal to noise ratio is used in determining the probability of detection. The equation for S/N is already given in Section 2.3.2. In this research, the threshold values below determine the probability of detection:

If $S/N < 0.1$ Probability of detection = 0

If $S/N \geq 0.1$ Probability of detection = 1

In calculating radar exposure, the dynamic model is used instead of the static model. Using a 3-D grid, the radar exposure is calculated only as the vehicle encounters each grid point. There are some advantages to this approach. No additional memory is required. Only the particles' best routes, location of the vehicle and the characteristics of the radar site are stored. Calculations are done as needed and are not saved. Calculations are only performed on some of the route points. These points are the only ones that the flight path uses. This can be a significant time saving over the static approach.

3.2.3. Aircraft Model

As mentioned in Section 2.3.3, there are 26 possible moves an aircraft can make in a 3D space. Allowing certain type of moves for an aircraft is a design issue in which the aerodynamic capabilities of the aircraft must be considered.

In a previous research [23], the movement of the aircraft is restricted to nine discrete forward directions. Descents and climbs are limited to 45° from horizontal. The movement of the aircraft in research [23] is shown on Figure 7.

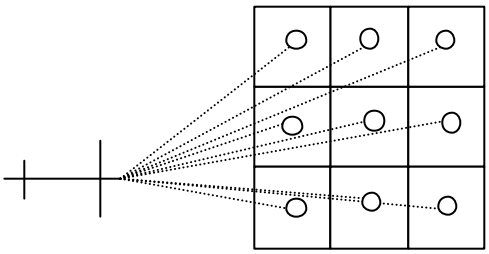

Figure 7. A pictorial representation of forward aircraft movement in research [23]

In this research, the vehicle is able to move to 17 discrete forward, sideway, up and down directions. No backward movements are allowed. In the case of a complex terrain, to prevent the vehicle from getting blocked due an obstacle, the ability of a

helicopter is given to the vehicle. Looking at the path created by the particles the moves that are considered impossible for an aircraft can be corrected later by applying local search techniques along the path. The possible moves for the aircraft in this research are shown in Figure 8.

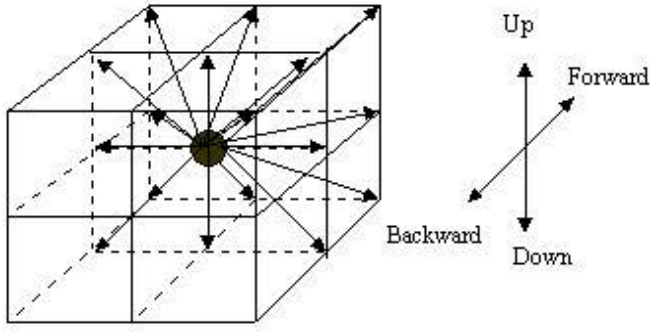

Figure 8. Pictorial representation of forward aircraft movement.

3.2.4. MRP Cost Model for Fitness Evaluation

Multiple factors affect the mission routing problem. To be able to solve the problem by using PSO_AS and still be able to represent the real world conditions, the number of criteria selected must be decreased. Therefore, only route distance and radar exposure are taken into consideration for cost evaluation. These two criteria are selected because they are used to calculate many of the other factors that affect the solution [23]. As mentioned in Section 2.3.4, the total cost C(s,g) of traveling along a path of N vertices from start s to goal g is simply the sum of the individual edge costs in the path as given by the following equation:

$$C(s,g) = \sum_{i=1}^{N-1} c_i = D(s,g) + R(s,g)$$

The length d(u,v) = d_i is the distance between u and v. The total length is $D(s,g) = \sum d_i$, and the total radar cost is $R(s,g) = \sum w_i r_i$ [23].

3.3. Designing the PSO Algorithm to the Problem

There is a direct data structure mapping from the algorithm domain to the problem domain. In order to produce a mapping of the input of the problem to the input of the algorithm, the cost of each path between the targets is calculated by the initial function. The initial function tries to calculate the cost of each path between targets by making a best first search. This gives more realistic approximation of the cost of the paths among the targets compared to the Euclidean distance. After the cost calculations, a symmetrical matrix c_{ij} is produced and used in the algorithm. This mapping of inputs is O (n^2)- the number of operations needed to fill the square matrix. For output, there is a direct mapping O (1) from the "target numbers" in the sequence of the tour to the original coordinates using the target number as an index to determine the coordinates.

3.4. Algorithm Design

Every Evolutionary Algorithm has its own way of searching the fitness landscape and trying to find the optimal solution. The difference stands out better by comparing their data structure. The data structure and symbolic notation for the PSO_AS is mentioned in the following sections while the structure and symbolic notations of other EA algorithms are presented in the Appendix D.

3.4.1. The Data Structure for the PSO

In PSO the swarm population is represented by the particles. These particles search the fitness landscape by the interaction among them. The interaction is the combination of the recombination, mutation, and selection events. At every generation the particles that are selected affect the interaction and all the members of the population continue to search by adapting themselves to other particles that have low fitness values. The fitness value is the result of the cost function and shows the quality of the route built by the particles. Different probabilities in PSO result in different convergence rates.

Representation : Particles

Fitness : Scaled objective function value

Self-adaptation : Velocity vector on the particles

Mutation : The probability of choosing global best, personal best and current

Recombination : Effect of global, local and the current on the velocity for the offspring, main operator

Selection : Probabilistic

Constraints : Limit of velocity

Theory : Different convergence rate for different probabilities

3.4.2. Symbolic Notation of PSO

t ← 0

P(t) ← initialize (Randomly create first tour for each particle)

F(t) ← evaluate (Evaluate cost function ΣC_{ij}, and determine Global Best)

While (Objective function ≠ true) do

P'(t) ← build a new tour for each particle using trust C_1, C_2, C_3

No mutation operator is used

F(t) ← evaluate cost function ΣC_{ij}, and determine Global Best)

P(t+1) ← Select (P''(t) = P'(t))

t = t + 1

end

3.4.3. PSO_AS Description

PSO_AS makes a stochastic search in the fitness landscape, which is the phenotype search space. Phenotype is the behavioral expression of the genotype in a specific environment [2]. In this research, the specific environment mentioned corresponds to the MRP search space. The genotype is the sum of inherited characters maintained within the entire reproducing population. It is often the genetic constitution underlying a single trait or set of traits [2]. The constraints, such as the terrain and the radar, form the genotype search space. In the genotype search space, the coordinates of the cities are $(C_{i,x}, C_{i,y}, C_{i,z})$ i ? (1…n). The particles of the swarm are represented with $X_1 \ldots Xm$. M is the total number of particles searching the fitness landscape for the optimal solution. The PSO_AS equations take place in the phenotype level. These equations are two simple equations representing the moves of the particles in the phenotype search space.

Equation 1 – Calculating a Single Particle's New Velocity

$$V_{t+1} = C_1 V_t \oplus C_2(P_{ig,t} - X_t) \oplus C_3(P_{\forall g,t} - X_t) \qquad (1)$$

Equation 2 – "Moving" a Single Particle in a Swarm

$$X_{t+1} = X_t + (?t)V_t \qquad ?t = 1 \qquad (2)$$

Where:

1) V_{t+1} – The particle's new velocity for the next generation
2) C_1 - A percentage of the number of "steps" in a velocity to be used. A measure of how much the particle "trusts" its own exploration.
3) V_t - The particle's current velocity. A list of permutation steps.
4) $\oplus$ - The concatenation of velocity steps.
5) C_2 - A percentage of the number of "steps" in a velocity to be used. A measure of how much a particle "trusts" its neighborhood's best velocity.
6) $P_{ig,t}$ - The neighborhood (from i to g) best position
7) "-" - The difference of two positions is the velocity that will transform the second position into the first position
8) X_t - The current position
9) C_3 - A percentage of the number of "steps" in a velocity to be used. A measure of how much a particle "trusts" the global velocity.
10) $P_{\forall g,t}$ - The global best position
11) "+" – The transformation of a position using the velocity (yields a position)
12) X_{t+1} - The particle's new "moved" position. The position in the next generation [2].

Fitness(x_t) = Function (x_t)

After applying both equations to a particle at each generation, the new fitness value of the particle is determined by the cost function. The equations applied to the particles' position (genotype level) produces the fitness values (phenotype level).

The particles of the swarm are first scattered randomly in the fitness landscape. After evaluating the particles' fitness values, the particles move to their new position in

the phenotype search space. Equation (1) determines each particle's move direction and amount and Equation (2) determines their new position. This process continues till the particles converge at a point or find an acceptable solution. This ongoing process is explained in more detail in the following sections.

3.4.3.1. Initialization (Initial Function)

In the genotype space, an aircraft knows the location of the targets so that it can make a move towards the target. On the way towards the target, there are some factors that prevent the aircraft from going straight to the target. These are the radar, the terrain, and the probability function. Of course, the probability function should not be considered as a problem but rather as a necessity in terms of providing randomness in the initialization phase. Without the probability function, all the particles make the same move, which results in building the same path. Of course, the amount of probability used in selecting the route is a parameter that can be played with to see how the particles move. The probability function can be both static and dynamic. With the static probability function, the values for probability of selecting the position from all the possible choices is the same. In a smooth physical landscape, a high probability of selecting the best route is better than having a lower probability for the best route. The reason is that with the high probability, the particles tend to go straight towards the target, which means having a shorter distance. In a chaotic physical landscape, this can cause many problems. When the particle gets stuck in a hole, it is very hard for the particle to escape that obstacle. The best way is to either choose a probability depending on the physical landscape type since we know what the physical landscape is like or to have a dynamic probability that decreases its probability of choosing the best path when it is stuck and increases its probability when it can escape the obstacle.

At the initialization phase, manipulating the particles just towards the target, even with low selection probability of the fittest next grid point, does not provide a good random initialization. One way to provide random initialization is to put pseudo targets evenly distributed throughout the physical landscape, manipulate the particles towards these pseudo targets, and then vector them to the real target. This is just one way of moving the particles throughout different parts of the physical landscape. Another way is to divide the physical landscape into smaller grids and make sure that the particles search different grids every time.

3.4.3.2. Fitness Evaluation

Each route that the particles follow provides a fitness value of that route. The fitness value of the route is determined by the cost function mentioned in Section 2.3.4. Each particle in the physical landscape represents an airplane that follows the route. But in the fitness landscape, a route represents a particle. The order of the coordinates of the route can be considered as the position of the particle in the fitness landscape. At the end of the initialization phase, every particle that completes its path produces a fitness value. By ranking these fitness values, a global best of all the particles and a personal best for each particle are chosen. The positions of these particles in the fitness landscape, which correspond to their routes in the physical landscape, are saved for later comparisons.

The second part of the evaluation comes into play after each generation of particles. The global best of all the particles and the personal best for each particle is updated if a better one is found.

3.4.3.3. Recombination

The recombination phase starts just after the initialization. It is an important ongoing process that continues till the end of the algorithm. This is the main part of the interaction of particles in the fitness landscape to get to better fitness values. The interaction of particles in the physical landscape is not very different from the interaction in the fitness landscape. As a result, each particle depending on its trust on the global, local or personal best is going to be affected by these three particles that help the particle find its new position in the fitness landscape. By mapping the point in the fitness landscape to the physical landscape, it is possible to get the best route found so far. The search of particles by interaction in the fitness landscape is shown on Figure 9.

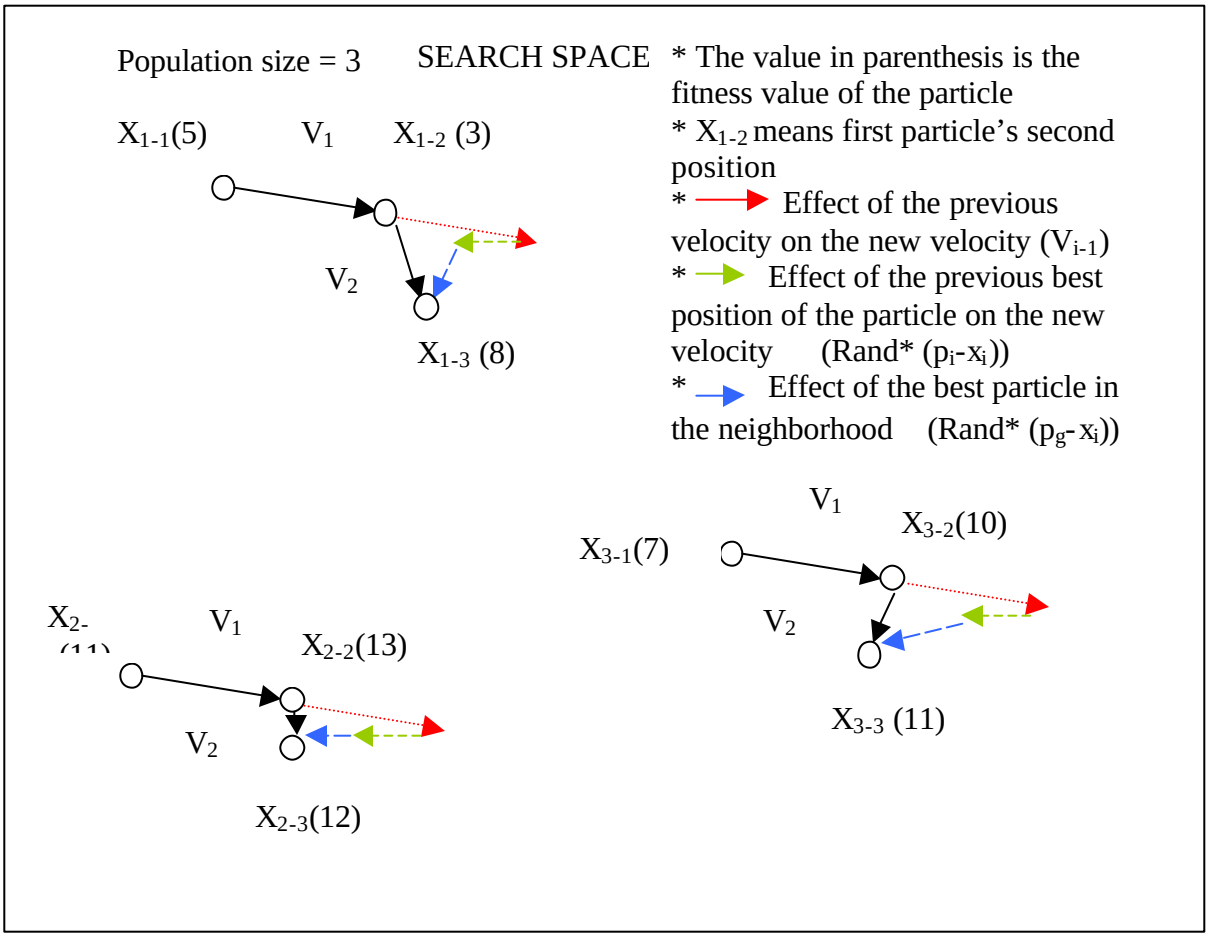

Figure 9. The Search of Particles In the Fitness Landscape

3.4.3.4. Convergence

As the particles move in both the physical and the fitness landscape, as a result of the interaction, the particles eventually converge. The routes become the same and no better solution is produced. The best solution found by the particles may not be the global best solution of the problem. It may be a good local point that the particles have converged to so a no-hope condition occurs which means the reinitialization or the end of the algorithm depending on the termination condition.

3.5. Graphical Interface

Most researches do not provide easily used programs for the people who may want to use the program for their own purposes. For a program to be easily used by the pilots or the flight path decision makers, an easy to use graphical interface such as a menu is desired. In most programs, the user has to provide the inputs that the program needs from the command line. For users who are not good at using computers, this may become a time consuming event. But a graphical interfaced program design can help the users overcome this problem. For this reason a graphical menu was prepared to help users with the inputs. The program starts with the display of the first menu. Figure 10 shows the first menu.

Figure 10. Menu 1

On the first menu, the user decides which scale factor to use. As mentioned before in Section 3.2.1, there are three scale factors, which are 100 m, 1 km, and 100 ft. Clicking on the radio button of the selected scale factor and pressing the start menu button starts the second menu. Figure 11 shows the second menu.

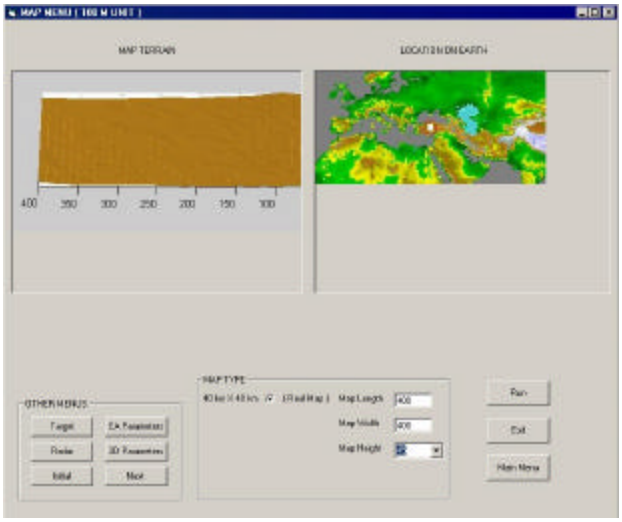

Figure 11. Menu 2

There are 3 different types of terrain data. There is one set of terrain data for each scale factor. The picture on the upper left corner of the menu shows the terrain and the picture on the upper right corner shows the location of the terrain on world. As the radio button for the map type is clicked, the map length and the width are shown. The user enters the map height using the map height combo box. To get to the next or any other menu, any one of the buttons on the bottom left corner of the menu is pressed. After getting through all the menus, the user should return to the second menu to run the program by pressing the Run button. Clicking on the next button starts the third menu. Figure 12 shows the third menu.

