DYNAMIC TASK ALLOCATION AND PATH PLANNING FOR MULTI-AIRCRAFT MISSIONS

ÇOK UÇAKLI GÖREVLER İÇİN DİNAMİK GÖREV TAHSİSİ VE ROTA PLANLAMASI

DOĞAN CANDEMİR

PROF. DR. SUAT ÖZDEMIR

Supervisor

Submitted to

Graduate School of Science and Engineering of Hacettepe University

as a Partial Fulfillment to the Requirements

for the Award of the Degree of Master of Science

in Computer Engineering

January 2024

ABSTRACT

DYNAMIC TASK ALLOCATION AND PATH PLANNING FOR MULTI-AIRCRAFT MISSIONS

Doğan CANDEMİR

Master of Science, Computer Engineering

Supervisor: Suat ÖZDEMİR

January 2024, 79 pages

Military aircraft are advanced vehicles capable of performing various missions, primarily in air-to-air and air-to-ground scenarios. Before takeoff, a mission plan is prepared by ground crews, and pilots are expected to follow the planned route during flight. Adherence to the mission plan is critical, especially for combat aircraft, for the mission's success and the pilots' safety and health. However, due to unknown threats and dynamic environmental changes during flight, it may only sometimes be possible to follow the pre-planned route. Especially in missions involving multiple aircraft, a new plan must be prepared for each aircraft. In such situations, the leader pilot in charge of the mission is expected to assign tasks to other pilots and update the route for each aircraft accordingly. This thesis focuses on this problem and presents a study on dynamic task assignment and route planning to be used when the pre-prepared mission plan becomes invalid.

To tackle the challenges posed by both single and multiple aircraft missions, we have developed a new approach called Narrowed Regions-based Bidirectional Rapidly Exploring Random Tree (Narrowed-BiRRT). This method involves a matrix-based target assignment process for dynamic task allocation, followed by steps for route planning for the aircraft. The

obtained routes are then optimized using a height optimization algorithm to prevent sudden altitude changes. This enhances the traceability of the routes generated by our developed method, making them more likely to be followed by real aircraft.

We tested the developed methods in scenarios, beginning with single-aircraft missions and progressively escalating threat levels in multi-aircraft situations. The algorithm showcased a 100% convergence rate in all test scenarios, highlighting its capability to generate routes in any environment where a solution was identified. Our method was tested in a real scenario involving two aircraft and three threats, producing an optimal route of up to 10 km for each aircraft in a total of 1.4 seconds according to RRT, RRT* and RRT-Connect.

As a result, the methods developed in this thesis consistently produce optimal, environmentally responsive, threat-resistant, and adaptable routes tailored to different systems. The presented methodology offers practical solutions for single and multiple aircraft missions, making it applicable in autonomous flight systems for military and potentially civilian applications.

Keywords: RRT, RRT-Connect, Path Planning, Mission Planning, Task Assignment

ÖZET

ÇOK UÇAKLI GÖREVLER İÇİN DİNAMİK GÖREV TAHSİSİ VE ROTA PLANLAMASI

Doğan CANDEMİR

Yüksek Lisans, Bilgisayar Mühendisliği

Danışman: Suat ÖZDEMİR

Ocak 2024, 79 sayfa

Askeri uçaklar hava-hava ve hava-yer başta olmak üzere geniş yelpazede birçok görevi icra edebilen gelişmiş uçaklardır. Havalanmadan önce yer ekipleri tarafından bir görev planı hazırlanır ve pilotlardan uçuş sırasında bu planda yer alan rotayı takip etmeleri beklenir. Görev planına bağlılık, özellikle savaş uçakları için, görevin başarısı ve pilotların güvenliği ve sağlığı açısından kritiktir. Ancak, uçuş sırasında bilinmeyen tehditler ve dinamik çevresel değişiklikler nedeniyle daha önce planlanmış rotayı takip etmek her zaman mümkün olmayabilir. Özellikle birden fazla uçağın yer alacağı görevlerde her bir uçak için yeni plan hazırlamak gerekir. Böyle bir durumda görevi komuta eden lider pilotun diğer pilotlara görev ataması ve buna göre her uçağın rotasının güncellenmesi beklenir. Bu tez kapsamında bu probleme odaklanarak daha önce hazırlanmış görev planının geçersiz olduğu durumlarda kullanılmak üzere dinamik görev atama ve rota planlama çalışması yapılmıştır

Hem tek uçaklı hem de çok uçaklı görevlerin zorluklarıyla başa çıkmak için Daraltılmış Bölge Tabanlı Çift Yönlü Hızlıca Keşfeden Rastgele Ağaç adlı yeni bir yaklaşım geliştirilmiştir. Bu yöntem, dinamik görev tahsisi için matris tabanlı bir hedef atama işlemi yapıldıktan sonra uçaklar için rota planlamanın adımlarını içermektedir. Elde edilen

rotalar daha sonra yükseklik optimizasyon algoritması ile optimize edilerek ani yükseklik değişiklikleri engellenmiştir. Bu da geliştirdiğimiz yöntem ile elde edilen rotaların gerçek uçaklar tarafından takip edilebilirliğini arttırmıştır.

Geliştirilen yöntemler, tek uçaklı görevlerle başlayarak tehdit seviyeleri artacak şekilde çok uçaklı senaryolarda test edilmiştir. Algoritma, tüm test senaryolarında %100 yakınsama oranı göstererek çözüm bulunan her ortamda rota üretebildiğini göstermiştir. İki uçaklı ve üç tehditli gerçek bir senaryoda yöntemimiz test edilmiş ve her iki uçak için de toplam 1.4 saniyede uzunluğu 10 km'yi bulan RRT, RRT* ve RRT-Connect algoritmalarına göre daha optimal bir rota ürettiği görülmüştür.

Sonuç olarak, bu tezde geliştirilen yöntemler, tutarlı bir şekilde optimal, çevresel etkenlere karşı duyarlı, tehditlerden etkilenmeyecek ve farklı sistemlere uyarlanabilir rotalar üretmektedir. Sunulan metodoloji, tek ve çoklu uçaklı görevlerde pratik çözümler sunarak askeri ve potansiyel olarak sivil uygulamalarda otonom uçuş sistemlerde kullanılabilir.

Anahtar Kelimeler: RRT, RRT-Connect, Rota Planlama, Görev Planlama, Görev Atama

ACKNOWLEDGEMENTS

I want to express my gratitude to my advisor Suat Özdemir, whose contributions were instrumental in completing this study. His support during my undergraduate and graduate education has been immensely helpful in bringing me to this point. I also thank İbrahim Kök for his academic support in this work and his consistent willingness to assist.

I extend my special thanks to Turkish Aerospace, a source of inspiration for me in this study, and my esteemed colleagues. Writing this thesis was much easier with their support, guidance, and assistance during challenging times. I also want to thank my friends during my undergraduate years. Their friendship and contributions were instrumental throughout this period.

I reserve my most profound appreciation for my family. I cannot forget the unwavering support shown by my mother, Perizade Candemir, and my father, Yılmaz Candemir, who have brought me to where I am today. Their teachings have always been a guiding light in both my social and academic life. If I could write these words now, their support would be the most significant contributor, without which none would have been possible. My siblings, Canan, Emrecan, and Yusuf, merit a heartfelt acknowledgment for their unwavering support and trust, a source of strength throughout my journey.

Finally, I want to express my sincere love to Selcen, who unconditionally entered my life and accompanied me through this journey. Her warm and gentle feelings have calmed me during the most tense moments. I am grateful to her for reminding me of what I can achieve and for her support during challenging times.

CONTENTS

Page

TABLES

FIGURES

ABBREVIATIONS

2D	:	**T**wo **D**imensional
3D	:	**T**hree **D**imensional
CPU	:	**C**entral **P**rocessing **U**nit
VUHF	:	**V**ery **U**ltra **H**igh **F**requency
RRT	:	**R**apidly **E**xploring **R**andom **T**ree
Narrowed-BiRRT	:	**N**arrowed **R**egions-Based **B**idirectional **RRT**
AC-I	:	**A**ircraft-I
AC-II	:	**A**ircraft-II
AC-III	:	**A**ircraft-III
A*	:	**A S**tar
PRM	:	**P**robabilistic **R**oadmaps
RRT*	:	**R**apidly **E**xploring **R**andom **T**ree **S**tar
EST	:	**E**xpansive **S**pace **T**rees
SRT	:	**S**ampling **B**ased **R**oadmap **O**f **T**rees
N/A	:	**N**ot **A**vailable
RTCA	:	**R**adio **T**echnical **C**ommision **F**or **A**eronautics
DO-178B	:	**D**coument **1**78B
DO-178C	:	**D**coument **1**78C

1. INTRODUCTION

Fighter aircraft, often considered the epitome of military aviation, have been instrumental in shaping the dynamics of modern warfare [1]. These highly sophisticated machines are purposefully designed and equipped to execute various missions, including air-to-air combat, air-to-ground strikes, and reconnaissance [2]. The standard operating procedure involves meticulous pre-mission planning, formulating a comprehensive strategy, and a specific flight path is determined based on mission objectives and operational requirements [2].

However, modern warfare's fluid and dynamic nature presents formidable challenges to the precise execution of pre-planned missions. Combat environments are characterized by inherent uncertainty, rapidly changing conditions, and unpredictable enemy actions, necessitating real-time adjustments to the original mission plan [3]. In this context, making decisions on the fly and exhibiting flexibility is paramount. Commanders and pilots must possess the agility to make in-the-moment adjustments in the field, altering targets, reallocating resources, and adapting flight paths in response to evolving circumstances.

The limited availability of timely and accurate information further complicates the decision-making process, as commanders must contend with processing and analyzing incoming data, assessing situational developments, and considering factors such as enemy movements, weather conditions, and resource availability [4]. Simultaneously, pilots are tasked with interpreting and implementing real-time instructions while ensuring the safe execution of operational requirements. The need for swift decision-making and adaptability places tremendous pressure on commanders and pilots alike, requiring a delicate balance between achieving mission objectives and managing operational constraints and risk [5].

In response to these challenges, there is a compelling need for advanced decision support systems that facilitate dynamic mission planning in modern fighter aircraft. These systems should be able to generate new flight paths on the fly if deviations from the initial plan are necessary during the mission. Such adaptability ensures that missions can be executed

effectively, even in the face of unforeseen circumstances, thereby enhancing the overall effectiveness of fighter aircraft in contemporary combat scenarios.

1.1. Scope Of The Thesis

The primary objective of this thesis is to address the challenges associated with dynamic task assignment and route planning in multi-aircraft military missions. Multi-aircraft missions involve complex tasks that demand high coordination among aircraft. Despite preparing a mission plan before takeoff, unforeseen actions by hostile elements and the dynamic nature of the combat environment necessitate the redefinition of missions and the determination of new routes for the aircraft. This vibrant and unpredictable context underscores the importance of developing effective strategies to adaptively allocate tasks and plan optimal flight routes during multi-aircraft military operations.

In this thesis, we developed the Narrowed Regions-based Bidirectional Rapidly Exploring Random Tree (Narrowed-BiRRT) algorithm, a method applicable to single and multi-aircraft missions in 3D dynamic and changing environments. The algorithm takes parameters such as the current positions of aircraft, target points, and identified threats in the environment, along with the instantaneous velocities of the aircraft, which are used to determine the narrowed regions.

We developed a target assignment method for multi-aircraft missions requiring task distribution among the aircraft. Calculating distances between aircraft and targets and creating a matrix, each aircraft is assigned to its nearest target. This assignment occurs at the beginning of each sampling iteration during the route planning process, allowing our method to facilitate real-time task sharing among aircraft in response to environmental changes.

Following generating a route for each aircraft with Narrowed-BiRRT, we apply altitude optimization to the waypoints constituting the route. This optimization aims to ensure a smoother increase in altitude for aircraft following the route, as abrupt changes in altitude can increase G-forces, posing a risk to the pilot's health. Therefore, achieving a smoother transition in altitude is crucial for the traceability of the route.

1.2. Contributions

- The proposed approach restricts sampling to relevant regions, regardless of the problem domain's size. This targeted sampling increases the likelihood of calculating the shortest path between the starting and ending points, optimizing the search process and offering computational advantages. Thus, it presents a promising method for path planning in diverse applications.

- This study leads to a shorter path planning time compared to other RRT-based algorithms. By excluding nodes from irrelevant regions within the generated tree, the total number of nodes is reduced. As a result, fewer searches are needed to derive the route.

- This study enables dynamic target assignment, eliminating the need for re-planning when missions must be reassigned among aircraft. This capability reduces the overall route planning time, offering increased flexibility in adapting to changing mission requirements.

