Stealth path planning for unmanned aerial vehicle operations in radar zones

Kan, Ee May

2013

Kan, E. M. (2013). Stealth path planning for unmanned aerial vehicle operations in radar zones. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/54887>

<https://doi.org/10.32657/10356/54887>

**STEALTH PATH PLANNING FOR UNMANNED
AERIAL VEHICLE OPERATIONS
IN RADAR ZONES**

KAN EE MAY

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy

2013

Acknowledgements

First and foremost, I would like to thank my supervisor Prof Lim Meng Hiot for providing me with the opportunity to study and conduct research at School of Electrical and Electronic Engineering, Nanyang Technological University. His constant guidance and encouragement have motivated me to reach my utmost accomplishment towards professional maturity. For all his support and concern for me and my family, I am deeply indebted to him for positive influence he has had on my life.

During the phase of this research work, I would like to acknowledge the contributions of Prof Yeo Swee Ping (National University of Singapore), Mr Ho Jiun Sien (Temasek Polytechnic), Dr Shao Zhen Hai (University of Electronic Science Technology of China), Mr Lam Chian Poh (Singapore Technologies Aerospace) and Mr Leong Yew Kong (Singapore Technologies Aerospace) with whom I was able to carry on this research work. My research collaborators and I gratefully acknowledge the funding support from Temasek Defence Systems Institute, Singapore for this research work. Additionally, I would like to acknowledge all of my lab-mates at Intelligent Systems Centre for their help. I have benefitted tremendously from valuable discussions with them. At this point I would like to thank all of my postgraduate friends who have supported and encouraged me morally. I would also like to recognize the efforts of Nora and Nazri at Intelligent Systems Centre, who provided IT supports and administrative assistance.

I would like to express my deep gratitude with all my heart to my parents. Even from hundreds of miles away, their endless support and encouragement have strengthened me to complete this long journey for Ph.D. I am also thankful to my other family members who have always wished for my success. I don't think I could have made the journey without their constant support throughout my whole academic years. Last but certainly not least, I would like to give thanks to my partner, Shawn Tay.

Your embrace lends me courage and strength, your passion inspires me and your devotion spurs me to work hard.

Table of Contents

Acknowledgements.....	I
Table of Contents.....	III
List of Figures.....	VI
List of Tables.....	VIII
Abstract	IX
Chapter 1: Introduction	1
1.1 Problem Significance	1
1.2 Motivation	5
1.3 Major Contributions	8
1.4 Organisation of the Thesis	11
Chapter 2: A Brief Literature Survey on Path Planning Methods	13
Chapter 3: Radar-Aware B-Spline Trajectory Generation for UAV.....	27
3.1 Introduction	30
3.2 Problem Model	31
3.3 Representation of Navigational Space.....	36
3.4 Solution Methodology.....	38
3.4.1 Connectivity Relationship between Nodes	38
3.4.2 Radar-aware Path-Planning Algorithm.....	43
3.4.3 Comparison.....	49
3.5 B-spline Path Smoothing	52
3.5.1 Flight Envelope	53
3.5.2 Weighting Functions.....	56
3.6 B-spline Path Patterns	61
3.7 Path-replanning Algorithm in Radar Zones.....	63
3.7.1 Algorithm Steps.....	65
3.7.2 Experimental Results	66
3.8 Summary	70

Chapter 4: Memetic Computation in Unmanned Aerial Vehicle (UAV) Operations	71
4.1 Introduction	73
4.2 Problem Formulation.....	75
4.2.1 Discretizing the Problem Domain.....	75
4.2.2 Objective Formulation	78
4.3 The Proposed Approach	81
4.3.1 Crossover Operator	83
4.3.2 Mutation Operator	85
4.3.3 Path-Repair Mechanism.....	87
4.3.4 Cooperative Individual Learning.....	89
4.3.5 Strategy 1: Clustering	91
4.3.5.1 Waypoints Relocation	91
4.3.5.2 Waypoints Exchange.....	92
4.3.5.3 Route Exchange	92
4.3.6 Strategy 2: Route Reduction.....	95
4.3.7 Strategy 3: TSP Optimization.....	97
4.3.7.1 Reinsertion.....	97
4.3.7.2 2-exchange.....	97
4.4 Simulations	99
4.5 Performance Analysis and Discussions.....	107
4.6 Computational Complexity.....	108
4.6.1 Individual Learning.....	108
4.6.2 Objective Function.....	109
4.7 Summary	110
Chapter 5: Multi-criteria Mission-planning.....	111
5.1 Introduction	113
5.2 Problem Formulation.....	116
5.2.1 Minimizing the Total Flight Time	116
5.2.2 Minimizing the Fuel Consumption	117
5.2.3 Maximizing the Safety Level	118
5.2.4 Constraint Functions	118

5.3 Solution Methodology	120
5.3.1 Particle Swarm Optimization (PSO)	120
5.3.2 Multi-objective Optimization	122
5.4 The Proposed Algorithm	123
5.4.1 Sequence of waypoints candidates	125
5.4.2 Crowding Distance Assignment Operator.....	126
5.4.3 Generic-evaluation Operator	128
5.4.4 Constraint Handling	130
5.5 Experimental Results.....	131
5.5.1 Clustering Approach	135
5.5.2 Pseudo-weight Vector Approach.....	137
5.6 Implementation	138
5.7 Summary	142
Chapter 6: Concluding Remarks.....	143
6.1 Conclusions	143
6.2 Future Research	146
References	150
List of Publications	159
Journals	159
Conference Papers	159

List of Figures

Chapter 1

Figure 1: Navigational space with known radar sites	1
---	---

Chapter 2

Figure 2: UAV operations in radar zones.....	26
--	----

Chapter 3

Figure 3. 1: Flow chart of the radar-aware path-planning method	27
Figure 3. 2: Navigational space with known radar sites.....	31
Figure 3. 3: Reduction in the received power of target with range.....	33
Figure 3. 4: Graphing of the different risk levels based on knowledge of the radar zone.....	36
Figure 3. 5: Numerical assignment of risk cost to cells in the navigational space.....	37
Figure 3. 6: Splitting of generated path for the purpose of adding middle knots	57
Figure 3. 7: A smooth curve in the route polygon	59
Figure 3. 8: A shortest curve in the route polygon.....	59
Figure 3. 9: Generated spline path	62
Figure 3. 10: Path evolution and re-planning using the proposed algorithm.....	68
Figure 3. 11: Illustration of the actual path followed by the UAV. Dashed lines show the currently tentative optimal path obtained from the algorithm, based on the risk cost from the current location of the UAV.	69

Chapter 4

Figure 4. 1: Navigational space 1 with known threats	76
Figure 4. 2: Navigational space 2 with known threats	77
Figure 4. 3: Illustration of the navigational space 1 with 8 waypoints	101
Figure 4. 4: Illustration of the multi-route solution representation $h = r_1, r_2, r_3$ for 3 UAVs to visit 8 waypoints at navigational space 1	101

Figure 4. 5: Illustration of the multi-route solution representation $h = r_1, r_2, r_3, r_4$ for 4 UAVs to visit 8 waypoints at navigational space 1	102
Figure 4. 6: Illustration of the multi-route solution representation $h = r_1, r_2, r_3, r_4, r_5$ for 5 UAVs to visit 8 waypoints at navigational space 1	102
Figure 4. 7: Illustration of the navigational space 1 with 14 waypoints.....	103
Figure 4. 8: Illustration of the multi-route solution representation $h = r_1, r_2, r_3$ for 3 UAVs to visit 14 waypoints at navigational space 1	103
Figure 4. 9: Illustration of the multi-route solution representation $h = r_1, r_2, r_3, r_4$ for 4 UAVs to visit 14 waypoints at navigational space 1	104
Figure 4. 10: Illustration of the multi-route solution representation $h = r_1, r_2, r_3, r_4, r_5$ for 5 UAVs to visit 14 waypoints at navigational space 1.....	104
Figure 4. 11: Illustration of the navigational space 2 with 8 waypoints.....	105
Figure 4. 12: Illustration of the multi-route solution representation $h = r_1, r_2, r_3$ for 3 UAVs to visit 8 waypoints at navigational space 2.....	105
Figure 4. 13: Illustration of the navigational space 2 with 14 waypoints.....	106
Figure 4. 14: Illustration of the multi-route solution representation $h = r_1, r_2, r_3$ for 3 UAVs to visit 14 waypoints at navigational space 2	106

Chapter 5

Figure 5. 1: Navigational space with start and target locations (circles) as well as known threats (diamonds).....	132
Figure 5. 2: Non-dominated solutions obtained using the proposed PSO algorithm and evolutionary algorithm for mission-planning problem. The two objectives are f_1 - path length and f_2 - safety factor.....	134
Figure 5. 3: Representation of clustered non-dominated solutions.....	136
Figure 5. 4: Illustration of the optimum solutions	138
Figure 5. 5: RCS plot of a target in our study.....	139
Figure 5. 6: The flight simulation environment.....	141
Figure 5. 7: The UAV following generated optimum set of waypoints.....	141

List of Tables

Chapter 3

Table 3. 1: Test results	48
Table 3. 2: Computational cost of the proposed algorithm	50

Chapter 4

Table 4. 1: Time elapsed of different planners on navigational space 1 (sec).....	100
Table 4. 2: Time elapsed of different planners on navigational space 2 (sec).....	100

Chapter 5

Table 5. 1: Classifications of UAV constraints and performance indices.....	128
Table 5. 2: Resulting statistics by the proposed PSO algorithm and evolutionary algorithm for the mission-planning problem. In SC(A,B), A is the proposed PSO algorithm and B is the multi-objective evolutionary algorithm. The best performing algorithm is indicated by bold numbers.	133

Abstract

A typical mission of a UAV is characterized by the requirement of navigating through multiple waypoints as dictated by the mission requirements. These waypoints are usually known *a priori*, and the mission must be completed satisfying certain criteria such as minimum time, minimum fuel or/and minimum danger exposure as dictated by the mission scenario. Such mission demands a high level of stealth, which has a direct implication on the safety and success of the mission. Assuming the domain of interest is fully characterized *a priori*, an offline trajectory planning optimization can be used to derive a satisfactory solution. However, in the case of UAVs flying over radar prone environment, UAVs necessarily require more advanced navigation and guidance capabilities.

In this thesis we first propose a radar-aware path planning method for UAV in radar zones. This method focuses on trajectory generation algorithms that take into account the stealthiness of autonomous UAV; generating stealth based paths through a region laden with radars. It involves generating a preliminary cost-effective path consisting of a series of straight-line segments. Upon the optimal path being represented as a sequence of straight line segments, we smoothen it by using of a series of B-spline path patterns. For application in UAV path-planning, the smoothened B-spline path patterns are incorporated to replace parts of the path while preserving continuity to obtain a feasible path. To this end, an optimum path traceable by the UAV with radar-aware capability is

generated. It has been demonstrated that the UAV manages to accomplish the mission with minimal radar exposure.

For cooperative control of multiple UAVs, we developed an efficient evolutionary framework to optimize their operations in radar zone. The proposed approach is capable of solving cooperative mission-planning problem for multiple UAVs. Taking advantage of memetic intelligence, the proposed planner is competent of handling large-scale optimization problems for multiple UAVs. To achieve robustness, a cooperating individual learning strategy is proposed in which local search strategies or memes, each having different learning roles and search features, work together to accomplish the shared optimization goal.

Next, our work focuses on aspect of optimizing a multi-criteria mission-planning problem for multiple UAVs, catering to the requirement of different mission constraints. The objective is to compute an optimum solution for each UAV to complete its mission, taking into account the pre-specified conditions. We demonstrate the effectiveness of the proposed approach by yielding a broad range of Pareto optimum solutions. This approach could be easily extended for different scenarios by appending different terms in objective function and constraints. Lastly, the proposed framework is experimentally validated through a flight simulation environment, showing that the applicability of our proposed approach on the actual system.

Chapter 1: Introduction

In this chapter, the background of the research is presented. The scope of research and the objectives are also described.

1.1 Problem Significance

The popularity of unmanned vehicles is evident from the recent craze and novelty in sophisticated fun toys. On a more serious note, unmanned aerial vehicles (UAVs) find ready applications in military and law enforcement operations [1] [2] [3] such as environmental surveillance, battlefield monitoring, ordnance delivery, resource assessment, etc. UAVs on reconnaissance missions have to constantly evade detection to avoid drawing unnecessary attention. To describe the problem, consider the navigational space with known radar sites (blue circle) as shown in Figure 1.

Figure 1: Navigational space with known radar sites

One of the challenges for UAV to successfully complete its mission hinges on its capability to minimize its exposure and visibility to radar. There are two aspects one may consider to achieve such capability - design of radar invisible UAVs or radar-aware mission-planning. Our work considers the latter; focusing on aspect of path-planning. The objective is to compute an optimum solution for UAVs to complete their mission, taking into account the constraints of flight orientation, path length, fuel consumption and flight safety factor. It is desirable to provide a UAV with a methodology with which it can satisfy certain objectives while intelligently and autonomously travelling between various distinct points in space.

One approach to achieve this objective is to deploy specially designed UAVs which are literally invisible to radar. A more practically viable option involves pre-emptive initiatives on the path-planning side. This involves computing flight paths of minimal radar visibility, capitalizing on the features inherent in the terrain that UAVs operate in and the physics of the UAVs manifested as flight orientation and flight formation (in the case involving several UAVs).

Over the years, advances in software and computing technology have fuelled the development of a variety of new guidance methodologies for UAVs. It is useful to endow UAVs with more advanced guidance capabilities, in particular capabilities that increase the vehicle's autonomy to allow for more complex missions or tasks. The availability of various guidance technologies and an understanding of operational scenarios create opportunities towards refining and implementing such advanced

guidance concepts. The basic difficulties include partially known and changing environment, extraneous factors such as threats, evolving mission elements, tight timing and positioning requirements. Moreover, these must be tackled, while at the same time, explicitly accounting for the actual vehicle maneuvering capabilities, including dynamics and flight-envelope constraints. Additionally, the time needed for completing a mission is a critical factor in most military reconnaissance scenarios. The endurance of tactical UAVs can be significantly enhanced by careful flight path-planning. The number and the type of objectives a UAV must satisfy to navigate from a start location to a target location; the total length of the planned path along with information of the flight dynamics and the flight speed of the UAV serves as good indicators of the time and fuel needed to complete the mission.

There has been extensive research in the area of path-planning, especially in the field of artificial intelligence and optimization, with most restricted to two-dimensional (2D) space [4] [5]. One such example is the A* algorithm [6]. Although one of the more popular path-planning algorithms in artificial intelligence (because of its ability to judge or value the best path from point to point), it is ineffective when dealing with the dynamics of battlefield scenarios [7]. Most of the search methods do not incorporate the physics of the vehicles; this, in turn, results in certain local path segments being physically difficult for the aircraft to follow under kinematic constraints. Several authors have proposed to smoothen these segments by taking the kinematic constraints into account [8] [9] but no explicit analysis on the optimality variation due to local path smoothening has been furnished. Generally, there are several criteria to be considered

simultaneously during optimization. It is also typical that these objectives tend to be conflicting in nature and finding an optimal solution involves trade-off of several criteria.

The problem we consider in this work is on the basis of path-planning of UAVs for reconnaissance. The level of UAV autonomy has to be enhanced in order to accomplish more complex tasks in radar prone environment, and eventually allowing coordination with other UAVs to accomplish the mission.

1.2 Motivation

For military missions, UAVs may need to operate in hazardous environments, and therefore, it is crucial to consider the threats, radars and restricted zones. As such, the optimum routes are intended to minimize the risk of destruction of the UAVs, fulfilling all the constraints imposed by the proposed mission, the physical characteristics of the UAVs, and the environment. It is highly desirable in time-critical reconnaissance operations that the mission time be minimized. Additionally, planning efficient paths can significantly increase the operational endurance of low endurance UAVs.

The purpose for this research stems from the need to develop radar-aware algorithms that maximize the stealth of these UAVs when they fly over radar zone. In such situations, it may be necessary that the algorithms are embedded in the on-board control module. Thus, they must be computationally efficient. Also, it is desirable to equip each UAV with the capability to meet certain objectives while intelligently and autonomously navigate between various points in space — based on considerations such as avoidance of collision and no-fly zones, and optimization of various mission-specific parameters. These may be achieved in a hierarchical manner by first computing optimum mission waypoints amongst multiple UAVs and then computing optimal paths for each UAV taking into account the relevant UAV-specific criteria. However, there are several criteria to be considered simultaneously during optimization. It is normal that these objectives are conflicting in nature and solving this optimization problem is stimulating from the multi-objective perspective, involving trade-off among several criteria.

The classical point-by-point optimization approaches are not appropriate to solve multi-criteria optimization problems. For example, the weighted-sum method converts a multi-criteria optimization problem into a single criterion optimization, yielding an individual optimal solution. These conventional optimization approaches such as linear programming, non-linear programming and dynamic programming require the algorithm to be executed iteratively to ensure the utilization of all the weight combinations. It is necessary for the algorithms to be effective in balancing the appropriate range of weights.

In line with these observations, it is demonstrated that an optimum solution of the path-planning problem is non-deterministic polynomial-time (NP)-complete rendering the use of exhaustive or deterministic search methods to be ineffective [10] [11]. Problems in discrete combinatorial optimization domain are usually highly constrained and characterized by many local optima. However conventional local optimization techniques tend to fail in locating high quality solutions.

In these cases, modern stochastic technique such as evolutionary computation is often regarded as an effective and practical alternative of tackling the NP-hard problem. Evolutionary approaches are less prone to being trapped in local minima as they evaluate potential solutions in parallel and generate optimum solutions by interacting through genetic operators.

Creative application of the evolutionary computation concept rather than rigid imposition of a standard algorithm proves to be more effective in solving specific types

of real problems [12] [13] [14]. Pellazar [15], Nikolos *et al.* [16] and Yi *et al.* [17] make use of evolutionary algorithms (EAs) to develop route planners for aircrafts. Nevertheless, those route planners are not adaptable to deal with dynamic constraints on the resulting path. This is due to fact that for evolutionary approaches which deal mainly with encoding, genetic operations and selection mechanisms, the tuning of parameters is important in order to achieve desirable results. While there are approaches to tune parameters online, the practice of trial and error to tune an evolutionary algorithm is still very common.

Motivated by these observations, we aim to develop radar-aware path planning algorithms to deal with multiple UAVs navigating through the radar zone to accomplish their missions, catering to the requirements of different vehicle types and constraints. Our proposed approach overcomes the shortcomings of existing path-planning methods through computational intelligence approach, involving novel genetic operators and problem-specific representation of candidates. Furthermore, an evolutionary approach being stochastic in nature incorporates large spaces of candidate solutions in the search; hence variations in solutions generated and at the same time satisfying some pre-specified requirements or constraints are highly desirable. By making use of computational intelligence approach, the problem can be scaled up, able to deal with large-scale mission-planning for UAVs. In our study, we consider UAV flight envelope restrictions in terms of enforced maximum flying height and minimum turning angle, threats as well as the terrain profile (mountains, valleys, etc.). Details on the formulation

of the UAV path-planning problem as well as the proposed solution methodologies are presented in the following chapters.

1.3 Major Contributions

In this thesis we present our theoretical developments of radar-aware path-planning algorithms, UAVs cooperative operation procedures, multi-criteria evolutionary framework and experimental results to substantiate our approach. The following summary lists the primary contributions of this work.

- Route planning for multiple UAVs imposes severe limitations on the development of path control algorithm, stemming from the condition for real-time implementation as well as the UAV kinematic constraints. To overcome this problem, we propose a radar-aware path-planning method by employing a multi-level risk cost decomposition scheme to approximate a radar zone. A cost-effective path traceable by the UAVs with radar-aware capabilities can be constructed directly from the representation by taking into account the available computational resources. The proposed algorithm is suitable for the on-board, and real-time implementation.
- For guidance and navigation control of UAVs, we propose a path smoothening algorithm using a set of path patterns which take into account the kinematic constraints of the UAV. Instead of smoothening the entire path from an initial

location to the target location, we smoothen the path segments with respect to the location of the UAV so as to minimize the radar exposure. By incorporating the path patterns from off-line optimization, the proposed smoothening scheme has the advantage of minimal on-line computational cost.

- We developed a general evolutionary framework to optimize the operations of multiple cooperating UAVs. Taking advantage of memetic intelligence, our proposed approach is capable of handling large-scaled optimization problems for multiple UAVs. We investigate the number and placement of control points within a navigational space by considering the complexity of the terrain and identify the waypoints based on the topographic information of the terrain. The solution generated by our proposed framework should comply with the predefined stealth considerations and the vehicle constraints.
- A multi-criteria optimization approach is proposed for mission-planning problem. This approach is developed to generate solutions by incorporating the principle of Pareto dominance into the optimization algorithm. The solutions yield a compromise among the criteria; distinguish a set of alternatives that define optimum solutions to the mission-planning problem. Subsequently, to ease straightforward execution for the mission planner, an effective selection of resulting solution is demonstrated. The experimental results prove that our proposed optimization approach is a feasible alternative for solving the UAV mission-planning problem.

- We propose a multi-criteria evaluation method based on priorities and constraints of the UAV. The evaluation method that combines different priorities and constraints uses the generic multi-criteria Pareto-evaluation function to permit the change of the levels of priority as well as the constraints of the mission. By doing so, the proposed planner can simply be applied for various types of missions. This method could be easily extended for different scenarios by appending different terms in objective function and constraints.
- To demonstrate the applicability of the proposed algorithm on the actual system, our proposed path planner is experimentally validated through a flight simulation environment. Additionally, we describe the orientation of the UAV associated with the implementation of the proposed algorithm, by taking into consideration the limitations of the UAV.

1.4 Organisation of the Thesis

This thesis consists of six chapters. The remaining chapters are as follows.

Chapter 2 reviews the past literature for similar or related work. It introduces the general path-planning problem, together with some of its properties and its relation to current work. In Chapter 3, we present a radar-aware path-planning algorithm. We then introduce a risk cost representation scheme to approximate a radar zone. An efficient algorithm for generating a stealthy path for UAV is shown. Given a discrete segmented path sequence, a smoothening algorithm is proposed to generate a smoothened path. In Chapter 4, we present an evolutionary cooperative mission planner to optimize the operations of multiple UAVs. With appropriate domain-specific representation, the proposed memetic approach can easily be applied for cooperative mission-planning. In Chapter 5, an optimization approach is proposed for solving multi-criteria mission-planning problem. The generated solutions yield a compromise among the criteria by distinguishing a set of solutions to the mission-planning problem. A simple but effective selection of resulting solution is proposed to facilitate straightforward execution for the mission planner. Additionally, we demonstrate the flight orientation of the UAV associated with the implementation of the proposed method, by taking into consideration the constraints of the UAV. Finally, Chapter 6 summarizes the main results of this thesis, and lists several recommendations for future research. Moreover, a preliminary study based on machine learning techniques for terrain based navigation is presented in Chapter 6.2. To deal with loss of GPS (Global Positioning System) information, terrain

based navigation can be considered. It is necessary to derive real-time positional reference information for UAVs to ensure the continuity of their mission during the outage of GPS.

Chapter 2: A Brief Literature Survey on Path Planning Methods

Path-planning is the process of automatically generating traversable paths for an autonomous moving object based on a set of predefined criteria, avoiding obstacles along the way. Path-planning can be broadly categorized as motion planning and trajectory planning [18]. In motion planning, the focus is on getting the object from a starting position to an end position, considering primarily on the translation and rotations needed to get the object there. Recent work has also included the aspects of dynamics, constraints, and optimality. Trajectory planning on the other hand focuses on taking the solution from the motion planning process and then determining a process of moving the object along that path, while still considering the mechanical limitations of that object [18].

The path planner reported in the literature implement waypoint following schemes assuming the UAVs are given a sequence of waypoints *a priori* in the region of interest. These waypoints are connected in order to generate smooth path segments [19] [20] in a manner that preserve the continuity of the curvature between line and arc segments, while minimizing the maximum curvature on the curve. In two-dimensional vehicle motion subject to non-holonomic constraints, it has been proved in [21] and [22] that the optimal shortest path exists which are either straight line segments or arcs of circle of radius if the vehicle has a maximum turning rate constraint. This set of path elements is termed as “Dubins path” and has inspired many researchers on how to design a feasible

path by connecting way-points. Based on this idea, Bui and Soueres [23] proposed new optimality conditions on these paths by computing a partition space from the two-dimensional plane of the configuration space while Yang and Kapila [24] exploited parameter optimization methods to solve optimal path-planning problems. A similar geometry-based path-planning algorithm is also reported in [25].