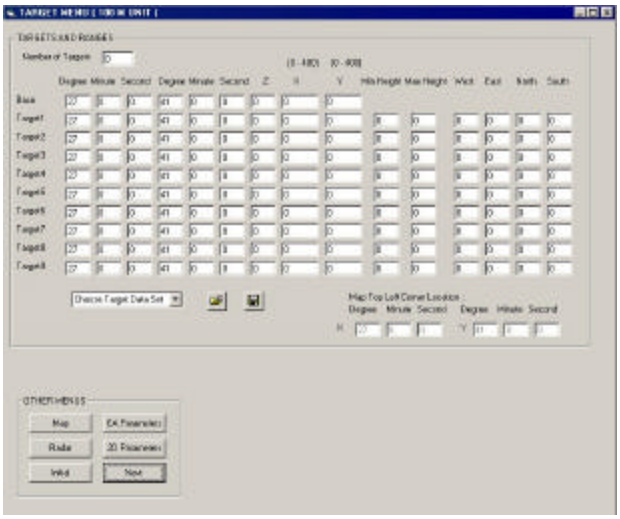

Figure 12. Menu 3

The third menu is the target menu. The targets on the terrain should be located within the map area of the selected terrain. To help the user place the targets, the coordinates of the Northwest corner of the map is provided in terms of geocentric units on the middle left part of the menu. The number of targets is written into the Number of Targets edit box on the left top corner of the menu. The number of targets does not

include the starting point, which is called the base. Below the number of targets edit box, there are many edit boxes for the user to input the target locations. By entering the target locations in geocentric units (latitude and longitude), the menu automatically converts them to Cartesian format without the need of a button and displays them under the X and Y static labels. The user has to enter the height of the targets located under the static label Z. The height entered is multiplied by the scale factor used. For a scale factor of 1km, a height of 20 means 20 km. The buttons to the right of the Y static label show the range of each target. The user specifies how close the UAV needs to get to each target to take the picture needed. The Save and Open buttons are put on each menu to help the user avoid having to enter the same inputs every time. Figure 13 shows the fourth menu.

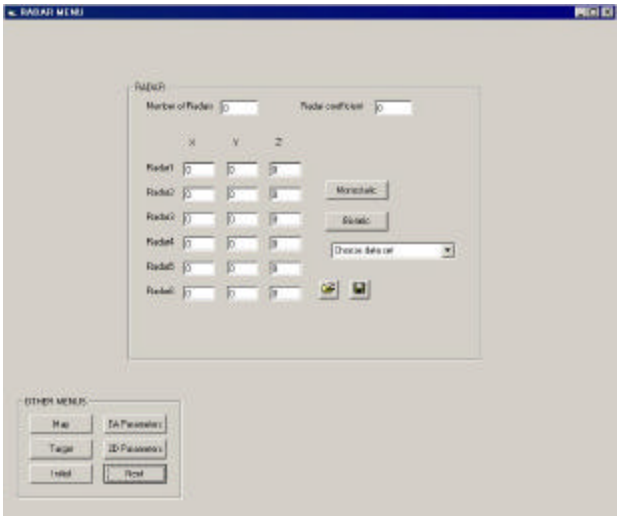

Figure 13. Menu 4

The fourth menu is designed to enter the radar locations. But only Cartesian format is available, so the user has to make the necessary conversions. This is the only not user-friendly part of the menu. The radar coefficient edit box is to specify the importance of radar avoidance compared to distance. The higher the radar coefficient, the

more the UAVs avoid the radars. The Monostatic and Bistatic buttons are put on the radar menu to specify each radar type. Figure 14 shows the fifth menu.

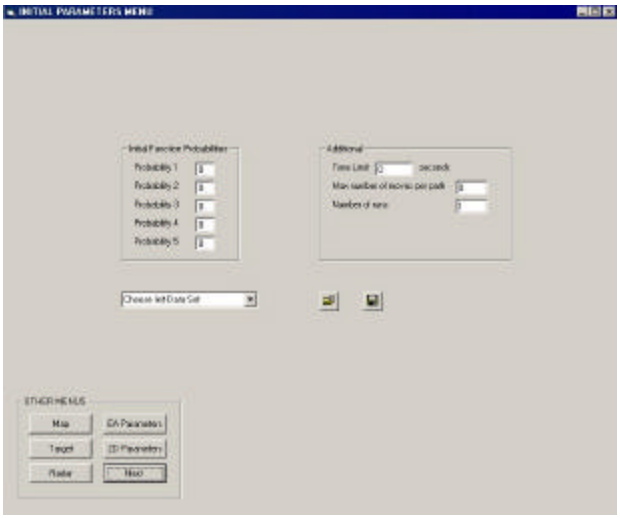

Figure 14. Menu 5

The initial function parameters are specified on the fifth menu. The probabilities for choosing the next candidate grid point are entered. Probability 1 shows the probability of the UAV choosing the best next grid point, Probability 2 shows the probability of the UAV choosing the second best next grid point, and so on. The parameters under the Additional static label are not currently used. Figure 15 shows the last menu.

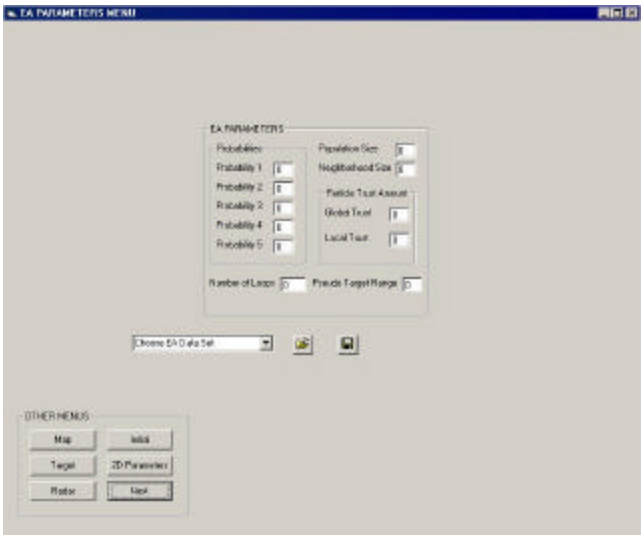

Figure 15. Menu 6

The sixth menu is where the EA parameter values are entered. The probabilities specified serve the same purpose as the ones on the Initial function parameters menu except this time these probabilities are used for the PSO function instead of the Initial function. The population size, the neighborhood size, the amount of trusts required for recombination, the termination condition, and the pseudo target ranges are all specified on menu 6.

3.7. Summary

In this chapter, the MRP is described by incorporating its problem domain with the algorithm domain since the first step of solving a problem is to have a thorough understanding of the problem. The design methodology is discussed, which is centered on the PSO_AS algorithm. Some of the implementation decisions are also discussed. We need to follow software engineering principles in designing a new system, because they offer a wide variety of tools that we can use to create an efficient, effective, and

maintainable system. Following the introduction of the PSO_AS algorithm design, the graphical interface is explained by using the snapshots of the menu.

4. LOW LEVEL DESIGN AND IMPLEMENTATION OF MRP

4.1. Introduction

This chapter discusses issues introduced in the previous chapter in greater detail. The low level design and implementation of the PSO_AS algorithm is the focus of this chapter. First, some representation issues of 3D maps are explained and then the methodology this thesis uses to solve the MRP is explained.

4.2. Low Level Design

The 3D map representation uses standard x, y, and z coordinates. The starting points for each x, y, z coordinate is 0. The last point on the x, y, z axis indicates the length, width and, the height respectively of the map selected. The vehicle's base (starting point) and the targets are each represented by unique 3D location coordinates on the map.

In a 2D map, a vehicle is able to make 8 moves as described in Section 2.3.3. Every move on a 2D map has a number associated with it to be able to keep track of the move directions during the execution of the program. The numbers in Table 4 are used to represent directions for the 2D MRP.

Table 4. Move Numbers for 2D Directions

Move Direction	Move Number
North	0
Northeast	1
East	2
Southeast	3
South	4
Southwest	5
West	6
Northwest	7

On a 2D discretized map the Unmanned Air Vehicle (UAV) uses the grid points on the map to get to the target desired. To show the flight path, a line connects every successive grid point that is visited. Figure 16 shows a path in a 2D map.

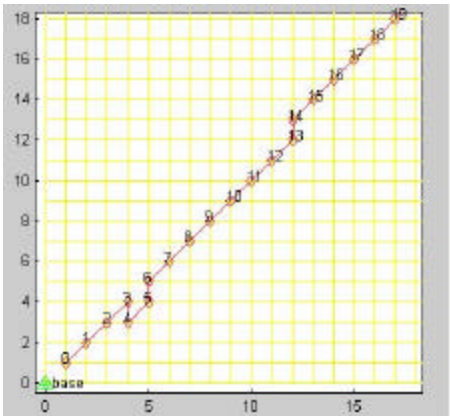

Figure 16. Picture of a path on 2D map

On Figure 16, the green triangle shows the starting point. The UAV follows the red line until it reaches the target. The numbers on the red line show the cumulative number of steps it takes to get to that point.

Adding another dimension to the 2D space, which is the elevation, creates 3D space. As with 2D representations, the 3D move directions also have numbers associated with them. Table 5 shows these numbers (U: UP, D: DOWN, S: SOUTH, W: WEST, N: NORTH, E: EAST).

Table 5. Move Numbers for 3D Directions

Direction	Number	Direction	Number	Direction	Number
UN	0	N	9	DNE	18
UNE	1	NE	10	DE	19
UE	2	E	11	DSE	20
USE	3	SE	12	DS	21
US	4	S	13	DSW	22
USW	5	SW	14	DW	23
UW	6	W	15	DNW	24
UNW	7	NW	16	D	25
U	8	DN	17		

In 2D or 3D discretized space, the vehicle has to use the grid points adjacent to the grid point it is on to get to the target desired. It cannot jump directly to another grid point that is not adjacent. The distance between adjacent coordinates in the 3D map is one distance unit multiplied by either 1, $\sqrt{2}$, or $\sqrt{3}$ depending on which direction the move is. One distance unit is equal to the scale factor of the terrain. As seen in Table 5, the distances of the move directions represented with a letter are 1. The distances of the directions with 2 letters are $\sqrt{2}$ and the distances of the 3 letter directions are $\sqrt{3}$. The smaller the scale factor the higher the resolution is. Figure 17 shows a path in a 3D map.

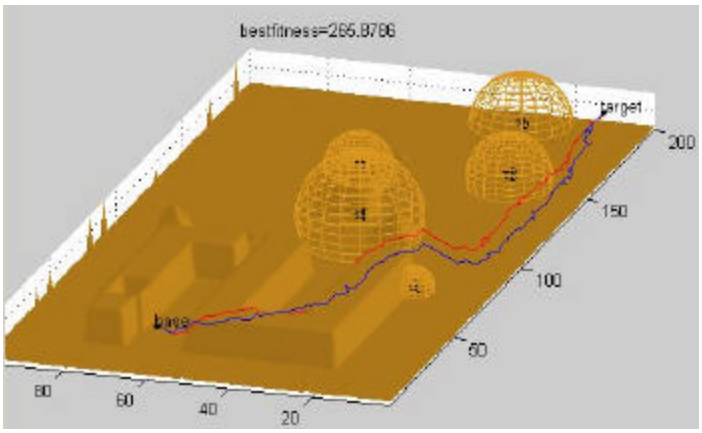

Figure 17. Picture of a route in a 3D Map

Solving the Multi-target MRP

As explained in Section 2.3, the single target MRP is a step in the multi-target MRP. In the single target MRP the particles look for the best path between the starting point and the target. In the multi-target MRP, an order of targets must be specified so that the particles know between which targets to look for an optimal path. With a few targets it is easy to find the best order of targets. All permutations of the order of targets can be tried and the best order can be found but with a 100 target MRP, this means trying 100! different target orders. Trying to build this many flight paths of different target orders would take years.

In the TSP, the salesman must visit all the cities from 1 to n once so as to minimize the distance traveled. The multi-target MRP is very similar in that the UAV must visit all the targets from 1 to n exactly once so as to minimize the distance traveled and the detection risk. In the TSP the distances between the cities are known and they are provided in an nxn distance matrix. In the multi-target MRP the optimal routes are unknown so a cost matrix of the approximate costs of the routes between the targets must be created.

Another challenge comes in finding the cost of the routes between each target including the starting point. Accepting the Euclidean distance between each target is a solution but not a good one. The radar and the obstacle constraints between targets increase the fitness values by increasing the risk and the distance of the path. The simple Euclidean distance obviously does not take these factors into account. The path between every target including the starting point must be built so that it uses the more realistic, constrained, distance and risk detection.

The path between each site is built by applying a best first search technique. In best first search, the best grid point from the next 17 candidate grid points is always chosen. By building paths with best first search between targets, a cost matrix is prepared. Recall, cost includes risk as well as distance. The best next grid point of the 17 is the one that increases the fitness value the least. Equation 1 shows the number of paths built for n targets including the starting point.

Equation 1. Number of paths necessary for a cost matrix

n = number of targets

$$total\#\ paths = \frac{(n+1)(n+2)}{2}$$

After creating the cost matrix, the problem becomes a TSP. To determine the best order of targets any algorithm used for TSP can be applied. Since the algorithm used in this research is PSO_AS, the same algorithm is used for finding the best order of targets. The code of a previous research for TSP [2] using PSO_AS algorithm is directly integrated into the implementation since there is no point in reinventing the wheel.

With the order of targets to visit provided, the next step is to find acceptable paths between each target beginning with the starting point. An initial function is applied to the

first search space, which is the space between the starting point and the first target. After the initialization phase, the PSO function is applied to improve the path until the termination condition is met. The same process is repeated between every consecutive target until the last optimal path between the last target and the starting point is found. At the end, all the paths between the consecutive targets are connected to get the total flight path.

Initialization (Initial Function):

From the starting point, the particle does an evaluation of the adjacent grid points. Every point in the 3D grid has 9 coordinates in front of it, 8 coordinates around it and 9 coordinates at the back, relative to the target. The algorithm does not allow back moves so 9+8=17 moves are allowed. Every time the vehicle gets to a new position while on its way to the target, it evaluates the next 17 coordinates using the cost function. The cost function calculates the risk level of the next grid point and how much closer the aircraft gets to the target with that grid point. The cost function does not evaluate coordinates that are under the terrain or outside the map, so there is no way the plane can go out of the map. Depending on the risk and the distance, the algorithm ranks the next available coordinates from best to worst. Then, based on the probability distribution one of the candidate is chosen using a random number input. The user specifies the probability distribution before running the algorithm.

Pseudo Targets:

A constraint on 3D routing produces a tendency for the particle to move toward the target and prevent it from making a whole exploration of the map (genotype space). This does not cause many problems where there are only a few radars or when the radars are widely spread with gaps between them. If there are many radars that form a wide strong block between the starting point and the target, the particles are not able to explore the sides of the map where there may be gaps for the particle to slide through without getting caught by the radar.

Since all the particles at the initialization create paths within this limited area it is not possible for the PSO to move the particles towards the sides. This is true for both the genotype and the phenotype space. It cannot be said that there is a direct mapping from the genotype to phenotype space, but this exploration logic works the same for both genotype and the phenotype space. Figures 18 and 19 show the interaction of particles in both phenotype and genotype space to get a better appreciation of the particle movement.

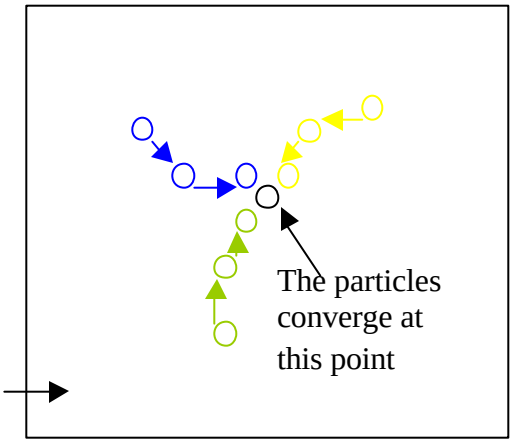

Figure 18. Phenotype level particle interaction

None of the particles are able to explore the area that is pointed to by the arrow at the left bottom of Figure 18 unless there are particles that are around that area by chance.

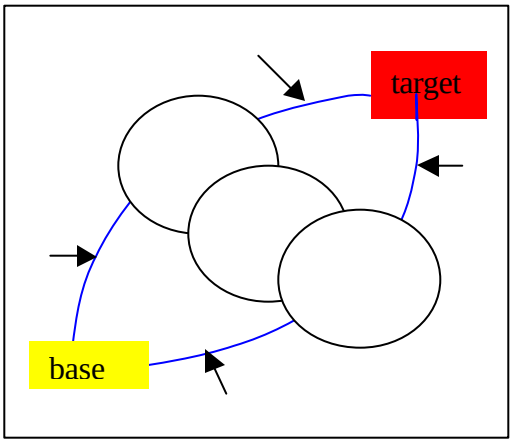

Figure 19. Genotype level particle interaction

On Figure 19 the particles make moves toward the target. The elliptic area, pointed to by the arrows, is the area that the particles use to get to the target. The particles hardly go outside this area since the interaction is not able to allow particles to explore the area outside the elliptic area. The three circles represent the radar block between the base and the target.

To make a better exploration of the genotype search space, pseudo targets are evenly scattered in the search space between the base and the target. There are 20 pseudo targets located in between the base and the target. Particles first go toward one of the pseudo targets until they get within some range of the pseudo target and then head toward the real target. This range is one of the algorithm's input. So each particle creates an additional route for each of the 20 pseudo targets. As a result of this the population size is multiplied by 21. A population of 10 at the initialization phase becomes 210 at the end of the initialization. Figure 20 shows the exploration by the particles in the genotype space with pseudo targets.