- This study pioneers the exploration of multi-aircraft missions centered explicitly around combat aircraft. This work breaks new ground in the field, unlike previous research, which did not delve into generating multiple routes using the RRT algorithm with multiple aircraft or robots. Addressing the unique challenges and complexities inherent in multi-aircraft missions provides valuable insights that pave the way for future research in autonomous path planning for military applications.

- We extensively tested the recommended method across numerous scenarios, and the results underscore the practical applicability of Narrowed-BiRRT in real-world scenarios. The outcomes affirm that the proposed approach provides viable solutions for practical implementation, validating its efficacy and relevance in addressing complex challenges in various scenarios. This empirical validation contributes to the credibility and utility of the Narrowed-BiRRT algorithm, emphasizing its potential for effective real-world deployment.

1.3. Organization

The organization of the thesis is as follows:

- **Chapter 1** presents our motivation, contributions and the scope of the thesis.
- **Chapter 2** provides insights into mission planning, detailing the methods employed in single and multi-aircraft scenarios. Distinctions between single and multi-aircraft missions are highlighted, and the feasibility of route planning for aircraft is discussed. The section concludes with an overview of commonly used path-planning methods.
- **Chapter 3** presents a comprehensive literature review of existing route planning studies. The studies are mainly consists of sampling-based approaches.
- **Chapter 4** delves into the developed methodologies, outlining the step-by-step process for creating a route from scratch. This section include Environment Creation, Task Assignment, Narrowed Regions Strategy, Path Creation, and Altitude Optimization.
- **Chapter 5** focuses on the experimental evaluation of the proposed methodology. Initial assessments are conducted for single-aircraft missions, followed by an in-depth analysis of multi-aircraft missions. We evaluated the study in completeness, efficiency, smoothness, optimality, robustness, and adaptability.
- **Chapter 6** discusses the outcomes of the study. Achievements are highlighted, and the section addresses the study's limitations and potential areas for improvement.

2. BACKGROUND OVERVIEW

Route planning has garnered significant attention as a research focus for various vehicles. Although the field of robotics boasts a plethora of studies on route planning, research specific to military aircraft remains relatively sparse. Military aircraft, being vehicles with unique mission profiles distinct from other counterparts, require a comprehensive understanding of their roles. So, this section initiates by delineating the characteristics of single and multi-aircraft missions, thus providing a more precise problem definition. Subsequently, we discuss the applicability of our developed method, considering the distinctive features of military aircraft missions. Finally, we review existing methods in the literature, elucidating the rationale behind the necessity for our proposed approach.

2.1. Mission Planning

Mission planning [2, 6] for aircraft involves systematically determining the optimal course of action to achieve specific objectives within a defined operational context. This intricate task encompasses a range of considerations, such as defining mission objectives, assessing environmental factors, determining routes, allocating resources, and addressing potential threats. The planning process integrates various parameters, including aircraft capabilities, fuel efficiency, payload capacity, and mission constraints. In a military context, mission planning for aircraft often involves:

- The coordination of multiple assets to accomplish complex objectives.
- Considering factors like airspace restrictions.
- Threat assessments.
- Strategic goals.

Precision and efficiency are paramount in mission planning, as they directly impact the success of operations and the safety of the aircraft and its crew. Advanced technologies,

such as route optimization algorithms and real-time data analysis, are crucial in enhancing the effectiveness and adaptability of mission planning for modern aircraft.

Missions in military operations can be divided into single-fighter and multi-fighter missions. Despite their differences, these two approaches are essential for military strategies and tactics.

2.2. Single-Aircraft Missions

In single-fighter missions, a single aircraft is used to engage enemy targets, while in multi-fighter operations, several aircraft are coordinated to achieve a common objective. Single-fighter missions focus on individual pilot skills, situational awareness, and adaptability. The individual aircraft operates independently, using its speed, maneuverability, and armament to engage and neutralize enemy threats. Figure 2.1 depicts a single-aircraft mission scenario. The terrain includes air defense systems and a tank. As part of its mission, the aircraft must destroy the tank. The hemispherical areas covering air defense systems also show their coverage area. If aircraft flies through these areas, it is likely to be shot down. Therefore, the mission should be performed outside these areas. The locations the aircraft must follow to hit the tank and the path it must take between the locations are marked with dashed lines.

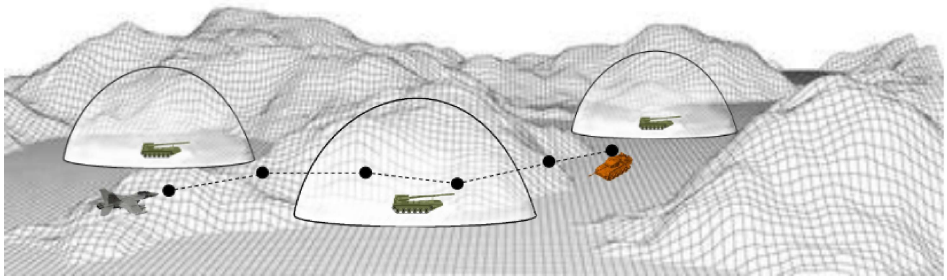


Figure 2.1 Single-aircraft mission scenario.

2.3. Multi-Aircraft Missions

Multi-aircraft missions emphasize teamwork, coordination, and synergy between multiple aircraft. These missions are designed to utilize the combined capabilities of the aircraft, maximize combat power, and improve survivability. Multi-aircraft missions may include tasks such as escorting friendly aircraft, conducting complex attack maneuvers, or engaging multiple enemy targets simultaneously. Figure 2.2 depicts a multi-aircraft mission scenario. Unlike Figure 2.1, this scenario has two aircraft and two tanks. Each aircraft moves forward by pursuing the determined locations towards the target assigned to it. The paths that aircraft must follow are determined according to the threats and each other.

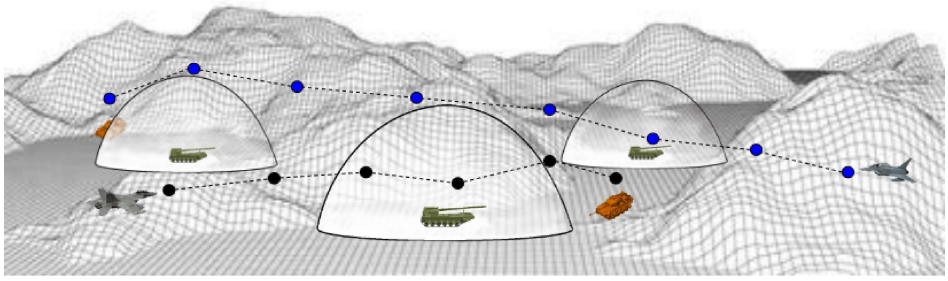


Figure 2.2 Multi-aircraft mission scenario.

In both scenarios, a real three-dimensional environment is considered. It was determined that the flight paths followed by the aircraft are determined through mission planning before aircraft takeoff. The flight paths shown in Figure 2.1 and Figure 2.2 illustrate the trajectory that the aircraft will follow during a specific time interval within the course of their missions. The trajectory itself consists of a sequence of waypoints and legs. Each individual point, shown in black and blue, illustrates the specific locations that the aircraft must fly through, representing waypoints. The dashed lines between these waypoints describe the flight path segments that the aircraft must follow, called legs. The threats are shown as hemispheres symbolising the coverage areas of the stationary air defence systems. Flying through these

regions not only endangers the mission, but above all the safety of the pilot. While certain threats are assumed to be known during mission planning, others are assumed to be detected by the aircraft's sophisticated radar and sensor systems during mission execution. It is assumed that the execution of the existing mission plan is dangerous in view of the new threats, so that a new trajectory must be derived to reduce the risk of mission execution.

2.4. Applicability of Path Planning for Aircraft

The applicability of path planning for aircraft is a critical facet of modern aviation, encompassing the strategic orchestration of trajectories to navigate through complex and dynamic environments. As aviation technology advances, the need for efficient and optimized path planning becomes increasingly pronounced. Path planning for aircraft involves systematically determining a route from an initial point to a designated goal while accounting for various constraints and environmental factors. This process is integral to optimizing flight paths, ensuring collision-free navigation, and mitigating potential risks, such as conflicts with other air traffic or hazardous terrain. Furthermore, path planning contributes significantly to enhancing fuel efficiency, minimizing operational costs, and improving overall flight safety. In both single and multi-aircraft missions, the judicious application of path planning algorithms aids in achieving mission objectives, responding to dynamic threats, and optimizing the allocation of resources. The ongoing development of path planning methodologies highlights its crucial role in shaping the trajectory of modern aviation, propelling advancements in autonomous flight, and bolstering the operational capabilities of fighter aircraft. As technology progresses, the significance of efficient and optimized path planning becomes increasingly pronounced in enhancing the strategic navigation of fighter aircraft through complex and dynamic environments. This evolution optimizes flight paths for individual fighter aircraft and contributes to the overall effectiveness of mission planning, ensuring collision-free and secure navigation in dynamic scenarios. The focus on fighter aircraft underscores the importance of path planning in maximizing operational efficiency, minimizing risks, and improving the overall effectiveness of military missions [7–10].

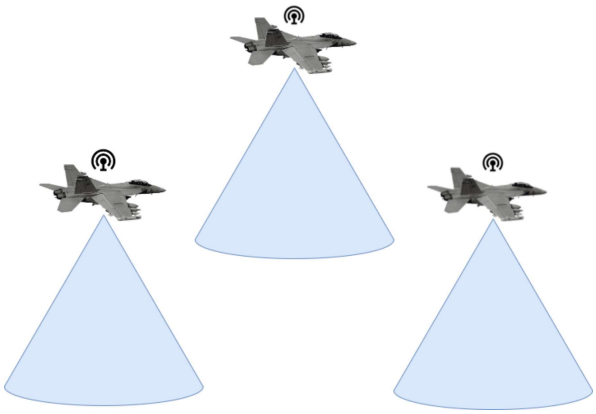


Figure 2.3 Aircraft Radar and Electro Optic Sensing System.

Achieving heightened situational awareness in modern aviation is made possible by integrating advanced radar and electro-optic systems [11, 12]. These cutting-edge sensor technologies empower aircraft to explore vast regions, comprehensively understanding the operational environment. Radar systems enable the detection of objects over extended distances, allowing for early threat identification and avoidance. Simultaneously, electro-optic systems enhance visibility by utilizing visual and infrared sensors, further expanding the spectrum of detectable elements. Furthermore, the seamless integration of these systems with sophisticated avionics facilitates real-time data processing. This processing capability enables the fusion of information from various sensors and supports executing path-planning algorithms. The combination of extensive coverage, precise detection, and swift data processing equips aircraft with unparalleled situational awareness, enhancing their adaptability and decision-making capabilities in dynamic and challenging operational scenarios.

In military aviation, the exchange of critical information among aircraft is orchestrated through a sophisticated amalgamation of communication technologies. VUHF [13] radios, serving as a fundamental component, enable secure and instantaneous voice communication, facilitating essential coordination and ensuring real-time responsiveness during missions. Furthermore, Data Link Systems [14], a pivotal advancement, empower aircraft to transfer an extensive array of digital data seamlessly. Link 16 [15], a tactical data link system widely adopted in military aviation, stands out for its ability to enhance interoperability by providing

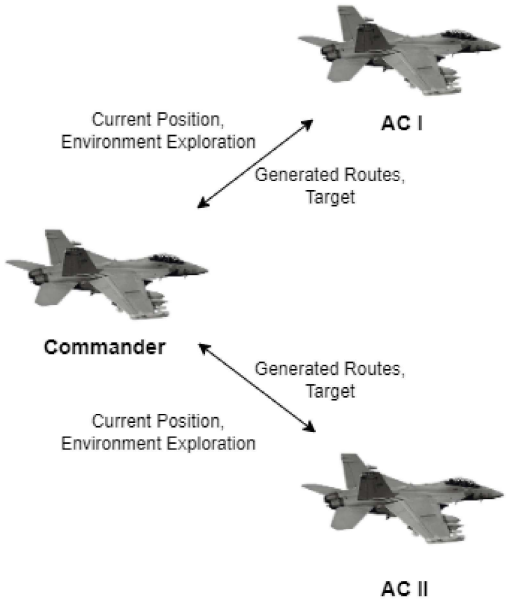


Figure 2.4 Data transfer between aircrafts.

a standardized platform for sharing crucial information among airborne platforms. This sophisticated data link system enables the real-time transmission of tactical information, including tracks, targeting data, and mission updates, fostering a networked environment among participating aircraft. The integration of Link 16 not only enhances communication efficiency within a formation but also contributes to a comprehensive and shared situational awareness, ultimately bolstering the effectiveness and success of military missions.