Many traditional path planners implement a brute force search for the next best alternate path, which is inefficient [18]. Later implementations went on to build a tree structure of possible alternate paths, and then using a branch and bound method to locate the best solution. There has been extensive research in the area of path-planning especially in the artificial intelligence, and optimization with most restricted to two-dimensional (2D) paths [4] [5]. An example would be the A^* algorithm [6], one of the more popular path-planning algorithm in the artificial intelligence. The strength of the A^* algorithm lies in the ability heuristically to judge or value the best path from point to point. However, with the dynamic nature of a battlefield scenario, the method relies on is constantly changing. This results in the A^* method to be inaccurate [7], thus making it ineffective for mission with critical path-planning purposes. An improvement to the A^* algorithm is the D^* [26] and the D^* *Lite* algorithm [27]. The D^* algorithm is a dynamic version of the A^* search algorithm, where it performs multiple searches by modifying previous search results locally to obtain significant search speed up of one to two orders of magnitudes. The D^* *Lite* implements a similar search strategy as the D^* but is substantially shorter and simpler to use. Studies performed by the developers showed that even with D^* *Lite*'s simplicity, it is at least equally as efficient as the original D^* .

D Lite* has been successfully implemented in robot path-planning situations in unknown terrain [28] [29] [30]. Wu et al. [31] developed a multi-objective path planner that uses a fuzzy logic system in the *D* Lite* graph search. Their case studies have shown that the method was able to successfully generate alternate paths that satisfy the required criteria.

The D^* algorithm, proposed by Stentz [32] [33], is capable of planning a path in unknown or partially known environment, as it adopts an efficient incremental search scheme to produce an optimal path. The incremental search scheme incorporates the information from previous searches to find a solution much faster than solving it from scratch. Subsequently, Koenig and Likhachev proposed the *Lifelong Planning A** (LPA) [34] which has been derived from A^* and adopted for dynamic environment by a reusing scheme. Furthermore, Koenig and Likhachev presented a *D* Lite* algorithm, derived from the LPA algorithm, which implements the similar planning approach as D^* although is algorithmically different.

Recently, many realized the limitations of two-dimensional (2D) path-planning and began implementing three-dimensional (3D) path planners, taking advantage of the additional two degrees of freedom when generating alternate paths [35] [36] [37] [38] [39]. However, the additional information from one extra dimension can create new computational and visualization challenges. An additional degree of freedom results in more alternate path choices, which need to be explored. Another technique of handling multi-dimensional motion planning is the use of probabilistic roadmaps (PRM), introduced by Kavraki et al. [40], which overcame the restrictions of higher dimensionality in traditional grid search algorithms. Lavelle introduced the concept of

rapidly exploring random trees (RRT) [41] that iteratively expanded the tree by adding new samples to the closest existing sample in the tree. Improved computation time and more flexible control of the alternate path criteria were observed in instances when optimization has been applied during the path-planning process [42] [43] [44]. Methods such as routing scheduling and risk-analysis-based routing [45] [46] have been catered for the transportation of hazardous materials, to optimize the objectives of minimizing risk and at the same time also minimizing the transportation cost.

More recently, a three-dimensional extension of Dubins path was developed by Shanmugavel et al, [47]. Although the path-planning strategies discussed above yield the optimal (shortest) paths between way-points for non-holonomic vehicles and yield easy implementation, one should note that they do not explicitly deal with obstacles possibly existing along the path segments. This disadvantage becomes apparent in the case when the vehicle navigates in an environment filled with obstacles where the designed path segments would result in collision with the obstacles. As discussed in [48], it is then necessary to consider a path-planning algorithm that takes into account obstacles avoidance. One technique for obstacle avoidance is to use the “artificial potential field” concept. Khatib [49] presented the earlier result of real-time obstacle avoidance for mobile robots using this concept. In this method, a mobile robot moves in a potential field such that the goal is modelled as an attractive force and the obstacles and other vehicles are modelled as repelling forces. In [50], local minima of a potential field are connected in order to build an incremental graph, and the graph is searched to attain the final goal configuration. In [51] [52], an artificial potential field serves as a local

holonomic planner to avoid obstacles, of which the output is modified in a least-squares sense to generate actual commands for the desired motion. Although potential field methods can produce smooth and dynamically feasible trajectories at low computational cost, they have several drawbacks.

Detecting and avoiding the obstacles are mostly influenced by the proximity of obstacles, and the vehicle can get trapped in a local minimum. Another issue arises from the fact that obstacles are not modelled as hard constraints. The potential fields are typically modelled as continuous and differentiable functions, leading to an imprecise description of the obstacle's shape and dimension. This restriction could result in unsatisfactory obstacle avoidance when the vehicle manoeuvres through tight environment. In contrast to the local potential field techniques, the so-called global path-planning algorithms, by which the overall workspace is accounted for during motion planning, can overcome not only the local minima problem but also guarantee tight obstacle avoidance. These algorithms begin by first obtaining a workspace representation explicitly incorporating the obstacles. A basic and simplest representation is to decompose the workspace into cells via a regular grid [53], but more efficient representations have been developed to reduce the memory cost for storing representations: Quadtrees [54] have been utilized for two dimensional spaces, and oct-tree [55] for three dimensional spaces. Roadmap type representations are utilized to extract a skeleton graph from the workspace, which keeps the connectivity between the nodes of feature points. Voronoi diagram [56] and Freeways [57] are widely used representations for global motion planning problem. The visibility graph [58] [59] is

another method for obtaining a graph by connecting the edges (or, vertices) of obstacles from a given start and end points. Collin's decompositions [60] or probabilistic roadmaps [61] also become efficient representation methods for global path-planning.

Having the workspace represented by a partition of finite cells (region) or a complete graph, a path-planning problem is then reduced to finding a sequence of neighboring cells or nodes in the graph satisfying certain extremal criteria. Several optimal searching algorithms have been proposed in the literature. The use of dynamic programming was proposed in [62] for a robot manipulator. An evolutionary search method was successfully used to compute near-optimal paths through obstacles and dynamically changing environment [63] [64]. The randomized rapid exploring heuristic search in [65] forms a subset of stochastic optimization [66] for a global path-planning, in which the search tree is expanded online towards randomly generated target states. If a graph is obtained for workspace representation, the A* algorithm [6] [67] or the Dijkstra's algorithm [68], are the most common algorithms to find the shortest path from a known initial location to a target location on the graph.

Another technique used to effectively represent the alternate paths generated by a path planner is to use B-splines. B-splines allow a parametric curved path to be described by only a few control points rather than the entire curve segmented which could be as many as thousands of sections, depending on the length and curvature of the path [69]. Recent methods include modelling the trajectory using splines based on elastic band theory [70] [71] and interactive manipulation of control points for spline trajectory generation [72] [73]. A series of cubic splines was employed in [74] to connect the straight line

segments in a near-optimal manner, whereas the algorithm presented in [75] yields extremal trajectories that transition between straight-line path segments smoothly in a time-optimal fashion. A path-planning algorithm over a dynamic environment has also been proposed in the literature. In order to deal with a dynamically changing environment, it is necessary for the path-planning algorithm to re-plan as often as possible in accordance with the new information about the environment. Several replanning strategies have been proposed for locally-directed wandering [76], local modification of initial path [77], and obstacle perimeter detouring [78].

Most of the search methods discussed above do not incorporate dynamic models of the vehicles. This in turn, results in certain local path segments being hard for the vehicles to follow under kinematic constraints. Several authors have proposed to smoothen these segments by taking the kinematic constraints into account [8] [9], but no explicit analysis on the optimality variation due to local path smoothing has been given. Recently, mathematical optimization has been adopted by many researchers to accommodate dynamic vehicle models in the path-planning problems. Nonlinear programming was used in [79] [80] and the use of mixed integer linear programming (MILP) in the path-planning problem was introduced in [81] [82]. MILP allows inclusion of discrete logic and integer variables in a continuous linear optimization problem. These variables can be applied to model logical constraints such as collision avoidance and obstacle, while dynamic equations of the vehicles are formulated as dynamic constraints. Although the MILP algorithm in trajectory planning can systemically handle hard obstacles of rectangular shapes over an entire workspace and

provide a feasible trajectory if one exists, it cannot handle the computational complexity for a large size trajectory planning problem, and the algorithm is not very useful in dynamically changing environment. The computational complexity arising from the path-planning can be reduced by repeatedly solving a constrained optimization problem online over a finite planning horizon. This scheme is so called model predictive control (MPC), or receding horizon control (RHC), originated from the chemical process industry and has received great attention due to its broader applications over various research fields. A good survey on MPC/RHC can be found in [83] [84] and the relevant texts [85] [86].

In recent years, receding horizon control has been introduced to solve the problems of trajectory generation and collision avoidance of multiple vehicles: Fuller et al [87] presented a MPC-based trajectory generation method for a flight vehicle with nonlinear dynamics by using a feed forward nominal trajectory to pose a time-varying linear, convex optimization problem. This result is extended in [88] [89] to account for obstacle avoidance trajectory planning. More recent publications regarding the implementation of nonlinear MPC for UAVs are found in [90] [91]. The benefit of the MPC approach is that it can allow one to effectively handle multi-variable systems with inputs and states constraints while less computational cost requires. On the other hand, an alternative receding horizon approach based on MILP was introduced in [92] [93]. In this approach, a discrete point mass model was employed as a finer dynamics model for a two-dimensional vehicle under the velocity and force limits, which correspond to state and control constraints, respectively. A finer level of optimization was executed over a fine

resolution of planning horizon using MILP solver, whereas a coarser level of optimization for the shortest path was done by searching a visibility graph that was expanded from a goal point through edges of each hard obstacle. These two-step optimizations achieve the shortest path over a long time-scale while planning a dynamically feasible path over a short time-scale at the reduced computational cost. Bellingham et al and Kuwata et al [94] [95] proposed a stable trajectory for highly complex environment by putting turning circles at each obstacle's corner, which takes into account the kinematic constraints of the maximum turning rate of vehicles. In addition, a safety condition of the algorithm was imposed by the form of loiter circles as a backup plan in case of the algorithm failure [96]. Although this algorithm has been successfully tested and executed for UAV guidance [97] [98], the algorithm has certain underlying weaknesses. First, since the discrete MILP solver incorporates linearized equations, in conjunction with binary variables as the constraints of turning circles, velocity and force limits, and so on, a better accuracy of solution could be obtained from using more binary variables to accommodate detailed constraint equations. This results in increased computation time that may not be suitable for real-time implementation. Also, since the obstacles are limited to rectangular shapes so as to apply the visibility graph search, various shapes of obstacles other than rectangles will require a conservative modelling of obstacles by polygons that completely enclose such obstacles, which might end up with a conservative path.

The key limitations and benefits of the existing methods are summarized below.

Cell decomposition approaches [53] [54] [60]: For successful implementation of exact cell decomposition, it is required that the geometry of each cell has to be simple in order to generate a connectivity graph. This requires computing the adjacency of two cells to find a path crossing the portion of the boundary shared by the two cells. A path can easily be found by following the adjacent, decomposed cells through free space. The trade-off of this method is that it involves a complex mathematical process. Cell decomposition methods are accurate, yet computationally expensive. They are only effective when the environment contains a countable number of obstacles.

Dijkstra's algorithm [68]: A polygonal path is generated from the Voronoi graph by applying Dijkstra's algorithm and the length of the path to compute the cost is considered. However the generated paths do not represent optimal paths as they considered only the distance and the paths may not be safe and smooth. This approach is lacking in adaptation capability as it is limited to stationary obstacles in a dynamic environment. In addition, it is computationally intensive and not suitable for real-time applications. Thus there is a need for a highly adaptive route planning instead of the rigid conventional approaches.

Mixed integer linear programming (MILP) [81] [82]: The issues of path planning and task timing constraints are addressed using agent availability time windows in the MILP formulation. However for more complex problems involving multiple vehicles and targets, the solution time of the MILP [92] [93] becomes impractical for on-line implementation as it requires the cost function to be linear. This introduces a time constraint that is hard to meet when the number of vehicles and the number of obstacles

increase. The number of binary variable is directly related to the complexity of the environment and the complexity of the task. Thus the computational time grows exponentially with the number of binary variables involved in the optimization problem. The success of UAV missions is highly dependent on the effectiveness and robustness of the algorithms.

Probabilistic roadmap methods [40] [41] [61]: Although they are quite useful in situations involving repeated motion in static domains, these techniques are difficult to apply in real environments. When one builds a roadmap of nodes and edges in the free configuration space, there is the issue of updating the roadmap whenever a change in the environment occurs.

Linear Programming (LP) [79] [80]: Linear programming has been widely adopted in small-scale path planning problem. However, because of its computationally intensive nature, the computational complexity of linear programming may increase exponentially with the size of the problem domain.

Dynamic Programming (DP) [62]: Given an optimization problem, dynamic programming seeks to determine a global optimum solution. It tries to limit the computational complexity by partitioning an optimization problem into stages. However, its memory intensive nature limits its applications in problems with high dimensions.

Biologically inspired algorithm: Meta-heuristic algorithms, such as evolutionary algorithms [12] [14] and genetic algorithms [13], are global optimization algorithms which are capable of providing decent results even when the problem dimension is high.

This characteristic makes EA and GA highly favorable in solving high dimensional path planning problems. However, in order to avoid local optimum solutions, meta-heuristic algorithms require extra caution in choosing their tuning parameters. Furthermore, the time required for convergence may increase as the size of the search space increases, thus rendering them unsuitable for real-time large-scale optimization problems. In Memetic Algorithms [121], the ability of a local improvement procedure to locate the local optima with high accuracy can complement the ability of Evolutionary Algorithms to capture a global view of the search space. Incorporating local improvement introduces explicit refinement operators that can produce high quality solutions by enhancing the exploiting ability of the search algorithm without impeding on exploring capabilities. MA aims to exploit the complementary advantages of EAs (i.e., generality, robustness, global search efficiency), and problem-specific local search/improvement (i.e., exploiting application-specific problem structure, rapid convergence toward local optima). The metaphorical parallels on the one hand, to Darwinian evolution; on the other hand, between memes and domain specific individual learning heuristics are captured within Memetic Algorithms, rendering a methodology that balances well between generality and problem-specificity. MAs have been demonstrated to converge to high quality solutions more efficiently than their conventional counterparts [121].

After reviewing various current methods and research being done for path-planning of UAV, we consider the goals of the current research relative to the existing literature works. Our focus in this thesis is on evolutionary approach to investigate path-planning and mission-planning problems for multiple UAVs. The nature inspired evolutionary

approach which is based loosely on the evolution theory has the most potential to deal with path optimization and mission-planning problem of UAVs. It is a form of population-based meta-heuristic that mimics the process of evolution. The natural selection and genetic dynamics of the evolution process have inspired the field of evolutionary computation. The evolutionary agents use randomness to generate optimum solutions in a more direct and intelligent manner than the conventional path-planning approaches. Unlike point-to-point conventional approaches mentioned above, the search emphasis is on a complete solution encompassing all the waypoints that constitute the overall mission. The flexibility of such an approach is evident when one has to further undertake tasks division or allocation to multiple vehicles basing on the overall solution attained by the evolutionary method. Furthermore, an evolutionary approach being stochastic in nature incorporates inherent randomness in the search; hence variations in solutions generated and at the same time satisfying some pre-specified requirements or constraints are highly desirable. With an evolutionary path planner, the problem can be scaled up, able to deal with large-scale path-planning for multiple UAVs.

In this thesis, we summarize the operations of the UAVs (as shown in Figure 2) as follows:

- 1) Path-planning to maneuver an UAV through a series of way points in a radar prone environment;
- 2) Cooperative mission-planning to control multiple UAVs to perform mission-specific tasks at different way points;

3) Multi-criteria mission-planning to optimize the overall mission in a radar zone.

Figure 2: UAV operations in radar zones

These UAV operations are illustrated in detail in the following chapters. The first UAV operation (to be discussed in Chapter 3) is known as the path-planning problem. A path-planning problem refers to the problem of finding optimum paths between waypoints in a radar prone environment. The other UAV operations (which will be presented in Chapter 4 and 5) are characterized as mission-planning problems. In Chapter 4, we look into the cooperative control of mission-planning for multiple UAVs so as to allow them to accomplish their missions in radar zones. In Chapter 5, we draw our focus on optimizing a multi-criteria mission-planning problem, catering to the requirements of different mission constraints. The mission has to be optimized according to some aspects such as minimum time, minimum fuel or/and minimum danger exposure since a mission must be completed satisfying certain criteria as dictated by the mission scenario.

Details on the formulation of the problem which includes aspects of metric for stealth in a mission are presented in the following chapters. These chapters will also discuss the design and theory involved in the evolutionary computation for our path-planning and mission-planning problem.

Chapter 3: Radar-Aware B-Spline Trajectory Generation for UAV

In this chapter, we present a radar-aware path-planning method (as illustrated in Figure 3.1) in guidance of autonomous UAVs; generating stealthy paths through a set of radar sites of known locations. A preliminary path consisting of a series of straight-line segments is first generated. Subsequently, the path is refined using a series of B-splines in lieu of the straight line segments, resulting in a cost-effective path traceable by the UAVs. Simulated results are presented in Section 3.4, illustrating the potential benefits of such algorithms.

Figure 3. 1: Flow chart of the radar-aware path-planning method

For convenience and ease of referencing, the symbols and abbreviations used in our discussions are summarized in the listing below:

UAV	Unmanned aerial vehicle
RCS	Radar cross section
S	Signal energy received by the radar
P_{avg}	Average power transmitted by the radar
G	Gain of the radar antenna
σ	RCS of the target
A_e	Effective area of the radar antenna
t_{ot}	Time of the radar antenna pointed at the target
R	Range to the target
S_{min}	Minimum signal energy detected by a radar
R_{max}	Maximum range determined by the minimum signal energy
$\aleph$	Navigational space
r_j	Memory range of the UAV
G	A topological graph
V	Set of nodes v
u	A preceding node
$\mathcal{E}$	Connectivity between the nodes
$\oplus$	Direct sum of combining several objects into one general object
ΔC_R^j	A union of risk costs $c_{k,l}^j$
$c_{k,l}^j$	Risk cost of dimension $1/2^j \times 1/2^j$
$\ x\ _\infty$	Maximum norm
$\ x\ _2$	Euclidean norm
$cell_G(v)$	Centre of the corresponding node edge (u, v) of G
w	Cost weighting parameter
$b(u)$	One-dimensional b-spline function
b_j	B-spline control point
$N_j^q(u)$	B-spline functions of degree d which are specified over a non-decreasing knot sequence $\{u_k\}$ such that $u_0 \leq u_1 \leq \dots \leq u_{m+q+1}$

m	Sum of the control points
q	B-spline order
l_i	Length of edge i
N	Number of radars
M	Number of discrete point
$k(u)$	Curvature of the B-spline curve at each knot parameter
$d_{1/6,i,j}$	Distance from the $\frac{1}{6}$ th point on the i^{th} edge to the j^{th} radar
$S(i)$	Known region up to $t = t_i$
$W(x_i, r_j)$	Effective sensory region at t_i
x_i	Current location of the UAV at t_i
r_j	Sensor range

3.1 Introduction

UAVs are usually mobilized to carry out critical missions in high-risk environments, particularly in situations where it may be hazardous for human operators. However, such missions demand a high level of stealthiness, which has a direct implication on the safety and success of the mission. Therefore it is important to minimize the risk of detection or more specifically, the probability of detection of the UAVs by radars. Currently UAVs use rudimentary guidance technologies, such as following pre-planned or manually provided waypoints. In order to deal with more complex mission or task, UAVs will require more advanced guidance capabilities, in particular ones that increase the vehicle autonomy [18].

In this chapter, we describe a novel approach of generating paths within radar zones by considering path optimality based on radar risk cost. The navigational path between the waypoints is considered when minimizing the exposure to a radar source. Another consideration is the distance between the UAV and the radars, which should be optimized in order to minimize the energy received at the radar. Also, the generated trajectory should be of minimal length, subject to the stealth constraint and satisfying constraints imposed by the aircraft's dynamics. The following section gives a description of the problem model used for measuring stealth and illustrates the calculation of the extent of radar risk. Section 3.4 presents the solution approach of the route planner, substantiated with simulated results. The summary of this chapter is presented in Section 3.8.

3.2 Problem Model

To describe the problem, consider a 2-dimensional navigational space scenario shown in Fig. 3.1 The objective is to find a stealthy path which minimizes the UAV's exposure to radar sites (small triangles) from the current UAV position (circle) to the target location (diamond) by taking into account the UAV kinematic constraints as well as available computational resources.

Figure 3. 2: Navigational space with known radar sites

To generate a stealthy path for the UAV, we have to take into account the power energy reflected to radar site as the radar tracks the location, speed, and direction of an aircraft [99]. The range at which radar can detect an object is related to the power transmitted by the radar, the fidelity of the radar antenna, the wavelength of the radar signal, and the radar cross section (RCS) of the aircraft [100] [101]. RCS is integral to the development of radar stealth technology, particularly in applications involving aircraft. For example, a stealth aircraft which is designed to be undetectable will have design features that give it a low RCS. Low RCS offers advantages in detection range reduction as well as increasing the effectiveness of decoys against radar-seeking threats. The size of a target's

image on a radar screen is measured by the RCS and denoted by the symbol σ and expressed in square meters. The RCS is the area a target would have to occupy to produce the amount of reflected power that is detected back at the radar [18]. The computation of the energy received by a radar is given by the radar range equation [100]

$$S \cong \frac{P_{avg} G \sigma t_{ot} A_e}{(4\pi)^2 R^4} \quad (3.1)$$

where S is the signal energy received by the radar, P_{avg} is the average power transmitted by the radar, G is the gain of the radar antenna, σ is the RCS of the target, A_e is the effective area of the radar antenna, t_{ot} is the time the radar antenna is pointed at the target, and R is the range to the target. Every radar has a minimum signal energy that it can detect, S_{min} . This minimum signal energy determines the maximum range (R_{max}) at which a given radar can detect a given target.

$$R_{max} \cong \sqrt[4]{\frac{P_{avg} G \sigma A_e t_{ot}}{(4\pi)^2 S_{min}}} \quad (3.2)$$

According to Eq. (3.2), the distance at which a target can be detected for a given radar configuration varies with the fourth root of its RCS. Fig. 3.2 shows a plot to give an indication of just how little radar power is typically reflected back from the target and received by the radar. In this case, the target presents the same aspect to the radar at ranges from 1 to 50 miles [69]. At a range of 50 miles, the relative power received by the radar is only 1.6×10^{-5} % of the strength at one mile. This figure explicitly

demonstrates the significance of the energy dissipation effect with the distance, as well as the sensitivity of radars to detect targets at short ranges.

Figure 3. 3: Reduction in the received power of target with range

We involve the aircraft range, radar cross section and transmission power in this study; therefore G , A_e , t_{ot} are considered inconsequential and the energy received can be approximated by

$$S_{min} \propto \frac{P_{avg}\sigma}{R_{max}^4} \quad (3.3)$$

Given a single radar located at the origin and a single aircraft travelling from initial point to the destination, Pachter [102] showed that an objective function for minimizing P_{avg} , is

$$J = \int_0^{\frac{l}{v}} \frac{1}{R^4(t)} dt \quad (3.4)$$

where v is the constant speed of the aircraft and l is the path length. A closed form solution to this problem was obtained using the calculus of variations, with the limitation

that the aircraft traverses an angle with respect to the radar of less than 60° . Beyond this limit, the optimal path length is infinite but the cost remains finite [99]. The objective cost function, Eq. (3.5), is augmented to include the rest of the radar at the navigational space, giving

$$J = \int_0^{\frac{l}{v}} \sum_{i=1}^m \frac{\alpha_i}{R_i^4(t)} dt \quad (3.5)$$

where constant α_i is defined by the type of radar, specifically by its power, radar cross section, and operational settings [99]. It is assumed that the radar zones are known *a priori*; the details of the radar zone can be acquired using satellite data or from surveillance data. We assume that the UAV's radar signature is uniform in all directions and is proportional to $1/R^4$, where R is the distance between the UAV and the radar; the UAVs are assigned simplified flight dynamics and performance constraints in two-dimensions; radars are monostatic; and all tracking radars are given simplified detection properties [18]. The radar risk cost involves the amount of energy received by radar at several locations. For the sake of simplicity, we assume that the radar is located at (x, y) of the navigational space. Let Eq. (3.6) denote R , the aircraft range from the radar to the aircraft [18].

$$R = \sqrt{(\Delta x)^2 + (\Delta y)^2} \quad (3.6)$$

We use Cartesian coordinates as the formulation of UAV distance from the radars and take into account a transformation of the integral by defining the path as $f = f(x, y)$.

Letting $v = \frac{ds}{dt}$ or $dt = \frac{ds}{v}$, the path length ds is specified in Cartesian coordinates by

$$ds = \sqrt{1 + \left(\frac{dy}{dx}\right)^2} dx = \sqrt{1 + \dot{y}^2} dx \quad (3.7)$$

The distance from each radar to the current location of UAV (x, y) is simply,

$$R_i(x, y) = \sqrt{(x - x_i)^2 + (y - y_i)^2} \quad (3.8)$$

where (x_i, y_i) are the known radar locations. Eq. (3.5) is then transformed into Cartesian coordinates, yielding the cost associated to radar risk as:

$$J = \int_{x_0}^{x_f} \left(\sum_{i=0}^m \frac{\alpha_i}{R_i^4(x, y)} \right) \sqrt{1 + \dot{y}^2} dx \quad (3.9)$$

The boundary conditions are the UAV's base location (x_0, y_0) and the target location (x_f, y_f) . The next step is to compute a stealthy path, steering the UAV clear and around known radar locations.