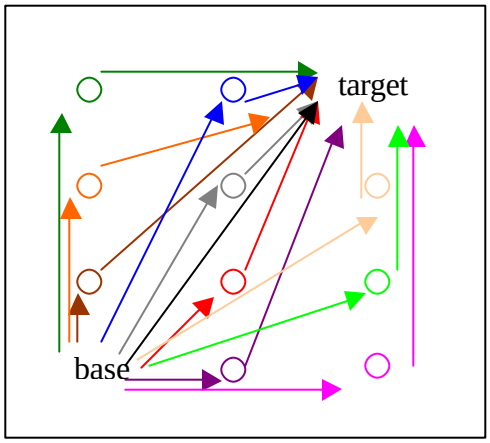

Figure 20. Exploration with pseudo targets

Particle Swarm Optimization Algorithm Description:

Instead of a direct PSO algorithm, a revised PSO algorithm called PSO_AS is used. PSO_AS algorithm is designed to make a better exploration of the search space [2]. It is expected that the PSO_AS can do the same type exploration within the area that the particles can explore.

After the initialization phase, every particle has a personal best and there is a global best, which is the best of all the personal bests. The initialization phase produces the first generation of particles. The following generations are the production of the interaction of the particles called PSO.

Move choices are based on the global best particle and the local best particle as in PSO_TSP. At every grid point for each particle the algorithm generates a random number between 1 and 100. This random number determines if the particle attempts to follow the global best, the local best, or takes a pseudo-random path based on the particle's position before commencing a new tour. If any of these moves do not work because they cause the particle to go out of map or go below the terrain, the initialization function decides the next grid point of the particle. If the global best is chosen, the next grid point is

determined by the move that the global best has taken. If the next grid point of the global best is not acceptable, then the local best move is applied. If the local best does not work either, then the last move of the particle is applied. If this does not work then the initialization function decides which grid point to move on to, and this is an acceptable move.

The initialization function is also applied when a particle does not reach the target within a number of moves equal to the least move number. The least move number is the least number of moves of the global best, personal best and the last tour. The moves with the interaction of particles ends when the number of moves is equal to the least move number. If the particle still has not reached the target, the initialization function takes over and finishes the particle path to the target.

Pseudo Algorithm

The Pseudo algorithm provides a better understanding of the PSO_AS algorithm. The MRP PSO_AS pseudo algorithm is as follows:

1 Initialization
   (a) Read the data/parameters necessary for the initial function.
   For 1 to number of targets:
   (b) Create paths between each target and the base by the initial function.
   (c) Calculate the path cost for each path and put the fitness into a integer matrix

   End For 1 to number of targets

2 Perform PSO_AS for TSP Algorithm (PSO_AS for 2D)
   (a) Read the data/parameters(fitness matrix created by Initial function) necessary for 2D PSO_AS Algorithm.

   (b) With PSO_AS, determine what the visiting order of targets should be.

3 Perform PSO_AS for 3D
   (a) Read the data/parameters (order of targets created by 2D PSO_AS).

For 1 to number of targets:
For 1 to population size:
For 1 to 21(20 pseudo targets + 1 real target):
(b) Create initial paths between the next set of two targets with initializing function.
(c) Calculate the initial costs of the path.
(d) Determine the best path of each particle, the global best, and their fitness.
End of For 1 to 21(pseudo targets + no pseudo target):
End of For 1 to population size:

While (no improvement by PSO_AS less than n)
For 1 to population size:
For 1 to 21(pseudo targets + no pseudo target):
(e) Create a new PSO_AS path with particle interaction.
(f) Evaluate the costs of the PSO_AS path.
End For 1 to 21(pseudo targets + no pseudo target)
End For 1 to population size
* If there is improvement compared to global best set noimprovement = 0
* Else noimprovement = noimprovement+1
End While (no improvement by PSO_AS less than n)

End For 1 to number of targets:

4.3. Constraints

There are many constraints in the 3D MRP. Terrain elevation on 3D is the main difference that does not exist in TSP/VRP/ 2D routing problems. One of the problems caused by terrain is when the airplane fronts an obstacle. To avoid this problem, the only way is to allow the vehicle to be able go straight up. Even though no vehicle can make a 90-degree move, it is allowed here to make that move so that it can complete its path and let the next particle create a route. While not implemented in this research, it is possible to correct these types of moves by applying local search techniques along the path to prevent moves outside the aerodynamic capability of the aircraft.

Another constraint is the radar. The aircraft has to avoid the radar and this requirement adds to the complexity of the problem. The range capability of the radar and

the probability of detection are both factors in the cost function and require an increased amount of input data.

Another constraint is having more than one target. As explained in Section 4.2, in an addition to the MRP problem, a TSP application must be applied to find a good order of targets.

4.4. Summary

In Chapter 4 the low level design and implementation of the PSO_AS algorithm on MRP is explained in detail. The pseudo algorithm is provided to give an understanding of the low level design. Last the constraints in MRP are discussed.

5. EXPERIMENTAL DESIGN PROCESS

5.1. Introduction

This chapter defines a set of MRP experiments. The experiments are designed to see if the design goals are met. The implementation is expected to produce effective and efficient results. The experiments are conducted to validate these objectives:

- Generating acceptable solutions for different data sets.
- Showing the importance of parameter value variations in Evolutionary Algorithms.
- Presenting improvement produced by Particle Swarm Optimization compared to Initial Function (Best First Search) for complex data sets.

The following sections give a detailed description of the experiments. Before giving the results, the design of experiments and performance metrics are described.

5.2. Design of Experiments

5.2.1. Hypothesis Testing

The experiments are grouped into 3 sets. Each set has a different number of experiments. These experiments are conducted to validate the objectives stated in Section 5.1.

The first experiment set is conducted to present the importance of parameter variations in the EA algorithms. The EA parameter values are changed each experiment to be able make better analysis of the effects of EA parameters on the MRP. There are 6 different EA parameters for the MRP and these are the move probabilities, trusts, population size, neighborhood size, pseudo target range, and the termination condition.

The statistical techniques mentioned in Section 2.4 are used in evaluating the results of parameter value changes. The mean of the results of each experiment are compared since the distribution of the experiment result data is not skewed. To give an understanding of the distribution of the data, Figure 27 is the main figure that shows all the averages of the experiment results of set 1 with the maximum and minimum values.

After validating that the PSO_AS algorithm makes improvements to the fitness values found by the Initial function (Best First Search), the second experiment set is conducted to show that the PSO_AS algorithm makes better improvements when there are more constraints.

The third experiment set is conducted to compare the PSO_AS algorithm with the A* algorithm. The same terrain, radars, and targets used as the test case in a previous research [1] of A* implementation is used. Visual results of the paths found by both implementations and their execution times are compared.

Other than PSO_AS and A* there are many algorithms suitable to find solutions to the MRP. The structure for some of the EA algorithms and their applicability on MRP is discussed in Appendix D.

5.2.2. Experiment Set 1

The experiments in experiment set 1 are designed to evaluate the performance of PSO parameters on MRP implementing a single aircraft against multiple targets. A real map of 400x400 has been used as the terrain [27]. Since 1 unit is expressed as 100 m for this map type, the map size corresponds to 40 km x 40 km. The elevation data is taken from a web site where real elevation data of the whole world is provided in ASCII format. The map height is 45 in Cartesian format, which corresponds to 4500m. The elevation data belongs to a middle part of Turkey. 3 targets and 6 radars are located on

the terrain. Throughout the 96 experiments the initial function parameters and the locations of the targets and radars are kept the same. Different EA parameter values are used for each of the 96 experiments. The aircraft model is the same where 17 of 26 moves are allowed. Each parameter has two different values except the trusts, which has 3. So, the number of experiments is 2x3x2x2x2x2 = 96. Each experiment is repeated 10 times so the total number of runs is 960. The number of replications is 10 since it is suggested that number of replications is kept small, for example, 10 [14]. The 96 experiments are divided in 6 groups and 16 experiments are conducted at a time. Table 6 shows the values of parameters.

Table 6. EA Parameter Values of Experiment Set 1

Parameters	**Value 1**	**Value 2**	**Value 3**
P1, P2, P3, P4, P5 (PSO)	100, 0, 0, 0, 0	90, 4, 2, 2, 2	
Global trust	50	10	80
Local trust	30	10	10
Personal Trust	20	80	10
Population size	5	10	
Neighborhood size	5	10	
Pseudo target range	10	5	
Termination condition	20	150	

The target locations are shown in Table 7.

Table 7. Target locations in Experiment Set 1

Target Number	Location- Geocentric	Location- Cartesian (x, y, z)
Starting point	27? 0'3''E 40? 38'25''N	1, 1, 16
1	27? 6'18''E 40? 41'3''N	100, 50, 23
2	27? 3'8''E 40? 58'24''N	50, 370, 25
3	27? 22'0''E 40? 49'12''N	350, 200, 18

The terrain data used in the first experiment set is a real elevation data. It allows the user to test different aspects of the program execution, such as if the algorithm is making an effort for terrain masking to evade radar exposure, if it is taking a route away from terrain and radar, if it is taking into consideration the distance. It is a V-shaped terrain with different altitudes that an algorithm might choose to exploit. Figure 21 shows this terrain and Figure 22 shows the location of the terrain in Turkey.

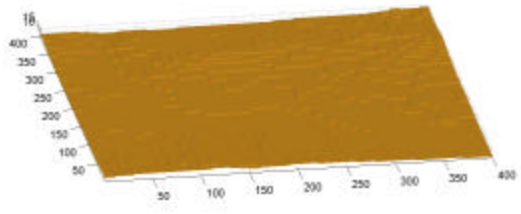

Figure 21. Visualization of the terrain used for first experiment set.

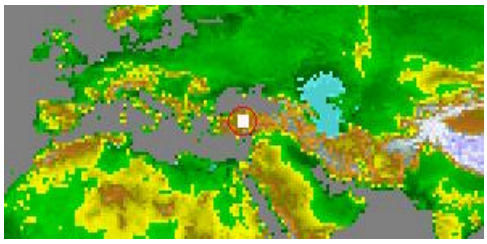

Figure 22. Location of the terrain in the world

6 radar sites are distributed over the terrain. All 6 radars are Monostatic and their technical specifications are considered to be the same for all the sites. Figure 23 visualizes the locations of the radar sites. The radar locations are also presented in Table 8.

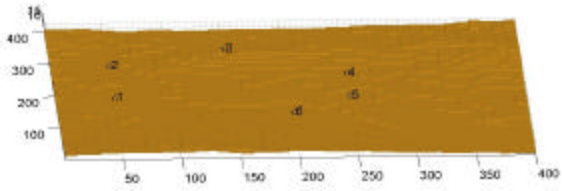

Figure 23. Radar locations on the terrain.

Table 8. Radar Locations in Experiment Set 1

Radar No	Location-Cartesian-(x, y, z)
1	50,150, 20
2	50, 250, 20
3	150, 300, 20
4	250, 220, 20
5	250, 150, 20
6	200, 100, 20

5.2.3. Experiment Set 2

Experiment set 2 consists of 2 experiments. On the first experiment there is only one target, no radars and no obstacles between the target and the starting point. It is designed to show that the initial function produces good solutions for problems without constraints or very few constraints. On the second experiment, the starting point is the same. The target is located further from the starting point than the first experiment. There are obstacles and three radars located in between the starting point and the target. This experiment is designed to show the PSO improvement made to initial fitness values when the problem has many constraints. An artificial map of 200x100 is used as the terrain. The scale factor is 100 ft. Since 1 unit is expressed as 100 ft, the map size corresponds to 20,000 ft x 10,000 ft. The map height is 45 in Cartesian format, which corresponds to 4500 ft. The aircraft model is the same. The terrain data used in the second experiment set for both of the experiments is an artificially created terrain with an S shaped mountainous area. Figure 24 shows this terrain. The target locations are shown in Table 9.

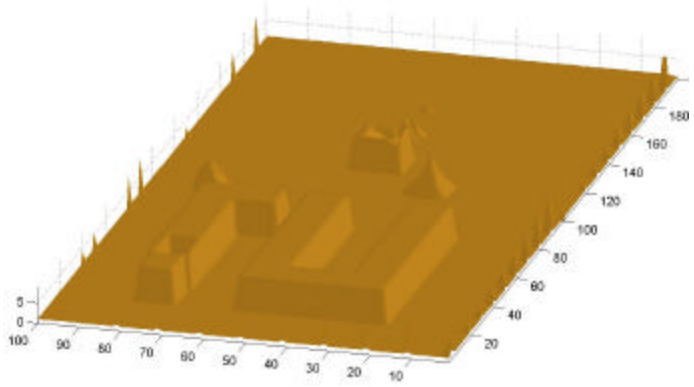

Figure 24. Visualization of the terrain used for experiment set 2.

Table 9. Target locations in Experiment Set 2

Experiment 1	**Location- Geocentric**	**Location- Cartesian (x, y, z)**
Starting point	9? 59’50’’E 20? 0’10’’N	10, 90, 2
Target	9? 59’50’’E 20? 2’10’’N	120, 90, 2
Experiment 2		
Starting point	9? 59’50’’E 20? 0’10’’N	10, 90, 2
Target	9? 58’30’’E 20? 2’0’’N	150, 20, 2

The first experiment has no radar sites and there are 3 radar sites located on the artificial terrain in the second experiment. Their technical specifications are considered to be the same for all the sites. In Figure 25, the locations of the sites are visualized.

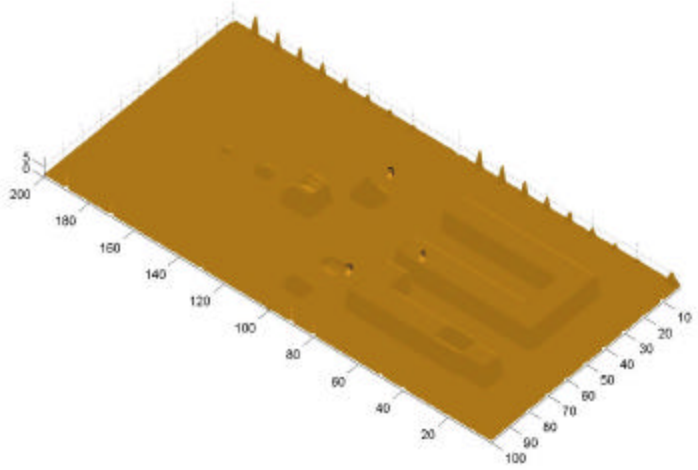

Figure 25. Radar sites of Experiment Set 2

5.2.4. Experiment Set 3

The third set of experiments is designed to evaluate the performance of the PSO_AS algorithm with a single aircraft against a single target and compare its results to an A* implementation that uses the same data set [23]. The same type of aircraft is used, as is the case for the radar sites. Figure 26 shows the radar sites location.

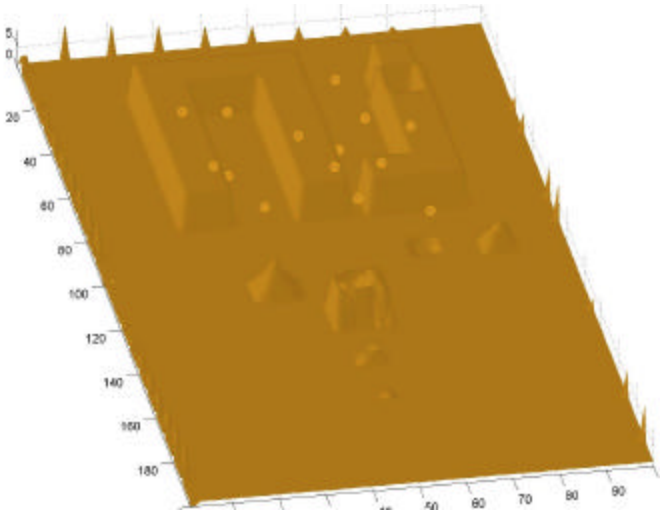

Figure 26. Radar sites of Experiment Set 3

The starting point for the map is chosen to be 75, 15, 2. The target is at location 60, 65, 3. The minimum altitude is 100' and maximum altitude is set to 4000'.

5.2.5. Initial Function Parameters

The initial function parameters for all 3 set of experiments are the same and shown in Table 10.

Table10. Initial Function parameters

Parameter	Value
Target range (HMIN, HMAX, WEST, EAST, NORTH, SOUTH)	0, 0, 0, 0, 0, 0
Max moves per path	500
P1, P2, P3, P4, P5	100, 0, 0, 0, 0

5.3. Benchmark

Even though there is no standard benchmark for 3D MRP, the benchmarks are created to test the results of the algorithm. Besides graphical output of the 3D MRP, charts presenting data are important to understand the pictorial result. Just the plain data does not give the decision maker a good understanding of the result. The mistakes are not easily spotted with some bunch of numbers, so a pictorial result is a great aid for MRP. The visualization of the 3D MRP gives an insight to understanding the result, but it should be supported with some analysis of data. Even though a route may seem better than the other, it may not be. Looking at the data, we can analyze where the route goes through risk areas and how risky the total flight path is.

5.4. Metrics

The metrics that are used in this research are the execution time, and the fitness value of the selected route. This was a minimization problem and the minimum fitness value is wanted.

Execution Time

Execution time is one of the main metrics for evaluation. The main requirement is to reduce the overall execution time to get a reasonable execution time when implementing the program for real world scenarios. The factors affecting execution time are the computation complexity, the search space, and the termination condition.

Fitness Value

The fitness value is the result of the cost function. In this research since risk and distance are the only two important criteria, the fitness value is determined by the equation below.

$$C(s,g) = \sum_{i=1}^{N} c_i = D(s,g) + R(s,g)$$

Fitness (Cost) = Distance of the route + Risk of the route

Solution Variation

One of the ways of determining acceptable solutions desired is to run experiments a sufficient number of times and compare the results. The evolutionary algorithms, since

they are stochastic, can provide different solutions for the problem with the same parameters.