Utilizing the specified radar and electro-optical system capabilities, the system can detect threats in the environment and take appropriate actions. Furthermore, in multi-aircraft missions, sharing data obtained by the sensors of each aircraft enhances situational awareness. All acquired data is centralized on the commanding aircraft, processed using advanced avionic equipment, and used to formulate new mission plans. This enables the mission commander to promptly assign tasks to other pilots and transmit new flight routes, facilitating the seamless execution of the mission under challenging combat conditions.

2.5. Methods

Several path planning methods have been developed for various fields of application. grid-based and sampling-based algorithms are commonly used algorithms. Among the grid-based path planning algorithms, A*, Dijkstra, and D* Lite algorithms are widely used [16]. However, it is common to use sampling-based methods in dynamic and unknown environments. Probabilistic Roadmap (PRM) [17] and Rapidly-exploring Random Trees (RRT) [18] are the most common sampling-based algorithms.

2.5.1. Grid-Based Methods

Grid-based path planning constitutes a prominent methodology in robotics and autonomous systems, offering a structured framework for navigating complex environments. This approach discretizes the environment into a grid of cells, each representing a distinct region within the space. This division facilitates the representation of obstacles and free spaces in a binary manner, simplifying the computational complexity of path-planning algorithms. The process involves generating a grid map that characterizes the traversability of each cell, distinguishing between areas occupied by obstacles and those accessible for movement. The inherent simplicity of grid-based representations allows for integrating various path-planning algorithms, making it a versatile and widely employed technique. Notably, grid-based path planning shares an affinity with the cell decomposition method, wherein the environment is decomposed into simpler geometric shapes, such as polygons or cells. This decomposition aids in the identification of feasible paths and the navigation of intricate terrains.

$VoronoiDiagrams$: Voronoi diagrams [19] are geometric structures that partition a space into cells based on proximity to a set of seed points. In a Voronoi diagram, each cell encompasses the region closer to a specific seed point than any other set point. This partitioning creates polygons, called Voronoi cells, where each cell represents the area associated with a particular seed point. Voronoi diagrams find applications in various fields, including computational geometry and path planning. In the context of cell decomposition,

Voronoi diagrams provide a means to divide an environment into distinct regions, simplifying the representation of spatial relationships. In grid-based path planning, the Voronoi diagram can be employed to create a graph representing the connectivity between regions, aiding in efficient pathfinding. The boundaries of Voronoi cells can serve as guides for defining regions of influence and navigation pathways, contributing to the overall efficiency and effectiveness of grid-based path-planning algorithms.

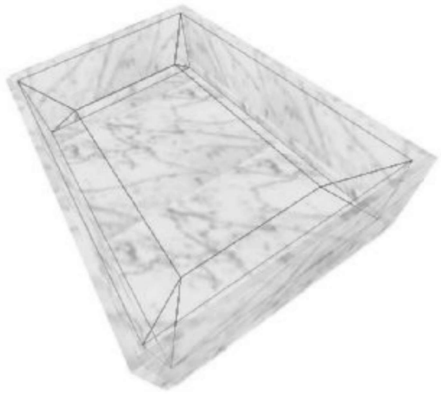


Figure 2.5 Example of Voronoi Diagram [16].

$DijkstraAlgorithm$: Dijkstra's Algorithm, a fundamental method in graph theory and pathfinding, is widely employed in network and spatial analysis. It efficiently determines the shortest paths between nodes in a graph with non-negative edge weights.

Let G be the graph, V be the set of vertices, and E be the set of edges. Each edge (u, v) has a non-negative weight $w(u, v)$. The Algorithm maintains two sets: S (the set of vertices whose shortest distance from the source is known) and Q (the set of vertices whose shortest distance is yet to be determined).

Initially, the distance from the source vertex s to itself is 0, and it is set to infinity for all other vertices. The Algorithm repeatedly selects the vertex u from Q with the minimum distance, explores its neighbors, and updates their distances if a shorter path is found. This process continues until all vertices are in S.

$A * Algorithm$: The A* [20] algorithm is a pathfinding algorithm that efficiently searches a graph to find the optimal path from a starting node to a goal node. A* incorporates a heuristic

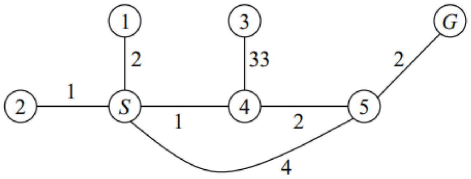


Figure 2.6 Example of Dijkstra [16].

function, which estimates the cost from the current node to the goal. The algorithm uses both the known cost from the start node and the estimated cost to the goal to prioritize nodes for exploration. Unlike uninformed search algorithms like depth-first [21] and breadth-first [22] search, A* introduces an informed approach by considering the heuristic information. The heuristic guides the search toward nodes likely to lead to the goal, based on the available local information. The algorithm dynamically evaluates nodes, considering both the cost from the start and the heuristic estimate. A* guarantees optimality because the heuristic is admissible, meaning it never overestimates the actual cost to reach the goal. An admissible heuristic ensures that A* explores nodes in a manner that minimizes the total estimated cost, leading to the discovery of the shortest path.

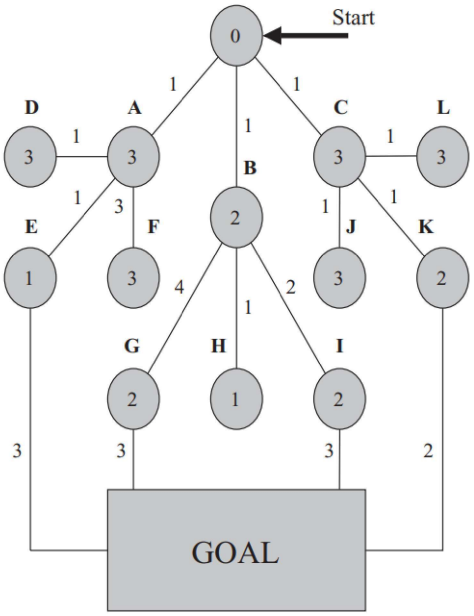


Figure 2.7 Example of A* [16].

In practical terms, A* is particularly useful for problems involving graphs where the cost of moving between nodes can vary and where an efficient and optimal path is desired. The algorithm efficiently balances exploration and exploitation, making it a versatile and widely used tool in pathfinding applications.

2.5.2. Sampling-Based Methods

Sampling-based algorithms represent a category of path-planning methods that operate on the principle of random sampling within the configuration space of a robot or vehicle. These algorithms are particularly effective for solving single-query path planning problems, where the objective is to find a collision-free path between a start and goal configuration. The core idea involves generating a set of random samples in the configuration space, checking their feasibility, and connecting them to construct a roadmap or tree structure that represents possible paths. Several well-known sampling-based algorithms include Probabilistic Roadmaps (PRM) [23], Expansive Space Trees (EST) [24, 25], Rapidly Exploring Random Trees (RRT) [26], RRT* [27], RRT-Connect [28] and Sampling-Based Roadmap of Trees (SRT) [29].

$Probabilistic Roadmaps (PRM)$: PRM, introduced by Kavraki [23], has proven to be a pioneering approach in sampling-based methods for robotic motion planning. PRM takes advantage of the cost-effectiveness of checking if a robot configuration is in a collision-free space, focusing on arrangements within this free space. The algorithm employs coarse sampling to generate roadmap nodes and fine sampling to establish edges representing collision-free paths between these nodes. Subsequently, planning queries are addressed by connecting user-defined initial and goal configurations to the roadmap, utilizing it to solve path-planning problems efficiently. Initially relying on uniform random distribution for node sampling, known as basic PRM, subsequent studies revealed the versatility of other sampling schemes, including importance sampling and deterministic methods like quasirandom sampling and grid-based sampling. PRM was initially designed as a multiple-query planner, but modifications enable its use for single queries, where the roadmap construction is done

incrementally and stopped upon answering the query. While PRM ensures probabilistic completeness and demonstrates effectiveness across diverse problems, it may not always be the fastest option for single-query scenarios.

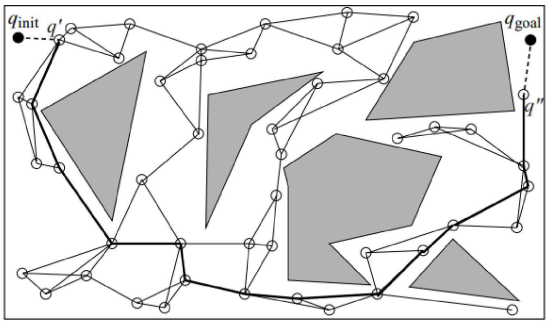


Figure 2.8 An example of how to solve a query with PRM [16].

$ExpansiveSpaceTrees(EST)$: EST algorithm is a sampling-based method employed in motion planning for robotic systems. Introduced as an alternative to the PRM, EST focuses on generating a tree structure within the configuration space, specifically in regions deemed more likely to contribute to feasible paths. Unlike PRM, which relies on a roadmap of discrete configurations, EST constructs a tree incrementally by selecting nodes based on their expected contribution to free space exploration. EST originated as an efficient single-query planner designed to quickly traverse the space between the initial configuration q_{init} and the goal configuration q_{goal}. The developers initially did not use the acronym EST, adopting it later, inspired by the concept of "expansive" space from the algorithm's theoretical analysis. Primarily tailored for kinodynamic problems, EST is adept at constructing a single tree for such problems. Recent planners have also leveraged or incorporated EST-based principles. The algorithm has been demonstrated to be probabilistically complete. The construction of trees involves selecting a configuration q from the tree, sampling a random configuration q_{rand} in its neighborhood, and attempting a connection using a local planner. If successful, q_{rand} becomes a vertex in the tree, and the connection (q, q_{rand}) is added as an edge. This process repeats until a predetermined number of configurations are added to the tree, ensuring an effective exploration of the configuration space.

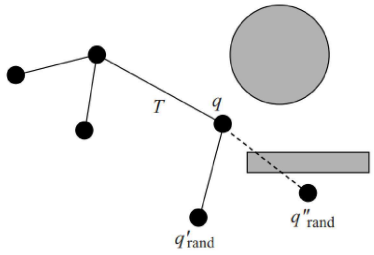


Figure 2.9 Adding a new configuration to an EST [16].

$RapidlyExploringRandomTrees(RRT)$: RRT algorithm is a powerful and widely-used motion planning technique suitable for high-dimensional configuration spaces. Initially proposed as a single-query algorithm, RRT incrementally grows a tree rooted at the initial configuration by repeatedly extending its branches towards randomly sampled configurations. The algorithm's key strength lies in its ability to rapidly explore vast configuration spaces, efficiently finding feasible paths in complex environments.

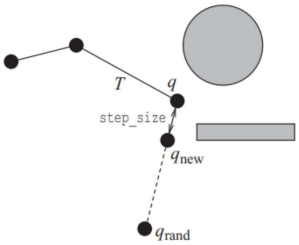


Figure 2.10 Adding a new configuration to an RRT [16].

RRT* extends the capabilities of RRT by incorporating a cost-to-come function during the tree expansion, enhancing the optimality of the generated paths. This modification enables RRT* to produce solutions that are feasible and near-optimal with respect to a given cost metric. The addition of cost-awareness makes RRT* well-suited for scenarios where optimizing the path based on specific criteria, such as minimizing travel time or energy consumption, is crucial.

RRT-Connect is a variant of RRT designed for multi-query scenarios and is particularly effective in solving problems requiring the connection of two configurations in a high-dimensional space. By growing two trees simultaneously—one from the initial

configuration and the other from the goal—RRT-Connect efficiently explores the configuration space and rapidly connects the two trees when a feasible path is found. This characteristic makes RRT-Connect advantageous for scenarios involving frequent start and goal configuration changes, demonstrating its versatility in dynamic environments.

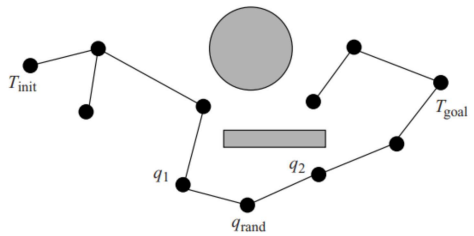


Figure 2.11 Merging two RRTs [16].

Dynamic step size selection in the RRT algorithm involves adjusting the step size based on the distance between the nearest node, q_{near}, and the randomly sampled configuration, q_{rand}. This adaptive strategy, driven by the chosen distance function, optimizes exploration efficiency by employing larger step sizes for distant, collision-free configurations and smaller ones for closer configurations. The sensitivity of RRT to the distance function underscores its crucial role in guiding both the step size and sampling process, requiring a careful balance between exploration depth and the number of added nodes, particularly in high-dimensional scenarios.