3.3 Representation of Navigational Space

We classify the radar region into different level of risk costs [103] as shown in Fig. 3.3. Namely, the path from an initial state to a final state may comprise of cells over different radar risk costs, depending on the distance from the current position of the UAV. At the low radar risk level, the path is resolved through feasible states hence minimizing UAV's exposure to radars. Subsequently, we employ our radar-aware path-planning method to find an optimal path on the radar region. The optimal path should carry minimal radar risks, and adhere to physical and motion constraints [69].

Figure 3. 4: Graphing of the different risk levels based on knowledge of the radar zone

We construct approximation of the radar sites at distinct risk levels $J_{low} \leq j \leq J_{high}$ at ranges r_j from the position of the UAV, (x_0, y_0) in the sense that the risk j is measured for all waypoints in the region,

$$W(x_0, r_j) \triangleq \{x \in \mathbb{R}^2 : \|x - x_0\|_\infty \leq r_j\} \quad (3.10)$$

where $\aleph$ is a navigational space and r_j denotes the memory range of the UAV. By this, we imply that the lower risk level J_{low} is applied for waypoints close to the low risk cost region depending on the position of the UAV to the radar. The following shows the risk cost decomposition on $\aleph$:

$$C_R = \Delta C_R^{J_{low}} \oplus \dots \oplus \Delta C_R^{J_{high}} \quad (3.11)$$

where ΔC_R^j is a union of risk costs $c_{k,l}^j$ of dimension $1/2^j \times 1/2^j$. The cost for travelling along an edge is derived from the radar risk cost and the cost associated to radar risk involves the amount of energy received by radar based on the distance from the UAV to the radar [18]. With the risk cost determined for each of the cells as shown in Fig. 3.4, the waypoints and connectivity links can be searched using our proposed algorithm to find a preliminary lowest cost path between the UAV's initial and target location.

Figure 3. 5: Numerical assignment of risk cost to cells in the navigational space

3.4 Solution Methodology

3.4.1 Connectivity Relationship between Nodes

We first assign a topological graph $G = (V, \mathcal{E})$ to C_R as follows. The nodes which belong to the set V represent the risk cost $c_{k,l}^j$ in C_R and the edges in the set $\mathcal{E}$ represent the connectivity between these nodes. We construct the link directly from the nodes that are associated with risk costs $c_{k,l}^j$ and compute the adjacency list of the nodes. More specifically, consider a node $c_{k,l}^{j_0}$ at risk level j_0 , whose dimension is $\frac{1}{2^{j_0}} \times \frac{1}{2^{j_0}}$ and located at (k, l) . A node will be identified as safe if it is associated with low radar risk. We then examine each node in $W(x, r)$ recursively until all safe nodes are found. Next, we look for the adjacent nodes so as to compute the adjacency list of the nodes with the current safe node. It is essential to devise a generic method to find the adjacency relationship between the nodes since we deal with nodes of different risk costs. The risk costs of the nodes are examined so as to search the low risk cost node that is adjacent to the current node. A pseudo-code implementation of the search algorithm is given in Listing 3.1. Note that the subroutine Reconstruct Scan is called recursively to identify safe nodes intersecting with the $W(x_0, r_j)$. After a node identified as a safe node, the search algorithm continues on establishing the links from the current node to adjacent safe nodes along Left, Left-Top, Top, and Right-Top direction using the recursive refinements of search level.

BEGIN ADJACENCY SEARCH ALGORITHM

```

{
  Initialize graph structure  $G = (V, E)$ ;
   $C_R \leftarrow$  Compute risk cost ( $W, j$ );
   $(K(j), L(j)) \leftarrow \aleph(x_0, r_j)$ ;
  for ( $k = 0: k_{r_j}$ ) {
    for ( $l = 0: l_{r_j}$ ) {
      if  $(k, l) \in (K(j), L(j))$ 
        Reconstruct_Scan( $k, l, j, C_R$ );
      else
        AddNode_EstablishLinks( $k, l, j, C_R$ );
    }
  }
}

```

END ADJACENCY SEARCH ALGORITHM

Reconstruct_Scan(k, l, j, C_R)

```

{
   $C_R \leftarrow$  Compute risk cost ( $C_R(k, l; j)$ );
   $(K(j+1), L(j+1)) \leftarrow \aleph(x_0, r_{j+1})$ ;
  for ( $\bar{k} = 2k: 2k+1$ ) {
    for ( $\bar{l} = 2l: 2l+1$ ) {
      if  $(\bar{k}, \bar{l}) \in (K(j+1), L(j+1))$ 
        Reconstruct_Scan( $\bar{k}, \bar{l}, j+1, C_R$ );
      else
        AddNode_EstablishLinks( $\bar{k}, \bar{l}, j+1, C_R$ );
    }
  }
}

```

AddNode_EstablishLinks(k, l, j, C_R)

```

{
  /*Add Node*/
   $v \leftarrow c_{k,l}^j \sim a_{j,k,l}$ ;
   $V(G) \leftarrow v$ ;
  Set  $d_{j',k,l}^i \leftarrow 0, \forall j' \geq j$ ;

   $c_{k,l}^j \leftarrow$  Get current node ( $k, l, j, C_R$ );
   $j_s \leftarrow$  Determine basic search level ( $c_{k,l}^j$ );
}

```

```

 $(k_s, l_s) \leftarrow \text{Get parent node index } (c_{k,l}^j, j_s);$ 

/*Establish Links*/
LinkLeft( $k, l, j, C_R$ ){
     $k_s = k_s - 1;$  } /*To examine left adjacent node*/
LinkLeftTop( $k, l, j, C_R$ ){
     $k_s = k_s - 1;$ 
     $l_s = l_s - 1;$  } /*To examine left-top adjacent node*/
LinkTop( $k, l, j, C_R$ ){
     $l_s = l_s - 1;$  } /*To examine top adjacent node */
LinkRightTop( $k, l, j, C_R$ ){
     $k_s = k_s + 1;$ 
     $l_s = l_s + 1;$  } /*To examine right-top adjacent node */
}

```

Listing 3.1: Pseudo-code implementation of the adjacency search algorithm

Subsequently, we make use of heuristic search algorithm to find a path from an initial location to the target location. The algorithm utilizes a heuristic estimate $h(v)$ that ranks each node v by a best cost estimate to reach the goal from the current node [104]. The algorithm visits the nodes in the order of the heuristic estimate, so the A^* algorithm is known as a best-first search algorithm. The key element of the A^* algorithm is that it expands each node from the priority queue that is ordered by (lower value has higher priority)

$$f(v) = g(v) + h(v) \quad (3.12)$$

where the cost $g(v)$ is the actual cost of the path up to v , i.e. the sum of the edge costs from the initial node, and $h(v)$ is the heuristic estimate at v . When a node u is expanded, the adjacent nodes to the current node are exploited. Let v be the adjacent node, then it follows that we evaluate the actual cost $g(v)$ to see if the transition from u

to v results in lower cost than any other transitions to v . The algorithm then sets a back-pointer $\pi(v)$ by its preceding node u . This process iterates until the goal node is reached and no other nodes have a lower cost to the goal. The A^* algorithm is complete in the sense that it is always guaranteed to find a solution if a solution exists. In addition, if the heuristic function $h(v)$ is admissible, that is, it uses an underestimate of the actual cost of reaching the goal, then A^* is optimal. Details about the implementation of the A^* algorithm can be found, for instance, in [105].

We associate with each node $v \in G$ a cell $c_{k,l}^j$. Since G is a topological graph, we may associate each node v with some point $x \in c_{k,l}^j$. Moreover, we decide on the centre of the node without loss of generality. Let $cell(v)$ denote the centre of the corresponding node. If $x \in c_{k,l}^j$, we write $v = node_G(x)$ followed by assigning an edge cost to each directed edge (u, v) of G as follows:

$$\mathcal{E}(u, v) = L(cell(v)) + \alpha ||cell(u) - cell(v)||_2 \quad (3.13)$$

where $\alpha \geq 0$ is a weight constant. The first term in Eq. (3.13) is proportional to the probability that the target node is close to radars, while the second term penalizes the distance between $cell_G(u)$ and $cell_G(v)$.

Suppose that we are given a path of $q + 1$ consecutive nodes in G as follows:

$$P = (v_0, v_1, \dots, v_q) \quad (3.14)$$

We can then assign a traversal cost to each node in the path P , induced by

$$g(v_i) = g(v_{i-1}) + \mathcal{E}(v_{i-1}, v_i), \quad i = 1, \dots, q \quad (3.15)$$

The value of $g(v_k)$ represents the (accumulated) cost of the path from v_0 to v_k ($k \leq q$), i.e., the weight of the edges followed up to v_k . We use the following heuristic estimate

$$h(v) = ||cell(v) - cell(v_f)||_\infty \quad (3.16)$$

where $v_f = node_G(x_f)$.

The A^* algorithm then finds a path that minimizes the cost in Eq. (3.15) to the goal node, or determines that such a path does not exist. However, the efficiency of the search process in a graph search algorithm depends on the number of nodes that are expanded. In order to enhance the search process, the algorithm is further improved by including a cost weighting parameter w . Thus we enhance the evaluation function by adding the cost weighting parameter w to the $f(v)$ as follows [18]:

$$f(v) = w * g(v) + (1 - w) * h(v) \quad (3.17)$$

where w is a value in the range of $[0,1]$. Based on Eq. (3.12), the search algorithm may not guarantee the consistency of permissible condition; therefore the generated path may not be optimal. In spite of this, by introducing w in Eq. (3.17), the $h(v)$ plays a significant role in the search processes when w value is smaller; the number of nodes to be expanded would be decreased. The choice of w between 0 and 1 gives the flexibility

to assign weight on exposure to threats or fuel expenditure depending on the particular mission scenario. For example when $w = 0$, the search process would proceed from the initial location to the target location by expanding the least number of nodes. However the generated path is not stealthy and may cause UAV to enter into high radar exposure region. A w value closer to 0 puts emphasis on shortest paths, with little regard for the exposure to radars, while a value closer to 1 put emphasis on threat exposure at the expense of longer path length. For this case, the cost weighting parameter w was chosen to give a reasonable trade-off between the path length and proximity of the path to threats [106]. The experimentally defined value for w has to be in line with our algorithm objective which can be computed in linear time. The detailed algorithm substantiated with experimental results is demonstrated in the following section.

3.4.2 Radar-aware Path-Planning Algorithm

The proposed radar-aware path-planning algorithm proceeds as follows. Starting from $x(t_0) = x_0$ at time $t = t_0$, we construct using the approach of a risk cost decomposition $C_R(t_0)$ of W . A topological graph, and the adjacency list of its nodes are obtained using the approach of Section 3.4.1. Let the corresponding graph be $G(t_0)$ and let $v_1^0 \in G(t_0)$ and $v_f^0 \in G(t_0)$ be the initial and the goal nodes such that $v_1^0 = \text{node}_{G(t_0)}(x_0)$ and $v_f^0 = \text{node}_{G(t_0)}(x_f)$, respectively. The algorithm [18] traverses various paths from the initial node to the goal node. Starting with the initial node, it maintains a priority queue of nodes to be traversed. At each step of the algorithm, the node with the lowest $f(v)$ value is removed from the queue, the f and h values of its neighbours are updated

accordingly, and these neighbours with the least risk cost are added to the queue. The conditions of adding a node to the queue are as follows:

- The node is movable and does not exist in the queue;
- The distance between the starting node and the current node is shorter than the earlier estimated distance;
- The node is not in high radar risk region.

If the three conditions are satisfied, then the current node will be added to the queue. Using the proposed algorithm we compute a path $P(t_0)$, of free and mixed nodes from v_1^0 to v_f^0 in $G(t_0)$ assuming that such a path exists. Let $P(t_0)$ be given by an ordered sequence of $l_0 + 1$ nodes as follows:

$$P(t_0) = (v_0^0, v_1^0, \dots, v_{l_0-1}^0, v_{l_0}^0 = v_f^0) \quad (3.18)$$

We assume that v_1^0 is a safe node owing to the low risk cost representation of W close to x_0 . The UAV subsequently navigates from v_0^0 to v_1^0 . Let now t_1 be the time the UAV is at the location $x(t_1) = cell_{G(t_0)}(v_1^0)$ and let $C_R(t_1)$ be the multi-level risk cost decomposition of W around $x(t_1)$ with a corresponding topological graph $G(t_1)$. Applying again the proposed algorithm we compute a path in $G(t_1)$ from $v_0^1 = node_{G(t_1)}(x(t_1))$ to $v_f^1 = node_{G(t_1)}(x_f)$ if such a path exists. Let $P(t_1)$ be given by the ordered sequence of $l_1 + 1$ nodes as follows:

$$P(t_1) = (v_0^1, v_1^1, \dots, v_{l_1-1}^1, v_{l_1}^1 = v_f^1) \quad (3.19)$$

The UAV subsequently navigates to v_1^1 at location $x(t_2) = cell_{G(t_1)}(v_1^1)$ at time t_2 . In general, at time t_i , it is assumed that the UAV is at location $x(t_i)$. A risk cost decomposition $C_R(t_i)$ of W is constructed at $x(t_i)$ with a corresponding graph $G(t_i)$. The algorithm yields a path $P(t_i)$ in $G(t_i)$ of length $l_i + 1$,

$$P(t_i) = (v_0^i, v_1^i, \dots, v_{l_i-1}^i, v_{l_i}^i = v_f^i) \quad (3.20)$$

where $v_0^i = node_{G(t_i)}(x(t_i))$ and $v_f^i = node_{G(t_i)}(x_f)$. This iteration process terminates at some time t_f when the UAV navigates from $x(t_f)$ to x_f . A pseudo code of the radar-aware path-planning algorithm is given in Listing 3.2.

```

BEGIN PATH PLANNING ALGORITHM
{
   $i = 0$ ;
   $x(t_i) \leftarrow x_0$ ;
  while ( $\|x(t_i) - x_f\|_\infty \neq 0$ )
  {
    compute  $L(x, i)$  for all  $x \in W$ ;
    construct  $C_R(i)$  at level  $J_{low}$ ;
    construct  $G(i) = (E(i), V(i))$ ;
     $v_1^i \leftarrow node_{G(i)}(x(t_i))$ ;
     $v_f^i \leftarrow node_{G(i)}(x_f)$ ;
     $P(i) \leftarrow ComputePath(v_1^i, v_f^i, G(i))$ ;
    if  $P(i) = \emptyset$ 
      report FAILURE; break;
     $x(t_{i+1}) \leftarrow cell_{G(i)}(v_2^i)$ ;
    Move to  $x(t_{i+1})$ ;
     $i \leftarrow i + 1$ ;
  }
}
END PATH PLANNING ALGORITHM

```

Listing 3.2: Pseudo-code implementation of proposed path-planning scheme

Note that the actual path followed by the UAV is given by the sequence of nodes $\{node_{G(t_0)}(x(t_0)), node_{G(t_1)}(x(t_1)), \dots, node_{G(t_f)}(x(t_f))\}$. The proposed radar-aware path-planning algorithm accommodates the need of the real time implementation by managing the amount of the information to process, thus computing an immediate optimum path within the allowable time scale. The objective of the UAV is to follow a safe path to the target location while circumventing the radars. Samples of experimental results are shown in Fig. 3.5(a)-(c).

Figure 3.5(a): Generated path with minimal radar risk at time step $t = 5$

Figure 3.5(b): Generated path with minimal radar risk at time step $t = 21$

Figure 3.5(c): Generated path with minimal radar risk at time step $t = t_f$

Specifically, Figs. 3.5(a)–(c) illustrate the path evolution at distinct time steps while navigating to the target location. Fig. 3.5(a) shows the UAV's position at time step $t = 5$ along with the optimum path to the final destination at that time. The actual path followed by the UAV is drawn by a solid line. Similarly, Fig. 3.5(b) shows the UAV's position at time step $t = 21$. In particular, Fig. 3.5(b) shows that as the UAV gets nearer to the radar zone, it identifies the presence of radars and plans the path to minimize the

exposure from radars. Finally, the UAV reaches the target location x_f in a radar-aware manner, as seen in Fig. 3.5(c).

The evaluation functions, Eq. (3.12) and Eq. (3.17) are tested empirically. The test results are compared in terms of the number of nodes visited [18]. During the test, the cost weighting parameter w is varied from 0.0 to 1.0 and the test result is demonstrated in Table 3.1.

Cost Weighting Parameter, w	Expanded Nodes	Cost of the Generated Path
w is not used	2405	103
0	235	117.5
0.1	219	112.5
0.2	215	110.5
0.3	215	106.5
0.4	318	104
0.5	2255	103
0.6	4975	101
0.7	7073	100.5
0.8	7521	99
0.9	7766	97
1	8085	97

Table 3. 1: Test results

The evaluation function without cost weighting parameter w leads to a global search and hence it is computationally inefficient. By including cost weighting parameter w , the expanded nodes change according to w . As illustrated in the table, when w increases, the $f(v)$ plays the dominant role and the cost of the generated path decreases. At the same time the number of expanded nodes increases. A w of closer to 0 results in the shortest path, with little consideration for the exposure to radars; whereas a value closer to 1 favours paths that avoid threat exposure at the expense of longer path length. The cost weighting parameter w was chosen in between 0.1 to 0.3 to give a reasonable trade-off between proximity of the path to threats and the path length. The illustration of the generated path shown in Fig. 3.5(c) is neither the safest path nor the shortest path, but represents a compromise between the two objectives [18].

3.4.3 Comparison

We give special attention to the computational speed during implementation as a typical micro-controller (E.g. Rabbit RCM-3400 microcontroller) has limited computational resources (10,000 instructions per second, and 512 KB RAM for handling variables). Table 3.2 shows the computational cost of the proposed path-planning algorithm. Given a risk measure L , the radar risk cost computation provides the approximation that is used to construct the adjacency relationship between nodes. The proposed algorithm generates an optimum path in the graph associated with the risk cost decomposition.

Multi-level risk cost decomposition	452 [msec]
Construct the connectivity relationship and G	292 [msec]
Compute a path using the proposed algorithm	202 [msec]
Average number of nodes of each G	~200

Table 3. 2: Computational cost of the proposed algorithm

Each iteration of the path-planning algorithm takes 946 [msec] for execution. With the knowledge of the execution time of the proposed algorithm, we actually choose to implement the proposed radar-aware path-planning algorithm every three seconds. Hence, the autopilot manages not only to execute the basic tasks such as data acquisition and processing, inner loops control, and etc., but also to plan a path in a seamless manner. The simulations were carried out on Core 2 Quad 2.67 GHz central processing unit (CPU) and 2 GB double-data-rate random access memory (DDR RAM), derived from codes written in Visual C ++ for implementing both algorithms. The radar-aware path-planning algorithm manages to accomplish the mission objective of reaching the target location in a fewer number of iterations, than the A^* algorithm. This is due to the fact that our proposed algorithm efficiently manipulates the risk cost information so that a preferred path is computed over the approximation of W . The A^* algorithm, however, relies on the path cost information by exploring the environment. Moreover, the computational cost of the radar-aware path-planning algorithm is compared with the original A^* algorithm. The total computation time of the proposed algorithm is obtained by adding the computation time throughout each iteration. For the radar-aware path-planning algorithm, the overall computation time consists of the time for initializing and updating, which is shown to be smaller than that of the original A^* algorithm. The

proposed algorithm is computationally efficient in the sense that it reuses information from the previous step. By the inherent benefit from the risk cost decomposition, the proposed algorithm requires less memory as compared to the original A^* algorithm. Additionally, our proposed algorithm possesses the advantages of scalability for real-time path-planning. The next step is to smoothen the generated path by using a series of B-spline to optimize the path by moving away from high radar risks and improving the navigability of the path for the UAV.

3.5 B-spline Path Smoothing

A better implementation of the guidance and navigation control incorporates a smoothed path by taking into account the dynamic constraints of UAVs. The waypoints can then be connected to generate smoothed path segments, preserving the continuity of curvature between line and arc segments while minimizing the curvature on the curve [107] [108]. In contrast to the explicit consideration of the kinematic constraint of the UAV, a spline-based path generation method has been widely adopted when computing the smoothed, dynamically feasible trajectory for UAVs. A series of cubic splines was employed in [109] to connect straight line segments in a near-optimal manner. The authors in [110] presented an implicit time-parameterization of the trajectory using a B-spline representation. The advantage of employing the B-splines in generating a smoothed path is dictated by the fact that the path can be represented by using smaller number of parameters than using complete description of path. Accordingly, both for path optimization and for on-line implementation, it is straightforward to deal with small number of parameters when generating paths that are subject to the given obstacle constraints at the minimum computational cost. The obstacle avoidance path-planning problem using B-splines involves a constrained optimization such that the path should not only avoid forbidden regions but also become a flyable trajectory. We make use of B-spline function [111] [112] [113] in our algorithm. The key idea is to build the optimal path in small segments. B-spline interpolation requires specification of the curve constraints as well as a suitable degree

of curvature. Hence the curve is made up of an arbitrary number of control points serving as reference points for the evaluated curve [18].

The path is smoothened using splines [112] [114] to join the path segments in order to make the path dynamically feasible for the UAV. Each segment of B-spline curves are then stitched together to the B-spline curve that corresponds to the next path sequence, so that the overall path will remain feasible. To this end, we formulate an optimization problem with constraints being given as risk cost as well as the dynamic constraints of the UAV. Instead of smoothening from an initial position to the goal position of the entire path, we smoothen the path segments according to the position of the UAV. Although the explicit dynamics of the UAV is not directly dealt with for smoothening the path segments, this approach has the advantage of having minimal on-line computational cost since most of the computation is done off-line.

3.5.1 Flight Envelope

An one-dimensional spline function b is expressed by:

$$b(u) = \sum_{j=0}^m b_j N_j^q(u) \quad (3.21)$$

where b_j are control points and $N_j^q(u)$ are the B-spline functions of degree q which are specified over a non-decreasing knot sequence $\{u_k\}$ such that $u_0 \leq u_1 \leq \dots \leq u_{m+q+1}$.

The number of knots is determined by the sum of the control points $(m + 1)$ and the B-spline order $(q + 1)$ [99]. The first and last knots of the sequence should have multiplicity $(d + 1)$ for a B-spline to pass the first and the last control points, in such

that $u_0 = u_1 = \dots = u_q$ and $u_{m+1} = u_{m+2} = \dots = u_{m+q+1}$, respectively. The B-spline basis functions are computed by the well-known *Cox-de Boor* recursion formulas [111] from the degree 0 to q as follows:

$$N_j^0(u) = \begin{cases} 1 & \text{if } u_j \leq u \leq u_{j+1} \\ 0 & \text{otherwise,} \end{cases} \quad (3.22)$$

$$N_j^q(u) = \frac{u - u_j}{u_{j+q} - u_j} N_j^{q-1}(u) + \frac{u_{j+q+1} - u}{u_{j+q+1} - u_{j+1}} N_{j+1}^{q-1}(u) \quad (3.23)$$

The B-spline basis function has several useful properties. Among them, it is well known that B-spline basis function has local support [115], that is $N_j^q(u)$ is a non-zero polynomial over a knot span $[u_j, u_{j+q+1})$, or given any span $[u_j, u_{j+1})$, at most $(q + 1)$ B-spline basis functions of degree q are non-zero. This local support property is important to curve design, since it allows to modify the B-spline curve locally without transforming the entire shape. A planar B-spline curve $b(u) = [b^1(u)b^2(u)]^T$ is expressed in terms of the B-spline basis functions as stated in Eq. (3.21) where $b_j = [b_j^1 b_j^2]^T$ are the control points. It is also assumed that the B-spline curve is clamped at the first and last control points by assigning the $(q + 1)$ multiple knots at the first and last knots. We formulate an optimization problem to calculate a smooth curve which avoids the high radar risk region. We adopt a B-spline curve, due to the inherent smoothness property of B-spline, as a reference path to the UAV. Hence, the path given by a B-spline curve is rendered a flyable path by the UAV. In particular, we let the path

be placed inside a feasible region, thus avoiding the radars. The region separates the radar and feasible regions to yield constraints for optimization problem.

Let the B-spline l be defined by piecewise line segments connecting the control points b_j at the Greville abscissae [116],

$$u_j^* = \sum_{i=j+1}^{j+q} \frac{u_i}{q} \quad (3.24)$$

such that it satisfies $l(u_j^*) = b_j$. Accordingly, an envelope of the B-spline specifies a bound on the distance between b and its l . This envelope should provide a good estimate of the shape of the B-spline by approximating the spline using less information.

The envelopes are expressed in terms of the weighted second difference of the control points,

$$\Delta_2 b_j \triangleq b'_{j+1} - b'_j \quad \text{where} \quad b'_j = \frac{b_j - b_{j-1}}{u_j^* - u_{j-1}^*} \quad (3.25)$$

and over the interval $[u_k^*, u_{k+1}^*](k = 0, 1, \dots, m)$.