5.5. Parameter Variations

The values of the parameters are very important for evolutionary algorithms. It is like tuning an analog radio to get the clearest sound of music. Obtaining good fitness values with some parameter values does not mean it is going to work the next time. A change in problem domain such as the number of radars, locations of targets, and terrain altitudes is like listening to the radio while driving a car. The parameters must be tuned to find the good solution. Of course tuning the parameters are not as easy as tuning a radio. One has to play with about 20 parameters and the parameters are in interaction with each other. Also it is expected but never known that the algorithm gets an acceptable solution. In this research the parameters can be divided into 3 groups:

The initial function parameters:

- P1, P2, P3, P4, P5 (Initial Function probabilities)
- Pseudo target range
- Number of targets
- Target locations
- Target ranges (east, west, north, south, minimum height, maximum height)
- Map length, width, height
- Number of radars
- Radar locations
- Radar function parameters

2D MRP Parameters [22]:

- Global, local, personal trust
- Population size
- Neighborhood size
- Number of runs
- Rehope2, rehope3
- Movetype
- Moveheuristic

3D MRP Parameters:

- Population size
- Neighborhood size
- Global, local, personal trust
- No improvement (Terminating condition)
- Max number of moves per path
- Number of runs
- Pseudo target range
- Target ranges (east, west, north, south, minimum height, maximum height)

5.6. Number of Tests

Replications are obtained by repeating the simulation with a different seed value. This method is based on the assumption that the means of independent replications are independent even though observations in a single replication are correlated.

The method consists of conducting m replications of size $n + n_0$ each, where n_0 is the length of the transient phase. The first n_0 observations of each replication are discarded. Further, the confidence interval width is inversely proportional to $\sqrt{mn}$. Thus, a narrower confidence can be obtained equally well by increasing either m or n. However, to reduce waste (mn_0 initial observations), it is suggested that m be kept fairly small, for example 10. The length of the replications n should be increased to obtain the desired confidence [14].

In this research, the number of tests chosen are related to the number of parameters and how many different values each parameter can assume. Since it is impossible to do the number of tests for all the values of all the parameters, some vital values that are the breakpoints for those parameters are chosen. With the values that are important for the parameters, the experimental testing is done. The number of runs for each experiment is 10. Since the results of the experiments are consistent and the distribution of the values of the results is not skewed, 10 runs per experiment satisfies the requirements.

5.7. Computational Platform

Two different platforms have been used for testing purposes in [1]. The platforms used in [1] were clusters of workstations and PCs. The workstation was Sun Solaris with Ethernet and Myrinet connection and PCs with Linux operating system. This research is tested on a single PC with Windows operating system. The PC used for the experiments is a Pentium III 1 GHz. Visual C++ 6.0 is the language of the code. C++ is preferred to Java since it is faster and low execution time is required in MRP. The graphical menu for easy input of the parameters has been prepared with Visual Basic 6.0. Visual Basic is preferred due to its easy use in preparing menus.

5.8 Summary

This chapter presented the design of experiments, the benchmark, the metrics used to evaluate the performance, input data necessary to run the program, the parameter variations, hypothesis, and the number of tests necessary.

6. ANALYSIS OF EXPERIMENTS

6.1. Introduction

This chapter examines the results of the MRP experiments. The results of three different experiment sets are evaluated and detailed explanation of the results is provided. In experiment set 1, the effects of PSO_AS parameters on MRP are discussed. In experiment set 2, the conditions for the improvement of the PSO_AS over the initial function are stated. In experiment set 3, an explicit comparison between the results of this research and a previous research [23] is made.

6.2. Statistical Analysis of Experiment Set 1

6.2.1. Analysis of Effects of PSO Parameters on Fitness Values

The average fitness values produced by the initial and the PSO function are shown in Figure 27:

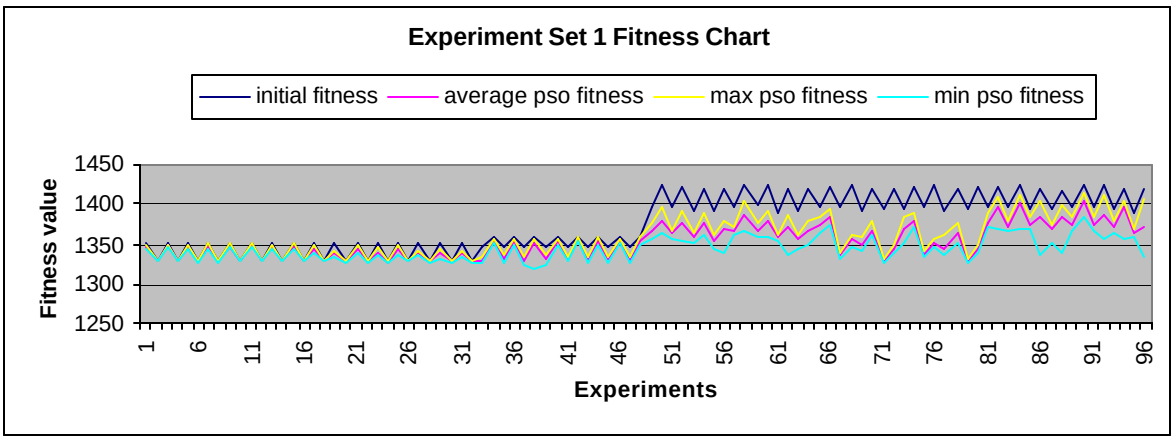

Figure 27. Comparison of Initial Function and PSO Results

The fitness values on Figure 27 are the average fitness values found by both the initial and the PSO_AS function at the end of each experiment. The minimum and the maximum fitness values are also displayed to give a better understanding of the results produced by the PSO_AS function. The initial fitness values are the result of the initial function or the fitness values produced at the end of the initialization phase. The cost function in MRP is a minimization function and the PSO_AS tries to improve (minimize) the fitness value produced by the initial function. Every experiment is given different parameter values. This provides the ability to learn with which EA parameter values better results are obtained for this specific problem. Every experiment, 0 to 8 percent improvement has been gained by the PSO_AS function compared to the initial function. The average fitness value as the result of 960 runs of the initial function is 1375.489. The average fitness value of 960 runs of the PSO function is 1352.716. The overall average improvement gained by PSO function is % 0.6. The Euclidean distance of the route is 1181.206. But this fitness value is the result without accounting for radar risk and terrain.

6.2.1.1. Pseudo Target Range and Move Probabilities (PSO P1, P2, P3, P4, P5)

As explained in Section 4.2, before moving towards the target, the particles first move towards one of the pseudo targets. After getting within specified range of the pseudo target, the particles then move towards the target. This is one of the initial function parameter values that affect the initial fitness values. The range of a pseudo target is found by dividing the map height, width, and the length by the pseudo target range value. The bigger the parameter value for the pseudo target range, the smaller the range of the pseudo target is. In 64 of 96 experiments, the pseudo target range value of 10 has produced better results for the initial function. It can be concluded that getting closer to the pseudo targets before going to the real target gives better results for this specific

problem. Getting closer to radars while building a path increases the fitness value so a larger pseudo target range is better for this specific problem. When the initial function probabilities P1, P2, P3, P4, P5 are consecutively 100, 0, 0, 0, and 0, the route is built by choosing the best of the next 17 grid points at every grid point. In this case since the route is always the same, all the experiments with initial probabilities of 100, 0, 0, 0, 0 must result in better solutions with either pseudo target range value of 10 or 5. For this to happen, the order of the targets has to be the same in all the experiments. Otherwise, it may not be so. The order of the targets is different in 32 of the 48 experiments with initial probabilities of 100, 0, 0, 0, 0 and this enables initial fitness values to be different. Looking at Figure 27 it is easily seen that for the first 32 experiments, the initial function values are different than the experiments from 33 to 48. For the experiments from 49 to 96, where the initial function probabilities are 90, 4, 2, 2, 2, the order of the targets are the same. A little disorder in the initial fitness values can be seen in Figure 27 from experiments 49 to 96. This is due to 4 % probability of the particle choosing the 2^{nd} best, 2 % probability of choosing the 3^{rd} best, 2 % probability of choosing the 4^{th} best, and 2 % probability of choosing the 5^{th} best instead of the best among the 17 possible grid points. It is seen that for this data set at the end of the initial function, the particles that got closer to the pseudo targets before moving towards the real target gave better results in 64 of the 96 experiments. This can be better understood by looking at the initial fitness values in Figure 28. Even though the pseudo targets have a similar effect on PSO fitness values, the same conclusion cannot be made. The reason for the similar effect is that the particles start the PSO function with the fitness values produced by the initial function. They try to improve these fitness values by the interaction among the particles of the swarm. The interaction may enable a particle's higher initial fitness value than the other particles' initial fitness values to be lower than the other particles' fitness values at the end of the PSO function. This can be better seen in Figure 29. This is not the case in most

experiments. Better start with the initial function usually ends with a better result of PSO_AS function.

With Figures 28 and 29 a better analysis of the pseudo target range effect on initial function results can be made:

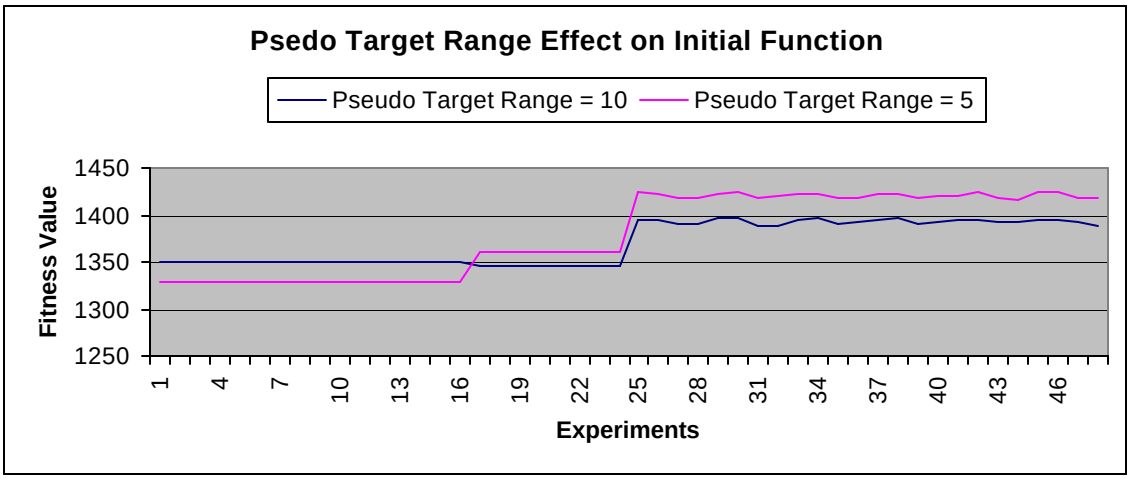

Figure 28. Pseudo Target Range Effect on Initial Function

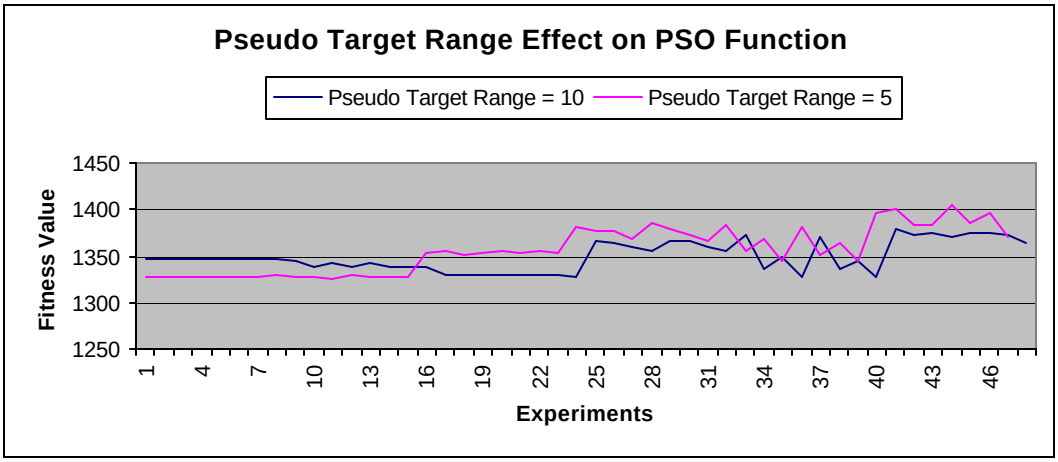

Figure 29. Pseudo Target Range Effect on PSO Function

Since the experiments are conducted 16 at a time, the order of targets is not the same in all the experiments. By looking at the results of the first 24 experiments in Figure 28 it is clearly seen. The order of the targets in the first 16 experiments is different than the target order in the experiments from 17 to 24. This causes the initial function to produce different fitness values. By comparing the results of the initial and PSO_AS on Figures 28 and 29 it is understood that most of the time a good start with initial function ends with a better result of the PSO function. The average of 48 experiments with pseudo target range value of 10 is 1349.293 and the average of 48 experiments with pseudo target range value of 5 is 1356.140. As explained before, these average results of two pseudo target ranges depend on the problem and cannot be generalized.

In Figure 30 the effect of move probabilities on Initial function is shown:

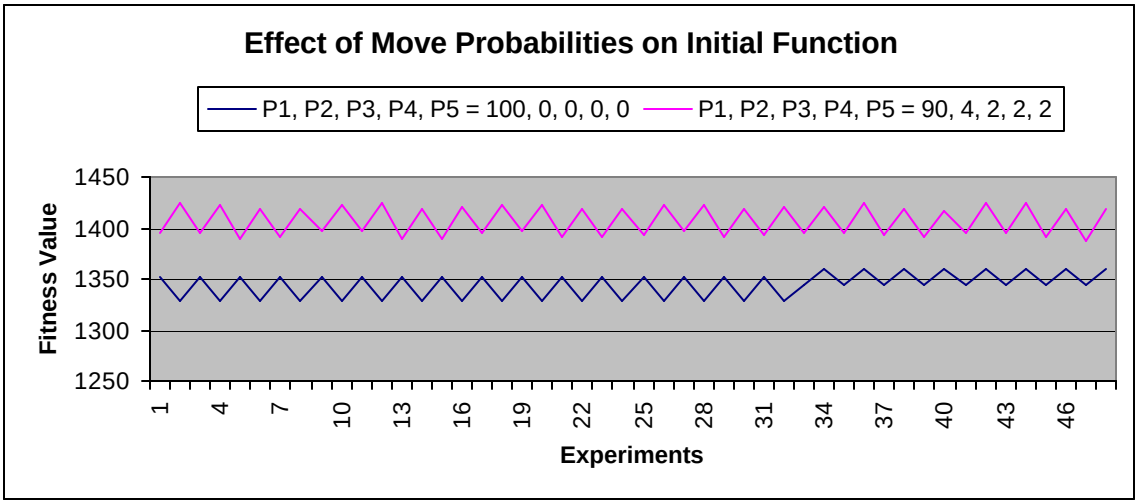

Figure 30. Effect of Move Probabilities on Initial Function

On Figure 31 the effect of move probabilities on the PSO Function is shown:

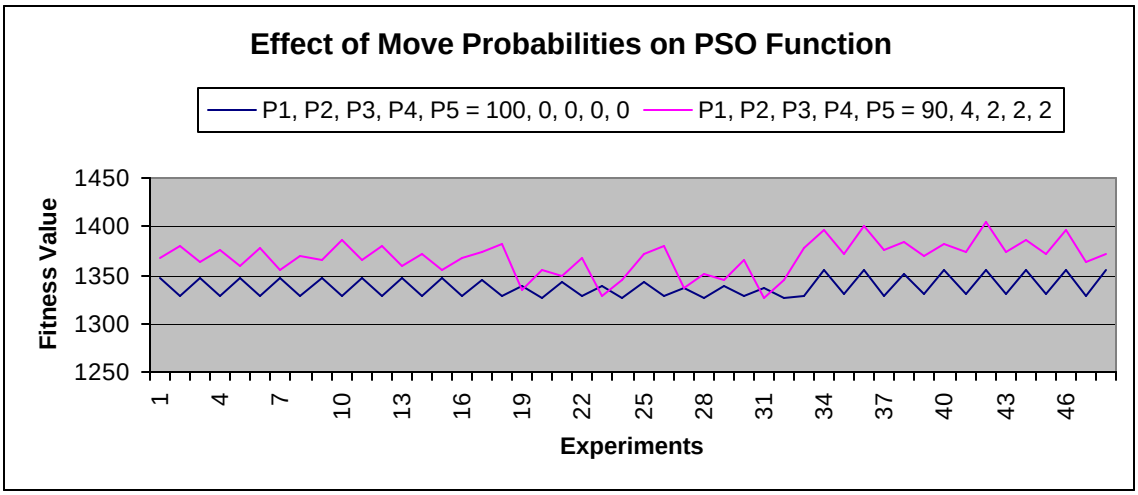

Figure 31. Effect of Move Probabilities on PSO Function

The average of 48 experiments with move probabilities of 100, 0, 0, 0, 0 is 1337.686523 and the average of 48 experiments with move probabilities of 90, 4, 2, 2, 2 is 1367.746908. As seen on Figure 30 for this specific problem, all of the fitness values produced at the initial function with higher probability of choosing the best next grid point were better. And in Figure 31 most of the fitness values produced by the PSO function with higher probability of choosing the best next grid point produces better results. It is concluded that for this specific problem, most of the time, the results produced with higher probability of choosing the best next grid point are better.