$Sampling - BasedRoadmapofTrees(SRT)$: The SRT planner is a strategic integration of sampling-based methods tailored for both multiple-query planning, exemplified by PRM, and single-query planning, represented by tree planners like EST and RRT. SRT optimally combines these techniques, leveraging the local sampling schemes of tree planners to construct a PRM-like roadmap efficiently. SRT can be viewed as a hybrid planner capable of answering multiple queries by utilizing the preconstructed roadmap. It also functions as a single-query planner for challenging problems where its construction and query-solving costs surpass those of dedicated single-query planners. The roadmap construction involves sampling tree roots uniformly in the free configuration space (Q_{free}) and growing the trees with a sampling-based tree planner. Subsequently, edges between trees are added based on local path connections and random tree neighbors, forming an undirected graph that captures

the connectivity of Q_{free}. SRT's versatility is underscored by its adaptability to various sampling-based tree planners beyond RRT and EST, offering flexibility in achieving efficient roadmap construction and query-solving.

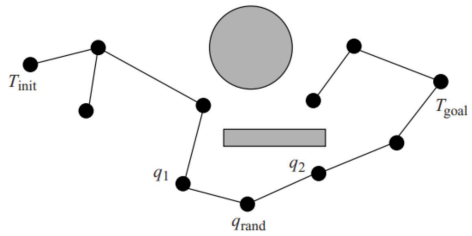


Figure 2.12 An example of a roadmap for a point robot in a two-dimensional workspace [16].

Grid-based methods can provide fast results in a known environment. If the positions to be included in the flight path are known, heuristic algorithms such as A* and Dijkstra can be applied. Alternatively, a graph can be constructed using the Voronoi diagram, and path planning can be performed using heuristic algorithms. However, in a dynamic environment, pre-determined waypoints may end up in threat zones. Moreover, when a new graph is generated using the Voronoi diagram, positions the aircraft cannot pass through may be produced. Sampling-Based methods generate waypoints in configurations where aircraft can fly through. However, as the problem domain grows, the area to be sampled also increases. Sampling in regions where the aircraft is unlikely to pass through can increase the path planning time. Therefore, there is a need to propose an improved method considering the size of the problem domain.

3. RELATED WORK

Sampling-based techniques, notably including PRMs, RRTs, and RRT-Connect, have garnered significant attention in path planning due to their adaptability and efficacy in navigating intricate configuration spaces. PRMs intelligently sample the space to construct a roadmap, facilitating efficient path identification, while RRTs rapidly explore the domain to create feasible trajectories. RRT-Connect further refines this approach by establishing connections between trees, enhancing path optimization. So, following related works is systematically classified into three principal categories, delineated by their focal exploration of PRM, RRT, and RRT-Connect algorithms.

3.1. PRM Studies

Following PRM-based studies, showcasing innovations in sampling, collision checking, and optimization for enhanced path planning outcomes. Amato et al. [30] undertook a comprehensive analysis, focusing on PRM-based methods, to evaluate diverse distance metrics and local planners within cluttered three-dimensional workspaces. They introduced a novel local planning strategy, termed ”rotate-at-s,” which surpasses the conventional straight-line approach in configuration space, particularly excelling in crowded settings. Shifting attention to sampling strategies, Boor et al. [31] introduced the Gaussian sampler for PRMs, significantly enhancing coverage within intricate sections of the free configuration space. Despite its effectiveness, this study did not delve into the specifics of the Gaussian strategy's implementation. Lazy PRM [32], emerged as a solution to reduce collision checks during planning, expediting execution. This strategy initiates with an assumption of collision-free nodes and edges, navigating the pre-existing roadmap for a shortest path. Nodes and edges along the path undergo collision assessment, leading to pruning if obstacles are encountered, ultimately iterating until a collision-free path is identified. In a pursuit to optimize path planning in high-dimensional spaces, Branicky et al. [33] advocated for the use of quasi-random sampling techniques, introducing both classical PRM with quasi-random sampling and quasi-random Lazy-PRM variants. These techniques offer a

more uniform distribution of points compared to traditional random sampling, aiding path planning in complex spaces. The comparison study conducted by [34] delved into various aspects of PRMs, encompassing collision checking, sampling techniques, and node addition methodologies. However, the dimensionality of the environment under scrutiny remains undisclosed. Moving towards deterministic approaches, LaValle's et al. investigation [35] highlighted deterministic PRM variants' superior performance compared to traditional PRMs, and identified certain grid search variants as comparable alternatives. Real-time UAV path planning in intricate 3D environments was proposed by [36], incorporating modified PRMs and free voxel utilization for roadmap node distribution, coupled with an A* algorithm to ensure feasible paths. Narrow passages gained attention in the work by Cao et al. [37], where an enhanced PRM strategy was introduced. This method strategically sampled dense obstacle areas, yielding improved adaptability in narrow passages. Lastly, Xu et al. [38] introduced a dynamic exploration planner (DEP) leveraging incremental sampling and PRM. This planner, guided by the Euclidean Signed Distance Function (ESDF) map, facilitates safe exploration in unknown environments by generating alternative paths to circumvent dynamic obstacles.

3.2. RRT Studies

The subsequent studies showcased herein are firmly grounded in the RRT framework, collectively contributing to the advancement and diversification of this paradigm within the realm of motion planning. Karaman et al. [39] introduced an anytime algorithm based on RRT* that bridges initial feasibility and optimality through committed trajectories and branch-and-bound tree adaptation for real-time implementation. Gammell's et al. [40] Informed RRT* enhances convergence rates and solution quality by sampling within a hyperspheroid region, demonstrating reduced state dimension dependence. Moreover, Salzman's et al. [41] LBT-RRT achieves near-optimal paths through an approximation factor, mitigating computational costs of the local planner. Noreen's et al. [42] survey emphasizes the need for robust and automated heuristics parameters, especially in sampling strategy-based solutions. Xinyu's et al. [43] Potential Function-based RRT*-connect

leverages bidirectional artificial potential fields to efficiently address narrow channels. Meanwhile, Wang's et al. [44] Kinematic Constrained Bi-directional RRT* integrates kinematic constraints and efficient branch pruning for rapid path discovery for differential drive mobile robots. Lastly, Liao's et al. [45] F-RRT* algorithm expedites convergence, though contingent on the triangular inequality. These studies collectively underscore the evolving landscape of RRT-based techniques, catalyzing advancements in optimal path planning and real-time implementations.

3.3. RRT-Connect Studies

The RRT-Connect algorithm and its various derivatives have been widely studied and improved in the context of path planning for autonomous systems, with applications ranging from UAVs to mobile robots. Several novel approaches have been proposed, aiming to enhance planning efficiency, reduce flight costs, and improve convergence towards optimal solutions. In this regard, Zhang et al. [46] introduced an innovative algorithm that combines the artificial potential field method with RRT-Connect, effectively guiding the random tree towards the goal for UAVs. However, while RRT-Connect has primarily been used for mobile robots, Kang et al. [47] addressed this limitation by proposing a new RRT-Connect based method that ensures quicker planning and shorter path lengths, bringing it closer to optimality. In response to this, Chen et al. [48] presented IRRT-Connect, which introduces a third node in the configuration space and utilizes a guidance method to bias the algorithm towards the target point during tree expansion. Additionally, Klemm et al. [49] introduced RRT*-Connect, which combines the advantages of RRT-Connect and RRT* to address single-query path planning problems. However, it acknowledged that further improvements can be made to the connect step in future work. Mashayekhi et al. [50] proposed the Informed RRT*-Connect, an informed version of RRT*-Connect that employs direct sampling to evaluate only the states capable of potentially providing better solutions than the current solution, applicable in both 2D and 3D environments. This method confines the search within an ellipsoidal subset, resulting in improved solutions with fewer iterations compared to the standard RRT*-Connect. For local path planning of autonomous vehicles in

dynamic obstacle avoidance scenarios, Zhang et al. [51] introduced the Bi-RRT algorithm. This approach takes into consideration the driver's driving habits and employs a target bias method to facilitate the growth of the random tree in a biased direction. This enables more effective and quicker convergence of the initial state and the goal state. Furthermore, Lau et al. [52] developed Smooth RRT-Connect, which expands along a curve while adhering to velocity and acceleration limits, instead of using straight-line trajectories for a two-dimensional environment. The proposed algorithm demonstrates superior performance in terms of computation time and algorithm reliability when compared to standard RRT solutions.

Many existing studies primarily focus on sampling the entire configuration space within a 2D environment. However, a notable limitation among these studies is the omission of the vehicle's speed, a crucial factor in real-time path planning scenarios. Also, many studies consider all configuration space to create a feasible path. Moreover, the specific context of path planning for fighter aircraft remains a significant research gap, and no one considers target assignment for multi-vehicle path planning. To address these issues, this study proposes a novel approach that shifts the emphasis from an exhaustive exploration of the entire configuration space to a more tailored strategy. By incorporating vehicle-specific characteristics and constraining the sampling process to a smaller feasible area that accommodates the aircraft's maneuverability, this approach aims to enhance the effectiveness of path planning for fighter aircraft.

4. PROPOSED METHOD

In this section, the proposed methodology developed within the scope of the thesis will be elucidated. This study begins by elucidating how to construct the problem environment and specifying its constituent elements. Next, it will expound on a method used to assign tasks among multiple aircraft in the context of multi-aircraft missions. After completing the task assignments, it will explain the sampling strategy developed for route planning, paving the way for the subsequent step of crafting routes for each aircraft. Finally, it will present an optimization method to facilitate aircraft movements, concluding with the comprehensive methodological framework proposed in this study.

4.1. Environment Creation

In route planning for aircraft, given a 3D environment E, we are concerned with finding a collision-free path combined with waypoints and legs. Some notations definitions used in our solution is given in Table 4.1.

Firstly, creating a rectangular environment is necessary. We express the minimum and maximum values that the rectangle can take on the x and y axes as follows:

$$x_{min} = \left(\frac{T_{init}(x) + T_{goal}(x)}{2} - C \right) \tag{1}$$

$$x_{max} = \left(\frac{T_{init}(x) + T_{goal}(x)}{2} + C \right) \tag{2}$$

$$y_{min} = \left(\frac{T_{init}(y) + T_{goal}(y)}{2} - C \right) \tag{3}$$

$$y_{max} = \left(\frac{T_{init}(y) + T_{goal}(y)}{2} + C \right) \tag{4}$$

After specifying the values that each corner point can take on the x and y axes, we calculate the minimum and maximum values to be taken along the z-axis as follows:

Table 4.1 Notations and Descriptions

Notation	**Description**
E	E defines rectangular prism environment that includes all objects. E_1, E_2, E_3, and E_4 denotes the corner points of E
$E_N(x, y, z)$	Shows the corners of the environment E with following labels $E_1(x_{min}, y_{max}, Z_{top})$ $E_2(x_{max}, y_{max}, Z_{top})$ $E_3(x_{max}, y_{top}, Z_{top})$ $E_4(x_{min}, y_{min}, Z_{top})$ $E_5(x_{min}, y_{max}, Z_{bottom})$ $E_6(x_{max}, y_{max}, Z_{bottom})$ $E_7(x_{max}, y_{top}, Z_{bottom})$ $E_8(x_{min}, y_{min}, Z_{bottom})$
$T_{init}(x, y, z)$	T_{init} denotes current location of aircraft in coordinate system with X, Y, and Z axis
$T_{goal}(x, y, z)$	T_{goal} denotes destination point in coordinate system with X, Y, and Z axis
C	$\|T_{init} - T_{goal}\|$, euclidean distance between T_{init} and T_{goal}
H	$\|T_{init}(z) - T_{goal}(z)\|$, altitude difference between T_{init} and T_{goal}. If difference equal to 0, a custom value can be taken as H
Z_{bottom}	Minimum altitude value of E. It must be 0 to take ground as reference.
Z_{top}	Maximum altitude value of E
T_i	T_i denotes the i. threat in E
$W_i(x, y, z)$	W_i denotes the i. waypoint on X, Y, and Z axis

$$Z_{\text{bottom}} = 0 \tag{5}$$

$$Z_{\text{top}} = \frac{T_{init}(z) + T_{goal}(z)}{2} + H \tag{6}$$

We express the rectangular prism environment E as follows, with variables x, y, and z in the coordinate planes.

$$
\begin{aligned}
x &\in [x_{\text{min}}, x_{\text{max}}] \\
y &\in [y_{\text{min}}, y_{\text{max}}] \\
z &\in [Z_{\text{bottom}}, Z_{\text{top}}]
\end{aligned}
\tag{7}
$$

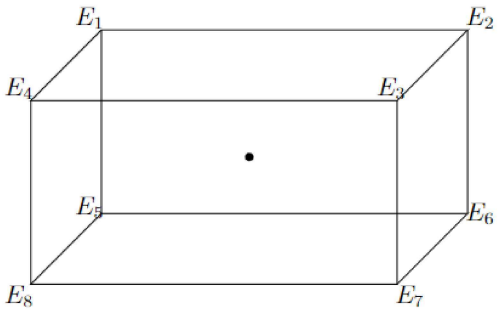


Figure 4.1 Environment E

Once we define the environment, characterizing the threats within it becomes essential. We delineate these threats along the X, Y, and Z axes, describing them using the concept of coverage areas. Each threat is associated with a coverage area denoted by R and can be mathematically expressed as follows:

$$
\text{Threat}_i = (X_i, Y_i, Z_i, R_i) \tag{8}
$$

Here, X_i, Y_i and Z_i represent the spatial coordinates of the threat in E, and R_i signifies its coverage area, extending in X, Y and Z directions. If we conceptualize threat regions as hemispheres, we can mathematically articulate a threat region within the environment. Consider a threat region T_i represented by its center (X_i, Y_i, Z_i), where X_i and Y_i denote the spatial coordinates, and Z_i signifies the altitude along the vertical axis. The radius of the hemisphere is denoted by R_i, reflecting the extent of the threat's coverage. Consequently, we can mathematically express the threat region T_i as:

$$
T_i : (x - X_i)^2 + (y - Y_i)^2 + (z - Z_i)^2 \leq R_i^2 \tag{9}
$$

In this equation, (x, y, z) represents any point within the environment. The inequality condition enforces that all points (x, y, z) falling within a distance R_i from the center (X_i, Y_i, Z_i) encompass all points within the threat region T_i.