We consider designing a smooth curve using a quartic B-spline, whose basis functions are computed from Eq. (3.23) as degree 4 polynomials in terms of the knot parameter u . Hence, the quartic B-spline basis function preserves its continuity up to the third order derivative, thus resulting in the continuity of the derivative of curvature. Without loss of generality, the knot parameter is selected $u \in [0, 1]$, and the first and last knots have multiplicity 5 such as $u_0 = \dots = u_4 = 0$ and $u_{m+1} = \dots = u_{m+5} = 1$. Accordingly, the

B-spline curve will be clamped at the first and last control points. For our optimization problem, we manipulate $(m + 1)$ B-spline control points as $b_j = [b_j^1 \ b_j^2]^T$ ($j = 0, \dots, m$), which have direct influence on the shape of the curve. The quantity of control points is chosen along the complexity of the curve shape. In general, the curve shape is closely related to the given route geometry, subsequently the quantity of control points can be opted for the minimum number required to get a B-spline curve inside the route. By taking into account the number of control points, the knot sequence is initially given arbitrary non-decreasing numbers in $(0, 1)$ and can be altered with knot insertion if the envelopes of the B-spline curve need to be refined [117]. We define four control points P_0 through P_3 , for positioning and modifying the curve. Each control point has a very local effect on the shape of the curve. The two intermediate points P_1 and P_2 are used to specify the endpoint tangent vectors. Hence the curve interpolates P_0 and P_3 while approximating the two other control points. To do this, a weighting function which returns the control points at a given point on the curve is required. The objective is to vary the location of the control points so as to determine a path that minimizes exposure to the radar [18].

3.5.2 Weighting Functions

Three different weighting functions are adopted to compute curves for attaining distinct optimization goals. As a first step, the path is parameterized in both x and y directions. An initial spline knot is fixed at the UAV starting location, while the final spline knot is fixed at the first edge of the path. In the context of spline interpolation, a knot is simply

a point the spline is required to go through [18]. We begin by splitting each edge of the path into three segments as shown in Fig. 3.6.

Figure 3. 6: Splitting of generated path for the purpose of adding middle knots

In this work, we calculate the radar risk cost at several locations along each edge and take the length of the edge into account [99]. The radar risk cost was calculated at three points along each edge: $\frac{l_i}{6}$, $\frac{l_i}{2}$, and $\frac{5l_i}{6}$, where l_i is the length of edge i . The radar risk cost for each edge may be calculated by

$$\mathcal{J}_1 = l_i \sum_{j=1}^N \sum_{i=1}^M \left(\frac{1}{d_{1/6,i,j}^4} + \frac{1}{d_{1/2,i,j}^4} + \frac{1}{d_{5/6,i,j}^4} \right) \alpha_j \quad (3.26)$$

where N is the number of radars; M is the number of discrete point and $d_{1/6,i,j}$ is the distance from the $\frac{1}{6}^{th}$ point on the i^{th} edge to the j^{th} radar. Next, in order to keep the B-spline curve as close as possible to a straight line, for which $\Delta_2 b_j = 0$, we employ the weighting function:

$$\mathcal{J}_2 = \sum_{i=1}^{m-1} (\Delta_2 b_j)^T (\Delta_2 b_j) \quad (3.27)$$

which implicitly minimizes the curvature variation of B-spline curve, thus resulting in a smooth B-spline curve. On the other hand, we assume that the arc length of the B-spline curve is approximately captured by the total length of the control polygon l . Hence, for the shortest path, we employ the weighting function as the sum of the length of the piecewise control polygon:

$$\mathcal{J}_3 = \sum_{i=1}^{m-1} \|l_i\|_2 \quad (3.28)$$

The constraints for the optimization problem comprise of equality constraints stipulate boundary conditions for position and curvature at each end point at $u_0 = 0$ and $u_{m+5} = 1$ as follows:

$$\begin{aligned} b(0) &= p_0, & b(1) &= p_f \\ k(0) &= k_0, & k(1) &= k_f \end{aligned} \quad (3.29)$$

where $k(u)$ is the curvature of the B-spline curve at each knot parameter. The path optimization problem is formulated as follows. Given a knot sequence $\{u_k\}$, two polygonal lines for channel geometry, and boundary conditions for each end point, find a B-spline curve which minimizes the weighting function in Eqs. (3.27) or (3.28) subject to the equality constraints in Eq. (3.29).

Figure 3. 7: A smooth curve in the route polygon

Figure 3. 8: A shortest curve in the route polygon

Fig. 3.7 shows the optimization result using the weighting functions in Eq. (3.26) and Eq. (3.27). The constructed quartic B-spline curve is drawn by a solid line, and the envelopes are drawn by dashed lines. The B-spline as well as the envelopes stay inside the specified route polygon. For the case of the shortest path using the weighting function in Eq. (3.28), Fig. 3.8 reveals that the computed B-spline curve is rendered shorter than the previous case.

3.6 B-spline Path Patterns

In this section we propose to construct path patterns to be utilized for path smoothening. The patterns contain a set of planar B-spline curves which are then used for local path segments to avoid radars. A radar-aware discrete path sequence is provided by a path planner, which constructs a route as the safe region delimited by polygonal lines. Taking into consideration all possible combinations of path sequences, we solve a number of path optimization problems subject to different route constraints, computing a set of B-spline patterns. The path patterns are used for path smoothening in conjunction with the path-planning algorithm in order to generate a feasible path which minimizes radar risk exposure. The B-spline path patterns are composed of a set of B-spline curves which is supposed to be placed inside the feasible routes. By assumption of the low risk cost representation of W , the cells of the optimal path sequence are regarded as a feasible region for the UAV to fly through safely. Hence, the shape of the feasible route is determined from the optimal path sequence by taking into account the square cell geometry, as the border lines around the cells of the local path instance are joined together to yield a polygon consisting of polylines for the route constraints. The boundary conditions of each B-spline curve are chosen, for the convenience sake, such that the B-spline curve starts from the center of the first cell of the local path instance, ends at the center of the last cell of the local path instance. The heading angles at each end of the curve is chosen such that the tangent vector at the point directs toward the center of the next adjacent cell, whereas the curvature values are set to be all zero. We

manage to solve the optimization problem discussed in previous section for minimizing the cost in Eqs. (3.26), (3.27) and (3.28).

The path-planning strategy outlined above was tested in a simulation environment. As a test case, the following scenario was proposed. The threat layout in Fig. 3.1 is assumed as the initial knowledge of radars and a path is planned accordingly. The straight line segments in Fig. 3.5 correspond to the original path first planned when the UAV began its mission. In approximately 1.5 seconds, the best initial path has been found and the first segment of that path has been smoothened by the algorithm [18]. The UAV may now begin to fly this new optimum path. As it flies along this new optimum path, the UAV may optimize the remainder of the path. In approximately 3.8 seconds, the entire path has been optimized. An illustration of the solution generated is shown in Fig. 3.9. The thick line represents the spline path generated by this proposed approach.

Figure 3. 9: Generated spline path

3.7 Path-replanning Algorithm in Radar Zones

Several replanning strategies have been proposed to solve dynamic planning problems for locally directed wandering [118], local modification of initial path [119], and obstacle perimeter detouring [120]. Although these methods are complete, they are sub-optimal and computationally inefficient. On the contrary, we present an efficient method by adopting a cost-effective search scheme to reduce the computational time. In particular, this proposed method is more appropriate when dealing with an environment having numerous states, by recycling information from the preceding search to find the solution at the next step. We employ heuristic estimate like A^* , while reusing information from previous searches to find a solution much faster than solving each iteration from scratch. The proposed algorithm simplifies the maintenance of priority queues, thus it does not use complicated conditional statements, and ending up with shorter codes than the original A^* algorithm.

In the sequel, we employ the proposed algorithm to the path replanning problem. A uniform cell decomposition C_R^J at risk level J on W is comprised of cells $C_{k,l}^J$ of dimension $\frac{1}{2J} \times \frac{1}{2J}$. We adopt the eight-connectivity relationship between the cells. The connectivity relationship is easily found by keeping track of the location of each cell through the indices k and l . It should be noted that the adjacency relationship will remain the same throughout the re-planning, but the edge costs will change incrementally to incorporate the information from the previous step. Suppose the UAV is equipped only with a proximity sensor that senses the region close to the current

location. That is, the sensor provides information about classification of the neighbouring environment by a low radar risk region or threat region. Let $S(i)$ be the known region up to $t = t_i$,

$$S(i) = S(i-1) \cup W(x_i, r_j) \quad (3.30)$$

where x_i is the current location of the UAV at t_i with the sensor range r_j ; $W(x_i, r_j)$ signifying the effective sensory region at t_i . We assume that the UAV updates the map from the gathered information while navigating an unknown environment. For re-planning, we consider the relative location of the edges to S and assign a conditional cost to each edge (u, v) as follows,

$$\mathcal{E}(u, v) \begin{cases} L(\text{cell}_G(v)) + \|\text{cell}_G(u) - \text{cell}_G(v)\|_2, & \text{if } u, v \in S \\ \|\text{cell}_G(u) - \text{cell}_G(v)\|_2, & \text{if } u, v \in W \setminus S \end{cases} \quad (3.31)$$

It follows that for the edges outside S , we consider the uniform size of cells and merely impose the path cost. By using the path cost, a general path-planning approach such as Dijkstra's or A^* simply calculates a path with the shortest distance between an initial location and the target location which might pass through radars. Nevertheless, we update the corresponding cost of the edges accordingly by appending the radar risk cost to each edge (as given in Eq. (3.24)) whenever the map is updated using contingent information from the sensor.

3.7.1 Algorithm Steps

The proposed algorithm proceeds as follows. From the corresponding graph and the uniform risk cost decomposition, we solve for an initial path from $v_0 = v_{start}$ to v_{goal} assuming only the risk cost for edge weights. Let v_1 be the node next to v_0 in the path. The UAV subsequently navigates from v_0 to v_1 . When the UAV is located at v_1 at time $t = t_1$, the algorithm continues to search for changed edge costs and then update the corresponding edge weights. Subsequently, a new path is computed from v_1 to v_{goal} , by incorporating the updated edge weights. It should be noted that if no edge costs have changed, the UAV navigates to the successive node v' in the previous path that has the minimum cost $J(v_{last}, v') + g(v')$. Similar to A^* , the proposed algorithm also incorporates a heuristic estimate to choose the nodes from a priority queue. A pseudo code of the proposed algorithm is given in Listing 3. Note that the feasible path followed by the UAV is given by the sequence of nodes $\{v_0 = v_{start}, v_1, \dots, v_{l-1}, v_l = v_{goal}\}$.

BEGIN PATH REPLANNING ALGORITHM

```

{
   $i = 0$ ;
  compute  $L(x)$  for all  $x \in W$ ;
  construct  $C_R$  at level  $J_{low}$ ;
   $v_{start} \leftarrow node_G(x_0)$ ;
   $v_{goal} \leftarrow node_G(x_f)$ ;
  Initialize();
  ComputePath( $v_{start}, v_{goal}, G$ );
   $v_i \leftarrow v_{start}$ ;
   $v_{last} \leftarrow v_i$ ;
  while ( $v_i \neq v_{goal}$ )
  {
     $i \leftarrow i + 1$ ;
     $v_i = argmin_{v' \in Adj(v_{last})} (J(v_{last}, v') + g(v'))$ ;
    Move to  $v_i$ ;
    Scan graph; /* search for changed edge costs*/
    If any changed edge costs;
  }
}

```

```

{
     $k_m \leftarrow k_m + h(v_{last}, v_i);$ 
     $v_{last} \leftarrow v_i;$ 
    For all  $(u, v) \in E$  with changed edge costs
    {
        Update  $J(u, v);$  /* update edge cost*/
        UpdateVertex( $v$ );
    }
    ComputePath( $v_i, v_{goal}, G$ );
}
}
END PATH REPLANNING ALGORITHM

```

Listing 3.3: Pseudo-code implementation of proposed path-replanning

3.7.2 Experimental Results

The experimental results from the proposed radar aware path re-planning algorithm are presented in Fig. 3.10(a) – (d). It is assumed that the UAV navigates over the radar zones, while updating the map with the information gathered from a proximity sensor. Fig. 3.10(a) – (d) show the path evolution at distinct time steps as the UAV navigates to the target location. At each step, the best proposed path is drawn by a dashed-line. As shown in Fig. 3.11, the generated path differs significantly from the path calculated in Fig. 3.9. This is attributed to the fact that the UAV does not possess the complete information at time t_5 and t_{15} for upcoming location and the initial path is computed using the risk cost based on the current location of the UAV. Hence, as shown in Fig. 3.10, the UAV is unable to anticipate the existence of the radars at the navigational horizon. Nonetheless, as the UAV gets closer to the radars, the proposed algorithm efficiently replans the entire path avoiding the radars, reaching the target location.

(a): Generated path at time step $t = 5$

(b): Generated path at time step $t = 15$

Figure 3. 10: Path evolution and re-planning using the proposed algorithm

Figure 3. 11: Illustration of the actual path followed by the UAV. Dashed lines show the currently tentative optimal path obtained from the algorithm, based on the risk cost from the current location of the UAV.

3.8 Summary

In this chapter, we proposed a technique to solve the UAV stealthy path-planning problem in a radar laden environment. The development of a radar-aware path planner which generates a feasible path for a UAV has been presented. The key strengths of the method are: 1) the ability to plot a safe path to the target while minimizing exposure to the radars; 2) the generated paths are smoothened using B-splines to improve navigability 3) the distance between the UAV and the radars is optimized in order to minimize the energy received at the radar and 4) computational efficiency is achieved. By utilizing the risk cost representation of the radar region, the computational cost is significantly reduced. As an added benefit, the proposed method possesses the advantages of scalability and can be tailored to the computational resources of the UAV. Finally, the ability to generate feasible paths by using this path-planning method has been demonstrated through simulations.

Chapter 4: Memetic Computation in Unmanned Aerial Vehicle (UAV) Operations

We present an evolutionary framework which is to optimize the operations of multiple cooperating UAVs. Relying on a memetic computing approach [121], we draw our focus on optimizing the cooperative operations of multiple UAVs, catering to the requirements of different vehicle types and constraints. Some experimental results and comparisons with other evolutionary approaches to further demonstrate the potential benefits of such approach are presented. For convenience and ease of referencing, the symbols and abbreviations used in our discussions are summarized in the listing below:

W_i	k waypoints $\{W_1, W_2, \dots, W_k\}$
m	Number of UAVs $\{U_1, U_2, \dots, U_i, \dots, U_m\}$
U_i	UAV
C_U	Finite capacity
r_i	Route of UAV U_i
a_{ij}	j^{th} waypoint visited by U_i
l_{max}	Maximum travelled distance of UAV
h	Optimal solution $\{r_1, r_2, \dots, r_i, \dots, r_m\}$
$P(t)$	Genetic population at generation t
n	Number of candidate solutions
$d(a_{ij}, a_{i(j+1)})$	Distance between two consecutive waypoints a_{ij} and $a_{i(j+1)}$
$D(r_i)$	Total number of waypoints visited by vehicle U_i
waypoint (a_{ij})	Total number of waypoints a_{ij} to be visited
R	Aircraft range
$C(h)$	Overall distance of all m vehicles
R_{ik}	Distance from the UAV U_i to the radar
N	Number of radars at a navigational space
κ_{max}	Maximum curvature of the UAV dynamic constraints

ψ	Separating axis
θ	Random angle
h_c	Child solution created from a partial route of two selected parents
ρ	Infeasible route in solution h_{repair}
ρ'	Feasible route in h_{repair}
$\mathcal{N}(h)$	Neighbourhood of solution h
$l(r_i)$	Route length of r_i
Q	Candidate route pairs (r_i, r_k)
$\bar{I}$	Average number of iterations
M	Population size
n	Number of waypoints
g	Total number of search generations
$\setminus$	Exclusive of
ΔC	Change in fitness cost

4.1 Introduction

Dealing with changing environments entails a coordination of autonomous mission-planning among the participating UAVs. Autonomous mission-planning serves as first-order input to algorithms that control the UAV over its flight path that help to maintain the required flight formation throughout the flight. The UAV waypoints are usually specified *a priori* and the mission must be completed satisfying certain criteria provided by the mission scenario. It is essential that the mission planner be efficiently implemented so as to optimize the cooperative missions of the UAVs to the target locations. Our focus in this work is on memetic computation approach to optimize the cooperative operations of multiple UAVs. The focus of this research pertains to the investigation on Memetic Algorithms (MAs) [121], which represent the forefront of today's evolutionary computation research. The basic form of MA is inspired by models of natural systems that combine the evolutionary adaptation of a population with individual learning within the lifetime of its members. MAs represent a synergy of evolutionary or population-based approach with advance local procedures for problem search. It may also involve the use of learning procedures during search to uncover useful information about the solution landscape in order to enhance the problem-solving capability. In a broader context, MAs can be interpreted as a search encompassing a population of competing and cooperating optimizing agents that exploit the available knowledge about the given optimization problem. The search emphasis is on a complete solution encompassing all the waypoints that constitute the overall mission. The flexibility of such an approach is evident when one has to further undertake tasks

division or allocation to multiple vehicles basing on the overall solution attained by the proposed approach. There are several important components considered in the proposed approach, for instance, the representation of the mission-planning problem, the fitness function as well as the genetic search operators. With proper and domain-specific representation, we can easily apply our proposed memetic approach to solve the cooperative mission-planning problem. Our current study examines the process of developing cooperative mission planner through evolution (as illustrated in Figure 4.0). We will look at how multiple UAVs required for a higher level task interact to involve in multiples missions, and how multiple UAVs can be evolved simultaneously. Also, we take into consideration the vehicle constraints while assigning the mission to the UAVs.

Figure 4.0: The cooperative mission-planning

The remaining of this chapter is organized as follows. Section 4.3 and Section 4.4 present the solution and implementation, respectively, of the proposed approach substantiated with simulation results. This is followed by a detailed discussion on the performance of the proposed approach in Section 4.5. The computational complexity of the proposed approach is discussed in Section 4.6. Lastly, we summarize the main contributions of this work in Section 4.7.

4.2 Problem Formulation

4.2.1 Discretizing the Problem Domain

In our mission-planning problem, we assume all the UAVs take off from the same location, W_0 . By way of example, we consider the navigational spaces portrayed in Fig. 4.1 and Fig. 4.2. The objective is to find an optimum solution for each UAV to fly from its current position (diamond) to the target locations (circle) while minimizing energy consumption and maintaining minimal exposure to radars (triangle). The heading angles of the UAV at the waypoints and the flight times between consecutive waypoints are given. Hence, the mission is specified by initial and final conditions, mission time, waypoints, and flight times between waypoints. It is a general requirement that a UAV visits the useful vicinity of (if possible) each site and lands back safely to the take-off zone or one of the other benign landing zones that are within its flight envelope. The term flight envelope refers to the parameters within which an UAV may be safely flown under varying, though expected, wind speed, wing loading, wind shear, visibility and other flight conditions without resorting to extreme control measures such as abnormal spin, or stall recovery, or crash landing.

Often, the reconnaissance zone is the subject of varying degrees of hostile activities arising out of on-ground vigilance, air surveillance radar installations, or active enemy aircraft patrols. Also, some paths may be infeasible due to the nature of ground terrain or no-fly-zone restrictions. Therefore the closure of all ‘safe’ paths that a UAV can possibly take is limited (at times, severely) as opposed to being an extensive 3-D space, limited by only its flight envelope, the terrain and the locations of sites. Revisiting a site

may be redundant, can increase the time needed for reconnaissance activities, reduce the available time-on-target and reduce the useful flight envelope of the UAV and, is likely to compound mission risks. Thus, an important goal of mission-planning for UAV is to choose a feasible path along which it can fly to the useful vicinity of each target site exactly once and then, fly back to safe ground.

Figure 4. 1: Navigational space 1 with known threats

Figure 4. 2: Navigational space 2 with known threats

To manage the computational complexity of optimizing the operations of multiple cooperating UAVs, a complex or difficult problem is decomposed into several sub-problems. Individual path control algorithm is then methodically configured to fulfil the goals of each specific sub-problem. The overall goal is accomplished by appropriate decomposition and path construction methods. For example, in a mission-planning scenario, a top-level controller plans the grouping of available UAVs, an allocation of feasible routes, followed by generating an optimum mission plan for individual UAV. At a lower level, feasible paths for each UAV are generated based on the mission plan received from the top-level. Hence, multiple UAVs can utilize available resources and follow their routes individually to accomplish the overall mission objective.

4.2.2 Objective Formulation

In a mission-planning problem of multiple cooperating UAVs, a series of k waypoints $\{W_1, W_2, \dots, W_k\}$ is to be visited by a fleet of m UAVs $\{U_1, U_2, \dots, U_i, \dots, U_m\}$. Each UAV U_i has finite capacity C_U and travels a route $r_i = \{a_{i1}, a_{i2}, \dots, a_{ij}, \dots, a_{in}\}$, with a_{ij} representing the j^{th} waypoint visited by U_i . Additionally, each UAV is only capable of travelling a maximum distance of l_{max} . The goal is to find the optimal solution $h = \{r_1, r_2, \dots, r_i, \dots, r_m\}$ which minimizes the total mission time by taking into account that each UAV navigates to the useful vicinity of each target site exactly once and then, fly back to safe ground and the overall distance travelled by each UAV U_i does not exceed its capacity C_U .

For each UAV, the goal is to accomplish its mission while minimizing energy consumption and maintaining minimal exposure to radars. In a typical mission-planning scenario, the information required include the starting and target locations of the UAV, threat locations and mission specification. Here we describe the representation and problem formulation to be applied in the proposed memetic computation approach. The genetic population $P(t)$ at generation t includes a set of N candidate solutions:

$$P(t) = \{h_1, h_2, \dots, h_i, \dots, h_N\} \quad (4.1)$$

Solution $h \in P(t)$ defines a set of m routes:

$$h = \{r_1, r_2, \dots, r_i, \dots, r_m\} \quad (4.2)$$

where r_i represents the route visited by UAV, U_i and m is the number of UAVs in h .

The route consists of an ordered set of waypoints:

$$r_i = \{a_{i1}, a_{i2}, \dots, a_{ij}, \dots, a_{in}\} \quad (4.3)$$

where a_{ij} of route r_i representing the j^{th} waypoint visited and n is the total number of waypoints visited by U_i .

The distance travelled by UAV U_i is denoted here as $l(r_i)$ and given by:

$$l(r_i) = \sum_{j=1}^{n-1} d(a_{ij}, a_{i(j+1)}) + d(a_{i1}, a_{in}) \quad (4.4)$$

where $d(a_{ij}, a_{i(j+1)})$ represents the distance between two consecutive waypoints a_{ij} and $a_{i(j+1)}$. The total number of waypoints visited by vehicle U_i is denoted as $D(r_i)$ and given by

$$D(r_i) = \sum_{j=1}^n \text{waypoint}(a_{ij}) \quad (4.5)$$

where $\text{waypoint}(a_{ij})$ denotes the total number of waypoints a_{ij} to be visited.

We then assign risk cost and distance cost to each route. It is assumed that the UAV's radar signature is uniform in all directions and is proportional to $1/R^4$ where R is the aircraft range; the risk cost for travelling along a path is inversely proportional to the

distance (to the radar). The overall distance $C(h)$ travelled by all m vehicles is defined as:

$$\begin{aligned} C(h) &= \sum_{i=1}^m l(r_i) \\ &= \sum_{i=1}^m [\sum_{j=1}^{n-1} d(a_{ij}, a_{i(j+1)}) + d(a_{i1}, a_{in})] \end{aligned} \quad (4.6)$$

To generate a stealthy path, steering the UAV clear and around known radar locations; we add the radar risk cost into Eq. (4.6) as follows:

$$\begin{aligned} C(h) &= \sum_{i=1}^m [\sum_{j=1}^{n-1} d(a_{ij}, a_{i(j+1)}) + d(a_{i1}, a_{in}) + \\ &\quad \sum_{k=0}^N \frac{1}{R_{ik}^4}] \end{aligned} \quad (4.7)$$

where R_{ik} denotes the distance from the UAV U_i to the radar and N is the number of radars located at the navigational space. The objective of the mission-planning problem is to minimize the cost of the total distance travelled by UAVs with minimal radar exposure and subjected to following constraints:

$$\begin{aligned} l(r_i) &\leq l_{max} \quad \forall r_i \in h \\ D(r_i) &\leq C_U \quad \forall r_i \in h \\ \kappa(t) - \kappa_{max} &\leq 0, \\ \kappa(t) &= \frac{|\frac{dx}{dt} \frac{d^2y}{dt^2} - \frac{dy}{dt} \frac{d^2x}{dt^2}|}{((\frac{dx}{dt})^2 + (\frac{dy}{dt})^2)^{3/2}} \quad \forall \kappa(t) \in r_i \end{aligned} \quad (4.8)$$

where l_{max} is the maximum allowable distance travelled by each vehicle; C_U is the capacity of each vehicle and κ_{max} represents the maximum curvature as determined from the dynamic constraints of the UAV.

4.3 The Proposed Approach

In this section, we show the essential components of the proposed memetic computation approach for solving mission allocation problem for multiple UAVs. The basic steps of the proposed approach involving the use of the genetic operators are outlined in Listing 4.1.