6.2.1.2. Global, Local, and Personal Trust

In Table 11, the trust values for experiment set1 are shown:

Table 11. Trust Values for Experiment Set1

Parameter/ Experiment	1-16	17-32	33-48	49-64	65-80	81-96
Global Trust	50	10	80	50	10	80
Local Trust	30	10	10	30	10	10
Personal Trust	20	80	10	20	80	10

The trust values do not affect the initial function results. The trust values show how much the particles trust their global, local, and personal best. A particle that trusts another particle gets closer to it in the fitness landscape. This is why trust is an important factor in the interaction of the particles. In Figure 32, the effect of trust values on PSO the function is shown:

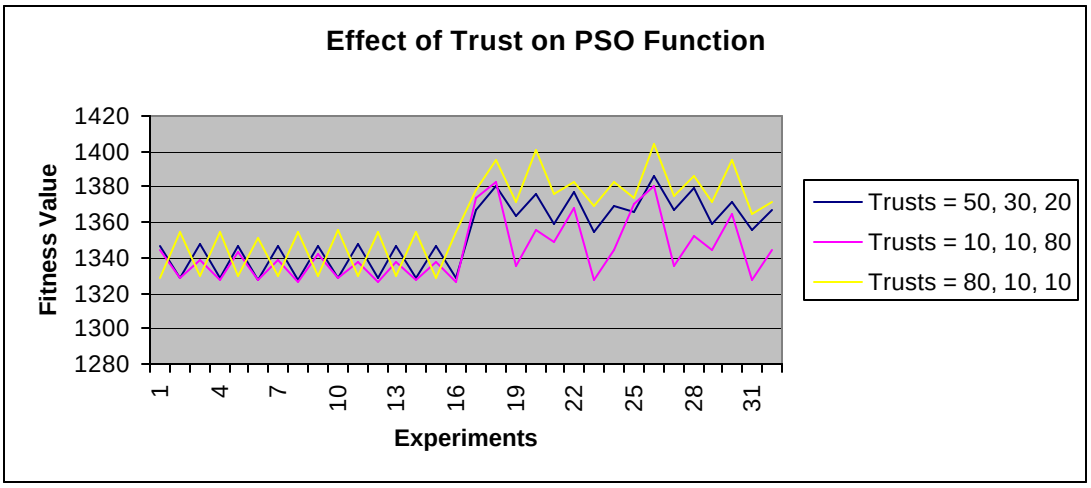

Figure 32. Effect of Trust on PSO Function

In Figure 32, the first 16 experiments for 3 trust values correspond to experiments 1-48 and the experiments from 17-32 correspond to experiments 49-96. In the first 16 experiments, the move probabilities are 100, 0, 0, 0, 0. In the second 16 experiments, the move probabilities are 90, 4, 2, 2, 2. The effect of move probabilities can be seen better on Figure 37, where most of the final PSO function results are better with move probabilities of 100, 0, 0, 0, 0 due to a good start of PSO. In the first 16 experiments, the particles had a good start, so the PSO does not get to improve the routes compared to the second set of 16 experiments. In the second set of 16 experiments, the trust values of 10, 10, and 80 have made the best improvement to the initial function. This is due to late

convergence of the particles. The late convergence of the particles enables better exploration of the search space. It is also seen that with the same trust values some improvements are much better. The termination condition plays an important role here. Since the particles converge late, if we do not give them enough time or number of loops to make an improvement and terminate the program, we may prevent the particles from finding a better local optimal or the global optimal in the fitness landscape. The effect of the termination condition is presented in more detail in the following sections. The average of 32 experiments with trusts of 10, 10, 80 is 1343.566, the average of 32 experiments with trusts of 50, 30, 20 is 1353.074, and the average of 32 experiments with trusts of 80, 10, 10 average is 1361.509. With these results and also by looking at Figure 32 it is concluded that for this specific problem the lower trust in the global best and the local best results in better fitness values.

6.2.1.3. Termination Condition

The termination condition and the trust values are related to each other. While the particles are interacting with each other and looking for better fitness values in the fitness landscape, the trust values play an important role. A late convergence is desired for better exploration of the fitness landscape, but late convergence may be unnecessary depending on the type of fitness landscape. While the particles look for the better fitness values, stopping them without allowing the particles to converge at a point may cause them to not find a better fitness value. Figure 33 displays a better evaluation of the effect of the termination condition on the PSO function:

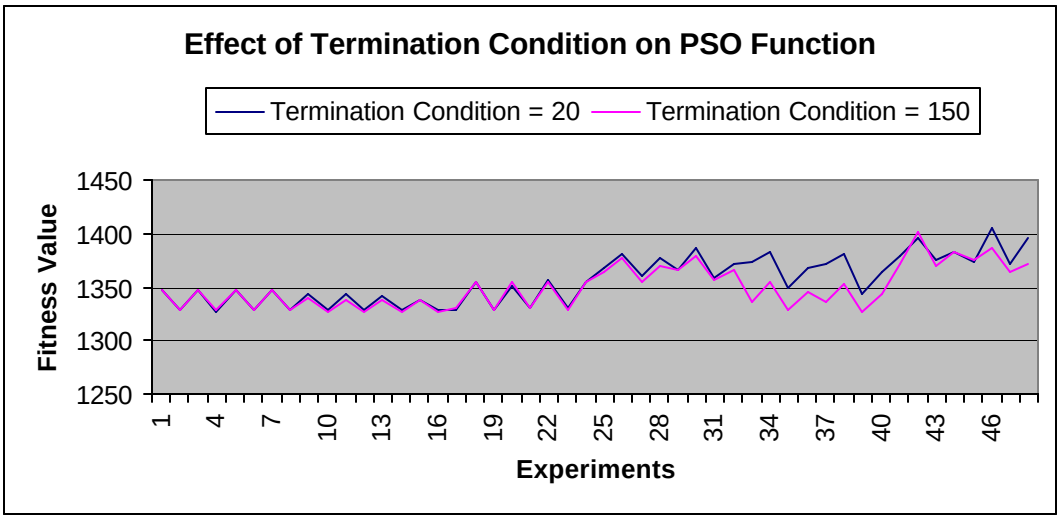

Figure 33. Effect of Termination Condition on PSO Function

It is clear that the termination condition of 150 provides better fitness values than the termination condition of 20. If the initial function produces good fitness values to start with, the PSO function does not improve the fitness value as much. On Figure 33 for the first set of 24 experiments, the initial function produces good solutions. The PSO function improves it little or none. On the second set of experiments, where the initial function produces higher values, the PSO function does a better job of improving the fitness values. As explained before with the termination condition of 20, the particles have to stop searching the fitness landscape before getting a chance to find better fitness values.

6.2.1.4. Neighborhood Size

The neighborhood size is one of the EA parameters that can affect the results provided by the program. In Figure 34, the effect of neighborhood size 5 and 10 on the PSO function is clearly expressed:

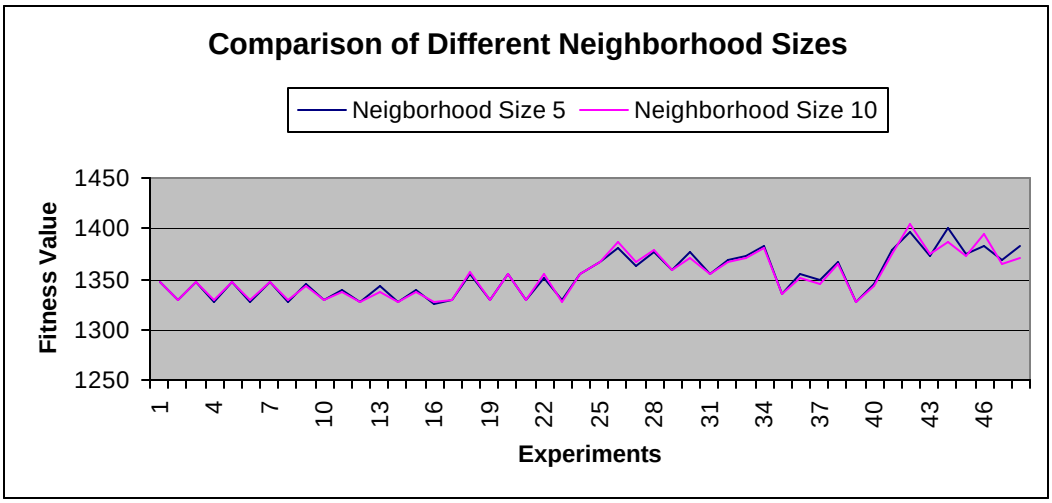

Figure 34. Effect of Neighborhood Size 5 and 10 on PSO Function

There is not a clear difference between neighborhood sizes of 5 and 10 on the PSO function. Overall the neighborhood size of 10 seems a little better but not enough to state that it is better. For this specific problem the neighborhood size does not affect the results which is also the case most of the time in general. The average fitness value of 48 experiments with neighborhood size of 5 is 1353.064 and the average fitness value of 48 experiments with neighborhood size of 10 is 1352.369.

6.2.1.5. Population Size

At the beginning of the algorithm the population size is n. Because of 20 pseudo targets the population size becomes 21n at the beginning of the PSO function. Every particle builds 21 paths using 20 pseudo targets and 1 without using a pseudo target by just using the actual target. At the beginning of the PSO function, all 21 paths built by a particle are accepted as separate particles each so that the population size becomes 21n. The population sizes 5 and 10 actually are 105 and 210. An alternative to this is only taking some of the particles into the swarm and ignoring the others. This is an

implementation issue that the designer should decide considering the problem domain. In Figure 35, the fitness values of population sizes 5 and 10 are presented:

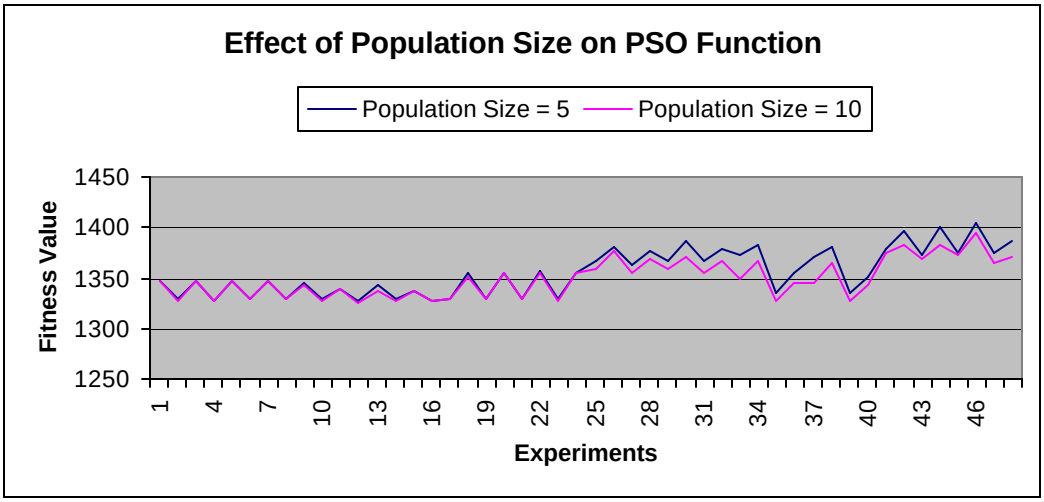

Figure 35. Effect of Population Size on PSO Function

On the second set of experiments (24-48) where the move probabilities are 90, 4, 2, 2, 2 the population size of 10 makes more improvement to initial fitness values. And overall the results with population size of 10 are better than 5. This is due to having more particles in the interaction. In general for EA algorithms, a limit on the population size must be specified because increases in the population size increase the computation time. Instead of increasing the population size, increasing the time limit or the number of iterations of termination condition may provide better solutions.

6.2.2. Analysis of Effects of PSO Parameters on Execution Time

The average execution times of 96 experiments are shown on Figure 36:

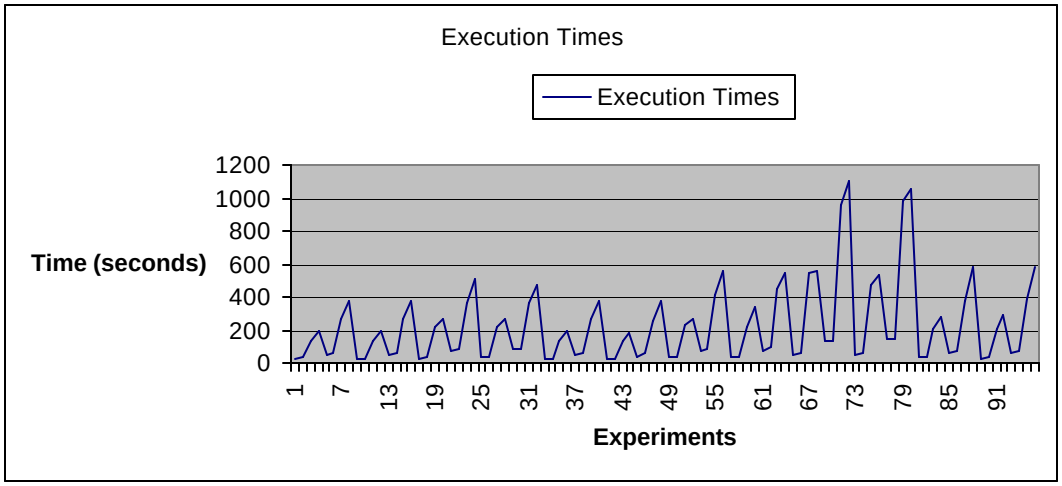

Figure 36. Average Execution Times

In Section 6.2.1 the statistical analysis of the fitness values is presented. For each experiment, since the initial function took very little time, only average total execution time of each experiment is taken into consideration.

The 96 experiments have been divided in 6 and the experiments were executed 16 at a time. For each portion of 96 experiments, which is once every 16 experiments, building the routes and determining the order of targets was made only one time. The same order of targets was used for each group of 16 experiments. This is why the order of the targets happened to be different on some of the experiments. The routes between every target including the starting point are built to determine the distances between every target including the starting point. For this data set, since there are 3 targets, a total of 4*3 / 2 = 6 routes are built. From previous experiments, it has been observed that for a problem with 99 targets it took approximately 5 seconds to build 99*100/2 = 4950 routes. For this problem, it took less than a second to build six routes. After determining the

distances between every target including the starting point, the distances are put in a distance matrix so that the best order of targets can be determined. With the distance matrix, the problem becomes a TSP. Using the code from a previous research [22], which again uses PSO_AS algorithm, the best order of targets is determined. Building the routes and determining the order of targets for experiment set 1 takes approximately 5 seconds. In Table 12, the total amount of time it takes to determine the order of targets is shown for all the experiments:

Table 12. Execution Times for Determining The Order of Targets

Experiment	**1-16**	**17-32**	**33-48**	**49-64**	**65-80**	**81-96**
Time (seconds)	6s	6s	5s	6s	6s	5s

After determining the order of targets, the initial function starts. As stated above, the initial function takes very little of the overall time (less than a second), so it does not affect the total execution time.

6.2.2.1. Pseudo Target Range

In Figure 37, the execution times for pseudo target range of 5 and 10 are compared:

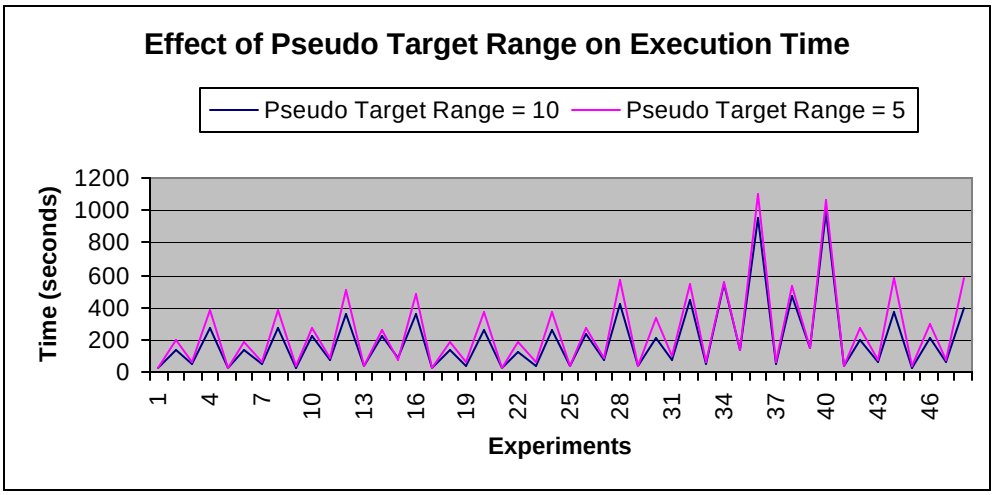

Figure 37. Effect of Pseudo Target Range on Execution Time

As explained before, the particles first move towards the pseudo targets before moving to the real target. The larger the pseudo target range value, the closer the particles get to the pseudo targets before moving to the real target. This can be better or worse in terms of both fitness values and execution times depending on the data set. In Figure 37, a better execution time is achieved with a pseudo target range value of 10.

6.2.2.2. Move Probabilities (PSO P1, P2, P3, P4, P5)

In Figure 38, the execution times for move probabilities of 100, 0, 0, 0, 0 and 90, 4, 2, 2, 2 are compared:

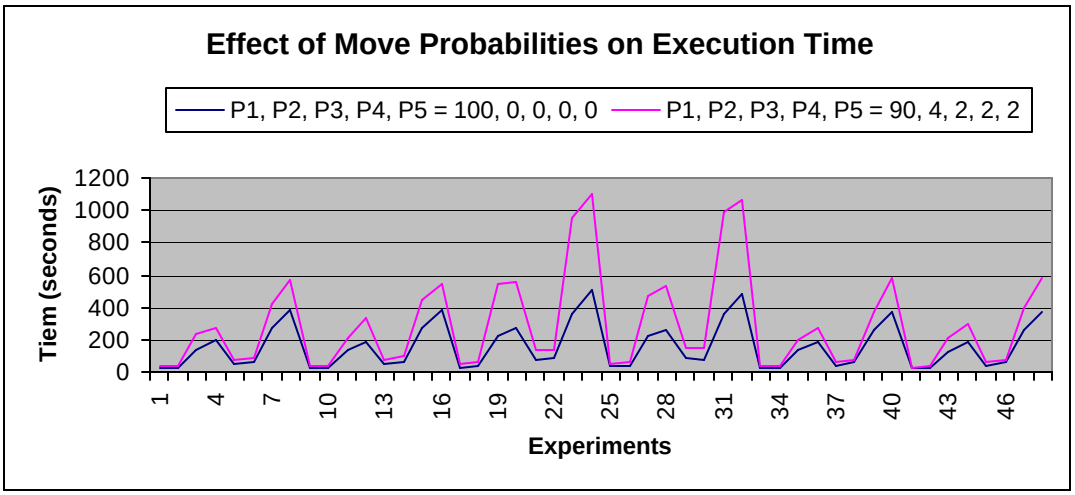

Figure 38. Effect of Move Probabilities on Execution Time

As seen on Figure 38, the PSO function cannot add much to the initial fitness values if the initial function produces good results. This causes the algorithm to terminate more quickly with less execution time. Since move probabilities of 100, 0, 0, 0, 0 produce better results in the initial function, an acceptable solution is reached more quickly and the algorithm ends quicker for this specific problem.