When considering E as a real environment, we denote the ground as the zero point on the Z-axis. Since each threat is placed on the ground, their Z-axis values are zero. Consequently, we can express the volumetric area V covered by a threat within E using the hemisphere formula.

$$V_i = \frac{2}{3}\pi r^3 \tag{10}$$

With the inclusion of threats, we have created a problem environment E. Figure 4.2 illustrates E resulting from mathematical calculations. Here, the blue dot represents the aircraft's current position, while the red dot marks the intended target. Transparent yellow hemispheres in E represent the threats and their coverage areas. In E, we must strategically plan a path from the blue to the red dot, avoiding threat zones marked by these yellow hemispheres.

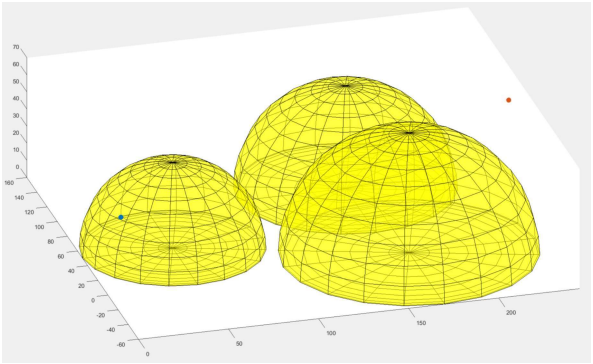


Figure 4.2 Sample Environment

4.2. Task Assignment

Assigning one or more tasks to each aircraft is imperative in multi-aircraft missions. Within the scope of this thesis, we developed and employed a method to allocate a specific task to each aircraft. Initially, we calculate the distance of each aircraft to all threats individually. We present the results of these calculations, based on three aircraft and three mission points, in Table 4.2. This approach ensures the systematic assignment of tasks to each aircraft, establishing a foundation for effective mission planning and execution in complex scenarios.

Table 4.2 Distances between Aircraft and Targets (in meters).

	Target-I	**Target-II**	**Target-III**
AC-I	30	40	50
AC-II	45	25	80
AC-III	35	40	45

In the subsequent step, we assign aircraft to each threat, commencing with the closest aircraft to the initial threat. For instance, starting with Target-I, we pair the aircraft most relative to Target-I, denoted as AC-I, with Target-I and remove it from the table. Consequently, we update Table 4.2, resulting in the derivation of Table 4.3.

Table 4.3 Distances between Aircraft and Targets after one iteration(in meters).

	Target-I	**Target-II**	**Target-III**
AC-I	N/A	N/A	N/A
AC-II	N/A	25	80
AC-III	N/A	40	45

In Table 4.3, we initiate the assignment process from Target-II. Disregarding AC-I, which has already been assigned to Target-I, we focus solely on comparing AC-II and AC-III. Proceeding in this manner, we assign Target-II to AC-II. Finally, with Target-III and AC-III remaining, we assign AC-III to Target-III. These assignments result in the acquisition of Figure 4.3.

After completing the task assignments, we conduct a sampling process and add a new node to the tree created for each aircraft. In the trees generated for each aircraft from the target

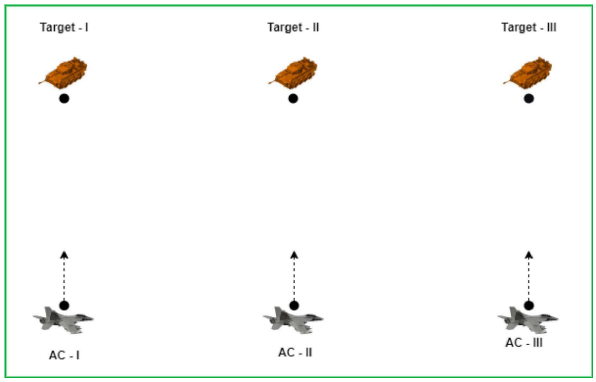


Figure 4.3 Target assignment for each aircraft.

nodes, we add new nodes by advancing toward the nearest nodes by the step size. Figure 4.4 depicts the sampling for each aircraft, with the addition of new nodes. Subsequently, we expand the target trees sequentially, starting from Target-I, by advancing towards the nearest nodes of the aircraft.

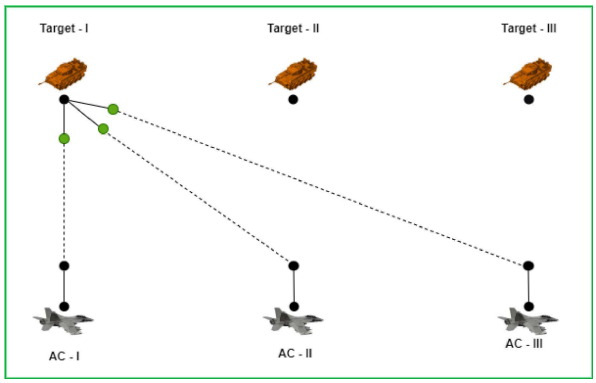


Figure 4.4 One iteration for aircraft-target pair after assignment.

After completing these steps, we finish one iteration of route planning. As we reassign tasks in each iteration, we also generate route segments for aircraft to reach other mission points. Furthermore, as each target tree extends toward all aircraft, the likelihood of finding the optimal route for an aircraft increases.

The Target Assignment Algorithm, $AssignTargetsToACs$, takes two separate arrays representing aircraft and threat positions as input and returns an array indicating the assignment of each target to an aircraft. The algorithm initiates by assigning values to N and K, representing the number of targets and aircraft, respectively, assuming that

these quantities are equal within the scope of this thesis. Following this, a $K \times N$ matrix is created, where each row corresponds to an aircraft, and each column signifies a target, storing the distances between aircraft and threats. Through an iterative process, the algorithm sequentially assigns each threat to the nearest aircraft, completing the assignment of all threats to aircraft. After extending all $Tree_{init}$ trees in each iteration, we call $AssignTargetsToACs$ again, facilitating task assignments among aircraft. So, dynamic task reassignment becomes possible if an aircraft's orientation shifts from one threat to another in subsequent iterations. This dynamic feature enables task reassignment without creating a new tree from scratch, allowing the system to adapt to various threats with the existing configuration. So, the algorithm provides flexibility in target assignments, accommodates changes in aircraft direction, and efficiently adapts to evolving threat scenarios.

4.3. Narrowed Regions Strategy

To establish a viable route between the initial position T_{init} and the desired destination T_{goal}, we must undertake sampling within the environment E. Sampling involves systematically selecting points within the spatial domain of E, thereby capturing a representative subset of the available configuration space. Unlike previous sampling-based techniques that typically encompass the entire environment, our approach introduces a novel strategy. In this method, we dynamically adjust the sampling environment's dimensions relative to the aircraft's current speed. The formula $x = vt$ plays a significant role, with x representing the radius of a sphere centered precisely at the aircraft's current position. Here, the aircraft's speed is denoted by v, and t is taken as a fixed value of 1 second. By utilizing this formula, we introduce a dynamic and adaptive sampling approach wherein the resulting sphere encapsulates the aircraft's immediate surroundings at a given time. By doing so, we consider the real-time velocity and trajectory of the aircraft, effectively tailoring the sampling process to the specific needs and capabilities of the flying system. This adaptive approach optimizes the utilization of computational resources and accelerates the planning process, as the sampling focuses primarily on areas relevant to the aircraft's immediate vicinity.

Algorithm 1: $AssignTargetsToACs(AC[], Target[])$

Input: AC[], Target[]
Output: targetToAC[]
$N \leftarrow$ Length of Target[]
$K \leftarrow$ Length of AC[]
$targetToAC[] \leftarrow \varnothing$
$distances[][] \leftarrow \varnothing$ // $K \times N$ matrix
for $i = 0$ *to* N **do**
  for $j = 0$ *to* K **do**
    $distances[i][j] \leftarrow$ norm$(AC[j] - Target[i])$;
for $i = 0$ *to* N **do**
  $minDistance \leftarrow N/A$;
  $targetIdx \leftarrow 0$;
  $acIdx \leftarrow -1$;
  for $j = 0$ *to* K **do**
    $minDist \leftarrow \min$(distances$[j]$); // Minimum distance of AC j
    $idx \leftarrow$ index of ;
    if $minDistance >$ **then**
      $minDistance \leftarrow$;
      $targetIdx \leftarrow$;
      $acIdx \leftarrow j$;
  Set all distances of $acIdx$ in distances to N/A; // Exclude this aircraft
  Set all distances of $targetIdx$ in distances to N/A; // Exclude this target
  $targetToAC[acIdx] \leftarrow$ targetIdx;
return $targetToAC$;

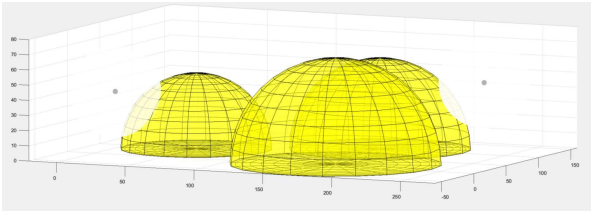


Figure 4.5 Sample Environment with sampling area around T_{init} and T_{goal}

In Figure 4.5, we can observe an example illustration of a sampling area. The depicted scenario features a transparent 3D sphere visually representing the concept of a sampling area. This sphere encapsulates a specific part of the environment, highlighting the designated space for careful analysis and strategic planning. Through this illustrative example, Figure 4.5 demonstrates the pivotal role of adaptive sampling in path planning, emphasizing its ability to focus computational effort on critical areas and ultimately improve the precision and efficiency of navigational pathfinding. Considering a fixed, larger region along the x, y, and z axes instead of our narrowed region approach, we would significantly expand our sampling region. Accounting for the aircraft's current position would lead to sampling at points that are either impractical or impossible to reach. We have observed that the expansion of the sampling region impedes the identification of optimal routes in test scenarios. The Narrowed Region technique facilitates the swift generation of a shorter and more efficient route, contrasting with considering the entire area, which could complicate the task by generating a longer route over an extended period.

Drawing inspiration from RRT-Connect principles, the approach establishes two distinct trees rooted in the initial point T_{init} and the goal point T_{goal}, respectively. The iterative process involves selecting the closest nodes from each tree stepwise. These chosen nodes play a pivotal role in shaping the growth of their respective trees, forming a bridge between the starting and target points. We systematically apply a predefined sampling strategy to the chosen nodes to optimize the search process. The algorithm converges towards an optimal trajectory by iteratively identifying the closest nodes and using the tailored sampling approach respect to traditional RRT approaches.

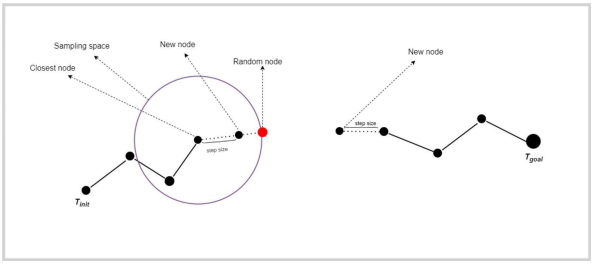


Figure 4.6 Sampling in E.