Algorithm Steps

BEGIN

Create an initial population of solutions $P(t = 0) = \{h_1, h_2, \dots, h_i, \dots, h_m\}$ For all $h \in P(t)$ $h \leftarrow \text{individual-learning}(h)$

End For

 $\text{Individual Count} \leftarrow 0$

While (stopping condition is not met)

If $\text{rand}[0,1] \leq p_c$ then $\text{Select } h_i, h_j \in P(t)$ $h_c \leftarrow \text{crossover}(h_i, h_j)$

Else

 $\text{Select } h_k \in P(t)$ $h_c \leftarrow \text{mutate}(h_k)$

End If

 $h_c \leftarrow \text{path-repair}(h_c)$ $h_c \leftarrow \text{individual-learning}(h_c)$ $h_{\text{worst}} = \max_{h \in P(t)} C(h)$ If $C(h_c) < C(h_{\text{worst}})$ then $P(t) \leftarrow P(t) \setminus h_{\text{worst}}$ $P(t) \leftarrow P(t) \cup h_c$

End If

 $\text{Individual Count} \leftarrow \text{Individual Count} + 1$ If $\text{Individual Count} \geq M$ $\text{Individual Count} \leftarrow 0$ $t \leftarrow t + 1$

End If

End While

END

Listing 4.1: The proposed memetic algorithm

In the first step, the initial population $P(t = 0)$ of candidate solutions for the mission allocation problem are generated randomly. The initial population then undergoes the individual learning operation to arrive at a population of local optima before proceeding with the reproduction operations, namely crossover and mutation. A path-repair operator is designed to recover any infeasible path produced by the genetic operators. The evolutionary search proceeds until a user-specified termination criterion is met or the computational budget is exhausted. The details of the problem-specific evolutionary operators, path-repair and cooperating multi-strategy individual learning operators are presented in the following sections.

4.3.1 Crossover Operator

We select a pair of candidate solutions (chromosomes) for the crossover operation. For conventional genetic algorithm (GA), different selection methods such as roulette wheel and rank selection are used. In general, the aim of any such selection method is to increase the chance for a chromosome of high fitness to be selected. In this work, we propose a space partition method so as to increase the efficiency as well as the quality of chromosomes selection. A separating axis ψ of random angle θ is initiated at the start location W_0 to divide the navigational space into two separate regions containing waypoints, H_+ and H_- . A child solution h_c is created from the partial routes of two selected parents $h_i, h_j \in P(t)$. For example, the child solution h_c is a combination of partial routes $\{r_1^i, r_2^i, r_3^i\} \in h_i$ and $\{r_1^j, r_3^j, r_4^j\} \in h_j$. The basic steps of the crossover operator are outlined in Listing 4.2.

```

Procedure::Crossover ( $h_i, h_j$ )
BEGIN
 $h_c \leftarrow \emptyset$ 
Create separating axis  $\psi$  such that
 $\psi$  passes through the start location
 $\psi$  is slanted at random angle  $\theta$ 
 $\psi$  divides the navigational space into separate regions  $H_+$  and  $H_-$ 

  For each route  $r_k^i \in h_i$ 
    If  $r_k^i \cap H_+ \neq \emptyset$  then
       $h_c \leftarrow h_c \cup r_k^i$ 
    End If
  End For

  For each route  $r_k^j \in h_j$ 
    If  $r_k^j \cap H_- \neq \emptyset$  then
       $r_{repeated} = h_c \cap r_k^j$ 
       $r_k^j \leftarrow r_k^j \setminus r_{repeated}$ 
       $h_c \leftarrow h_c \cup r_k^j$ 
    End If
  End For

END

```

Listing 4.2: Crossover operator

4.3.2 Mutation Operator

We adopt a multiple heuristics-based mutation operator including swap, inversion, insertion and displacement, defined as r_{swap} , $r_{inversion}$, $r_{insertion}$, $r_{displacement}$ respectively in our study. The mutation process maintains diversity within the solution pool and prevents solutions from being trapped in local optimum points. During mutation, one of the four mutation heuristics (r_{swap} , $r_{inversion}$, $r_{insertion}$, $r_{displacement}$) is selected at random for each solution $h \in P(t)$ and each heuristic has an equal probability of being chosen. The basic steps of the mutation operator are outlined in Listing 4.3.

```

Procedure :: Mutate ( $h$ )
BEGIN
 $r \leftarrow rand[0,1)$ 

 $r_t \leftarrow r_{swap} + r_{inversion} + r_{insertion} + r_{displacement}$ 

 $p_{swap} \leftarrow \frac{r_{swap}}{r_t}$ 

 $p_{inversion} \leftarrow \frac{r_{swap} + r_{inversion}}{r_t}$ 

 $p_{insertion} \leftarrow \frac{r_{swap} + r_{inversion} + r_{insertion}}{r_t}$ 

    If  $r \leq p_{swap}$  then
        Given:  $a_{ij} \in r_i, a_{kl} \in r_k$  with  $r_i, r_k \in h$ 
        Do:  $swap(a_{ij}, a_{kl})$ 

    Else If  $r \leq p_{inversion}$  then
        Given:  $a_{i(j+1)}, a_{i(j+2)}, \dots, a_{i(j+\mu)} \in r_i$  with  $r_i \in h$ 
        Do:  $swap(a_{i(j+u)}, a_{i(j+\mu+1-u)}), \forall u = 1, 2, \dots, \frac{\mu}{2}$ 

    Else If  $r \leq p_{insertion}$  then
        Given:  $a_{ij} \in r_i, r_i \neq r_k$  with  $r_i, r_k \in h$ 
        Do:  $r_i \leftarrow \frac{r_i}{a_{ij}}, r_k \leftarrow r_k \cup a_{ij}$ 

    Else
        Given:  $a_{i(j+1)}, a_{i(j+2)}, \dots, a_{i(j+\mu)} \in r_i, r_i \neq r_k$  with  $r_i, r_k \in h$ 
        Do:
             $r_i \leftarrow r_i \setminus \{a_{i(j+1)}, a_{i(j+2)}, \dots, a_{i(j+\mu)}\}$ 
             $r_k \leftarrow r_k \cup \{a_{i(j+1)}, a_{i(j+2)}, \dots, a_{i(j+\mu)}\}$ 

    End If
END

```

Listing 4.3: Mutation operator

4.3.3 Path-Repair Mechanism

In our proposed algorithm, a path-repair mechanism is designed to revive any infeasible solution as the mutation and crossover operators may generate infeasible solutions due to the vehicle constraints. For an infeasible solution h_{repair} , some UAV(s) U_i may either violate the capacity constraint or distance constraint or both. Let ρ represents an infeasible route in solution h_{repair} . Thus $\rho' = \frac{h_{repair}}{\rho}$ is the set of feasible routes in h_{repair} . In this procedure, we remove an infeasible route r_i from ρ ; followed by removing the edge between consecutive waypoints a_{ik} and $a_{i(k+1)}$ to split r_i into two routes r_a and r_b . It is to ensure that the augmentation $\delta(k)$ due to the split is minimal. The basic steps of the path-repair mechanism for the proposed method are listed as below:

```

Procedure:: Path-Repair ( $h$ )

 $\rho \leftarrow$  infeasible routes in  $h$ 
 $\rho' \leftarrow$  feasible routes in  $h$ 
While  $\rho \neq \emptyset$ 
    Given  $r_i \in \rho$ 
    For  $j = 1$  to  $m$  Do
         $\delta(j) =$  distance increase by splitting  $r_i$  between  $a_{ij}$  and  $a_{i(j+1)}$ 
    End For
     $k \leftarrow \min_j \delta(j)$ 
    Split  $r_i$  between  $a_{ik}$  and  $a_{i(k+1)}$  to form  $r_a$  and  $r_b$ 
    If  $r_a$  is infeasible then
         $\rho \leftarrow \rho \cup r_a$ 
    Else
         $\rho' \leftarrow \rho' \cup r_a$ 
    End If
    If  $r_b$  is infeasible then
         $\rho \leftarrow \rho \cup r_b$ 
    Else
         $\rho' \leftarrow \rho' \cup r_b$ 
    End If
     $\rho \leftarrow \rho / r_i$ 
End While
END

```

Listing 4.4: Path-repair mechanism

4.3.4 Cooperative Individual Learning

A single local search having certain learning roles and search features, would work effectively on a given problem of certain properties but fail to improve solutions on problems that possess contradictory properties. Hence we propose a cooperative individual learning procedure so that multiple local search are hybridized and organized in a cooperative manner; they can work together to accomplish their shared optimization goal, thus making the proposed algorithm robust and effective. We describe the proposed individual learning procedure as three distinct cooperative strategies, each having unique learning roles and search features. The first strategy involves a clustering procedure to improve waypoint allocation between routes so as to reduce the overall path length. Next, a route reduction strategy is introduced to reduce the number of participating UAVs as well as the overall distance travelled. The final strategy of the individual learning involves a TSP optimization to improve the solution quality of each route. The complete individual learning operator is outlined in Listing 4.5 while the individual clustering, route reduction and TSP procedures are detailed in Listing 4.6, 4.7 and 4.8, respectively. The three strategies of the individual learning procedure are described in greater details in the following sections.

Procedure:: Individual_Learning

For $h \in P(t)$

Do (*Strategy 1: Clustering*)

$h' \leftarrow$ reallocate waypoints between $r_i \in h$ and $r_k \in h$ that maximizes $\Delta C = C(h) - C(h')$

If $\Delta C > 0$ then

$h \leftarrow h'$

End If

Until $(C(h) \leq C(h'), \forall h' \in \mathcal{N}(h))$

Do (*Strategy 2: Route reduction*)

$h' \leftarrow$ merge $r_i \in h$ and $r_k \in h$ that maximizes $\Delta C = C(h) - C(h')$

If $\Delta C > 0$ then

$h \leftarrow h'$

End If

Until $(C(h) \leq C(h'), \forall h' \in \mathcal{N}(h))$

For $r_i \in h$

Do (*Strategy 3: TSP Optimization*)

$h' \leftarrow$ reorder waypoints in r_i that maximizes $\Delta C = C(h) - C(h')$

If $\Delta C > 0$ then

$h \leftarrow h'$

End If

Until $(C(h) \leq C(h'), \forall h' \in \mathcal{N}(h))$

End For

$h \leftarrow h'$

End For

Listing 4.5: Individual learning procedure

4.3.5 Strategy 1: Clustering

We cluster the waypoints using a reassignment strategy, based on a composite edge exchange neighbourhood search. The composite edge-exchange neighbourhood search strategy is composed of three heuristics: waypoints relocation, waypoints exchange, and route exchange. These three heuristics employ greedy strategy that searches neighbourhood $\mathcal{N}(h)$ to obtain the best improved solution h' . The process is repeated at the new point h' and the heuristics continue until a local optimum is found, i.e., $\Delta C \leq 0, \forall h' \in \mathcal{N}(h)$. A detailed description of the composite edge-exchange neighbourhood search strategy in terms of the three heuristics is presented to illustrate the clustering procedure.

4.3.5.1 Waypoints Relocation

In waypoints relocation, waypoint a_{ij} of route $r_i \in h$ is removed and reinserted into a separate route $r_k \in h$ such that ΔC is maximized to arrive at h' . ΔC is defined as follows

$$\Delta l(a_{ij}, r_i, r_k) = l(r_i) + l(r_k) - l(r_i \setminus a_{ij}) - l(r_k \cup a_{ij}) \quad (4.9)$$

$$\Delta C = \max\{\Delta l(a_{ij}, r_i, r_k) | r_k \in h, a_{ij} \notin r_k\} \quad (4.10)$$

where $l(r_i)$ denotes the route length of r_i .

4.3.5.2 Waypoints Exchange

In waypoints exchange, waypoint a_{ij} of route $r_i \in h$ is swapped with waypoint a_{kl} of route $r_k \in h, r_k \neq r_i$ such that ΔC is maximized to arrive at h' . ΔC is defined as follows:

$$\Delta l(a_{ij}, a_{kl}) = l(r_i) + l(r_k) - l(\{r_i \setminus a_{ij}\} \cup a_{kl}) - l(\{r_k \setminus a_{kl}\} \cup a_{ij}) \quad (4.11)$$

$$\Delta C = \max\{\Delta l(a_{ij}, a_{kl}) | a_{ij} \in r_i, a_{kl} \in r_k, a_{ij} \notin a_{kl}\} \quad (4.12)$$

4.3.5.3 Route Exchange

In route exchange, a sub-route $r'_i = \{a_{i(j+1)}, a_{i(j+2)}, \dots, a_{i(j+\mu)}\}$ of route $r_i \in h$ is swapped with sub-route $r'_k = \{a_{k(l+1)}, a_{k(l+2)}, \dots, a_{k(l+v)}\}$ of route $r_k \in h, r_i \neq r_k$ such that ΔC is maximized to arrive at h' . ΔC is defined as follows:

$$\Delta l(r'_i, r'_k) = l(r_i) + l(r_k) - l(\{r_i \setminus r'_i\} \cup r'_k) - l(\{r_k \setminus r'_k\} \cup r'_i) \quad (4.13)$$

$$\Delta C = \max\{\Delta l(r'_i, r'_k) | r'_i \subset r_i, r'_k \subset r_k\} \quad (4.14)$$

Listing 4.6 outlines the basic steps of the proposed clustering strategy.

Procedure:: Clustering

BEGIN

Do

Waypoints relocation:

For $r_i \in h, r_k \in h, r_k \neq r_i$

For $a_{ij} \in r_i$

$$\Delta l(a_{ij}, r_i, r_k) = l(r_i) + l(r_k) - l(r_i \setminus a_{ij}) - l(r_k \cup a_{ij})$$

End For

$$(a_{i'j'}, r_{i'}, r_{k'}) = \max_{a_{ij}, r_i, r_k} \{\Delta l(a_{ij}, r_i, r_k)\}$$

$$\Delta C = \Delta l(a_{i'j'}, r_{i'}, r_{k'})$$

If $\Delta C > 0$ then

$$r_{i'} \leftarrow r_{i'} \setminus a_{i'j'}$$

$$r_{k'} \leftarrow r_{k'} \cup a_{i'j'}$$

End If

Waypoints exchange:

For $a_{ij} \in r_i, a_{kl} \in r_k, r_i \neq r_k$

$$\Delta l(a_{ij}, a_{kl}) = l(r_i) + l(r_k) - l(\{r_i \setminus a_{ij}\} \cup a_{kl}) - l(\{r_k \setminus a_{kl}\} \cup a_{ij})$$

End For

$$(a_{i'j'}, a_{k'l'}) = \max_{a_{ij}, a_{kl}} \{\Delta l(a_{ij}, a_{kl})\}$$

$$\Delta C = \Delta l(a_{i'j'}, a_{k'l'})$$

If $\Delta C > 0$ then

$$r_{i'} \leftarrow \{r_{i'} \setminus a_{i'j'}\} \cup a_{k'l'}$$

$$r_{k'} \leftarrow r_{k'} \cup a_{k'l'} \setminus a_{i'j'}$$

End If

Route exchange:

For $r'_i \in r_i, r'_k \in r_k, r_i \neq r_k$

$$\Delta l(r'_i, r'_k) = l(r_i) + l(r_k) - l(\{r_i \setminus r'_i\} \cup r'_k) - l(\{r_k \setminus r'_k\} \cup r'_i)$$

End For

$$(r'_{i'}, r'_{k'}) = \max_{r'_i, r'_k} \{\Delta l(r'_i, r'_k)\}$$

$$\Delta C = \Delta l(r'_{i'}, r'_{k'})$$

If $\Delta C > 0$ then

$$r'_i \leftarrow \{r'_i \setminus r'_{i'}\} \cup r'_{k'}$$

$$r'_k \leftarrow \{r'_k \setminus r'_{k'}\} \cup r'_{i'}$$

End If

Until $(C(h) \leq C(h'), \forall h' \in \mathcal{N}(h))$

END

Listing 4.6: Clustering strategy

4.3.6 Strategy 2: Route Reduction

A reduction strategy is introduced to reduce the overall travelled distance by exploring the routes merging. Let Q denote the set of candidate route pairs (r_i, r_k) that can be merged. The resultant reduction in overall travelled distance when merging r_i, r_k , is denoted here as $\Delta C(r_i, r_k)$, and derived as:

$$\Delta l(a_{ij}, a_{kl}) = d(a_{ij}, a_{i(j+1)}) + d(a_{kl}, a_{k(l+1)}) - d(a_{ij}, a_{kl}) - d(a_{i(j+1)}, a_{k(l+1)}) \quad (4.15)$$

$$\Delta C(r_i, r_k) = \max\{\Delta l(a_{ij}, a_{kl}) | a_{ij} \in r_i, a_{kl} \in r_k\} \quad (4.16)$$

Each route pair (r_i, r_k) in Q is subsequently ranked using criterion $\Delta C(r_i, r_k)$. Route pair of higher $\Delta C(r_i, r_k)$ has higher precedence of being merged. The procedure is repeated until the local optimum solution is reached. The route reduction strategy is also outlined in Listing 4.7.

```

Procedure:: Route_Reduction
BEGIN
 $Q \leftarrow \emptyset$ 

  For  $r_i, r_k \in h$ 
    For  $a_{ij} \in r_i, a_{kl} \in r_k$ 
      
$$\Delta l(a_{ij}, a_{kl}) = d(a_{ij}, a_{i(j+1)}) + d(a_{kl}, a_{k(l+1)}) - d(a_{ij}, a_{kl}) - d(a_{i(j+1)}, a_{k(l+1)})$$

    End For
     $\Delta C(r_i, r_k) = \max\{\Delta l(a_{ij}, a_{kl}) | a_{ij} \in r_i, a_{kl} \in r_k\}$ 
    If  $\Delta C > 0$  then
       $Q \leftarrow Q \cup (r_i, r_k)$ 
    End If
  End For

  Rank each  $(r_i, r_k)$  in  $Q$  by  $\Delta C(r_i, r_k)$  in descending order

  Do
     $(r_{i'}, r_{k'}) = \max_{(r_i, r_k) \in Q} \{\Delta C(r_i, r_k) | (r_i, r_k) \in Q\}$ 
    Merge  $r_{i'}, r_{k'}$  in  $h$ 
     $Q \leftarrow Q \setminus (r_{i'}, r_{k'})$ 
  Until  $(C(h) \leq C(h'), \forall h' \in \mathcal{N}(h))$ 

END

```

Listing 4.7: Route reduction strategy

4.3.7 Strategy 3: TSP Optimization

Each route in the solution h is subsequently cast as a Traveling Salesman Problem (TSP). Here, a composite search strategy is used to achieve the improvement in route length as outlined in Listing 4.8. The cooperative search strategy involves two heuristics: reinsertion and 2-exchange. Let $\Delta C = C(h) - C(h')$, which indicates the difference in overall distance travelled between current solution h and the best cost-improving solution $h' \in \mathcal{N}(h)$.

4.3.7.1 Reinsertion

In reinsertion, a waypoint $a_{ij} \in r_i$ is removed and reinserted after a separate waypoint $a_{ik} \in r_i$ such that ΔC is maximized to arrived at h' . ΔC is derived as follows

$$\begin{aligned} \Delta l(a_{ij}, a_{ik}) = & d(a_{i(j-1)}, a_{ij}) + d(a_{ij}, a_{i(j+1)}) + \\ & d(a_{ik}, a_{i(k+1)}) - d(a_{i(j-1)}, a_{i(j+1)}) - d(a_{ij}, a_{ik}) - \\ & d(a_{ij}, a_{i(k+1)}) \end{aligned} \quad (4.17)$$

$$\Delta C = \max\{\Delta l(a_{ij}, a_{ik}) | a_{ij}, a_{ik} \in r_i\} \quad (4.18)$$

4.3.7.2 2-exchange

In 2-exchange, a waypoint $a_{ij} \in r_i$ is joined with separate waypoint $a_{ik} \in r_i$ by reversing the visiting order of waypoints between a_{ij} and a_{ik} such that ΔC is maximized to arrive at h' . ΔC is then computed as follows

$$\begin{aligned} \Delta l(a_{ij}, a_{ik}) = & d(a_{ij}, a_{i(j+1)}) + d(a_{ik}, a_{i(k+1)}) - \\ & d(a_{ij}, a_{ik}) - d(a_{i(j+1)}, a_{i(k+1)}) \end{aligned} \quad (4.19)$$

$$\Delta C = \max\{\Delta l(a_{ij}, a_{ik})\} \quad (4.20)$$

```

Procedure :: TSP
BEGIN
For  $r_i \in h$ 
  Do
  Reinsertion:
    For  $a_{ij}, a_{ik} \in r_i, a_{ij} \neq a_{ik}$ 
       $\Delta l(a_{ij}, a_{ik}) = d(a_{i(j-1)}, a_{ij}) + d(a_{ij}, a_{i(j+1)}) + d(a_{ik}, a_{i(k+1)}) - d(a_{i(j-1)}, a_{i(j+1)}) -$ 
       $d(a_{ij}, a_{ik}) - d(a_{ij}, a_{i(k+1)})$ 
    End For
     $(j', k') = \max_{j,k}\{\Delta l(a_{ij}, a_{ik})\}$ 
     $\Delta C = \Delta l(a_{ij'}, a_{ik'})$ 
    If  $\Delta C > 0$  then
      Delete edges  $(a_{ij'}, a_{i(j'+1)}), (a_{i(j'-1)}, a_{ij'}), (a_{ik'}, a_{i(k'+1)})$ 
      Join edges  $(a_{i(j-1)}, a_{i(j+1)}), d(a_{ij}, a_{ik}), (a_{ij}, a_{i(k+1)})$ 
    End If
  2-exchange:
    For  $a_{ij}, a_{ik} \in r_i, a_{ij} \neq a_{ik}$ 
       $\Delta l(a_{ij}, a_{ik}) =$ 
       $d(a_{ij}, a_{i(j+1)}) + d(a_{ik}, a_{i(k+1)}) - d(a_{ij}, a_{ik}) - d(a_{i(j+1)}, a_{i(k+1)})$ 
    End For
     $(j', k') = \max_{j,k}\{\Delta l(a_{ij}, a_{ik})\}$ 
     $\Delta C = \Delta l(a_{ij'}, a_{ik'})$ 
    If  $\Delta C > 0$  then
      Delete edges  $(a_{ij'}, a_{i(j'+1)}), (a_{ik'}, a_{i(k'+1)})$ 
      Join edges  $d(a_{ij'}, a_{ik'}), (a_{i(j'+1)}, a_{i(k'+1)})$ 
    End If
  Until  $(C(h) \leq C(h'), \forall h' \in \mathcal{N}(h))$ 
End For
END

```

Listing 4.8: TSP optimization

It is worth noting that our approach can also be applied to other optimization problems with cooperating agents. The effectiveness of the proposed approach, substantiated with the experimental results will be presented in the following section.

4.4 Simulations

In this section, the proposed memetic computation approach will be compared against a Genetic Algorithm (GA)-based mission planner. To illustrate the impact of the size of a problem domain to the system performance, different numbers of way points are projected to each landscape model under study. All simulations are performed in MATLAB 7 and experimental results presented in this work are averaged over 100 simulations. For both planners, cost of the best solution is recorded at each generation. Figs 4.3-4.14 illustrate the multi-route solution representation $h = \{r_1, r_2, \dots, r_i, \dots, r_m\}$ for m UAVs, where each route $r_i = \{a_{i1}, a_{i2}, \dots, a_{ij}, \dots, a_{in}\}$ denotes the closed circuit that is visited by UAV U_i . To investigate the scalability issue, the averaged computational time of each planner is recorded and shown in Tables 4.1 and 4.2.