6.2.2.3. Global, Local, and Personal Trust

In Figure 39, the execution times for three different trust values are compared:

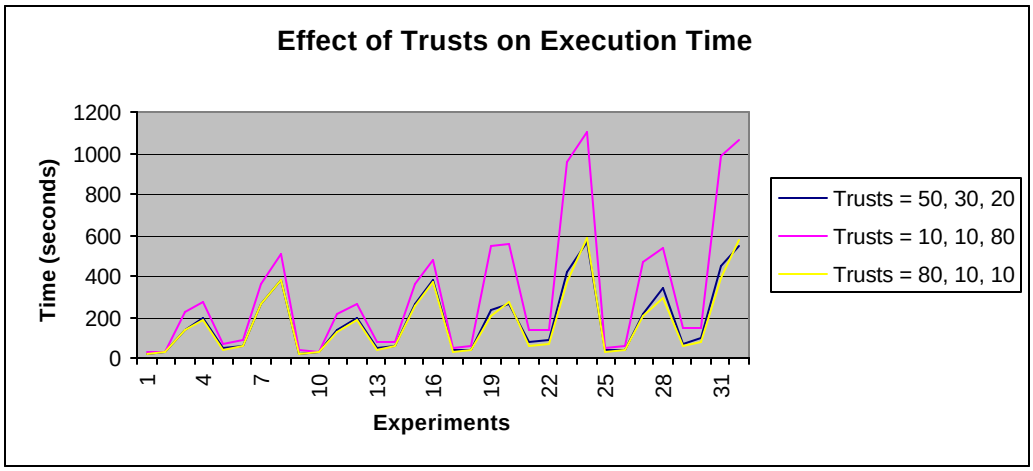

Figure 39. Effect of Trusts on Execution Time

The execution times for trusts of 50, 30, 20 and 80, 10, 10 are the same. The reason of high execution time for the trusts of 10, 10, 80 is the late convergence of the particles. Late convergence enables the particles to make a better exploration of the search space such that improvements are gained even in the later iterations of the algorithm.

6.2.2.4. Termination Condition

In Figure 40, the effect of termination conditions of 20 and 150 on execution time is compared:

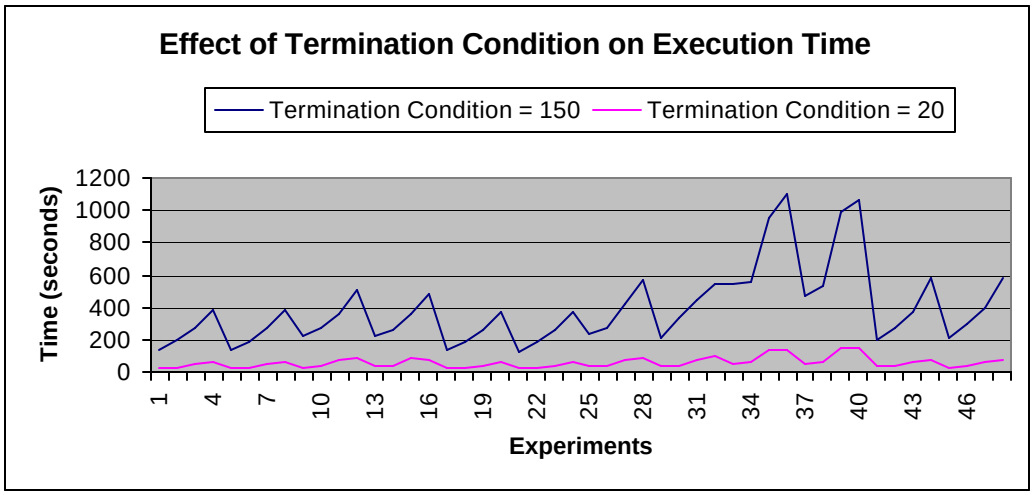

Figure 40. Effect of Termination Condition on Execution Time

The termination condition plays the most important role in the execution time. In the first set of 24 experiments where the initial function produces good fitness values, the PSO function does not improve the fitness values as much, so the algorithm ends more quickly compared to second set of 24 experiments where the initial fitness values are not as good as the first set. By looking at Figure 40, it can be concluded that the execution times for termination condition 150 are 5 to 10 times longer than the execution time for a termination condition 20. Looking at the improvement of PSO function in the fitness values with termination condition of 150, it is up to the user to decide if an 8% improvement is worth waiting 10 times as long.

6.2.2.5. Neighborhood Size

In Figure 41, the effect of neighborhood sizes of 5 and 10 on execution time is compared:

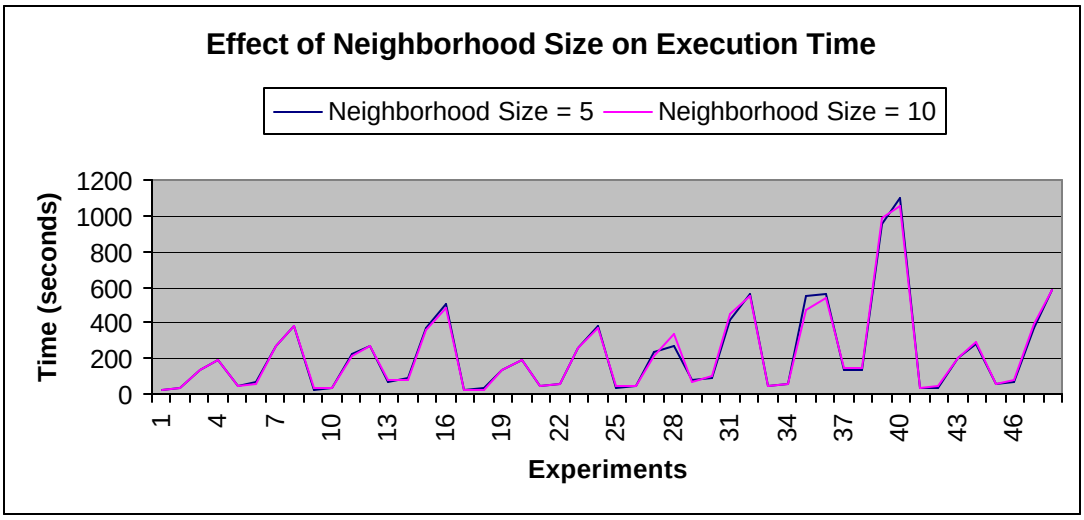

Figure 41. Effect of Neighborhood Size on Execution Time

The difference in the neighborhood size does not affect the execution time as seen in Figure 41.

6.2.2.6. Population Size

In Figure 42, the effect of population sizes of 5 and 10 on execution time is compared:

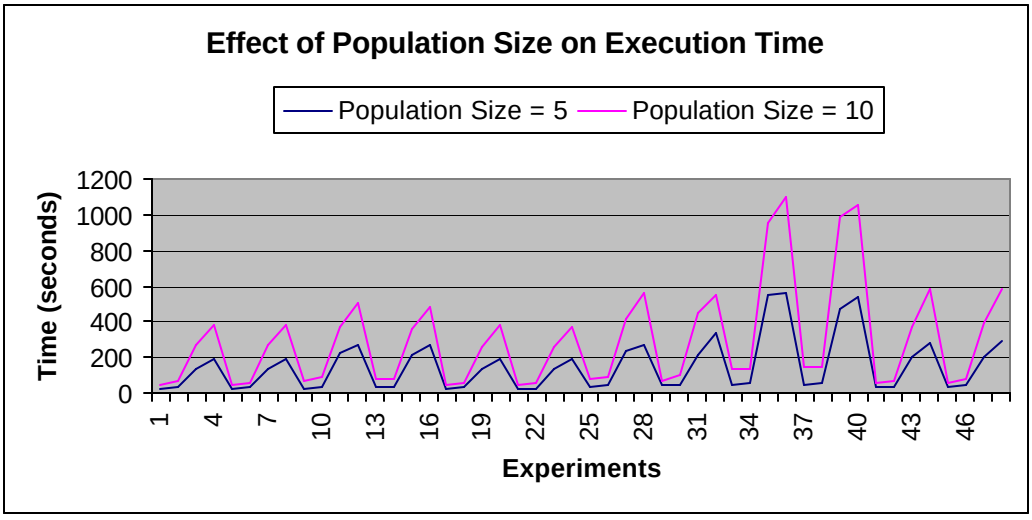

Figure 42. Effect of Population Size on Execution Time

The population size is one of the more important factors that increase the computation time. Doubling the population size doubles the number of particles that interact with each other. The execution times of population size 10 is twice as much as the execution times of population size 5 for this specific problem. In general, there is not a linear relationship between the population size and the execution time.

6.3. Statistical Analysis of Experiment Set 2

6.3.1. Experiment 1

The first experiment of the second experiment set is implemented using a single target and no radars. Since the target is located at a point where it is easy for the initial function to build an optimal path to, there is no need for PSO in trying to optimize the path. In Figure 43, the path built by the initial function is shown:

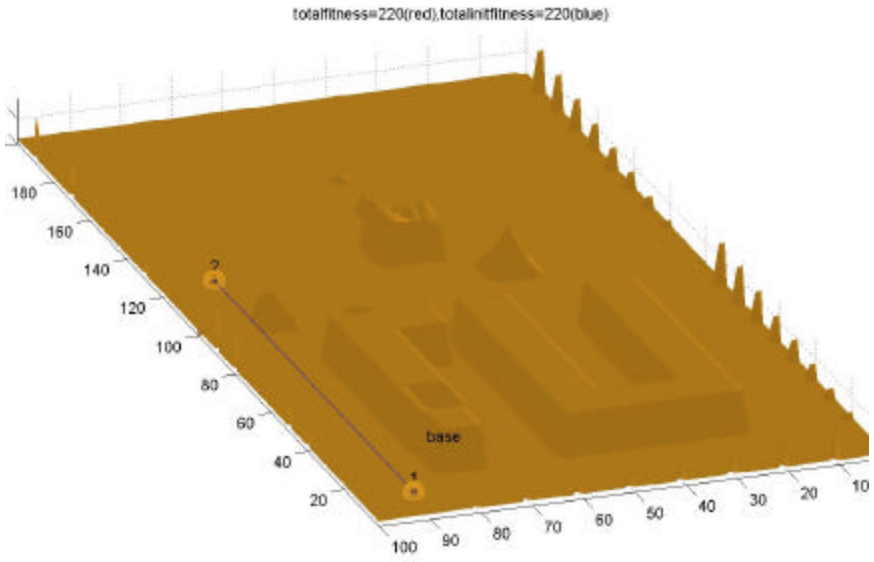

Figure 43. Path Built by Initial Function for Single Target

The path drawn by the initial function is not optimized by the PSO function since the initial function already found the optimal path. The initial fitness value at the end of initial function is 220. The Euclidean distance between the starting point and the target is 110. The UAV chooses the same path to go back to the starting point. The fitness value found by PSO function is also 220. It is not possible to see the path built by PSO since it is the same path. The execution times and fitness values for the initial and PSO function are as shown in Table 13:

Table 13. Comparison of Initial and PSO Function for Experiment 1

Metrics	Initial Function	PSO Function
Fitness Value	220	220
Execution Time	1s	43s

6.3.2. Experiment 2

In the second experiment, the starting point is the same as the first one. There is only one target, which is located farther from the starting point as compared to the first experiment. Three radar sites are located in between the starting point and the target. As compared to the first experiment, there are more constraints such as the radars and the mountainous area that forms a block in between the base and the target. The path built by the initial function and the PSO function are as shown below in Figure 44:

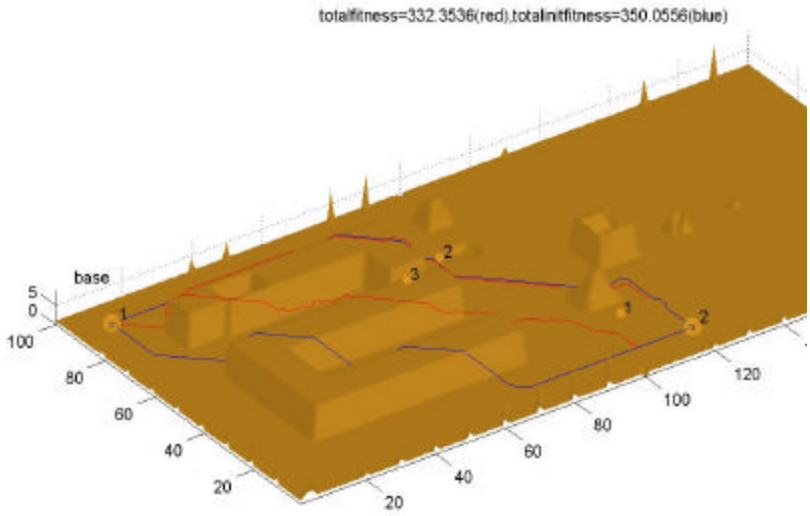

Figure 44. Optimized path with PSO_AS

In Figure 44, there are four different paths between the target and the starting point. The two paths in blue are the paths built from the starting to the target and back to the starting point. The red paths are the optimized paths by the PSO_AS. The initial fitness value at the end of initial function is 350. The fitness value found by PSO function is 332. Compared to the first experiment, the constraints in the second experiment between the starting point and the radar prevent the initial function to build a good path. This is where the PSO_AS function really works and optimizes the path built by the initial function. The execution times and fitness values for the initial and PSO function are as shown in Table 14:

Table 14. Comparison of Initial and PSO Function for Experiment 2

Metrics	Initial Function	PSO Function
Fitness Value	350	332
Execution Time	1s	210s

6.4. Statistical Analysis of Experiment Set 3

In the third experiment set, the PSO_AS is compared to a parallel A* implementation [23]. Even though the data set is the same, different approaches to the problem and different platforms used does not allow a fair comparison to be made. As a nature of the PSO_AS algorithm, it searches for the optimal solution in the fitness landscape while the A* algorithm searches the physical landscape. The PSO_AS algorithm needs initial solutions to be produced in order to start its stochastic search. So the PSO_AS implementation on MRP actually has two solutions. The first is the solution produced by the initial function, which has very low execution times since it is a best first search. The initial function is actually the initialization phase that provides the PSO_AS the necessary fitness values it needs in order to start its own search. The PSO_AS takes more time compared to the initial function. The amount of improvement the PSO_AS makes over the initial function depends on the problem constraints. It should also be considered that the PSO_AS implementation builds paths which return back to the starting point while the A* implementation does not do this. The A* implementation uses masking while the PSO_AS does not. For the data set provided in experiment set 3, the execution times are shown in Table 15.

Table 15. Execution Times in Experiment Set 3

	Parallel A* (2 processor)	PSO_AS	Initial Function
Execution time	23s	200s	1s
Fitness value	-	139	142

The optimality of the solutions can be compared using Figures 45 and 46.

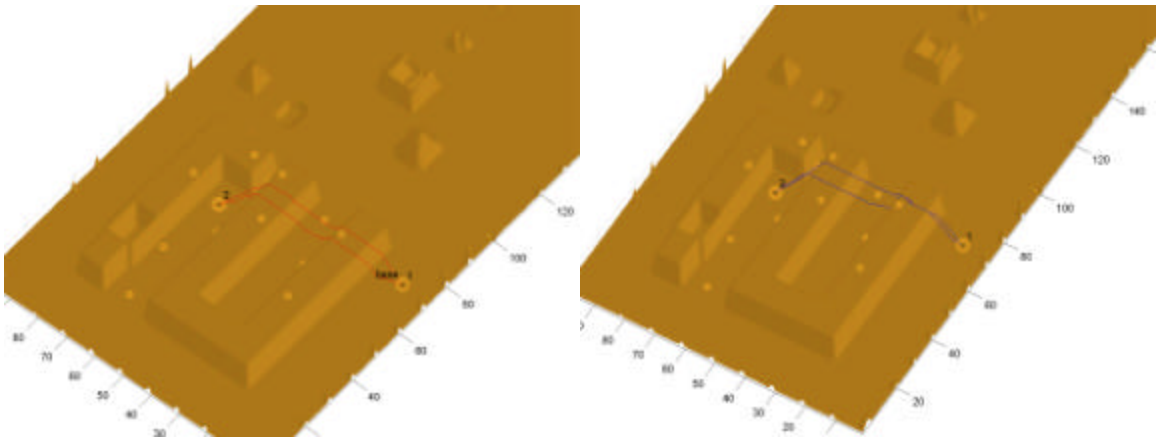

Figure 45. Paths produced by both the initial (right) and the PSO function (left)

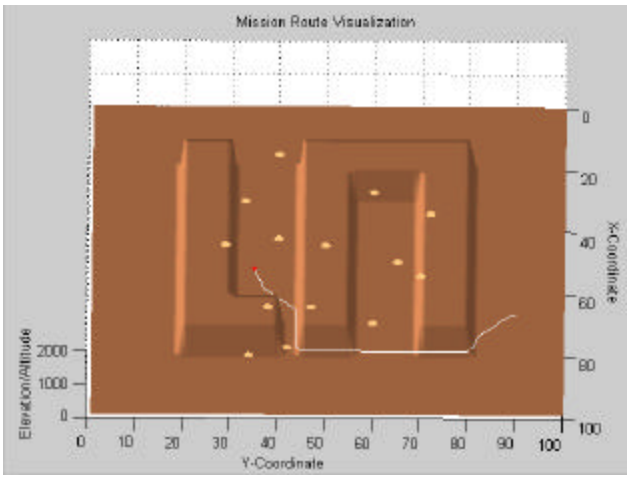

Figure 46. Path produced by the A* implementation [1]

6.5. Summary

In Chapter 5 the design of experiments is explained in detail. First, the reason for the experiments is indicated. In the following sections, the experiments are discussed.