Figure 4.6 illustrates a representative sampling scenario, depicting two distinct trees stemming from the initial configuration, denoted as T_{init}, and the goal configuration, denoted as T_{goal}. The nodes designated as $N_{closest}$ signify the most relative nodes within their corresponding trees. Notably, we establish a sampling region based on the spatial vicinity of these closest nodes. Within this delineated sampling region, we generate N_{random} in the search space of both most relative nodes. Upon traversing from $N_{closest}$ to N_{random} over the distance of the specified step size, we attain the resultant node N_{new}, signifying a progressive advancement within the trajectory space.

$$\mathbf{N}_{\text{closest}} = (x_{\text{cl}}, y_{\text{cl}}, z_{\text{cl}}) \tag{11}$$

$$\mathbf{N}_{\text{random}} = (x_{\text{rand}}, y_{\text{rand}}, z_{\text{rand}}) \tag{12}$$

$$\mathbf{N}_{\text{new}} = (x_{\text{new}}, y_{\text{new}}, z_{\text{new}}) \tag{13}$$

$$\text{step is the step size} \tag{14}$$

$$\text{d is the distance between } \mathbf{N}_{\text{closest}} \text{ and } \mathbf{N}_{\text{random}} \tag{15}$$

Distance d between $\mathbf{N}_{\text{closest}}$ and $\mathbf{N}_{\text{random}}$:

$$\text{d} = \sqrt{(x_{\text{rand}} - x_{\text{cl}})^2 + (y_{\text{rand}} - y_{\text{cl}})^2 + (z_{\text{rand}} - z_{\text{cl}})^2} \tag{16}$$

New point $\mathbf{N}_{\text{new}}$:

$$x_{\text{new}} = x_{\text{cl}} + \frac{\text{step}}{\text{dist}} \cdot (x_{\text{rand}} - x_{\text{cl}}) \tag{17}$$

$$y_{\text{new}} = y_{\text{cl}} + \frac{\text{step}}{\text{dist}} \cdot (y_{\text{rand}} - y_{\text{cl}}) \tag{18}$$

$$z_{\text{new}} = z_{\text{cl}} + \frac{\text{step}}{\text{dist}} \cdot (z_{\text{rand}} - z_{\text{cl}}) \tag{19}$$

Figure 4.6 illustrates the obtained $Tree_{init}$ and $Tree_{goal}$ through successive sampling iterations. Both trees are within one step of each other, indicating the completion of the sampling process. After connecting $Tree_{init}$ and $Tree_{goal}$ from their nearest nodes, we create a graph in region E that extends between T_{init} and T_{goal}. Subsequently, path planning can be facilitated using algorithms such as A*, Dijkstra, or other search methods.

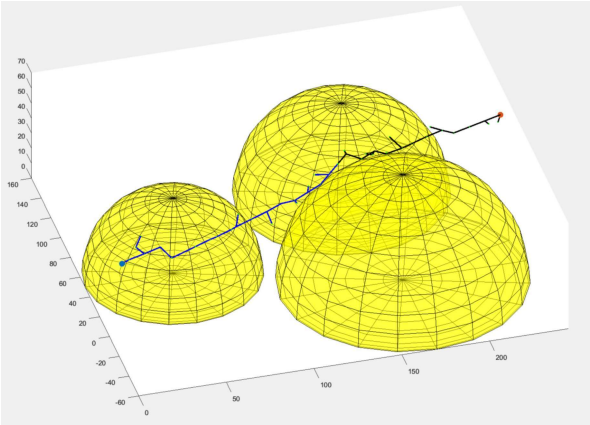


Figure 4.7 $Tree_{init}$ and $Tree_{goal}$ reaches after sampling processes.

4.4. Path Creation

This section formally defines our dynamic path-planning approach for both single-aircraft and multi-aircraft missions in complex 3D environments. At first, we use our method for single-aircraft missions. Our method aims to improve path planning for fighter aircraft missions by creating two trees, rooted at the start and goal positions (T_{init} and T_{goal}). It employs a narrowed sampling strategy guided by the nearest nodes in both trees, focusing exploration on restricted regions. This iterative process generates a connecting tree between the start and goal points. By combining bidirectional RRT with narrowed sampling, the algorithm effectively explores regions near the current node, enhancing path planning in constrained aircraft movement scenarios. The optimal trajectory is determined using the A* algorithm. Simulation results across diverse scenarios consistently demonstrate the algorithm's ability to generate smooth and adaptable paths, promising improved mission planning and execution for fighter aircraft in dynamic environments.

Algorithm 2: Narrowed-BiRRT

Data: Initial Configuration $T_{\text{init}}(x, y, z)$, Goal Configuration $T_{\text{goal}}(x, y, z)$, Threats $Threats$, Step Size M

Result: Trajectory

$Tree_{\text{init}} \leftarrow \varnothing$;
$Tree_{\text{goal}} \leftarrow \varnothing$;
$Trajectory \leftarrow \varnothing$;
$E \leftarrow$ CreateEnvironment$(T_{\text{init}}, T_{\text{goal}})$;
Set T_{init} as root of $Tree_{\text{init}}$;
Set T_{goal} as root of $Tree_{\text{goal}}$;
$node_{\text{init}} \leftarrow$ Nearest$(Tree_{\text{init}}, T_{\text{goal}})$;
$node_{\text{goal}} \leftarrow$ Nearest$(Tree_{\text{goal}}, T_{\text{init}})$;
$flag \leftarrow$ init;
while *dist*$(node_{init}, node_{goal}) > N$ **do**
    if *flag* $==$ *init* **then**
        $S_{\text{init}} \leftarrow$ GetSamplingRegion$(node_{\text{init}})$;
        $node_{\text{rand}} \leftarrow$ Sample$(S_{\text{init}}, Threats)$;
        TakeAStep$(Tree_{\text{init}}, node_{\text{rand}}, M, Threats)$;
        $node_{\text{init}} \leftarrow$ Nearest$(Tree_{\text{init}}, node_{\text{goal}})$;
        TakeAStep$(Tree_{\text{goal}}, node_{\text{init}}, M, Threats)$;
        $node_{\text{goal}} \leftarrow$ Nearest$(Tree_{\text{goal}}, node_{\text{init}})$;
        $flag \leftarrow$ goal;
    else
        $S_{\text{goal}} \leftarrow$ GetSamplingRegion$(node_{\text{goal}})$;
        $node_{\text{rand}} \leftarrow$ Sample$(S_{\text{goal}}, Threats)$;
        TakeAStep$(Tree_{\text{goal}}, node_{\text{rand}}, M, Threats)$;
        $node_{\text{goal}} \leftarrow$ Nearest$(Tree_{\text{goal}}, node_{\text{init}})$;
        TakeAStep$(Tree_{\text{init}}, node_{\text{goal}}, M, Threats)$;
        $node_{\text{init}} \leftarrow$ Nearest$(Tree_{\text{init}}, node_{\text{goal}})$;
        $flag \leftarrow$ init;
$Tree \leftarrow Tree_{\text{init}} \cup Tree_{\text{goal}}$;
$Trajectory \leftarrow$ A_star$(T_{\text{init}}, T_{\text{goal}}, Tree)$;

$Sample$: The "sampling" algorithm is designed to perform a random sampling within a narrowed area, denoted S, based on the given initial state and the all threat information contained in $Threats$. This process generates a random configuration by successively calculating the x, y and z coordinates, resulting in the creation of $node_{\text{rand}}$. Then, the algorithm evaluates the proximity of $node_{\text{rand}}$ to each threat by measuring their respective distances. To ensure safety, the algorithm compares these distances with the detection radius of each threat. If $node_{\text{rand}}$ falls within the coverage area of a threat, a new random configuration is calculated and the comparison process is repeated. This iterative process continues until a configuration is found that falls outside the coverage areas of all threats. It is worth noting that empirical experiments have consistently shown that this sampling method successfully prevents the algorithm from entering an infinite loop, which underlines its efficiency and reliability in practical applications.

Algorithm 3: $Sample(S, Threats)$

$node_{\text{rand}} \leftarrow \varnothing$;
$isCollisionOccured \leftarrow$ true;
while *isCollisionOccured* **do**
    $x \leftarrow S(x) + (S(x) - S(R)) \cdot random_{\text{number}}$;
    $y \leftarrow S(y) + (S(y) - S(R)) \cdot random_{\text{number}}$;
    $z \leftarrow S(z) + (S(z) - S(R)) \cdot random_{\text{number}}$;
    $node_{\text{rand}} \leftarrow (x, y, z)$;
    $isCollisionOccured \leftarrow$ false;
    for *each threat* $threat$ *in* $Threats$ **do**
        if *norm*$(threat(x, y, z)$ - $node_{rand}) \leq threat(R)$ **then**
            $isCollisionOccured \leftarrow$ true;
            break;
return $node_{\text{rand}}$;

$TakeAStep$: The algorithm works in the context of a tree structure, starting from T_{init} or T_{goal}, and considers a defined node within the environment E, together with the inputs M and $Threats$. The main function of this algorithm is to add a new node to the given tree. The process starts by identifying the $node_{nearest}$ configuration within the tree. Then the algorithm calculates the distance between $node_{nearest}$ and the input node. If the distance is less than M and the path between $node_{nearest}$ and $node_{new}$ does not intersect any threat zones, $node_{new}$ is

determined and added to the tree. Conversely, the calculated distance is greater than M, the algorithm continues by advancing $node_{nearest}$ by the distance M in the direction of the input node, which leads to the derivation of $node_{new}$. Again, $node_{new}$ is only included in the tree if the path between $node_{nearest}$ and $node_{new}$ is free of threatening areas.

Algorithm 4: $TakeAStep(tree, node, M, Threats)$

$node_{\text{nearest}} \leftarrow$ Nearest$(tree, node)$;
$node_{\text{new}} \leftarrow$ Nearest$(tree, node)$;
if *norm(*$node_{nearest} - node$*)* > 0 **then**
  if *norm(*$node_{nearest} - node$*)* $< M$ **then**
    $node_{\text{new}} \leftarrow node$;
    if *leg* $node_{nearest} \rightarrow node_{new}$ *not intersected by any threat in Threats* **then**
      add $node_{\text{new}}$ to tree;
  else
    $node_{\text{new}} \leftarrow node - node_{\text{new}}$;
    $node_{\text{new}} \leftarrow$ Nearest$(tree, node) + \left(\frac{node_{\text{new}}}{\text{norm}(node_{\text{new}})}\right) \cdot$ M;
    if *leg* $node_{nearest} \rightarrow node_{new}$ *not intersected by any threat in Threats* **then**
      add $node_{\text{new}}$ to tree;
return $tree$;

The same algorithm, with minor modifications, can be adapted for use in multi-aircraft missions. No alterations are required for the $Sample$ and $TakeAStep$ algorithms. The only necessary adjustments pertain to the sequence and structure of these algorithms, tailored to the context of multi-aircraft operations.

In the context of multi-aircraft missions utilizing the Narrowed Regions-based Bidirectional RRT (Narrowed-BiRRT) algorithm, the process commences by defining the environment variable E. Subsequently, we establish aircraft-to-target assignments. Following this assignment, we execute a loop until each aircraft reaches its target. The steps within this loop unfold as follows:

- Initially, for $Tree_{init}$, we create a sampling area S, considering each aircraft's current speed. Then, we apply the TakeAStep algorithm to both $Tree_{init}$ and $Tree_{goal}$ for