	Number of waypoints	
	8	14
Proposed memetic computation with 3 UAVs	5.36	7.39
Proposed memetic computation with 4 UAVs	10.57	14.59
Proposed memetic computation with 5 UAVs	18.28	25.23
GA approach with 3 UAVs	9.60	14.21
GA approach with 4 UAVs	18.26	25.75
GA approach with 5 UAVs	30.34	42.78

Table 4. 1: Time elapsed of different planners on navigational space 1 (sec)

	Number of waypoints	
	8	14
Proposed memetic computation with 3 UAVs	5.24	7.23
Proposed memetic computation with 4 UAVs	10.06	13.88
Proposed memetic computation with 5 UAVs	17.47	24.11
GA approach with 3 UAVs	10.06	14.18
GA approach with 4 UAVs	19.32	27.24
GA approach with 5 UAVs	33.31	46.97

Table 4. 2: Time elapsed of different planners on navigational space 2 (sec)

Figure 4. 3: Illustration of the navigational space 1 with 8 waypoints

Figure 4. 4: Illustration of the multi-route solution representation $h = \{r_1, r_2, r_3\}$ for 3

UAVs to visit 8 waypoints at navigational space 1

Figure 4. 5: Illustration of the multi-route solution representation $h = \{r_1, r_2, r_3, r_4\}$ for 4 UAVs to visit 8 waypoints at navigational space 1

Figure 4. 6: Illustration of the multi-route solution representation $h = \{r_1, r_2, r_3, r_4, r_5\}$ for 5 UAVs to visit 8 waypoints at navigational space 1

Figure 4. 7: Illustration of the navigational space 1 with 14 waypoints

Figure 4. 8: Illustration of the multi-route solution representation $h = \{r_1, r_2, r_3\}$ for 3 UAVs to visit 14 waypoints at navigational space 1

Figure 4. 9: Illustration of the multi-route solution representation $h = \{r_1, r_2, r_3, r_4\}$ for 4 UAVs to visit 14 waypoints at navigational space 1

Figure 4. 10: Illustration of the multi-route solution representation $h = \{r_1, r_2, r_3, r_4, r_5\}$ for 5 UAVs to visit 14 waypoints at navigational space 1

Figure 4. 11: Illustration of the navigational space 2 with 8 waypoints

Figure 4. 12: Illustration of the multi-route solution representation $h = \{r_1, r_2, r_3\}$ for 3

UAVs to visit 8 waypoints at navigational space 2

Figure 4.13: Illustration of the navigational space 2 with 14 waypoints

Figure 4.14: Illustration of the multi-route solution representation $h = \{r_1, r_2, r_3\}$ for 3 UAVs to visit 14 waypoints at navigational space 2

4.5 Performance Analysis and Discussions

Simulation results show that the proposed memetic computation approach can always obtain better results than its generic GA-based counterpart. The results are consistent for both navigational spaces. The computational time of the proposed approach is lower than that of the GA-based cooperative mission planner. The conventional GA approaches do not scale well with complexity due to an exponential increase in the search space when large number of waypoints is fed to mutation operator. They tend to converge towards local optima rather than the global optimum of the planning problem. Such a problem can nevertheless be alleviated by transforming the optimization problem. That is, the representation of the problem must be in the simplest form to facilitate such problem tractable to evolutionary search. Our proposed memetic computation approach uses a cooperating multi-individual learning strategy in which local search strategies or memes, each having different learning roles and search features, work together to accomplish the shared optimization goal. With the proposed approach, our cooperative mission planner can always obtain the best combination of solutions based on the given waypoints. The computational complexity of our algorithms is analysed in the following section.

4.6 Computational Complexity

We analyse the computational complexity of the proposed memetic computation approach for solving the mission-planning problem in this section.

4.6.1 Individual Learning

The most intensive computation of the proposed algorithm lies in the individual learning procedure which has a time complexity of $O(n^3 M \bar{I} g)$ where $\bar{I}$ denotes the average number of iterations to arrive at the local optima, M is the population size, n is the number of waypoints, and g denotes the total number of search generations incurred. To reduce the computational complexity, the fitness of new solution $h' \in \mathcal{N}(h)$, $C(h')$ is not directly computed using the fitness function defined in Eq. (4.6). Instead, $C(h')$ is estimated from $C(h)$ and derived as:

$$C(h') = C(h) - \Delta C \quad (4.21)$$

where ΔC represents the change in fitness cost and given in Eqs. (4.9), (4.11), (4.13), (4.15), (4.17) or (4.19) for different heuristics proposed in the individual learning, respectively. Note that Eq. (4.21) reduces the computations of $C(h')$ from $O(n)$ to a linear time complexity of $O(1)$. Hence the proposed individual learning procedure has an overall computational complexity of $O(n^2 M \bar{I} g)$. To further improve the computational complexity of the individual learning search, the search neighbourhood is constrained to size z where $z \ll n$. The eventual computational complexity of the individual learning thus further reduces from $O(n^2 M \bar{I} g)$ to $O(nz M \bar{I} g)$.

4.6.2 Objective Function

We study the time complexity of the proposed fitness function which has an expected time complexity of $O(n)$. However, it is worth noting that instead of evaluating the fitness of a newly evolved solution $h \in P(t+1)$, it is sufficient to compute the difference in total distance travelled by UAV U_j between h and its parents, where U_j is a UAV affected by the genetic operator, which is given by:

$$\Delta l(r_j) = l(r_j^{t+1}) - l(r_j^t) \quad (4.22)$$

where $l(r_j^t)$ is the distance travelled by UAV U_j of the parent solution (h) and $l(r_j^{t+1})$ is the distance travelled by UAV U_j of h . This can be computed in linear time $O(1)$. Thus the expected time complexity to arrive at the fitness of a potential solution h reduces to $O(K)$, where K denotes the average number of UAVs in solution h . Note that the fitness of each individual is estimated in every evolutionary operation, from mutation, crossover, individual learning to constraint validation, and since $K \ll n$, this translates to a significant saving in computational complexity when compare to a direct computation of the fitness values.

In summary, the proposed algorithm is estimated to have a reduction of expected time complexity from $O(n^3 M \bar{I} g) + O(n M g) = O(n^3 M \bar{I} g)$ to $O(n z M \bar{I} g) + O(K M g) = O(n h M \bar{I} g)$. This represents a reduction of n^2 times in computational cycles.

4.7 Summary

The evolutionary technique we presented in this chapter is capable of solving the cooperative mission-planning problem for multiple UAVs to accomplish their missions. We presented a solution approach with multi-route genetic representation and a path-repair mechanism. To achieve robustness, a cooperating individual learning strategy is proposed in which local search strategies or memes, each having different learning roles and search features, work together to accomplish the shared optimization goal. Simulation results show that the proposed cooperative mission planner is less computationally demanding than a GA-based mission planner. The results also highlighted the efficiency and robustness of the memetic computation approach, yielding consistent performance on cooperative mission-planning problems for multiple UAVs.

Chapter 5: Multi-criteria Mission-planning

This chapter focuses on aspect of multi-criteria mission-planning for UAVs. The objective is to compute an optimum solution for each UAV to complete its mission, taking into account the individual criteria and mission constraints. Relying on a swarm intelligence approach, we draw our focus on optimizing a multi-criteria mission-planning problem, catering to the requirements of different mission constraints. Some experimental results and comparisons with evolutionary algorithm to further demonstrate the potential benefits of such approach are presented. The symbols and abbreviations used in this chapter are summarized in the listing below:

W_{ji}	i^{th} waypoint of j^{th} UAV
t_{ji}	Time of j^{th} UAV arrives at its i^{th} waypoint
$d_{i,i+1}$	Distance between any two points in a space
R_j	Enemy threats where $j = 1, \dots, l$
T	Total mission time
v_l	Lower limit of the UAV velocity
v_u	Upper limit of the UAV velocity
z_l	Lower limit of the maximum altitude z
z_u	Upper limit of the maximum altitude z
ψ_l	Lower threshold of the heading angle between two waypoints, W_i and W_{i+1}
ψ_u	Upper threshold of the heading angle between two waypoints, W_i and W_{i+1}
X_i	i^{th} individual (particle) of the population
V_i	Velocity of the i^{th} particle
P_i	Best visited position of the i^{th} particle
g	Global particle guide in the population with superscripts indicating the iteration number
N	Size of the population
w	Inertial weight

$\mathcal{X}$	Restriction factor
c_1 c_2	Acceleration coefficients
n	Iteration number
Δt	Time step
r_1 r_2	Uniformly distributed random numbers
D	Search space
R^n	A sequence of real numbers in n -dimensional space
x	A sequence of n -dimensional decision variables (integer, continuous or discrete)
l_i	Lower bound of i -th decision variable
u_i	Upper bound of i -th decision variable
m	Number of objectives
SC	Set coverage metric
SP	Spacing metric
C_i	A separate cluster of non-dominated solution i
d_{12}	A gap between two individuals i_1 and i_2 in objective space
f_i^{min}	A minimum value of the i^{th} criteria
f_i^{max}	A maximum value of the i^{th} criteria
ϕ	Elevation
θ	Azimuth
r	Slant range
P_r	Probability of radar exposure

5.1 Introduction

A typical UAV mission is characterized by the need to navigate through multiple waypoints depending on the requirements and desired outcome. The execution of the mission may consider criteria such as flight orientation, flight time, fuel consumption or/and risk exposure. It is advantageous in time critical reconnaissance operations that the specific criteria be fulfilled. Our study is to compute an optimum solution for UAVs to accomplish its mission and to solve the multi-criteria mission-planning problem for multiple UAVs. The mission-planning problem is considered as a multi-objective optimization problem to minimize the flight time, the radar exposure and the distance of the total path length. It is highly desirable to equip each UAV with the capability to meet certain objectives while intelligently and autonomously navigate between various points in space — based on considerations such as avoidance of collision and no-fly zones, and optimization of various other mission-specific parameters. Additionally, planning efficient paths can significantly increase the operational endurance of low endurance UAVs. These may be achieved in a hierarchical manner by first computing optimum mission waypoints amongst multiple UAVs and then computing optimal paths for each UAV with its individual criteria. However there are several criteria to be considered simultaneously during optimization. It is normal that these objectives are conflicting in nature and finding an optimal solution involves trade-off among the criteria. The conventional point-by-point approaches for optimization are not appropriate for solving multi-objective optimization problems as the outcome of these classical optimization methods is a single optimal solution. That is, a weighted sum method would convert a

multi-criteria optimization problem into a single criteria optimization. All possible Pareto fronts must first be derived if one would like to obtain the global Pareto optimum since there is only one point on the Pareto front that can be obtained by using a single pair of fixed weights. In order to make sure that every weight combination has been utilized, the algorithms are required to be executed iteratively. Apparently, this is not a feasible way to reiterate the algorithm continually as it would exhaust all the weight combinations. Hence it is essential for the algorithms to gather information from previous performance so as to target the appropriate range of weights in further evolutions.

Our focus in this chapter is on computational intelligence approach – swarm optimization to mission-planning. The flexibility of such an approach is evident when one has to undertake tasks division or allocation to multiple vehicles basing on the overall solution attained by the computational intelligence method. Furthermore, a swarm optimization approach being stochastic in nature incorporates large spaces of candidate solutions in the search; hence variations in solutions generated and at the same time satisfying some pre-specified requirements or constraints are highly desirable. By making use of computational intelligence approach, the problem can be scaled up, able to deal with large-scale multi-criteria mission-planning for UAVs.

Details on the objective formulation of the multi-criteria mission-planning problem are presented in the following section. The remaining of this chapter is organized as follow. Section 5.3 and Section 5.6 present the solution and implementation, respectively, of the

mission planner substantiated with simulation results. Lastly, we summarize the main contributions of this study in Section 5.7.

5.2 Problem Formulation

The mission-planning problem for a UAV can be formulated in terms of a flight cost function subject to constraints of the mission and vehicle physics. We consider the mission-planning problem as an optimization problem with the conflicting objectives of minimizing the flight time, minimizing the fuel consumption and maximizing the satisfaction level of safety factor. For instance, some conflicts may encounter during mission-planning: for the minimization of flight time, shorter path length is to be planned to satisfy the criteria of minimal flight time; whereas for the maximization of safety factor, longer path length is required so as to avoid the threats or obstacles; and for maximizing the satisfaction level of safety factors, careful planning of fuel consumption is necessary. Thus, solving the optimization problems of this UAV mission-planning problem is stimulating from the multi-objective perspective. The conflicting objectives of the proposed mission planner are stated as follows:

5.2.1 Minimizing the Total Flight Time

We consider the primary objective of the mission, i.e. to minimize the time to execute the mission. First, we let $\sum_{i=0}^k W_i$ denotes the k waypoints which are required to visit by m UAVs, and $\sum_{j=0}^m U_m$ denotes the number of UAVs. Then we define W_{ji} as i^{th} waypoint of j^{th} UAV. The time when j^{th} UAV arrives at its i^{th} waypoint is denoted by t_{ji} as follows

$$\begin{cases} t_{j(i+1)} = t_{ji} + \frac{\sqrt{(W_{j(i+1)} - W_{ji})^2}}{v_{ji}} \\ t_{j(1)} = 0 \end{cases} \quad (5.1)$$

Next, we use f_1 to denote the cost according to the time associated to the sequence of points.

$$f_1 = \sum_{j=0}^m \sum_{i=0}^k t_{ji} \quad (5.2)$$

5.2.2 Minimizing the Fuel Consumption

The next criterion for optimization deals with flight distance denoted as f_2 . We consider the fuel consumption as being proportional to the flight distance and we use f_2 to denote the cost according to the distance associated to the sequence of points. For a given sequence $S = [W_1, W_2, \dots, W_k]$ where k corresponds to the number of points, the corresponding f_2 is expressed as:

$$f_2 = \sum_{i=1}^{k-1} d_{i,i+1} \quad (5.3)$$

where f_2 is equal to the distance traversed across the k points and $d_{i,i+1}$ denotes the distance between any two points in a 3-dimensional space which can be computed as:

$$d_{i,i+1} = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2}$$

5.2.3 Maximizing the Safety Level

Consider a navigational space with l enemy threats R_j , where $j = 1, \dots, l$. For a k -entry way-point sequence $S = [W_1, W_2, \dots, W_k]$, its corresponding f_3 is expressed as

$$f_3 = \sum_{i=1}^k \sum_{j=1}^l \text{dist}(W_i, R_j) \quad (5.4)$$

Here, f_3 is defined as the safety factor of a way point; $\text{dist}(W_i, R_j)$ is the distance between the i^{th} way point in a sequence and the j^{th} enemy threat. The navigational path between the way points is considered when avoiding the threats. Another consideration is the distance between the way points and the threats, which should be maximized in order to minimize the risk of collision.

5.2.4 Constraint Functions

The total mission time, T is divided into m sub-periods, with m equivalent to the number of UAVs. During each sub-period, the velocity of the UAV is planned cooperatively. The g_1 puts a threshold on the velocity of the UAV where v_l and v_u represent the lower and upper limits of the velocity of the UAV, respectively.

$$g_1 = \frac{1}{T^2} \left[\left(\frac{dx_i}{dt} \right)^2 + \left(\frac{dy_i}{dt} \right)^2 \right] \text{ where } \frac{v_l^2}{v_u^2} \leq g_1 \leq 1 \quad (5.5)$$

It is assumed that fuel efficiency will deteriorate if the UAV has to carry out frequent ascents and descents during its flight. Hence, the constraint value g_2 should then be proportional to the accumulated vertical displacements of the waypoint sequence. For a

k number of waypoint sequence $S = [W_1, W_2, \dots, W_k]$, the corresponding g_2 is thus expressed as

$$g_2 = \sum_{i=1}^k \frac{|z(W_{i+1}) - z(W_i)|}{\sqrt{(x(W_{i+1}) - x(W_i))^2 + (y(W_{i+1}) - y(W_i))^2}} \quad (5.6)$$

where $z_l \leq g_2 \leq z_u$

The overall maximum altitude z should not surpass or fall below a specified threshold, denoted by z_l and z_u respectively.

Subsequently, to make sure that the UAV is moving toward next waypoint, g_3 denotes a restriction on the heading angle of the UAV.

$$g_3 = \tan^{-1} \left[\left(\frac{\frac{dy_i}{dt}}{\frac{dx_i}{dt}} \right) \right] \quad \text{where} \quad \psi_l \leq g_3 \leq \psi_u \quad (5.7)$$

It is worth noting that ψ is the UAV heading angle between the two waypoints, W_i and W_{i+1} with the lower and upper thresholds represented by ψ_l and ψ_u , respectively. The problem to be solved in this work considers UAV flight envelope constraints in terms of imposed minimum heading angle and maximum altitude. Based on the constraints specified, the mission planner to be developed should generate solution that carry minimal path length, and adhere to physical and motion constraints in the sense that its position at any point along the path should not exceed a specified limit.

5.3 Solution Methodology

5.3.1 Particle Swarm Optimization (PSO)

Swarm intelligence [122] is inspired from the collective groups of simple agents such as ant colonies, bird flocking and fish schooling that lead to the emergence of intelligence global behaviour. A population of particles interact locally with one another and benefit from the findings of other particles in a group. Over time, particles are accelerated towards those particles with better fitness values to mimic the success of other individuals within their group. The key improvement of such an approach over other global minimization strategies is that the large numbers of individuals that form the particle swarm proceeding toward the most favourable region of the solution space and thus causing the approach remarkably resilient to the local minima problem.

Every particle signifies a candidate solution in the particle swarm. We indicate the i^{th} individual (particle) of the population as a D – dimensional vector, $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})^T$ and the velocity of this particle as another D –dimensional vector, $V_i = (v_{i1}, v_{i2}, \dots, v_{iD})^T$ if the search space is D –dimensional. Next, we specify best visited position of the i^{th} particle as $P_i = (p_{i1}, p_{i2}, \dots, p_{iD})^T$ based on the previous iteration. Subsequently, we define g as the global particle guide in the population with superscripts indicating the iteration number; followed by manipulating the population according to the following equations:

$$v_{id}^{n+1} = \mathcal{X}[w v_{id}^n + \frac{c_1 r_1^n (p_{id}^n - x_{id}^n)}{\Delta t} + \frac{c_2 r_2^n (p_{gd}^n - x_{id}^n)}{\Delta t}] \quad (5.8)$$

$$x_{id}^{n+1} = x_{id}^n + \Delta t v_{id}^{n+1} \quad (5.9)$$

where $d = 1, 2, \dots, D; i = 1, 2, \dots, N$; N specifies the size of the population; w denotes the inertial weight and often used as a parameter to manage the search in the solution space; $\mathcal{X}$ indicates a restriction factor to restrict and control the magnitude of the velocity; c_1 and c_2 are acceleration coefficients; n is the iteration number whereas Δt indicates the time step and normally set to 1; r_1 and r_2 are uniformly distributed random numbers. The effectiveness of the particle swarm optimization can be validated from its competence in producing superior solutions while involving a reduced amount of computational time [122]. Selection of the best particle guides in the population is considered as the main challenge in extending particle swarm optimization to multi-objective optimization problems. This is due to the lack of apparent conceptions of identifying global and local best particle guides when dealing with a multi-objective problem.

In this chapter, we make use of swarm intelligence approach for solving multi-criteria mission-planning problem. The proposed algorithm selects non-dominated solutions based on Pareto dominance criteria; searches for best solutions and stores them as a sequence of waypoint candidates; makes use of crowding distance operator to attain true Pareto optimal fronts; and lastly integrates an efficient evaluation method for effective evaluation of the solution space. In the following sections, we discuss our proposed approach in detail.

5.3.2 Multi-objective Optimization

We consider a general multi-objective optimization problem as minimizing a function $f(x)$ depending on q equality and p inequality constraints.

$$\min. f(x) = \{f_1(x)f_2(x) \dots f_m(x)\}^T \quad (5.10)$$

$$x \in D$$

$$\text{where } x \in R^n, f_i: R^n \rightarrow R \text{ and}$$

$$D = \begin{cases} x \in R^n : l_i \leq x \leq u_i, \forall i = 1, \dots, n \\ g_j(x) \geq 0, \forall j = 1, \dots, p \\ h_k(x) = 0, \forall k = 1, \dots, q \end{cases} \quad (5.11)$$

where D indicates a feasible search space; R^n indicates a sequence of real numbers in n -dimensional space; $x = \{x_1 x_2 \dots x_n\}^T$ denotes a sequence of n -dimensional decision variables (integer, continuous or discrete); l_i and u_i represent the lower bound and upper bound of i -th decision variable, respectively; and lastly m denotes the number of objectives. The optimization problem yields Pareto optimal solutions by optimizing the vector function simultaneously. In our study, we consider Pareto front as a collection of non-dominated (Pareto optimal) solutions. A solution x^* is signified as dominated by another solution x , if x is superior than x^* pertaining to all objectives [123].

5.4 The Proposed Algorithm

The proposed algorithm involves initialization, evaluation and exploration of population by integrating optimization operators with Pareto-dominance criteria. In our proposed algorithm, we first evaluate the particles and check for dominance relation among the population. Next, we search for non-dominated solutions and store them as waypoint candidates in order to lead the search particles. It is worth noting that the variable size of the sequence of waypoint candidates is to improve the computational efficiency of the algorithm during optimization. Additionally, we make use of crowding distance assignment operator to trim down the size of the waypoint candidates if it exceeds the allowable size. Lastly, we exploit an efficient evaluation strategy for evaluating the search space. By combining these operators, the algorithm is competent to sustain diversity in the population and successfully explore towards true Pareto optimal fronts. The main steps of the procedure are described as follow:

#1 The algorithm begins with the generation of potential solutions to form an initial population. The iteration counter t is set to 0.

- i. Apply real numbers in the specific range of decision variable randomly to initialize the existing location of the i -th particle x_i ;
- ii. Apply uniformly distributed random number in $[0, 1]$ to initialize every particle velocity vector v_i ;
- iii. Assess every particle in the population;

iv. Place the individual best location P_i to x_i .

#2 Classify non-dominated solutions among the particles and save those non-dominated solutions as waypoint candidates.

#3 $t = t + 1$.

#4 Iterate the loop as follows:

- i. Choose randomly a global best particle guide P_g for the i^{th} particle among waypoint candidates.
- ii. Compute the new v_i , and the new x_i based on Eq. (5.8) and Eq. (5.9), respectively.
- iii. Iterate the loop for every particle in the population.

#5 Assess all the particles in the current population.

#6 Examine the Pareto dominance for every particle in the population. Replace local best solution P_i with current solution if P_i is dominated by the current solution.

#7 Classify non-dominated solutions among the particles and save them as waypoint candidates.

#8 Restrict the size of the sequence of waypoint candidates by applying the crowding distance operator if it exceeds the allowable size.

#9 Implement generic evaluation on a particular number of particles.

#10 Return to step 3 if the termination criterion is not met; else output the set of non-dominated solutions from the sequence of waypoint candidates.

Subsequently, we present the main operators of the proposed procedure in the following sections.

5.4.1 Sequence of waypoints candidates

The performance evaluation and selection of the best global particle guide are vital procedures in a multi-criteria optimization approach. They sustain a superior range of non-dominated solutions and influence the convergence capability of the algorithm. In the preceding iteration of the procedure, any of the non-dominated solutions from the sequence of waypoint candidates can be chosen as a global guide. We propose a performance evaluation method to make sure every particle in the population moves towards the true Pareto optimal region. Moreover, the crowding distance operator is used to perform restriction on the waypoint candidates. This operator ensures the highest crowding distance values of non-dominated solutions are stored as waypoint candidates. Subsequently, we evaluate and select the best global particle guide from a restricted size of the waypoint candidates. During optimization, the variable size of the waypoint candidates manages to reduce the computational time. However the computing requirement for the sorting and crowding value becomes greater when the size increases. Hence in the initial steps of the procedure, we set the size to half of the maximum size of waypoint candidates; followed by increasing the value in a stepwise manner to explore the solution space effectively. This procedure evaluates distinct particle guides from a

restricted number of waypoint candidates and thus improves the algorithm performance by enabling the particles explore towards the true Pareto optimal region.

5.4.2 Crowding Distance Assignment Operator

The crowding distance assignment operator [124] specifies an estimate of the density of solutions. In a particular solution, the average distance of two neighbouring solutions is the crowding distance value. We compute the crowding distance value by first sorting the solution sequence in ascending order of the objective function values and followed by securing boundary solutions with the smallest and greatest values of the objective function. Subsequently, the final crowding distance value of a solution is computed by adding up the individual crowding distance values in every objective function. The detailed computation of crowding distance value is presented in Listing 5.1.

1. Obtain the size of non-dominated solutions from the sequence of waypoint candidates

$$l = |S| \text{ where } S = [W_1, W_2, \dots, W_k]$$

2. Initialize distance.

For $i = 1$ *to* l

$$S[i].dist = 0$$

3. Calculate the individual crowding distance value.

4. Sort each solution according to objective function value.

For each objective m

$$S = \text{sort}(S, m)$$

5. Set the boundary points to infinite to guarantee their selection.

$$S[1].dist = S[l].dist = \infty$$

For $i = 2$ *to* $(l - 1)$

$$S[i].dist = S[i].dist + \frac{S[i+1].m - S[i-1].m}{f_m^{max} - f_m^{min}}$$

Listing 5.1: Crowding distance assignment operator

5.4.3 Generic-evaluation Operator

In this work, we apply the generic evaluation operator involving priorities, goals and Pareto sets in which we can place the objectives in levels of priority and impose constraints on each of them. The operator evaluates all non-dominated solutions in the population, determines if one Pareto dominates the other, and ranks them accordingly. In order to set the priority levels of our objectives for the mission-planning problem, we classify them as follows:

- constraints that the UAV has to fulfil due to its physical restrictions, or mission requirements;
- mission-specific performance indices to be optimized.

Constraints			
	Velocity	Height limit	Turning angle
Priority Level	High	High	High
Performance Indices			
	Flight time	Fuel efficiency	Safety level
Priority Level	Med	Med	Med

Table 5. 1: Classifications of UAV constraints and performance indices

Based on the classification as presented in Table 5.1, the constraints that the UAV must fulfil are set as high-priority levels. On the other hand, the performance indices to be optimized are set at lower priority levels in line with the goal of the mission. The rows of Table 5.1 indicate priority levels for the objectives. By using this method, we can

simply modify the constraints or performance indices as required by the mission. However, to cater for certain situations such as sudden pop-up threats while UAV is navigating through a hostile environment, a simpler problem model is required to deal with the situation. Mitigation of threat exposure is more important than the other performance indices when replanning. Hence the priority level of threat exposure is set to high and the rest of the performance indices are reduced to low. This improves the performance of multi-objective optimization while extending from conventional optimization algorithm. The evaluation operator works on a specific number of particles and distributes the non-dominated solutions along the true Pareto optimal front. Initially, this operator strives to substitute the non-feasible solutions with the evaluated particles of waypoint candidates. Subsequently, it attempts to exploit the search space in the sequence of waypoint candidates along the Pareto fronts. The evaluation procedure is described as follows.