7. CONCLUSION AND RECOMMENDATIONS

7.1. Conclusion

The goal of this research effort is to develop and evaluate an aggregated multi-objective MRP system using the PSO_AS algorithm. This goal is achieved through the design, implementation, extensive testing, and analysis of a functional MRP system. All of the objectives presented in Section 1.2 are achieved in this research. The objectives associated with the goal and stated in Section 1.2 are:

- Integrating the PSO_AS to MRP by extending a previous PSO_AS effort
- Finding the shortest path, minimizing the flight time, minimizing the probability of being shot down, and consuming less fuel.
- Making radar sites more reflective of real world situations.
- Designing a map geocentric coordinate interface and integrating it with an existing visualization system.
- Increasing the number of targets
- Designing a set of experiments that allow testing and evaluation of the performance of the designed MRP algorithm.

The previous PSO_AS effort for TSP is extended by applying PSO_AS to MRP [22]. The PSO_AS for TSP is also combined with the PSO_AS for MRP in helping to find the best order of targets. This integration also helps meet another objective, increasing the number of targets. The number of targets can be increased to a very large number. This is a plus to the A* implementation for multi-target MRP where only two targets are considered [23]. Another improvement to previous A* implementation for MRP is the addition of the Bistatic radar equation into the monostatic radar model.. Section 2.3.2 provides detailed information about both the Bistatic and the Monostatic

radars. This radar model improvement makes the radar sites more reflective of real world situations. Mentioned in Section 3.5, a mapping from geocentric to Cartesian format is made possible by the graphical interface. In Chapter 5 three sets of experiments are designed to validate the effective functionality of the PSO_AS algorithm. The analysis of the results is made in Chapter 6 by using the metrics explained in Section 5.3.

Various design issues are studied and associated MRP implementations are improved. The algorithm is tested on real world data. One of the terrain data used is the middle part of Turkey. The elevation data is provided from a website [27]. By importing real world data, benchmarks that can be used as standard for future efforts are created. Three different maps of which two are real world terrain are created. Finding optimal flight paths in real world terrain is easier than in the artificially generated terrain. Since the elevation in real terrain is not as high as the artificially generated, it is easier for the particles of the swarm to build an optimal flight path with very little terrain constraint. Building a total flight path by visiting all the targets and returning is made possible. PSO_AS implementation on TSP is integrated into the design and testing to help build a total flight path with many targets.

With so many parameters, designing a graphical interface is a must. Otherwise, entering the inputs takes significant time of the planner. A graphical interface is also another addition to previous MRP efforts. Both the Geocentric and the Cartesian format are used and the conversion between them is made possible by the graphical interface. These two formats are used due to their simplicity. Section 3.5 explains the details of the graphical interface.

Visualization is an important part of MRP efforts. The planner cannot make the decisions without visualization of the flight path, radar sites, and the terrain. The visualization is designed using the Mat lab code in Appendix C. Mat lab has plots where zooming, and rotation of the visualized 3D terrain is possible. This gives the planner the

capability of looking at the flight path from all angles and makes better analysis of the flight path.

A set of experiments and metrics are defined. The first experiment set presents the importance of parameter variations in PSO_AS algorithm. The results of the first experiment reveal that the initial function finds acceptable solutions since initial function is similar to best first search. In best first search, the particle always chooses the best next path. In the initial function if the best next path is given a 100 % probability of being selected, then the initial function is the same as best first search. It is also seen that the PSO_AS algorithm makes improvements to the results of the initial function. The results are compared using the average and also showing the maximum and minimum values. The amount of improvements varied depending on the different PSO_AS parameter values.

The results of the second experiment set shows that the PSO_AS algorithm is more effective on MRP with more constraints. In the first experiment, where the target is located at an easy to find location, both the initial function and the PSO_AS algorithm find the optimal solution. In the second experiment with more constraints the PSO_AS improves the solution of the initial function.

In the third experiment, comparisons between the previous A* effort and PSO_AS are made. The optimality of the flight paths can be compared only by visualization since A* implementation does not provide fitness values of the paths. Since A* is a deterministic algorithm, the terrain used has to be small to find an optimal solution [23]. This is due to the increase of the search space. This increase does not affect the PSO_AS algorithm since it is stochastic and good solutions can be found in a bigger search space. The real world test terrain in the first experiment is 16 times bigger than the biggest test terrain used in A* implementation [23]. The execution time for the A* implementation for a single target is 23s and it is less than PSO_AS which is shown on Section 6.3.2. In

the implementation for two targets, the execution time of A* increases to 750s where the PSO_AS as shown on the first experiment set have lower execution times for even 6 targets. The execution times are 10s to 600s depending on the parameter values. So when it comes finding paths to many targets, the PSO_AS is much faster. In experiment set 3, the results for single aircraft against six targets showed that the optimal routes are found. The initial function found a solution within 1s. Creating the cost matrix and determining the order of targets took approximately 5s.

The PSO_AS is considered more of a UAV type application. When a surveillance mission is assigned for a UAV to take pictures of many targets the PSO_AS implementation for MRP provides a good flight path considering the risk and the distance. The A* implementation however only provides path to a single target. When two targets come into play, the A* divides the aircrafts into two and assigns different aircrafts to different targets, so it is not really a multi-target implementation like the PSO_AS where a good visiting order of targets and a safe return is provided.

7.2. Recommendations for Future Research

Research is a continuous process. As seen with most of the research investigations, the work is built upon previous efforts which are built upon another.

One of the improvements to this research could include adding multiple aircrafts. But this addition should not be like the multi-aircraft A* implementation. In A* implementation, only two targets and two aircrafts are specified where one aircraft is sent to a target and the other aircraft is sent to another target [23]. Instead, the best visiting order of targets among the aircrafts where a safe return is possible should be provided. This would add another constraint to the single aircraft MRP.

Time windows can be incorporated into the MRP where the aircraft should be over a target within a given time. This concept should be integrated into the implementation considering the day and night factor, radars that change locations and radars that are active at different times of the day.

To have better exploration of the search space, more pseudo targets between the starting point and the target can be located. Also, instead of a pseudo target an order of pseudo targets can be visited before going to the real target. Another issue for better exploration is to divide the search space into grids and evaluate each grid as the genes of a chromosome that are to be searched. This feature may also increase the efficiency of the code by decreasing the size of the explicit search space.

In the radar model, the probability of failure, probability of being hit by an overflying aircraft, and the probability of being jammed can be added. Some type of masking function can also be added. The Radar Cross Section (RCS) of the aircraft can be made more realistic. Instead of a spherical model in which the aircraft has the same RCS value on each side, more realistic approaches can be investigated. When making these improvements it has to be kept in mind that each of the improvements adds to the complexity of the code. So, there has to be some kind of tradeoff between complexity and performance.

Aircraft data is currently quite simple. Characteristics of UAVs may be simulated with greater accuracy. Maximum turn angle and maximum combat radius can be added to the aircraft model. In order to have a more realistic model, its thrust/lift ratio can be added to its maneuverability calculations. This value can also be used to statistically determine the probability that a UAV can survive a missile shot with its maneuverability.

The UAV could also have different RCS values for different configurations, and for its different position in reference to the radar site.

The algorithm should also take into consideration the fact that there are aircraft types that have very low radar reflections. A route that is not suitable for one aircraft type might be very suitable for another type of aircraft. Another issue is the addition of multiple objectives. A new design may be developed to enhance the existing code with more features.

The algorithm can be parallelized to reduce the execution time. Each processor can act as a particle of the swarm or each processor can be given a certain population size. The processors would then exchange information when better solutions are found. This means the interaction would take place among the processors. The parallel application decreases the computation time.

The algorithm can be improved. As stated in earlier researches, instead of giving one good solution, the algorithm may be changed to give a set of solutions. That approach gives flexibility to the planner according to other dictating criteria. Some other criteria can also be added in the evaluation of the routes such as, weather or fuel constraints. Some statistics can be added in the program. For example, a target might have been destroyed before with some probability, and then the program might take actions and find the second target on its mission from current position.

Memetic algorithms that make local searches in the fitness landscape can be added to the PSO_AS algorithm that searches the fitness landscape globally [28].

The graphical interface can be improved. The user can act with the program execution by clicking the icons or pull down menus. If the program is improved to allow

changing parameters during the program execution, the user can change some of the parameters such as the target, or the radar location, and examine the course of action that the program takes.

The data used to express terrain can be changed to another format. It is not easy to find simple ASCII format. By either mapping the ASCII to DEM files or by designing the algorithm to be able to use standard world format helps importing real world data and using the program for every part of the world. Some terrain data types are:

-SDTS

-Binary DEM

-USGS DEM

-Japan DEM

-DDL

-ASCII Regular

-ASCII Irregular

-ASCII (x, y, z)

- TGA Irregular

A 3D simulation of the flight path can be added to the program. This would give the pilot a cockpit view before flying the mission.

Different capabilities such as the payload capacity of the UAVs mentioned in Section 2.2 can be integrated into the UAV model to build a more realistic application.

Appendix A: Example: Moving a Particle

Let's look at an example to illustrate how PSO_AS works. The numbers in the parenthesis are the move types of 3D

Routing.

X_t ={2,4,6,5,3,7,1}

$P_{\forall g,t}$ = {3,7,6,4,5,1,2}

$P_{ig,t}$ = {7,6,2,3,5,4,1}

C_1 = 10%

C_2 = 10%

C_3 = 80%

Random numbers generated = {60, 82, 87, 26, 94}

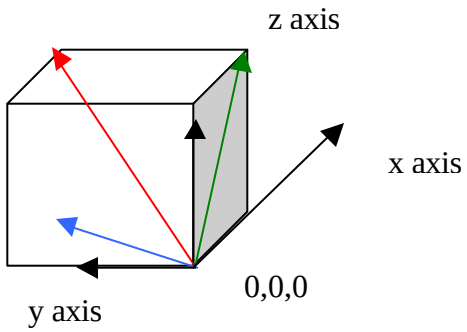

Figure 47. Making a move with particle interaction

The red arrow shows the global best move, the blue shows the personal best, and the green shows the last tour. The first random number generated is 62. Since our global trust percentage is between 0 and 80 the particle trusts the global move. And the next coordinates become 1,1,1. For the next move, the second moves of the global, local, and last tour are chosen.

This goes on till the particle gets to the target. If none works the initialization determines the next x, y, z coordinate.

Appendix B: More on statistical Techniques

While analyzing the results of a set of observations, the mean and the median always exist and on the contrary the mode may not exist. Even if the mode exists there may be more than one. The three indices are generally used for different distribution types. In a unimodal, symmetrical distribution all three exist and are equal. In a bimodal, symmetrical distribution the median and the mean are equal but the mode is not unique. In a uniform distribution the median and the mode are equal but there is no mode. In a distribution that is skewed to the right, the value of mean is greater than the median, which in turn is greater than the mode. Finally, in a distribution that is skewed to the left the mean is less than the median, which is less than the mode. Figures 7-11 gives better insight in terms of understanding the different distribution types:

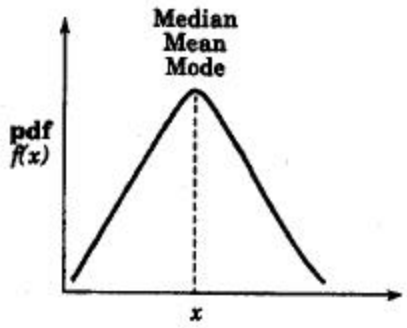

Figure 48. Uniform Symmetrical Distribution.

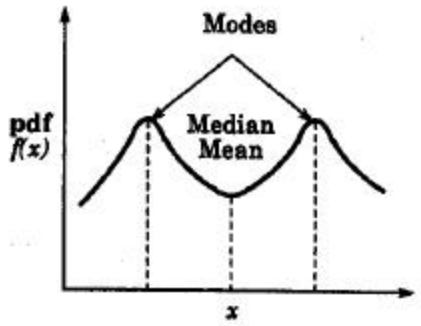

Figure 49. Bimodal Symmetrical Distribution

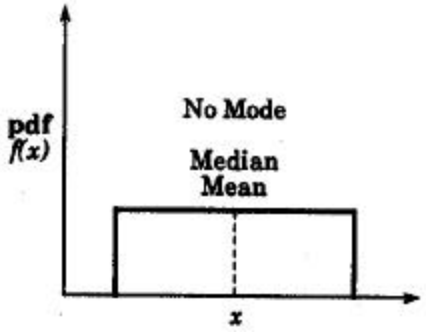

Figure 50. Uniform Distribution.

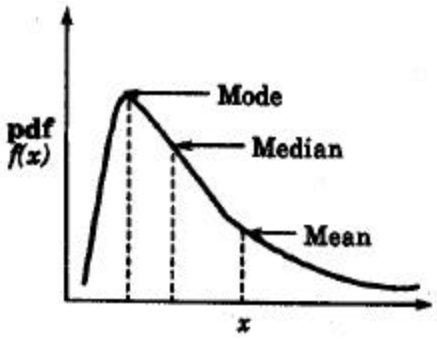

Figure 51. Distribution Skewed to the Right

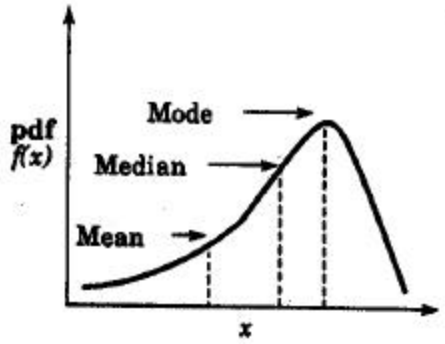

Figure 52. Distribution Skewed to the Left

Good charts are the ones that:

- Require minimum effort from the reader
- Maximize information
- Avoid unnecessary information
- Use commonly accepted practices
- Avoid ambiguity

Common mistakes in preparing charts:

- Presenting too many alternatives on a single chart
- Presenting many y-variables on a single chart
- Using symbols in place of text
- Placing extraneous information on the chart
- Selecting scale ranges improperly
- Using a line chart in place of a column chart

Some of the pictorial games played on the reader to deceive them:

- Using non-zero origins to emphasize the difference
- Using Double-Whammy graph for dramatization

- Plotting Random quantities without showing confidence intervals
- Pictograms scaled by height
- Using inappropriate cell size in histograms
- Using broken scales in column chart

Appendix C: Matlab Code for Visualization

```
clear all

load('hedefler.txt');
load('bigmaptakla.txt');
load('totalfitness.txt');
load('tumyol.txt');
load('test.txt');
load('deney.txt');
%load('psoroute.txt');
load('radar.txt');

bigmap=bigmaptakla;

s=size(tumyol,1);
s1=size(test,1);
s2=size(deney,1);

bas=tumyol(1,:);
tar=tumyol(s,:);

bas1=test(1,:);
tar1=test(s1,:);

bas2=deney(1,:);
tar2=deney(s2,:);

fitness=totalfitness(1,1);

nx=size(bigmap,2);
ny=size(bigmap,1);

%%%%%%
brown = [128 89 19]/255;
lightbrown = [128 89 19]/255*1.7;
earthColors = repmat(0,[64,3]);
for j = 1:64,
    earthColors(j,:) = (64-j)/64 * brown + j/64*lightbrown;
end;
%%%%%%
green = [10 128 10]/255;
lightgreen = [10 128 10]/255*1.7;
radarColors = repmat(0,[64,3]);
```

```
for j = 1:64,
    radarColors(j,:) = (64-j)/64 * green + j/64*lightgreen;
end;
%%%%%%

figure(11), clf

surfl(bigmap);
axis equal;
axis vis3d;
hold on;
shading interp;
colormap(earthColors);
rotate3d;

plot3(tumyol(:,1),tumyol(:,2),tumyol(:,3),'red'), hold on
plot3(test(:,1),test(:,2),test(:,3),'blue'), hold on

sos=20;
[X,Y,Z] = SPHERE(sos);
t=sos+1;
nr=size(hedefler,1);

kos=20;
[X,Y,Z] = SPHERE(kos);
t=kos+1;
kr=size(radar,1);

surf(0.5*X+tumyol(1,1)*ones(t,t),0.5*Y+tumyol(1,2)*ones(t,t),0.5*Z+tumyol(1,3)*ones(
t,t),ones(t,t)*1), hold on;
surf(0.5*X+tumyol(s,1)*ones(t,t),0.5*Y+tumyol(s,2)*ones(t,t),0.5*Z+tumyol(s,3)*ones(t
,t),ones(t,t)*64), hold on;

surf(0.5*X+test(1,1)*ones(t,t),0.5*Y+test(1,2)*ones(t,t),0.5*Z+test(1,3)*ones(t,t),ones(t,t
)*1), hold on;
surf(0.5*X+test(s1,1)*ones(t,t),0.5*Y+test(s1,2)*ones(t,t),0.5*Z+test(s1,3)*ones(t,t),ones
(t,t)*64), hold on;


surf(0.5*X+deney(1,1)*ones(t,t),0.5*Y+deney(1,2)*ones(t,t),0.5*Z+deney(1,3)*ones(t,t),
ones(t,t)*1), hold on;
surf(0.5*X+deney(s2,1)*ones(t,t),0.5*Y+deney(s2,2)*ones(t,t),0.5*Z+deney(s2,3)*ones(
t,t),ones(t,t)*64), hold on;
```

```
for i=1:nr
  plot3(hedefler(i,1),hedefler(i,2),hedefler(i,3),'kp'), hold on;

mesh(hedefler(i,4)*X+hedefler(i,1)*ones(t,t),hedefler(i,4)*Y+hedefler(i,2)*ones(t,t),hede
fler(i,4)*Z+hedefler(i,3)*ones(t,t)), hold on;
text(hedefler(i,1)+0.2,hedefler(i,2),hedefler(i,3),num2str(i));
```

Appendix D: Possible Algorithm Domains for Solution

1. Evolutionary Strategy (ES)

The Data Structure for the ES:

Representation : Real-valued

Fitness : Objective function value

Self-adaptation : Standard deviations and rotation angles

Mutation : Gaussian, main operator

Recombination : Discrete and intermediate, sexual and panmictic, important for self-adaptation

Selection : Deterministic, extinctive or based on preservation

Constraints : Arbitrary inequality constraints

Theory : Convergence rate for special cases (1+1)-ES,(1+,?)-ES, global convergence for (μ+ ?)-ES

Back's Symbolic Notation of ES [10]:

t=0;

Initialize P (0): = $\{\vec{a}_1(0),..,\vec{a}_m(0)\} \in I^m$

where I = $IR^n x IR^{n_s} x[-p,p]^{n_s}$

Evaluate P(0) : $\{\Phi(\vec{a}_1(0)),...,\Phi(\vec{a}_m(0))\}$

While (P(t)?true) do

Recombine:

Mutate:

Evaluate:

Select:

t=t+1;

od

The Applicability of ES to 3D UAV Routing:

For the initialization phase of the 3D UAV Routing, there has to be a complete initial population. To provide this population, it is possible that a single initial starting point can be used to by means of mutation.