Algorithm 5: $Narrowed - BiRRT - MultiAircraft$

Inputs: $T_{\text{init}}[]$, $T_{\text{goal}}[]$, Threats, Speeds, Step Size M

Output: $Trajectories[]$

$Tree_{\text{init}}[] \leftarrow \varnothing$;
$Tree_{\text{goal}}[] \leftarrow \varnothing$;
$E \leftarrow$ CreateEnvironment($T_{\text{init}}[]$, $T_{\text{goal}}[]$) ;
$targetsToACArr[] \leftarrow AssignTargetsToACs(T_{\text{init}}[], T_{\text{goal}}[])$;
$firstIterationFlag \leftarrow True$;
$K \leftarrow$ Number of ACs ;
$N \leftarrow$ Size of targetsToACArr ;
$nearestStartNode \leftarrow \varnothing$;
$nearestGoalNode \leftarrow \varnothing$;
$isACReachToTarget[K] \leftarrow \{0\}$;
while $0 \in isACReachToTarget$ **do**
  for $i = 1$ *to* N **do**
    if *firstIterationFlag is equal to True* **then**
      $goalNode_{\text{nearest}} \leftarrow Tree_{\text{goal}}[targetsToACArr[i]]$;
      $initNode_{\text{nearest}} \leftarrow Tree_{\text{init}}[i]$;
      $acId \leftarrow i$;
    if *isACReachToTarget*$[acId]$ *is not equal to True* **then**
      $initNode_{\text{nearest}}, goalNode_{\text{nearest}} \leftarrow$ Nearest($Tree_{\text{init}}[acId], node_{\text{goal}}$[targetID]) ;
      $S_{\text{goal}} \leftarrow$ GetSamplingRegion($initNode_{\text{nearest}}$, speeds$[acId]$) ;
      $node_{\text{rand}} \leftarrow$ Sample(S_{init}, Threats) ;
      TakeAStep($Tree_{\text{init}}[acId], node_{\text{rand}}, M, Threats$) ;
      TakeAStep($Tree_{\text{goal}}$[targetId], $node_{\text{rand}}, M, Threats$) ;
  for $i = 1$ *to* N **do**
    $acId \leftarrow i$;
    $targetId \leftarrow i$;
    if *isACReachToTarget*$[acId]$ *is not equal to True* **then**
      for *each* $Tree_{init}$ **do**
        $initNode_{\text{nearest}}, goalNode_{\text{nearest}} \leftarrow$ Nearest($Tree_{\text{init}}[acId], node_{\text{goal}}$[targetID]) ;
        TakeAStep($Tree_{\text{goal}}$[targetId], $initNode_{\text{nearest}}, M, Threats$) ;
  for $i = 1$ *to* N **do**
    $acId \leftarrow i$;
    $targetId \leftarrow$ targetsToACArr$[i]$;
    $initNode_{\text{nearest}}, goalNode_{\text{nearest}} \leftarrow$ Nearest($Tree_{\text{init}}[acId], node_{\text{goal}}$[targetID]) ;
    $T_{\text{init}}[acId] \leftarrow$ initNode$_{\text{nearest}}$;
    $T_{\text{goal}}[acId] \leftarrow$ goalNode$_{\text{nearest}}$;
  $targetsToACArr[] \leftarrow$ AssignTargetsToACs($T_{\text{init}}[], T_{\text{goal}}[]$) ;
  for $i = 1$ *to size of* $isACReachToTarget$ **do**
    $acId \leftarrow i$;
    $targetId \leftarrow targetsToACArr[i]$;
    if *isACReachToTarget*$[acId]$ *is not equal to True* **then**
      $initNode_{\text{nearest}}, goalNode_{\text{nearest}} \leftarrow$ Nearest($Tree_{\text{init}}[acId], node_{\text{goal}}[targetID]$) ;
      if *norm*$(initNode_{nearest} - goalNode_{nearest}) < M$ **then**
        $isACReachToTarget[acId] \leftarrow True$;
  $firstIterationFlag \leftarrow False$;

each pair of assigned aircraft and targets, propelling both trees towards each other by the specified step size.

- After completing the first step for all aircraft-target pairs, in the second step, each $Tree_{goal}$ progresses towards the nearest nodes of all $Tree_{init}$ structures, moving by the step size.

- Up to this point, we complete an iteration for all $Tree_{init}$ and $Tree_{goal}$ structures. Subsequently, if the assigned aircraft and target pairs are sufficiently close, we merge

$Tree_{init}$ and $Tree_{goal}$, shaping a tree from the aircraft's current position towards the target.

- After completing all the steps, considering the existing assignments, we identify the closest nodes between $Tree_{init}$ and $Tree_{goal}$. We then update the $T_{init}[]$ and $T_{goal}[]$ lists accordingly. Following this, we rerun the target assignment algorithm to update the assignments.

The Narrowed-BiRRT algorithm presents a versatile solution applicable to single and multi-aircraft missions. In single-aircraft scenarios, the algorithm efficiently navigates the vehicle through a dynamic environment, adapting its trajectory to avoid potential threats. In multi-aircraft missions, the algorithm further extends its utility by facilitating concurrent operations of multiple aircraft. Through an iterative process involving environment definition, aircraft-to-target assignments, and coordinated movement, the algorithm adeptly orchestrates the trajectories of multiple aircraft, addressing complex mission objectives. The adaptive nature of the algorithm allows it to seamlessly transition between single and multi-aircraft applications, demonstrating its effectiveness in diverse mission scenarios.

4.5. Altitude Optimization

In the realm of trajectory optimization for fighter aircraft, ensuring optimal mission execution and aircraft performance relies heavily on the nuanced consideration of altitude dynamics. Fighter jets require agile and adaptive altitude management strategies that align with the dynamic requirements of combat scenarios. Deliberate focus is warranted, as instantaneous changes in pitch angles are often unsuitable, emphasizing the need to achieve smooth and controlled altitude ascents and descents. Therefore, this section introduces the development of an altitude optimization algorithm. We designated waypoints for each aircraft and established a route, devising a method to optimize the altitudes of each waypoint while keeping the x and y coordinates of each waypoint constant in the route.

In the context of optimizing the trajectory for aircraft traversing from current position to point destination position, particular attention is directed towards achieving a smooth and controlled altitude transition. The optimization process focuses on minimizing the altitude difference between current and destination positions, thereby facilitating a seamless altitude change as the aircraft approaches its destination. This strategic approach seeks to mitigate abrupt variations in altitude and pitch angles, ensuring a more harmonious and efficient flight profile.

Figure 4.8 illustrates a position/altitude graph, where the horizontal axis represents the aircraft's position in the x and y coordinates, and the vertical axis depicts altitude information at each position. According to the graph, the aircraft currently maintains an altitude of 2000 ft at its present position, while the destination point is situated at an altitude of 5000 ft, resulting in an altitude difference of 3000 ft. The altitude optimization developed aims to sequentially adjust the altitude of each waypoint sequentially, forming the route to correspond to this altitude difference. This optimization process strives to align the altitude of each waypoint within the specified range, ensuring a smooth transition from the aircraft's current position to the destination point.

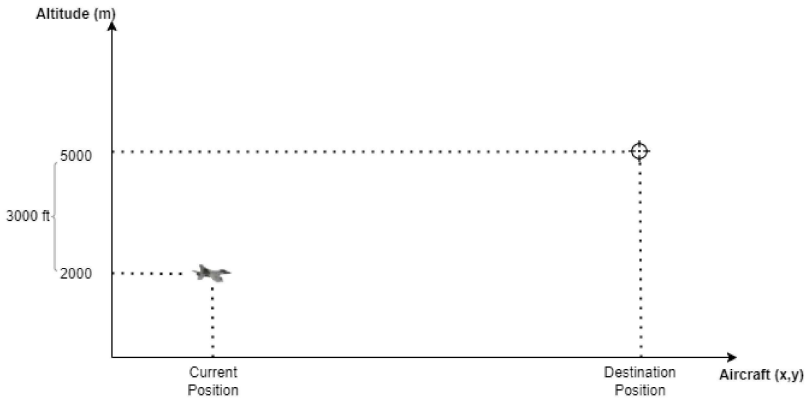


Figure 4.8 Aircraft position - altitude graph.

For instance, considering Figure 4.9, where it is assumed that there are no obstacles in the environment, the sequential passage through waypoints W_1, W_2, and W_3 is required to travel from the current position to the destination position. While following the route, the altitude is adjusted smoothly by setting the height value corresponding to each waypoint

as its z-axis value, as depicted in the figure. This approach ensures a gradual increase in altitude, enhancing the smoothness of the trajectory as the aircraft traverses through each waypoint from the current position to the destination.

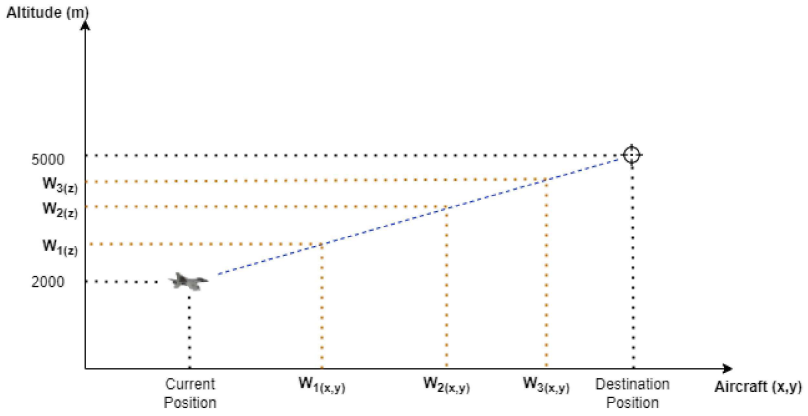


Figure 4.9 Waypoints in aircraft position - altitude graph.

Figure 4.10 illustrates the method utilized in the presence of an environmental threat. In this scenario, as the altitude of the threat surpasses that of W_2, we increase the altitude of W_2 to exceed that of the threat. Subsequently, we delineate a height curve from the current position to W_2 and calculate the value for W_1's z-axis. Following this, we draw a height curve from W_2 to the destination position, computing the altitudes of the subsequent waypoints. Continuously optimizing for altitude involves adjusting the start and end points as the number of threats increases.

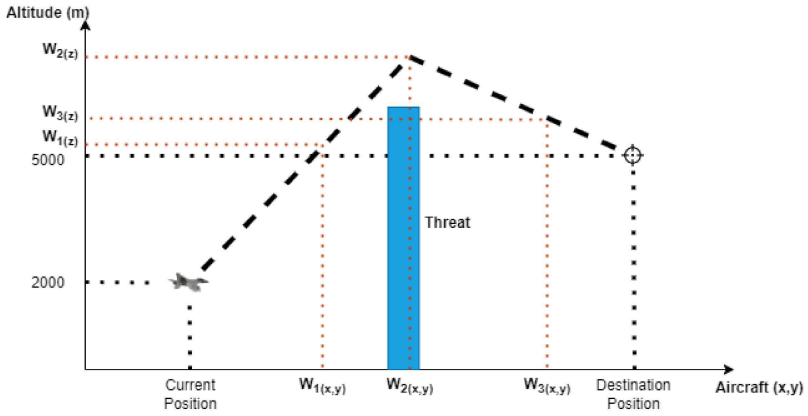


Figure 4.10 Optimized altitudes in aircraft position - altitude graph.

This approach outlines how altitude optimization is accomplished through Algorithms 4 and 5. As mentioned earlier, we identify waypoints to establish the route for each aircraft. Following this, we invoke the $OptimizeAltitude$ algorithm to determine optimized altitudes and waypoints, defining each aircraft's trajectory. The algorithm takes waypoints, the initial node, and the goal node as inputs and produces waypoints along with optimized altitudes as its output.

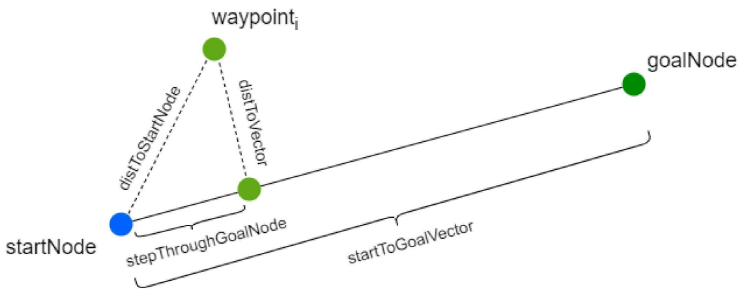


Figure 4.11 Altitude optimization.

The $GetAltitude$ function is the initial point of invocation for each waypoint, with specified parameters. It creates a vector from the starting point to the target point, calculating the distance of the waypoint to this vector. This process involves determining the projection of the waypoint onto the vector. The altitude value for the resulting new position is then calculated by normalizing the altitude difference between the starting and ending points, utilizing it as a parameter to ascertain the waypoint's altitude. Figure 4.11 illustrates the operational dynamics of this method. A vector is formed from $startNode$ to $goalNode$, and subsequently, we calculated the distances from the node to $startNode$ and $startToGoalVector$ to find $stepThroughGoalNode$. Progressing linearly from $startNode$ towards $goalNode$ by the length of $stepThroughGoalNode$ determines the point where the waypoint aligns. Lastly, we return the altitude value of this point, thereby establishing the altitude for smooth route tracking.

After calculating the optimized altitude, we check whether the route intersects with any potential threats. If an intersection with a threat zone is detected, we increase the altitude

Algorithm 6: $GetAltitude(startNode, goalNode, node)$

Set calcAltitude to 0;
distToStartNode $\leftarrow \|$startNode $-$ node$\|$;
startToGoalVector $\leftarrow$ goalNode $-$ startNode;
distToVector $\leftarrow$ GetDistanceToVector(Waypoints$[i]$, startToGoalVector);
stepThroughGoalNode $\leftarrow \sqrt{\text{distToStartNode}^2 - \text{distToVector}^2}$;
normalizedDirection $\leftarrow \frac{\text{startToGoalVector}}{\|\text{startToGoalVector}\|}$;
displacementVector $\leftarrow$ normalizedDirection $\times$ stepThroughGoalNode;
finalPosition $\leftarrow$ startNode $+$ displacementVector;
Set calcAltitude to altitude of finalPosition;
return calcAltitude;

of the relevant waypoint by a predetermined amount to prevent the route from intersecting with the threat. In this case, we need to re-optimize the altitude of the route segment up to the relevant waypoint. Failure to do so would require the aircraft to make an abrupt pitch adjustment. Instead, we set the last optimized waypoint as the new target point and recommence the optimization process from the starting point up to this designated waypoint.