1. Sort the particle fitness function and get the particle index number respectively.
2. Employ crowding distance assignment operator for the calculation of the density of solutions in the sequence of waypoint candidates. Sort them accordingly and choose one of the least crowded solutions from the waypoint candidates randomly as particle guide.
3. Evaluate a specific number of particles and rank them accordingly.

It is worth noting that the evaluated value is restricted to the upper limit or lower limit and the particle velocity vector remains unchanged during the steps of evaluation.

5.4.4 Constraint Handling

We adopt the constraint handling mechanism [124] to deal with the constrained optimization problems. In this simple and effective method, a solution i is considered as a constrained dominate solution j if it meets any of the conditions as follows:

1. Solution i is deemed as feasible while solution j is not.
2. Both solutions i and j are deemed as non-feasible, although solution i has a smaller overall constraint violation.
3. Both solutions i and j are deemed as feasible when solution i dominates solution j .

Lastly, we demonstrate the efficiency and applicability of the proposed approach in the following sections.

5.5 Experimental Results

We apply our proposed approach to the UAV mission-planning problem by making use of the following parameters. We set the initial population of the algorithm to 200; set c_1 and c_2 to 1.0 and 0.5 respectively; define w as 1.0, $\mathcal{X}$ as 0.9; and specify the size of non-dominated solutions as 200. We then run the algorithm for 500 iteration steps. In this study, we use set coverage metric (SC) and spacing metric (SP) [125] as performance measures to assess the performance of our proposed approach. The SC (A, B) metric produces the solution dominance and relative convergence between two solution vectors A and B; calculating the proportion of B, which is weakly dominated by A. The role of SP metric is to assess the distribution of vectors throughout the set of non-dominated solutions and indicate how evenly the generated non-dominated solutions in the approximation set are distributed in the search space. The computational complexity of the algorithm is dominated by the objective function computation, crowding distance computation and the non-dominated comparison of the particles in the population. If there are M objective functions and N number of solutions (particles) in the population, then the objective function computation has $O(MN)$ computational complexity.

For comparison purposes, we run the multi-objective evolutionary algorithm by specifying the initial population as 200; probability of crossover as 0.9; and probability of mutation as $1/n$ (n is the size of real variables). Next, we set the crossover and mutation distribution index values to 20 and 100 respectively and run the evolutionary

algorithm for 500 generations. By way of example, we consider the navigational space portrayed in Fig. 5.1.

Figure 5. 1: Navigational space with start and target locations (circles) as well as known threats (diamonds)

The main objective is to find an optimum solution which minimizes the UAV's risk of collision while flying from its current position to the target locations. The flight trajectory should also take into account the path length selected for the UAV, so as not to exceed prescribed fuel consumption at any point. The main idea is to have a generated solution with maximal level of safety factor while complying with physical and motion constraints.

To show the effectiveness of the proposed algorithm, first it is applied to path length and safety factor as two objectives of the planner for solving the mission-planning problem. The mission planner consists of minimizing path length (Eq. (5.3)) and maximizing the level of safety factor (Eq. (5.4)) subject to satisfying the constraints from Eq. (5.5) to (5.7). To examine the performance, we then carry out 10 independent runs for the mission planner by applying both the proposed algorithm and the evolutionary algorithm. The resulting statistics for both the algorithms is tabulated in Table 5.2.

Statistics	Performance metric			
	Set coverage (SC) metric		Spacing (SP) metric	
	SC(A,B)	SC(B,A)	A	B
Best	0.1670	0.0360	164.9022	196.9804
Worst	0.7031	0.5080	236.3021	630.2212
Mean	0.4466	0.2302	206.6202	402.4579
Variance	0.0349	0.0254	1136.9908	26066.7456
SD	0.1668	0.1426	30.1462	146.5188

Table 5. 2: Resulting statistics by the proposed PSO algorithm and evolutionary algorithm for the mission-planning problem. In SC(A,B), A is the proposed PSO algorithm and B is the multi-objective evolutionary algorithm. The best performing algorithm is indicated by bold numbers.

From Table 5.2, it can be seen that the mean value of SC (A, B) is higher than the mean value of SC (B, A) pertaining to set coverage metric. The SC (A, B) metric represents the proportion of B solutions that are weakly dominated by A solutions. Therefore we can conclude that our proposed algorithm is able to perform better than the multi-objective evolutionary algorithm. Likewise, the proposed PSO algorithm has lower spacing metric mean value than the multi-objective evolutionary algorithm; indicating that the proposed algorithm obtains the better distribution of Pareto optimal solutions. A sample of the experimental result corresponding to SC (A, B) median value is presented in Fig. 5.2 for illustration purposes.

Figure 5. 2: Non-dominated solutions obtained using the proposed PSO algorithm and evolutionary algorithm for mission-planning problem. The two objectives are f_1 - path length and f_2 - safety factor

It can be seen that the proposed algorithm is competent of generating a well-distributed set of Pareto optimal solutions. The remarkable performance of the proposed algorithm can be attributed to the application of the sequence of waypoint candidates and the generic evaluation operator. The evaluation operator distributes the non-dominated solutions along the true Pareto optimal front and improves the exploratory capabilities of the algorithm to prevent premature convergence. In the preceding iteration of the procedure, any of the non-dominated solutions from the sequence of waypoint candidates are chosen as a global guide. The selection of the global guide of the particle swarm is a crucial step in a multi-objective PSO algorithm. It affects both the convergence capability of the algorithm as well as the good spread of non-dominated solutions. Our proposed algorithm manages to produce boundary solutions easily in this case; whereas the multi-objective evolutionary algorithm fails to yield the boundary solutions.

It shows that our proposed algorithm is effective for mission-planning as there are several alternatives for selection at distinct objective satisfaction levels. The proposed planner can make a right decision by analysing the trade-off between the conflicting objectives based on the circumstances prevailing under the mission-planning problem. In the following sections, the selection of resultant solution for the mission-planning problem is presented in detail.

5.5.1 Clustering Approach

Our proposed approach is capable of finding optimal solutions by trading off the predefined criteria. The procedure is likely to find N non-dominated solutions after running for an adequate number of iterations. However, it is time-consuming to consider only one solution among the N non-dominated solutions. Therefore we exploit a clustering approach [125] in which only a few representative N non-dominated solutions are well-dispersed from each other on the solution space. Firstly, each solution is assumed to fit into a distinct cluster and the gap between each pair of clusters is then computed. Next, the clusters with the smallest gap are merged together to make up a significant cluster. Subsequently, the procedure is terminated after a satisfactory number of clusters are discovered and only one best solution from each cluster is stored while the rest of them are discarded. The detailed clustering approach is described as follows:

#1 Initialize a cluster group C ; Represent each non-dominated solution i as a separate cluster. i.e. $C_i = \{i\}$, $C = \{C_1, C_2, \dots, C_N\}$

#2 If, $|C| \leq \bar{N}$ proceed to #5, else proceed to #3.

#3 Compute the gap for each pair of clusters by Eq. (5.12),

$$d_{12} = \frac{1}{|C_1| \cdot |C_2|} \sum_{i \in C_1, j \in C_2} d(i, j) \quad (5.12)$$

where d_{12} denotes the gap between two individuals i_1 and i_2 in objective space.

#4 Search for (i_1, i_2) pair with respect to the smallest gap and followed by merging the two clusters C_{i_1} and C_{i_2} together. The size of C is reduced by one. Return to #2.

#5 Store the best solution with least average gap from each cluster and discard the rest.

We manage to reduce the great number of non-dominated solutions to 20 representative solutions by applying the clustering approach. The representative non-dominated solutions for the mission-planning problem are as shown in Fig. 5.3.

Figure 5. 3: Representation of clustered non-dominated solutions

Lastly, with the well-spread clusters available on the solution space, a composite measure is assigned to each cluster based on its location on the space so as to facilitate the selection of resulting optimum solution. The detailed Pseudo-weight vector approach is presented in the following section.

5.5.2 Pseudo-weight Vector Approach

The pseudo-weight vector [125] is assigned according to the relationship between the found solution and the individual criteria,

$$\omega_i = \frac{(f_i^{max} - f_i(x)) / (f_i^{max} - f_i^{min})}{\sum_{m=1}^M (f_i^{max} - f_m(x)) / (f_i^{max} - f_m^{min})} \quad (5.13)$$

where f_i^{min} and f_i^{max} are the minimum and maximum values of the i^{th} criteria, respectively. Subsequently, a straightforward approach is to select the desired solution with the computed weight vectors. This provides ease in selection of final solution for mission-planning problem. Suppose the mission planner decides to plan a mission with weights 0.4 and 0.6 for path length and safety factor respectively, then we can consider a solution associated to the set of weight vectors. Fig. 5.4 shows the optimum solutions avoiding the threats as well as optimizing the path length together with other predefined constraints.

Figure 5. 4: Illustration of the optimum solutions

5.6 Implementation

The proposed mission planner readily implements the corresponding mission. In order to successfully perform its mission, UAVs are required to minimize the exposure by radars based on the orientation of the UAV. With an optimum set of waypoints from an initial position to goal locations situated within radar regions, the UAV would simply need to maintain a safe distance by translating at a fixed orientation in the horizontal plane around the radars. However since certain state variables such as azimuth angle may have greater influence on radar cross section (RCS) [100] [101] the planner would need to search for a continuous sequence of azimuths that minimizes the probability of radar exposure. We enable the guidance and control solution to explicitly incorporate the empirical data that characterizes the UAV's RCS varying upon the direction of the illuminating radar.

Figure 5. 5: RCS plot of a target in our study

As shown in Fig. 5.5, it can be observed that the UAV beam has the highest RCS. This is because of increasing reflectivity of the perpendicular aspect and the exposure of the large physical area to the radar. The nose/tail area of the UAV has the second highest RCS due to echoes off the engines or propellers. The greatest reflection illustrated in the example is in the $90^\circ/260^\circ$ positions whereas the weakest reflection is in the $135^\circ/320^\circ$ positions. Generally, the RCS of a target is calculated or measured as a function of the azimuth and elevation angles. It is assumed that the UAV is navigating at a constant altitude with zero pitch and bank angle. Three scalars, namely elevation (ϕ), azimuth (θ), and slant range r , computed using the locations of the UAV and the radar. For each radar, they are defined as follows:

$$\theta = \tan^{-1}\left(\frac{-(x_r - x_i)y_i + (y_r - y_i)x_i}{(x_r - x_i)x_i + (y_r - y_i)y_i}\right) \quad (5.14)$$

$$\phi = \tan^{-1}\left(\frac{u_r - u_i}{\sqrt{(x_r - x_i)^2 + (y_r - y_i)^2}}\right) \quad (5.15)$$

$$r = \sqrt{(x_r - x_i)^2 + (y_r - y_i)^2 + (u_r - u_i)^2} \quad (5.16)$$

where the coordinates with subscript r are associated with the location of the radar. The first two variables determine the signature value s of the UAV with respect to the associated radar. Depending on the range r and the signature s of the UAV, with respect to each radar, there is a probability of radar exposure P_r . We consider P_r as $f(s, r)$ where $s = f(\theta, \phi)$ and $r = \sqrt{((x_r - x)^2 + (y_r - y)^2 + (z_r - z)^2)}$ depending on the distance to the radar site from a given position (x, y, z) in the navigational space. It is worth noting that P_r increases with increasing s and decreasing r . For instance, there is a region of low signature s when the nose of the UAV exposes directly away from the radar or straight to the radar and the elevation magnitudes are comparatively low. The minimum and maximum probabilities of radar exposure are 0.01 and 0.99, respectively. If the probability of radar exposure is maximal, the UAV could be at risk as it is completely exposed to the radars. Therefore, a sequence of commands that presents lower probability of radar exposure is desirable.

For each i^{th} point of the trajectory and j^{th} radar, the RCS_{ij} of the UAV and the distance R_{ij} between the UAV and the radar are considered. The RCS_{ij} depends on the orientation of the UAV with respect to the radar. We assume that the types, engagement zones, and locations of radars are known so as to facilitate the calculation of the azimuth and elevation angles. It is worth noting that we do not model the radar zones as obstacles in our study. The solution approach was implemented on a flight simulation environment as illustrated in Fig. 5.6.

Figure 5. 6: The flight simulation environment

The sequence of generated azimuth command simply orients the vehicle so that its exposure to the radars is minimal. Fig. 5.7 demonstrates that the UAV attempts to minimize its radar exposure by finding optimum entry approach to the radar zones towards the target locations.

Figure 5. 7: The UAV following generated optimum set of waypoints

It can be observed that in the optimum entry approach, the UAV moves with low azimuths. To reach target locations inside the radar zones, the UAV navigates through

these zones with optimized azimuths. For instance, “nose-in” for entry and “tail-in” for exit. Note that the RCS ranges from 0.01 when observed from within the plane of the wings to 0.5 when observed from a direction normal to the wings. The stealth solution, generated by the proposed mission planner, is shown with a path connecting the optimum set of waypoints. The resulting stealth solution successfully minimizes the exposure by radars. The standard mission-planning approach would simply generate a straight-line route that go directly through the centres of the radars in this simulation. In contrast, our proposed approach generates a stealth solution that skirts radar engagement zones. Note that in this example, the result shows the optimal solution avoiding the radar regions as well as optimizing the other predefined constraints.

5.7 Summary

In this chapter, we presented a multi-criteria optimization approach for mission-planning problem. This approach is developed to generate solutions by yielding a compromise among the criteria; distinguish a set of optimum solutions to the mission-planning problem. To ease straightforward execution for the mission planner, an effective selection of resulting solution is demonstrated. The experimental results demonstrate that our proposed approach is a feasible alternative for solving the mission-planning problem of UAV that has several criteria to be considered simultaneously while navigating through territory to accomplish the mission. Lastly, our proposed planner is experimentally validated through a flight simulation environment.

Chapter 6: Concluding Remarks

This chapter summarizes the core research contributions and also outlines some potentially challenging and interesting directions for the future research.

6.1 Conclusions

Unmanned Aerial Vehicles (UAVs) can be useful for missions in high-risk environments, particularly in situations where it may be hazardous for human operators. However, issues on maximum endurance, efficiency as well as minimum vital risks have to be considered. Therefore it is important to minimize the risk of detection or more specifically, the probability of detection of the UAVs by radars. Currently operating UAVs use rudimentary guidance technologies, such as following pre-planned or manually provided waypoints. In order to allow more complex mission or task, UAVs require more advanced guidance capabilities, in particular ones that increase the vehicle autonomy. The availability of various guidance technologies and the understanding of operational specifications create opportunities to work towards refining and implementing such advanced guidance concepts. The basic difficulties include partially known and changing environment, extraneous factors, such as threats, evolving mission elements, tight timing and positioning requirements. Moreover, these must be tackled, while at the same time, explicitly accounting for the actual vehicle maneuvering capabilities, including dynamics and flight-envelope constraints. Thus this thesis focused

on developing efficient approaches that aid in the stealth of these UAVs to accomplish their missions when they travel over radar zones.

The UAV operation presented in Chapter 3 is known as the path-planning problem. We proposed a radar-aware path-planning scheme which generates a feasible path for UAVs over radar territory. The key strengths of the method are: 1) the ability to plot a safe path to the target while minimizing exposure to the radars; 2) the generated paths are smoothened using B-splines to improve navigability 3) the distance between the UAV and the radars is optimized in order to minimize the energy received at the radar and 4) computational efficiency is achieved. The ability to generate feasible paths by using this path-planning approach has been demonstrated through simulations.

For guidance and navigation control of UAVs, we propose a path smoothening algorithm using a set of path patterns which take into account the kinematic constraints of the UAV. Instead of smoothening the entire path from an initial position to the goal position, we smoothen the path segments according to the position of the UAV so as to minimize the radar exposure as well as to improve navigability. By incorporating the path patterns from off-line optimization, the proposed smoothening scheme has the advantage of minimal on-line computational cost since most of computation could be done off-line.

The UAV operations described in Chapter 4 and 5 are characterized as mission-planning problems. In Chapter 4, a memetic computation approach has been introduced to optimize the operations of multiple UAVs. The proposed approach is capable of solving

cooperative mission-planning problem for multiple UAVs to accomplish their missions. Taking advantage of memetic intelligence, our proposed mission planner is competent of handling large-scaled optimization problems for multiple UAVs. We investigate the number and placement of control points within a navigational space by considering the complexity of the terrain and identify the waypoints based on the topographic information of the terrain. To achieve robustness, a cooperating individual learning strategy is proposed in which local search strategies or memes, each having different learning roles and search features, work together to accomplish the shared optimization goal. Simulation results show that the proposed cooperative mission planner is less computationally demanding than a genetic algorithm (GA) -based mission planner. The results also highlighted the efficiency and robustness of the memetic computation approach, yielding consistent performance on cooperative mission-planning problems for multiple UAVs.

Finally, a multi-criteria optimization approach is presented in Chapter 5 to generate Pareto-optimal solutions for UAV mission-planning. The proposed method made use of the generic multi-criteria Pareto-evaluation function to permit the change of the levels of priority as well as the constraints of the mission. By doing so, the proposed planner can simply be applied for various types of missions by appending different terms in objective function and constraints. Experimental results prove that the proposed approach is a feasible alternative to solve the mission-planning problem for UAV that has several criteria to be considered simultaneously while navigating through territory to accomplish the mission. Furthermore, the performance of the proposed planner is

validated through the flight simulation environment; indicating the applicability of the proposed approach on the actual system.

6.2 Future Research

To deal with loss of GPS (Global Positioning System) information, terrain based navigation can be considered. It is necessary to derive real-time positional reference information for UAVs to ensure the continuity of their mission during the outage of GPS. Further enhancements based on machine learning techniques are considered. Some suggestions for future work are as follows. For convenience and ease of referencing, the symbols and abbreviations used in our discussions are as follows:

$g(\bullet)$	nonlinear piecewise continuous activation function
W_{ji}	weight between the i^{th} input neuron and j^{th} hidden layer neuron
B_j	bias for the j^{th} hidden layer neuron
X_{ik}	input at the i^{th} input neuron of the k^{th} training case
H_{jk}	matrix containing activation of the j^{th} hidden layer neuron for the k^{th} training case
β	a vector representing weights between hidden layer neurons and the output layer neuron
$\mathbf{T}$	a vector representing targets for all training cases
$\mathbf{H}'$	Moore-Penrose pseudoinverse of matrix $\mathbf{H}$
a_i, b_i	hidden neurons parameters
O_i	real output of the network
t_i	target output corresponding to the i^{th} tested data point
N	number of tested data points
DT	Delaunay Triangulation
ELM	Extreme Learning Machine
GPS	Global Positioning System

During mission execution, UAVs rely on global positioning system (GPS) information to ascertain its position for navigation. However, GPS devices rely on information from Global Navigation Satellite System (GNSS) about global positional information, including the longitude, the latitude and the time when the signal was released from the satellites. When signals lapse, it can cause a breakdown of information, causing the GPS device to operate in an erroneous mode or possibly shutting down. Moreover, there are certain atmospheric indicators which can cause the inaccurate mapping of certain areas, such as vast waterways or terrain, large structures, electronic interference, or dense foliage. All these can affect signal reception resulting in positional errors or even false readings.

In the absence of GPS information, the capability of a UAV to carry out its intended mission is hindered. In this sense, the UAV is incapacitated and if the disruption is too serious, the success of the mission is jeopardized. It is therefore necessary to find alternative means for UAVs to derive positional reference information to ensure the continuity of the mission. In recent years, there have been significant progresses made in terrain-based navigation [126] [127] due to more widespread availability of sensors that can directly sense terrain while the platform carrying the sensor is in motion. The technique of terrain based navigation has been utilized by cruise missiles and aircrafts over land, aiding the navigation system based on terrain information in order to increase the estimation accuracy.

In principle, Delaunay triangulation (DT) interpolation is directly applicable to estimate elevations of points in terrain modeling. However DT requires more floating point

parameters to represent a terrain [128]. In the case of UAV, the amount of memory allocated for storage of terrain data is preferably small so that much of the memory resources can be dedicated to other components onboard the UAV. Not denying the fact that the current level of technology have resulted in cheap and readily available memory resources, a more compact and efficient representation of geographical and terrain data is still advantageous.

Our future research is to explore a connectionist approach as a basis for deriving positional reference information based on the terrain data captured by the UAV onboard sensor. The objective is to estimate the state of the UAV (position and attitude) and a map of the surrounding environment simultaneously based on limited sensing capabilities. The sensed terrain information is correlated with existing reference data to derive the navigation estimate. We aspire to present a scheme based on extreme learning machine (ELM) [129] [130] [131] as a mechanism for learning the stored digital elevation information to perform the estimation or map building for UAV navigation without the need for GPS or other derive positioning information. The advantages of the ELM are that it has only one tunable parameter, specifically the number of neurons, and its training algorithm consists of only a single step. The parameters of hidden neurons need not be tuned and can be randomly generated according to any continuous probability distribution. ELM achieves this by avoiding iterative optimization of the hidden neuron parameters (input weight vector and the biases for additive hidden neurons and the centers and impact factors for radial basis function hidden neurons). Thus the ELM needs far less memory than that needed by other approaches. In a

nutshell, the prominent features of machine learning approaches introduce competitive solution for terrain based navigation and will greatly facilitate the research in creating a real-time navigation plan for routing a fleet of UAVs during the outage of GPS. On the whole, our preliminary study [132] has shown how ELM can be used as representation of patterns or models. We make use of ELM as a mechanism for learning the stored elevation data which can introduce competitive solution for UAVs to navigate through terrain without the need of GPS. Further enhancements based on the representation of terrain models by creating a real time navigation plan to be used for routing a fleet of UAVs are considered in future.

References

- [1] A. Agarwal, M. H. Lim, M. J. Er and T. N. Nguyen, "Rectilinear workspace partitioning for parallel coverage using multiple UAVs," *Advanced Robotics*, vol. 21, no. 1, pp. 105-120, Jan 2007.
- [2] K. K. Lim, Y. S. Ong, M. H. Lim and A. Agarwal, "Hybrid ant colony algorithm for path planning in sparse graphs," *Soft Computing Journal*, vol. 12, no. 10, pp. 981-994, Aug 2008.
- [3] C. W. Yeu, M. H. Lim, G. B. Huang, A. Agarwal and Y. S. Ong, "A new machine learning paradigm for terrain reconstruction," *IEEE Geoscience & Remote Sensing Letters*, vol. 3, no. 3, pp. 382-386, Jul 2006.
- [4] I. Davis, "Warp Speed: path planning for Star Trek Armada," in *AAAI Spring Symposium on Interactive Entertainment & Artificial Intelligence*, Menlo Park, CA, Mar 2000.
- [5] E. Frazzoli, M. Dahleh and E. Feron, "Real-time motion planning for agile autonomous vehicles," *Journal of Guidance, Control & Dynamics*, vol. 25, no. 1, pp. 116-129, Jan 2002.
- [6] R. Dechter and P. Judea, "Generalized best-first search strategies and the optimality of A*," *Journal of Association for Computing Machinery*, vol. 32, no. 3, pp. 505-536, Jul 1985.
- [7] B. Stout, "Smart moves: intelligent path finding," *Game Developer Magazine*, pp. 28-35, Jul 1997.
- [8] S. Arinaga, S. Nakajima, H. Okabe, A. Ono and Y. Kanayama, "A motion planning method for an AUV," in *Symposium on Autonomous Underwater Vehicle Technology*, Monterey, CA, Jun 1996.
- [9] P. Chandler, S. Rasmussen and M. Pachter, "UAV cooperative path planning," in *AIAA Guidance, Navigation & Control Conference*, Denver, CO, Aug 2000.
- [10] J. F. Gilmore, "Autonomous vehicle planning analysis methodology," in *Proc. Assoc. Unmanned Veh. Syst. Conf.*, Washington, DC, 1991.
- [11] R. J. Szczerba, "Threat netting for real-time, intelligent route planners," in *Proc. IEEE Symp. Inf., Decision, Contr.*, Adelaide, Australia, 1999.
- [12] C. H. glu and A. C. Sanderson, "Planning multiple paths with evolutionary speciation," *IEEE Trans. Evol. Comput.*, vol. 5, p. 169-192, Jun. 2001.
- [13] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd ed, Berlin, Germany: Springer-Verlag, 1996.
- [14] J. Xiao, Z. Michalewicz, L. Zhang and K. Trojanowski, "Adaptive evolutionary planner/navigator for mobile robots," *IEEE Trans. Evol. Comput.*, vol. 1, no. 2, p. 18-28, Apr. 1997.
- [15] M. Pellazar, "Vehicle route planning with constraints using genetic algorithms," in *Proc. Nat. Aerosp. Electron. Conf.*, Dayton, OH, 1994.
- [16] I K Nikolos, K P Valavanis, N C Tsourveloudis and A N Kostaras, "Evolutionary algorithm based offlineonline path planner for UAV navigation," *IEEE Trans.*

- Syst. Man. Cybern. B, Cybern.*, vol. 33, no. 12, p. 898–912, Dec 2003.
- [17] M. Yi, M. Ding and C. Zhou, "3D route planning using genetic algorithm," in *Proc. SPIE Int. Symp. Multi-Spectral Image Process.*, Wuhan, China, 1998.
 - [18] E. M. Kan, M. H. Lim, S. P. Yeo, Z. H. Shao and J. S. Ho, "Radar Aware Path Planning with B-Spline Trajectory Generation for Unmanned Aerial Vehicles (UAVs)," *International Journal of Reasoning-based Intelligent Systems*, vol. 2, no. 3, p. 226 – 236, 2010.
 - [19] Y. Kanayama and B. I. Hartman, "Smooth Local Path Planning for Autonomous Vehicles," in *IEEE International Conference on Robotics and Automation*, May 1989.
 - [20] S. A. and C. Laugier, "Planning Sub-Optimal and Continuous Curvature Paths for Car-Like Robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Victoria, B. C., Canada, Oct. 1998..
 - [21] L. E. Dubins, "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents," *American Journal of Mathematics*, vol. 79, p. 497–516, 1957.
 - [22] Boissonnat, J. D., A. C´er´ezo and J. Leblond, "Shortest Paths of Bounded Curvature in the Plane," in *Proceedings of 9th IEEE International Conference on Robotics and Automation*, 1992.
 - [23] X.-N. Bui and P. Sou`eres, "Shortest Path Synthesis for Dubins Nonholonomic Robot," in *Proceedings of IEEE International Conference on Robotics and Automation*, San Diego, CA, May 1994.
 - [24] G. Yang and V. Kapila, "Optimal Path Planning for Unmanned Air Vehicles with Kinematic and Tactical Constraints," in *Proceedings of the 41st IEEE Conference on Decision and Control*, Las Vegas, NV, Dec 2002.
 - [25] A. R. Larson, M. J. Pachter and M. Mears, "Path Planning by Unmanned Air Vehicles for Engaging an Integrated Radar Network," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, CA, Aug. 2005.
 - [26] A. Stentz, "The Focussed D* Algorithm for Real-Time Re-planning," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995.
 - [27] S. Koenig and M. Likhachev, "D* Lite," in *Proceedings of the National Conference on Artificial Intelligence*, AAAI Press, Menlo Park, CA, 2002.
 - [28] S. Koenig, C. Tovey and Y. Smirnov, "Performance Bounds for Planning in Unknown Terrain," *Artificial Intelligence*, vol. 147, p. 253–279, 2003.
 - [29] Koenig, S. and M. Likhachev, "Fast Replanning for Navigation in Unknown Terrain," *IEEE Transactions on Robotics*, vol. 21, no. 3, p. 354–363, 2005.
 - [30] A. Ranganathan and S. Koenig, "A Reactive Robot Architecture with Planning on Demand," *Intelligent Robots and Systems*, vol. 2, p. 1462–1468, 2003.
 - [31] P. Wu, R. Clothier, D. Campbell and R. Walker, "Fuzzy Multi-objective Mission Flight Planning in Unmanned Aerial Systems," in *Proceedings of IEEE Symposium on Computational Intelligence in Multi-Criteria Decision Making*,

2007.