If the individuals cannot be created (by means of recombination and mutation) because the individual that is to be created violates one or more of the constraints (3D UAV Routing problem has many constraints), then it is a lethal mutant. It becomes a complex process for the algorithm to handle the constraint but Schwefel's original algorithm is able to handle inequality constraints.

The termination process can be done by the evaluation of the parent and choosing the best one.

(m, l) strategy might work better than the $(m + l)$ for fitness landscapes that are not chaotic that much because the global best will be around the many local bests and once the individuals get to one of the local bests around the global it will be very easy for them to find the global solution.

With many constraints, it may be better to not have high selective pressure that it wouldn't be able to escape the local optima.

Evolutionary strategy has self-adaptation which is similar to PSO where the particles adapt themselves toward the better fitness values.

Standard deviations and rotation angles in mutation can be used as local search technique to provide better solutions. PSO doesn't have mutation operators but it puts limit on the velocity of the particles so that it doesn't converge so fast. Here the ES may prevent the early convergence of the landscape may be even better exploration by using its mutation operators.

So, ES can be applied to Mission Routing. PSO uses adaptation just like the ES in the fitness landscape to get to the global.

2. Evolutionary Programming (EP)

The Data Structure for the EP:

Representation : Real-valued

Fitness : Scaled objective function value

Self-adaptation : None (standard-EP), variances (meta-EP), correlation coefficients

Mutation : Gaussian, only operator

Recombination : Discrete and intermediate,sexual and panmictic, important for self-adaptation

Selection : Probabilistic, extinctive

Constraints : Arbitrary inequality constraints

Theory : Convergence rate for special cases (1+1)-ES,(1+,?)-ES, global convergence for (µ+ ?)-ES

Back's Symbolic Notation of EP [10]:

t=0;

Initialize P (0): = $\{\vec{a}_1(0),..,\vec{a}_m(0)\} \in I^m$

where I = $IR^n xIR^{n_s}$

Evaluate P(0) : $\{\Phi(\vec{a}_1(0)),...,\Phi(\vec{a}_m(0))\}$

While (P(t)?true) do

Mutate:

Evaluate:

Select:

t=t+1;

od

The Applicability of EP to 3D UAV Routing:

It generates its start population by uniform random sampling, which can be applied to creating uniform paths in 3D space.

The lack of crossover operator causes the mutation to be the main operator and it is defended by the evolutionary programmers that actually the mutation plays more important role than the crossover and that the effect of the crossover operator is exaggerated.

In the evolutionary strategies, if the individuals cannot be created (by means of recombination and mutation) because the individual that is to be created violates one or more of the constraints (3D UAV Routing problem has many constraints), then it is a lethal mutant. It becomes a complex process for the algorithm to handle the constraint but Schwefel's original algorithm is able to handle inequality constraints. Handling inequality constraints is so far not been incorporated into Evolutionary Programming in general, but the method used in Evolutionary Strategies can be directly transferred to Evolutionary Programming.

The selection operator is probabilistic. The parent population size is equal to the offspring population size so the candidates for the next generation are determined on a probabilistic selection. It can be a tournament selection.

The termination process is usually controlled by the maximum number of generations or by the maximum time.

(m, l) strategy might work better than the $(m + l)$ for fitness landscapes that are not chaotic that much because the global best will be around the many local bests and once the individuals get to one of the local bests around the global it will be very easy for them to find the global solution.

With many constraints, it may be better to not have high selective pressure that it wouldn't be able to escape the local optima.

Since, mutation is the main operator there won't be any convergence so the limit on the algorithm is either determined by time or by the number of iterations. Not converging to a local optimum may provide good exploration but decreases the local search ability.

EP can be applied to Mission Routing. A deep thinking must be put in to consider the constraints of the UAV Routing.

3. Genetic Algorithm (GA)

The Data Structure for the GA:

Representation : Binary-valued

Fitness : Scaled objective function value

Self-adaptation : No self-adaptation

Mutation : Bit-inversion, background operator

Recombination : z-point crossover, uniform crossover, only sexual, main operator

Selection : Probabilistic, based on preservation

Constraints : Simple bounds by encoding mechanism

Theory : Schema processing theory, global convergence for elitist version

Back's Symbolic Notation of GA:

```
Begin
    t=0;
    Initialize P (t);
    Evaluate structures in P (t);
    While termination condition not satisfied do
    Begin
        t= t+1;
        select_repro C(t) from P(t-1);
        recombine and mutate structures in C(t) forming C'(t);
        evaluate structures in C'(t);
        select_replace P(t) from C(t) and P(t-1);
    end
end
```

The Applicability of GA's to 3D UAV Routing:

Olsan's GA implementation fits to the symbolic formulation presented by Thomas Back. He represents the routes as genetic strings. Each checkpoint is represented by three

dimensions: x,y,z. Each checkpoint corresponds to a gene in the chromosome. Consecutive genes in a chromosome map to adjacent points on a 3-dimensional grid representing air space. Each chromosome is large enough to handle varying length worst-case longest path. The genes at both ends of the chromosome are constrained most since they correspond to areas immediately around the route's starting and ending points. The genes in the middle are constrained since they can take on values anywhere in the search space [8].

Initialization: A standard operator is used. The value of the genes are independent, they are dependent only on the order since the order implies the sequence of the route.

Evaluation: The evaluation function measures the distance of each route and the radar detection probability.

Recombination: Standard crossover operators are used to cross the routes which represent genetic strings.

Mutation: Specialized mutation operators are used to change the route

Some parameters:

n = number of checkpoints

d_i = distance of between checkpoints i-1 and i

rp_i = radar detection probability between checkpoints i-1 and i

w_d = weight of distance criteria

w_r = weight of radar detection criteria

Constraints:

$w_d + w_r = 1.0$

Evaluation function = $w_d * \sum_{i=0}^{i=n} d_i + w_r * \sum_{i=1}^{i=n} d_i * rp_i$

$$\text{Cost} = w_d * \sum_{i=0}^{i=n} d_i + w_r * \sum_{i=1}^{i=n} d_i * rp_i$$

Olsan's thesis is not the only GA implementation on MRP but maybe a good one in terms of showing how GA can be applied to a real world problem like this.

Comparison with PSO:

A route in PSO is represented by each particle as a point in the fitness landscape while in GA a chromosome represents a route. The genes that make up the chromosome represent the genotype information in Olsan's work. Some approaches used in GA can be taken as an operator in terms of helping PSO. Such as the crossover operator in GA can be used in PSO to make local optimizations similar to Lin-Kerninghan.

4. Ant System (AS)

The Data Structure for the AS:

Representation : Ants
Fitness : Scaled objective function value
Self-adaptation : No self-adaptation
Mutation : Amount of drop in pheromone level from the paths taken + probability of choosing
Recombination : Amount of pheromone added to the global best
Selection : Probabilistic

Constraints : None

Theory : Different convergence rate for different probabilities

Back's Symbolic Notation of AS:

t=0;

Initialize → P (t);

For I^{max} iterations do:

t=t+1;

a. For all ants generate a new solution using Formula (1) and the candidate list
b. Improve all vehicle routes using the 2-opt –heuristic
c. Update the pheromone trails using Formula (2)

Until $t=I^{max}$

The Applicability of AS to 3D UAV Routing:

G=(V, A, d) where $V=\{v_0, v_1, \ldots\ldots, v_n\}$ set of vertices

$A=\{(v_i, v_j)$: i?j} vertex v_0 is the depot, the other vertices are visitors or customers, and the non-negative weights d_{ij}, represent the distance between v_i and v_j.

The order of cities to be visited is chosen randomly at the initialization. For the selection of a not yet visited city, two aspects are taken into account: how good was the choice of that city, an information stored in the pheromone trails t_{ij} associated with each arc (v_i, v_j), and how promising is the choice of that city.

With $\Omega = \{v_j \in V: v_j \text{ is feasible to be visited}\} \cup \{v_0\}$, city v_j selected to be visited after city v_i according to a random-proportional rule that is stated as follows:

$$P_{ij} = \{ \sum_{h \in \Omega} \frac{[t_{ij}]^a [h_{ij}]^b}{[t_{hi}]^a [h_{ih}]^b} \text{ if } v_j \in \Omega \quad \rightarrow \quad \text{Formula 1} \quad \text{Construction of a route}$$

$$P_{ij} = \{ \quad 0 \qquad \text{otherwise}\}$$

$$h_{ij} = \frac{1}{d_{ij}}$$

t_{ij} = Pheromone trail

a and b determine the relative influence of the trails and the visibility

a = importance of pheromone trail

b = importance of heuristic function

$$t_{ij}^{new} = r\, t_{ij}^{old} + \sum_{m=1}^{s-1} \Delta t_{ij}^{m} + s \Delta t_{ij}^{*} \quad \rightarrow \quad \text{Formula 2} \quad \text{Trail update}$$

r = trail persistence ; (r -1) = pheromone evaporation [9].

Local search techniques like 2-opt, 3-opt, Lin-Kerninghan can be added to shorten the distance. AS is applicable to 3D UAV Routing.

Comparison with PSO:

AS is very similar to PSO in many ways. Ants can also be considered a different type of swarm. The particles in the swarm correspond to the ants in the AS. Both choose their way according to a probability function. And as the particles or the ants tend to follow the better way the convergence takes place. Local search techniques are easily

applied to both algorithms. The mutation and the crossover operators are totally different. The ants use the pheromone in their crossover and mutation operations. Their amount of laying the pheromone decides whether they do a mutation or a crossover. The PSO moves and the particles are affected from each other in the fitness landscape. Their positions in the fitness landscape decide how the particles will be affected at the genotype level.

Bibliography

1. Back, Thomas, "Evolutionary Algorithms in Theory and Practice". Oxford University Press, 1996.

2. Back T., D. B. Fogel, T. Michalewicz, Evolutionary Computation 1 –Basic Algorithms and Operators, IOP Publishing Ltd, 2000.

3. Bahnij, Maj Robert B. A Fighter Pilot's Intelligent Aide for Tactical Mission Planning. MS thesis, AFIT/GCS/ENG/85D-1, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1985

4. Bullnheimer Bernd, Richard F. Hartl, Christine Strauss. "An improved Ant System Algorithm for the Vehicle Routing Problem". Institute of Management Science, University of Vienna.

5. Bradshaw, 2Lt Jeffrey S. A Pilot's Planning Aid for Route Selection and Threat Analysis in a Tactical Environment. MS thesis, AFIT/GCS/ENG/86D-11, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1986.

6. Bramanti-Gregor, Anna. Strengthening Heuristic Knowledge in A* Search. PhD dissertation Wright-State University, 1993.

7. Droddy, Vincent A. Multicriteria Mission Route Planning Using Parallelized A* Search. MS thesis, AFIT/GCE/ENG/93D-04, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1993.

8. Gregory M. Nielson, et al., Scientific Visualization, Overviews, Methodologies, and Techniques. IEEE Computer Society, Los Alamos, California, 1997.

9. Grimm, James J. Solution to a Multicriteria Aircraft Arouting Problem Utilizing Parallel Search Techniques. MS thesis, AFIT/GCE/ENG/92D-04, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1992

10. Golden, Bruce L., et al. "A Multifaceted Heuristic for the Orienteering Problem," Naval Research Logistics, 35;359-366 (1988).

11. Golden, Bruce L., et al. "The Orienteering Problem," Naval Research Logistics, 34:307-318 (1987).

12. Gudaitis, Michael S. Multicriteria Mission Route Planning Using a Parallel A* Search. MS thesis, AFIT/GCS/ENG/94D-05, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1994.

13. Harder, Robert W. "A Java Universal Vehicle Router In Support of Routing Unmanned Aerial Vehicles". MS Thesis, AFIT/GOR/ENS/00M-16, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, March 2000.

14. Jain, Raj. , "The Art of Computer Systems Performance Analysis". John Wiley & Sons,Inc. , 1991.

15. Kinney, Gary W. Jr. "A Hybrid Jump Search and Tabu Search Metaheuristic for The Unmanned Aerial Vehicle (UAV) Routing Problem". MS Thesis, AFIT/GOA/ENS/00M-5, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, March 2000.

16. Lawler, E.L., J.K. Lenstra, A.H.G. Rinooy Kan and D. B. Shmoys, *The Traveling Salesman Problem*, Wiley, Chichester, 1985.

17. Skolnik, M., Introduction to Radar Systems, Third Edition, McGraw Hill, 2000.

18. Pellazar, Miles B. "Multi-Vehicle Route Planning with Constraints Using Genetic Algorithms," National Aerospace and Electronics Conference. May 1994.

19. Pressman, Roger S., Software Engineering, A Practitioner's Approach, 4th edition, Mc Graw-Hill, 1997.

20. Olsan, James B. "Genetic Algorithms Applied to a Mission Routing Problem". MS Thesis, AFIT/GCE/ENG/93-12, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1993.

21. Schulmeyer, G. Gordon and James, I. MacManus, editors. Total Quality Management for Software. Van Nostrand Rienhold, 1992.

22. Secrest, Barry R. "Traveling Salesman Problem For Surveillance Mission Using Particle Swarm Optimization". MS Thesis, AFIT/GCE/ENG/01M-03, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, March 2001

23. Sezer, Ergin. "Mission Route Planning With Multiple Aircraft & Targets Using Parallel A* Algorithm". MS Thesis, AFIT/GCE/ENG/00M-04, School of

Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, March 2000.

24. Willis Nicholas J, Bistatic Radar, Artech House, 1991

25. www.gat.com - "General Atomics Aeronautical Systems, Inc."

26. Cormen, Thomes and others. Introduction to Algorithms, McGraw Hill, 1989.

27. http://www.ngdc.noaa.gov/cgi-bin/seg/ff/nph-newform.pl/seg/topo "Select your own area"

28. Dorigo Marco, David Corne, Fred Glovar, "New Ideas in Optimization", McGraw Hill,1999

29. Isesee, Ernst K "Multi-criteria Network Routing of Tactical Aircraft in a Threat Radar Environment". MS Thesis, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, Mar-91

30. Dyer, Douglas E. "Low-Level Aerial Route Planning for Detection Avoidance". MS Thesis, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, Jun-92

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
15-03-2002	Master's Thesis	Jun 2001 – Mar 2002

4. TITLE AND SUBTITLE

MULTI-OBJECTIVE MISSION ROUTE PLANNING USING PARTICLE SWARM OPTIMIZATION

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Yavuz, Kursat, 1st Lt., TUAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 P Street, Building 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GCE/ENG/02M-04

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Embedded Information Systems Engineering Branch, Information Technology Division, Information Directorate, Air Force Research Laboratory, AFRL/IFTA
2241 Avionics Circle
Wright Patterson AFB OH 45422
DSN: 785-6653x3592
Robert.Ewing@wpafb.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

Professor Gary B. Lamont, CIV, AFIT/ENG, Ph. 937 255-3450

14. ABSTRACT

The Mission Routing Problem (MRP) is the selection of a vehicle path starting at a point, going through enemy terrain defended by radar sites to get to the target(s) and returning to a safe destination (usually the starting point). The MRP is a three-dimensional, multi-objective path search with constraints such as fuel expenditure, time limits, multi-targets, and radar sites with different levels of risks. It can severely task all the resources (people, hardware, software) of the system trying to compute the possible routes. The nature of the problem can cause operational planning systems to take longer to generate a solution than the time available. Since time is critical in MRP, it is important that a solution is reached within a relatively short time. It is not worth generating the solution if it takes days to calculate since the information may become invalid during that time.

Particle Swarm Optimization (PSO) is an Evolutionary Algorithm (EA) technique that tries to find optimal solutions to complex problems using particles that interact with each other. Both Particle Swarm Optimization (PSO) and the Ant System (AS) have been shown to provide good solutions to Traveling Salesman Problem (TSP). PSO_AS is a synthesis of PSO and Ant System (AS). PSO_AS is a new approach for solving the MRP, and it produces good solutions. This thesis presents a new algorithm (PSO_AS) that functions to find the optimal solution by exploring the MRP search space stochastically.

15. SUBJECT TERMS

Mission Routing Problem, Particle Swarm Optimization, Evolutionary Algorithm, Ant System, Traveling Salesman Problem

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Lamont, B. Gary, Professor, CIV AFIT/ENG
U	U	U	UU	135	**19b. TELEPHONE NUMBER** (937) 255-3450x4718

X