- [32] A. Stentz, "Optimal and Efficient Path Planning for Partially-Known Environments," in *Proceedings of the IEEE International Conference on Robotics and Automation*, May 1994.
- [33] A. Stentz, "Map-based Strategies for Robot Navigation in Unknown Environments," in *Proceedings AAAI 1996 Spring Symposium on Planning with Incomplete Information for Robot Problems*, Menlo Park, CA, 1996.
- [34] S. Koenig and M. Likhachev, "Incremental A*," *Advances in Neural Information Processing Systems*, p. 1539–1546, 2002.
- [35] A. Ryan, M. Zennaro, A. Howell, R. Sengupta and J. K. Hendrick, "An Overview of Emerging Results in Cooperative UAV Control," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, Piscataway, NJ, 2004.
- [36] S. Scherer, S. Singh, L. Chamberlain and S. Saripalli, "Flying Fast and Low among Obstacles," in *Proceedings of IEEE International Conference on Robotics and Automation*, Piscataway, NJ, 2007.
- [37] J. Langelaan and S. Rock, "'Towards Autonomous UAV Flight in Forests,'" in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Reston, VA, 2005.
- [38] B. Sinopoli, M. Micheli, G. Donato and T. J. Koo, "Vision Based Navigation for an Unmanned Aerial Vehicle," in *Proceedings of 2001 IEEE International Conference on Robotics and Automation*, Piscataway, NJ, 2001.
- [39] J. Z. Sasiadek and I. Duleba, "3D Local Trajectory Planner for UAV," *Journal of Intelligent and Robotic Systems*, vol. 29, p. 191–210, 2000.
- [40] L. E. Kavraki, P. Svetska, J. C. Latombe and M. H. Overmars, "Probabilistic Roadmaps for Path Planning in High Dimensional Configuration Spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, p. 566–580, 1996.
- [41] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," *TR 98-11, Computer Science Dept., Iowa State University*, 1998.
- [42] Y. Q. Qin, D. B. Sun, N. Li and Y. G. Cen, "Path Planning for Mobile Robot using the Particle Swarm Optimization with Mutation Operator," in *Proceedings of the 3rd International Conference on Machine Learning and Cybernetics*, 2004.
- [43] P. Williams, "Real Time Computation of Optimal Three-Dimensional Aircraft Trajectories including Terrain-Following," in *AIAA Guidance, Navigation, and Control Conference*, Reston, VA, 2006.
- [44] Y. Li and X. Chen, "Mobile Robot Navigation Using Particle Swarm Optimization and Adaptive NN," *Advances in Natural Computation*, vol. 3612, p. 628–631, 2005.
- [45] G. F. List, P. B. Mirchandani, M. Turnquist and K. G. Zografos, "Modeling and Analysis for Hazardous Materials Transportation: Risk Analysis, Routing Scheduling and Facility Location," *Transportation Science*, vol. 25, p. 100–114, 1991.

- [46] P. Leonelli, S. Bonvicini and G. Spadoni, "Hazardous Materials Transportation: A Risk-Analysis-Based Routing Methodology," *Journal of Hazardous Materials*, vol. 71, p. 283–300, 2000.
- [47] M. Shanmugavel, A. Tsourdos, R. Zbikowski and B. A. White, "3D Dubins Sets Based Coordinated Path Planning for Swarm of UAVs," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, CO, Aug. 2006.
- [48] R. A. Larson, M. Pachter and M. J. Mears, "Path Planning by Unmanned Air Vehicles for Engaging an Integrated Radar Network," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, CA, Aug. 2005.
- [49] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *International Journal of Robotics Research*, vol. 5, no. 1, p. 396–404, 1986.
- [50] J. Barraquand, B. Langlois and J.-C. Latombe, "Numerical Potential Field Techniques for Robot Path Planning," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, p. 224–240, Apr. 1992.
- [51] A. D. Luca and G. Oriolo, "Local Incremental Planning for Nonholonomic Mobile Robots," in *Proceedings of IEEE International Conference on Robotics and Automation*, San Diego, CA, May 1994.
- [52] A. Bemporad, A. D. Luca and G. Oriolo, "Local Incremental Planning for a Car-Like Robot Navigating among Obstacles," in *Proceedings of IEEE International Conference on Robotics and Automation*, Minneapolis, MN, Apr. 1996.
- [53] C. E. Thorpe, "Path Relaxation: Path Planning for a Mobile Robot," in *National Conference on Artificial Intelligence*, 1984.
- [54] S. Kambhampati and L. S. Davis, "Multiresolution Path Planning for Mobile Robots," *IEEE Journal of Robotics and Automation*, vol. 2, p. 135–145, Sept. 1986.
- [55] C. L. Jackins and S. L. Tanimoto, "Oct-tree and Their Use in Representing Three Dimensional Objects," *Computer Graphics and Information Processing*, vol. 14, no. 3, p. 249–270, 1980.
- [56] R. Klein, *Concrete and Abstract Voronoi Diagrams*, Berlin, Germany:: Springer-Verlag, 1989.
- [57] R. A. Brooks, "Solving the Find-Path Problem by Good Representation of Free Space," *IEEE Transactions on System, Man, and Cybernetics*, vol. 13, no. 3, p. 190–193, 1983.
- [58] T. Lozano-Perez and M. A. Wesley, "An Algorithm for Planning Collision-Free Path Among Polyhedral Obstacles," *Communications of the ACM*, vol. 22, no. 10, p. 560–570, 1979.
- [59] M. Neus and S. Maouche, "Motion Planning using the Modified Visibility Graph," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, Tokyo, Japan, Oct. 1999.
- [60] J.-C. Latombe, *Robot Motion Planning*, Boston, MA: Kluwer Academic Publishers, 1991.

- [61] L. Kavraki and J.-C. Latombe, "Randomized Preprocessing of Configuration for Fast Path Planning," in *Proceedings of the IEEE International Conference on Robotics and Automation*, San Diego, CA, May 1994.
- [62] K. Shin and N. McKay, "A Dynamic Programming Approach to Trajectory Planning of Robotic Manipulators," *IEEE Transactions on Automatic Control*, vol. 31, no. 6, p. 491–500, 1986.
- [63] D. Rathbun, S. Kragelund, A. Pongpunwattana and B. Capozzi, "An Evolution Based Path Planning Algorithm for Autonomous Motion of a UAV Through Uncertain Environments," in *Proceedings of the 21st Digital Avionics Systems Conference*, 2002.
- [64] D. Jia and J. Vagners, "Parallel Evolutionary Algorithms for UAV Path Planning," in *AIAA 1st Intelligent Systems Technical Conference*, Chicago, IL, Sept. 2004.
- [65] T. Karata and F. Bullo, "Randomized Searches and Nonlinear Programming in Trajectory Planning," in *Proceedings of the 40th IEEE Conference on Decision and Control*, Orlando, FL, Dec. 1994.
- [66] K. Marti and S. Qu, "Path Planning for Robots by Stochastic Optimization Methods," *International Journal of Intelligent and Robotic Systems*, vol. 22, p. 117–127, June 1998.
- [67] P. E. Hart, N. J. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, p. 100–107, 1968.
- [68] E. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, vol. 1, p. 269–271, 1959.
- [69] E. M. Kan, M. H. Lim, S. P. Yeo, Z. H. Shao and J. S. Ho, "Contour Based Path Planning with B-Spline Trajectory Generation for Unmanned Aerial Vehicles (UAVs) over Hostile Terrain," *Journal of Intelligent Learning Systems and Applications*, vol. 3, no. 3, pp. 122–130, 2011.
- [70] J. Hilgert, K. Hirsch, T. Bertram and M. Hiller, "Emergency Path Planning for Autonomous Vehicles using Elastic Band Theory," *Advanced Intelligent Mechatronics*, vol. 2, p. 1390–1395, 2003.
- [71] T. Sattel and T. Brandt, "Ground Vehicle Guidance Along Collision-free Trajectories using Elastic Bands," in *American Control Conference*, 2005.
- [72] J. Hwang, R. C. Arkin and D. Kwon, "Mobile Robots at your Fingertip: Bezier Curve On-line Trajectory Generation for Supervisory Control," in *Proceedings of Intelligent Robots and Systems*, 2003.
- [73] J. Aleotti, S. Caselli and G. Maccherozzi, "Trajectory Reconstruction with NURBS Curves for Robot Programming by Demonstration," in *Proceedings of Computational Intelligence in Robotics and Automation*, 2005.
- [74] K. B. Judd and T. W. McLain, "Spline Based Path Planning for Unmanned Air Vehicles," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Montreal, Canada, Aug. 2001.

- [75] E. P. Anderson, R. W. Beard and T. W. McLain, "Real-Time Dynamic Trajectory Smoothing for Unmanned Air Vehicles," *IEEE Transactions on Control Systems Technology*, vol. 13, p. 471–477, May 2005.
- [76] A. Pirzadeh and W. Snyder, "A Unified Solution to Coverage and Search in Explored and Unexplored Terrains using Indirect Control," in *IEEE International Conference on Robotics and Automation*, May 1990.
- [77] R. E. Korf, "Real-Time Heuristic Search," *Artificial Intelligence*, vol. 42, p. 189–211, 1990.
- [78] V. Lumelsky and A. Stepanov, "Dynamic Path Planning for a Mobile Automaton with Limited Information on the Environment," *IEEE Transactions on Automatic Control*, vol. 31, p. 1058–1063, Nov. 1986.
- [79] A. U. Raghunathan, V. Gopal, D. Subramanian, L. T. Biegler and T. Samad, "Dynamic Optimization Strategies for Three-Dimensional Conflict Resolution of Multiple Aircraft," *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 4, p. 586–594, 2004.
- [80] B. R. Geiger, J. F. Horn, A. M. DeLullo, L. N. Long and A. F. Niessner, "Optimal Path Planning of UAVs Using Direct Collocation with Nonlinear Programming," in *AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO*, Aug. 2006..
- [81] A. Richards and J. How, "Aircraft Trajectory Planning with Collision Avoidance using Mixed Integer Linear Programming," in *Proceedings of the American Control Conference*, Anchorage, AK, 2002.
- [82] A. Richards, J. How, T. Schouwenaars and E. Feron, "Plume Avoidance Maneuver Planning Using Mixed Integer Linear Programming," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Montreal, Canada, Aug. 2001.
- [83] M. Morari and J. Lee, "Model Predictive Control: Past, Present, and Future," *Computers and Chemical Engineering*, vol. 23, no. 4, p. 667–682, 1999.
- [84] C. E. Garcia, D. M. Prett and M. Morari, "Model Predictive Control: Theory and Practice – A Survey," *Automatica*, vol. 25, no. 3, p. 335–348, 1989.
- [85] E. F. Camacho and C. Bordons, *Model Predictive Control*, London, UK: Springer-Verlag, 1999.
- [86] F. Allgower and A. Zheng, "Nonlinear Model Predictive Control," *Progress in Systems and Control Theory*, vol. 26, 2000.
- [87] J. Fuller, D. Seto and R. Meisner, "Optimization-Based Control for Flight Vehicles," in *AIAA Guidance, Navigation, and Control Conference*, Denver, CO,, Aug. 2000..
- [88] L. Singh and J. Fuller, "Trajectory Generation for a UAV in Urban Terrain, using Nonlinear MPC," in *Proceedings of the American Control Conference*, Arlington, VA, June 2001.
- [89] T. Lapp and L. Singh, "Model Predictive Control Based Trajectory Optimization

- for Nap-of-the-Earth (NOE) Flight Including Obstacle Avoidance," in *Proceedings of the 2004 American Control Conference*, Boston, MA, June-July 2004.
- [90] D. H. Shim, H. Chung, H. J. Kim and S. Sastry, "Autonomous Exploration in Unknown Urban Environments for Unmanned Aerial Vehicles," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, CA, Aug. 2005.
 - [91] Y. Kang and J. K. Hedrick, "Design of Nonlinear Model Predictive Controller for a Small Fixed-Wing Unmanned Aerial Vehicle," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, CO, Aug. 2006..
 - [92] J. Bellingham, A. Richards and J. P. How, "Receding Horizon Control of Autonomous Aerial Vehicles," in *Proceedings of the American Control Conference*, Anchorage, AK, May 2002.
 - [93] A. Richards, Y. Kuwata and J. How, "Experimental Demonstrations of Real-time MILP Control," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Austin, TX, Aug. 2003.
 - [94] J. Bellingham, Y. Kuwata and J. How, "Stable Receding Horizon Trajectory Control for Complex Environments," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Austin, TX, Aug. 2003..
 - [95] Y. Kuwata and J. P. How, "Stable Trajectory Design for Highly Constrained Environments using Receding Horizon Control," in *Proceedings of the 2004 American Control Conference*, Boston, MA, June-July 2004..
 - [96] T. Schouwenaars, J. How and E. Feron, "Receding Horizon Path Planning with Implicit Safety Guarantees," in *Proceedings of the 2004 American Control Conference*, Boston, MA, June-July 2004.
 - [97] T. Schouwenaars, M. Valenti, E. Feron and J. How, "Implementation and Flight Test Results of MILP-based UAV Guidance," in *IEEE Conference on Aerospace*, Mar. 2005.
 - [98] M. Valenti, T. Schouwenaars, Y. Kuwata, E. Feron and J. How, "Implementation of a Manned Vehicle - UAV Mission System," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Providence, RI, Aug. 2004.
 - [99] E. M. Kan, S. P. Yeo, Z. H. Shao and J. S. Ho, "Contour Based Path Planning for UAVs over Hostile Terrain," in *Soft Computing and Pattern Recognition*, Dec 2009.
 - [100] E. Knott, J. Shaeffer and M. Tuley, *Radar Cross Section*, 2nd ed, vol. pp. 231., Artech House, 1993.
 - [101] U. F. Knott, *Radar cross section measurements*, New York: Van Nostrand Reinhold, 1993.
 - [102] M. Pachter, D. R. Jacques and J. M. Hebert, "Minimizing radar exposure in air vehicle path planning," in *Proceedings of the 41st Israel Annual Conference on Aerospace Sciences*, Israel, Feb 2001.

- [103] E. M. Kan, M. Lim, S. Yeo, Z. Shao and J. Ho, "Radar Aware Path Planning with B-Spline Trajectory Generation for Unmanned Aerial Vehicles (UAVs)," *International Journal of Reasoning-based Intelligent Systems*, vol. 2, no. 3, p. 226 – 236, 2010.
- [104] P. E. Hart, N. J. Nilsson and B. Raphael, "A Formal Basis for theHeuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, p. 100–107, 1968.
- [105] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms 2nd ed., The MIT Press , Sept. 2001.
- [106] E. M. Kan, M. Lim, S. Yeo, Z. Shao and J. Ho, "Contour Based Path Planning with B-Spline Trajectory Generation for Unmanned Aerial Vehicles (UAVs) over Hostile Terrain," *Journal of Intelligent Learning Systems and Applications*, vol. 3, no. 3, pp. 122-130, 2011.
- [107] A. Scheuer and C. Laugier, "Planning Sub-Optimal and Continuous Curvature Paths for Car-Like Robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Victoria, B. C. Canada, Oct. 1998.
- [108] R. H. Bartels, J. C. Beatty and B. A.Barsky., An Introduction to Splines for use in Computer Graphics and Geometric Modeling, Morgan Kaufmann Publishers, 1987.
- [109] K. B. Judd and T. W. McLain, "Spline Based Path Planning for Unmanned Air Vehicles," in *AIAA Guidance, Navigation, and Control Conference and Exhibi*, Montreal, Canada, Aug. 2001.
- [110] G. B. Vazquez, A. H. Sossa and S. J. L. Diaz-de-Leon, "Auto Guided Vehicle Control Using Expanded Time B-splines," in *IEEE International Conference on Systems, Man, and Cybernetics*, San Antonio,TX, Oct. 1994.
- [111] C. De Boor, "A Practical Guide to Splines," *Applied mathematical sciences*, vol. 27, 1978.
- [112] M. Cox, "The numerical evaluation of B-splines," vol. 10, no. 134–149, 1972.
- [113] F. Yamaguchi, Curves and Surfaces in Computer Aided Geometric Design, Springer, 1988.
- [114] L. L. Schumaker, Spline Functions: Basic Theory, John Wiley and Sons, 1981.
- [115] L. Piegl and W. Tiller, The NURBS Book. Monographs in Visual Communication, Berlin Heidelberg: Springer-Verlag, 1997.
- [116] T. N. E. Greville, Theory and Applications of Spline Functions, New York: Academinc press, 1968.
- [117] D. Lutterkort and J. Peters, "Smooth Paths in a Polygonal Channel," in *Proceedings of the fifteenth Annual Symposium on Computational Geometry*, Miami Beach, FL, 1999.
- [118] A. Pirzadeh and W. Snyder, "A Unified Solution to Coverage and Search in Explored and Unexplored Terrains using Indirect Control," in *IEEE International Conference on Robotics and Automation*, May 1990.

- [119] R. E. Korf, "Real-Time Heuristic Search," *Artificial Intelligence*, vol. 42, p. 189–211, 1990.
- [120] V. Lumelsky and A. Stepanov, "Dynamic Path Planning for a Mobile Automaton with Limited Information on the Environment," *IEEE Transactions on Automatic Control*, vol. 31, p. 1058–1063, Nov. 1986.
- [121] R. Meuth, M. H. Lim, Y. S. Ong and D. C. Wunsch, "A proposition on memes and meta-memes in computing for higher-order learning," *Memetic Computing*, vol. 1, no. 2, pp. 85-100, Apr. 2009.
- [122] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*, San Mateo, CA: Morgan Kaufmann, 2001.
- [123] Y. Y. Haimes, K. Tarvainen, T. Shima and J. Thadathil, *Hierarchical Multiobjective Analysis of Large-scale Systems*, New York: Hemisphere publishing co., 1990.
- [124] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transaction on Evolutionary Computation*, vol. 6, p. 182–197, 2002.
- [125] K. Deb, *Multi-objective Optimization Using Evolutionary Algorithms*, Chichester: John Wiley and Sons, 2001.
- [126] J. Kim and S. Sukkarieh, "Autonomous Airborne Navigation in Unknown Terrain Environments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, pp. 1031-1045, 2004.
- [127] J. Kosecka and F. Li, "Vision Based Topological Markov Localization," in *23rd IEEE International Conference on Robotics and Automation*, 2004.
- [128] C. W. Yeu, M. H. Lim, G. Huang, A. Agarwal and Y. S. Ong, "A new machine learning paradigm for terrain reconstruction," *IEEE Geoscience & Remote Sensing Letters*, vol. 3, no. 3, pp. 382-386, Jul 2006.
- [129] G. -B. Huang and C. -K. Siew, "Extreme learning machine: RBF network case," in *Proc. Int. Conf. Control Autom. Robot Vis.*, Dec. 2004.
- [130] G. -B. Huang, Q. -Y. Zhu and C. -K. Siew, "Extreme Learning Machine: Theory and Applications," *Neurocomputing*, vol. 70, pp. 489-501, 2006.
- [131] G.-B. Huang, Q.-Y. Zhu and C.-K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," in *Proc. Int. Joint Conf. Neural Networks*, Jul 2004.
- [132] E. M. Kan, M. Lim, Y. Ong, A. Tan and S. Yeo, "Extreme learning machine terrain-based navigation for unmanned aerial vehicles," *Neural Computing and Applications*, vol. , pp. 1-9, 2012.

List of Publications

Journals

- [1] E. M. Kan, M.H. Lim, S.P. Yeo, Z.H. Shao and J.S. Ho, "Radar Aware Path Planning with B-Spline Trajectory Generation for Unmanned Aerial Vehicles (UAVs)," *International Journal of Reasoning-based Intelligent Systems*, vol. 2, no. 3, p. 226 – 236, 2010.
- [2] E. M. Kan, M.H. Lim, S.P. Yeo, Z.H. Shao and J.S. Ho, "Contour Based Path Planning with B-Spline Trajectory Generation for Unmanned Aerial Vehicles (UAVs) over Hostile Terrain," *Journal of Intelligent Learning Systems and Applications*, vol. 3, no. 3, pp. 122-130, 2011.
- [3] E. M. Kan, M.H. Lim, Y.S. Ong, A.H. Tan and S.P. Yeo, "Extreme learning machine terrain-based navigation for unmanned aerial vehicles," *Neural Computing and Applications*, vol. 22, pp. 469-477, 2012.

Conference Papers

- [1] E. M. Kan, M.H. Lim, S.P. Yeo, Z. Shao and J. Ho, "Radar Aware Path Planning for Unmanned Aerial Vehicles (UAVs)," in *International Symposium on Innovations in Intelligent Systems and Applications*, Trabzon, Turkey, June 2009.
- [2] E. M. Kan, S.P. Yeo, Z.H. Shao and J.S. Ho, "Stealth Path Planning for UAVs in Hostile Radar Zones," in *Proceedings of the 11th IASTED International Conference on Control and Applications*, Cambridge, UK, 2009.
- [3] E. M. Kan, S.P. Yeo, Z.H. Shao and J.S. Ho, "Contour Based Path Planning for UAVs over Hostile Terrain," in *Soft Computing and Pattern Recognition*, Dec 2009.
- [4] E. M. Kan, S.P. Yeo, Z.H. Shao and J.S. Ho, "An evolutionary algorithm for multiple waypoints planning with B-spline trajectory generation for Unmanned Aerial Vehicles (UAVs)," in *International Conference on Computational Problem-Solving*, 2010.
- [5] E. M. Kan, S.P. Yeo, Z.H. Shao and J.S. Ho, "Evolutionary approach for trajectory generation of Unmanned Aerial Vehicles (UAVs) over hostile terrain," in *International Conference on Computational Problem-Solving*, 2010.
- [6] E. M. Kan, M.H. Lim, S.P. Yeo, Z.H. Shao and J.S. Ho, "Path Planning for Unmanned Aerial Vehicle(UAV)," in *Aerospace Technology Seminar*, Singapore, February 2011.