Algorithm 7: $OptimizeAltitude(Waypoints, Threats, startNode, goalNode)$

Inputs: Waypoints, Threats, startNode, goalNode
Outputs: Optimized_Waypoints
Set collisionOccured to False;
for $i = 1$ *to size of Waypoints* **do**
    calcAltitude $\leftarrow$ GetAltitude(startNode, goalNode, Waypoints[i]);
    Set altitude of Waypoints[i] to calcAltitude;
    while *isCollisionOccured(Waypoints[i-1], Waypoints[i], Threats)* **do**
        Increase altitude of Waypoints[i] by 5 m;
        Set collisionOccured to True;
if *collisionOccured* $=$ *True* **then**
    Set startNode $\leftarrow$ Waypoints[0];
    Set goalNode $\leftarrow$ Waypoints[i];
    for $j = 2$ *to* i **do**
        node $\leftarrow$ Waypoints[j];
        calcAltitude $\leftarrow$ GetAltitude(startNode, goalNode, node);
        Set altitude of node to calcAltitude;
    Set startNode $\leftarrow$ Waypoints[i];
    Set goalNode $\leftarrow$ last Waypoint;
    Set collisionOccured to False;
return Optimized_Waypoints;

In conclusion, altitude optimization is a critical aspect of aircraft navigation that ensures efficiency and safety. Through the application of Algorithms 1 and 2, waypoints are meticulously defined, and the OptimizeAltitude algorithm is invoked to determine the optimal altitudes for each point along the route. The process involves calculating distances and projections and normalizing altitude differences to achieve a smooth and optimized trajectory. Furthermore, the system incorporates threat detection mechanisms, dynamically adjusting waypoint altitudes to mitigate potential risks. The iterative nature of altitude optimization, with its continuous monitoring and adjustment, underscores its significance in enhancing the overall performance and safety of aircraft navigation systems.

5. EXPERIMENTAL RESULTS

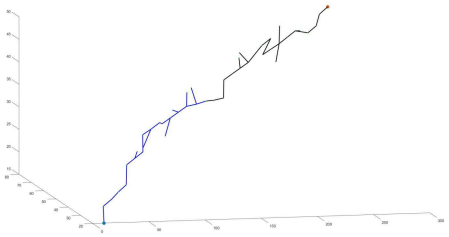
In this section, we meticulously evaluate the methodologies proposed in the thesis within the confines of the created environment E, characterized by the x, y, and z axes. We analyze the time taken for path planning and assess route distances for each scenario. Our rigorous evaluation begins with a focused examination of single-aircraft missions, specifically delving into the intricacies of Algorithm 2. In this particular context, we deliberately exclude considerations for altitude optimization. Following this, we extend our exploration to encompass scenarios involving multi-aircraft missions, seamlessly integrating the application of Algorithm 2. Altitude optimization becomes a pivotal component during this phase, factored into the evaluation alongside all algorithms.

To execute these evaluations, we craft simulations to assess the efficacy of the proposed approaches, implementing and visualizing algorithms tailored for single-aircraft missions using Matlab. Furthermore, to ensure computational robustness, we conduct tests on a computer with an 11th Gen Intel Core i5 12 CPU running at 2.75GHz, utilizing C++. Notably, evaluations for multi-aircraft missions are carried out exclusively using Matlab, maintaining consistency and coherence in our assessment approach.

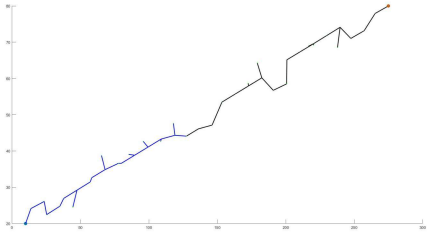
5.1. Single-Aircraft Path Planning

In Figure 5.1, a trajectory generated without any threats is depicted using the constructed $Tree_{init}$ and $Tree_{goal}$. The generation time for the route is 17 ms. Despite a significant difference in altitude between T_{init} and T_{goal}, the generated route exhibits a smooth increase in altitude. Figure 5.1(b) illustrates the path generated in the x and y axes, revealing minimal deviations between the starting and target points.

The results shown in Figure 5.2 are obtained upon introducing a threat into the environment. Examining both (a) and (b) visuals, it is evident that significant deviations are avoided. However, since altitude optimization is not applied in this scenario, $Tree_{init}$ abruptly increases its altitude when approaching the intersection point with the threat. Executing

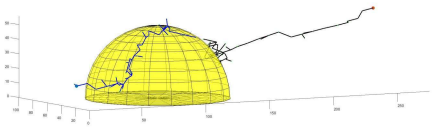


(a) No threat(3D)

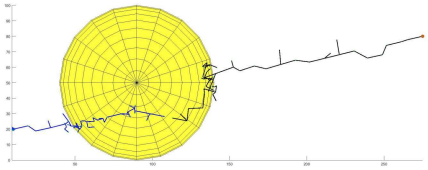


(b) No threat(2D)

Figure 5.1 Single Aircraft path planning in no-threat environment



(a) One threat(3D)



(b) One threat(2D)

Figure 5.2 Single Aircraft path planning in one-threat environment

such a maneuver is challenging for an aircraft. Nevertheless, Section 5.2 demonstrates that this issue is addressed by implementing altitude optimization.

As the number of environmental threats increases, the generated routes in Figures 5.3 and 5.4 exhibit no deviations. Naturally, as the threat count rises, the execution time also increases. Table 5.1 illustrates how the time spent on route generation increases as the threat scenarios are examined sequentially. However, this duration may vary based on the specific scenario. In environments with a high density of threats, the algorithm's runtime will inevitably increase.

It is crucial to highlight that the algorithm's promptness in producing results in threat-free areas aligns with expectations, given the absence of collisions. Conversely, the increased execution time with growing environmental threats is permissible due to the additional complexities involved. Moreover, variations in execution time across repeated executions of the same scenario affirm that the algorithm's outcomes are adaptable to specific conditions

(a) Two threat(3D) (b) Two threat(2D)

Figure 5.3 Single Aircraft path planning in two-threat environment

(a) Three threat(3D) (b) Three threat(2D)

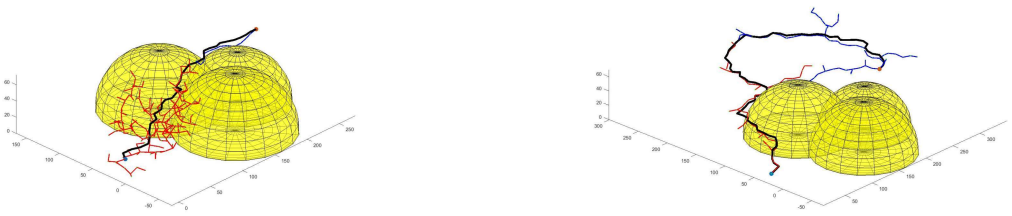
Figure 5.4 Single Aircraft path planning in three-threat environment

Table 5.1 Experimental Results

T_{init}	T_{goal}	Threat 1	Threat 2	Threat 3	Execution Time
(10, 20, 15)	(275, 80, 50)	-	-	-	17 ms
(10, 20, 15)	(275, 80, 50)	(90, 50, 0, 50)	-	-	34 ms
(10, 20, 15)	(275, 80, 50)	(160, 0, 0, 60)	(140, 95, 0, 60)	-	53 ms
(10, 20, 15)	(275, 80, 50)	(160, 0, 0, 60)	(140, 95, 0, 60)	(210, 45, 0, 60)	58 ms

rather than predetermined. In essence, the Narrowed-BiRRT demonstrates its ability to generate routes that evade detection by adversary elements when deployed in real aircraft. A meticulous examination of the results indicates that redundant sampling is effectively circumvented, supporting the assumption that the developed method provides the most efficient route in a dynamically changing warfare environment, accounting for current threats.

In Figure 5.5, we showcase results obtained from scenarios employing and not employing the Narrowed Regions method using the same setting. The $Narrowed - BiRRT - MultiAircraft$ algorithm is applied, and altitude optimization is incorporated into the



(a) Enabled narrowed regions

(b) Disabled narrowed regions

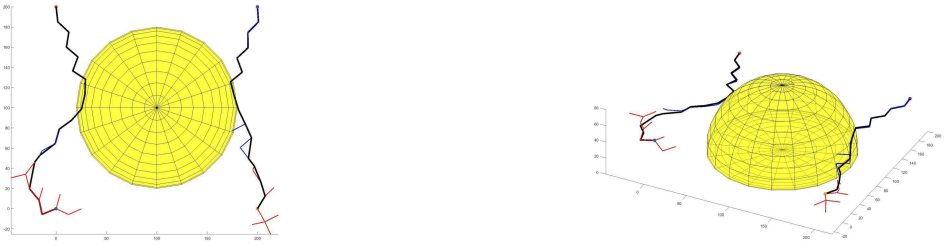
Figure 5.5 Single-Aircraft Path Planning Comparison

generated routes. Upon close examination, it is evident that the route in Figure 5.5(a) is shorter, measuring 377.102 m, compared to the route in Figure 5.5(b), which spans 826.864 m. Interestingly, despite the longer route in (b), the computation time is observed to be shorter than that in (a). A detailed analysis reveals that scenario (a) involves more samplings. Conversely, in scenario (b), fewer samplings and their locations farther from the current position result in longer route legs. This implies that more samplings extend the algorithm's route generation time. However, when the same scenario is tested in an environment with fewer and less dense threats, the Narrowed Regions method produces shorter and smoother routes more rapidly.

5.2. Multi-Aircraft Path Planning

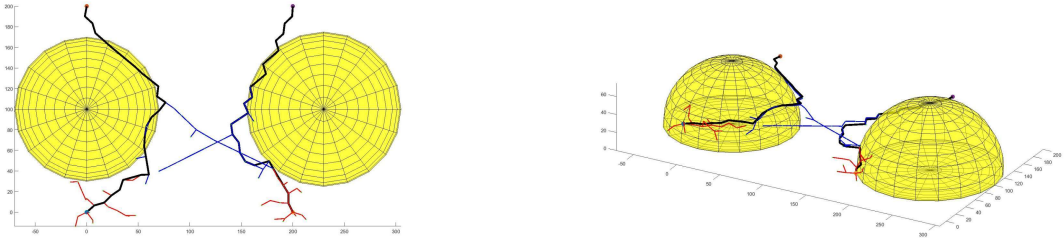
Multi-aircraft path planning has been evaluated using three scenarios involving two aircraft and two mission points. Table 5.2 provides information on the current positions of the aircraft, their target points, the locations of each threat, and the coverage area. Additionally, the last column presents the time taken to generate the route for each scenario, considering the provided scenario inputs.

In Figure 5.6, we executed the scenario with only Threat I present in the environment, resulting in the generation of paths shown in (a) and (b). While Figure 5.6(a) illustrates the two-dimensional representation of the path using the x and y axes, Figure 5.6(b) provides a three-dimensional view. Compared to the single-aircraft path planning scenario, the path has slightly more deviation. So $Tree_{goal}$ nodes take steps towards the assigned aircraft and



(a) One Threat(2D) (b) One Threat(3D)

Figure 5.6 Multi-Aircraft Path Planning with One Threat



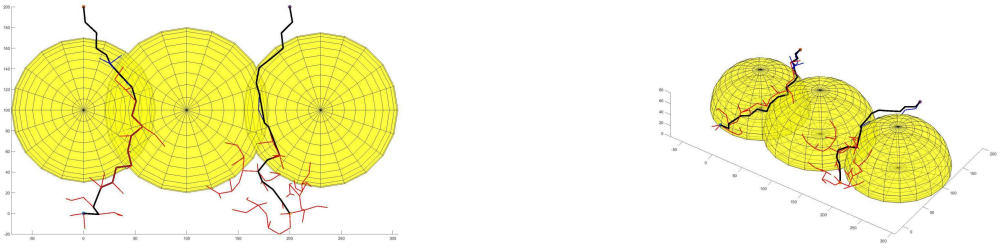
(a) Two Threat(2D) (b) Two Threat(3D)

Figure 5.7 Multi-Aircraft Path Planning with Two Threat

other aircraft, leading to deviations in $Tree_{goal}$. In Scenario I, we generated both paths in a total of 15.4 ms, with the respective lengths of the aircraft paths being 263.32 m and 225.89 m. Results highlight that the multi-aircraft algorithm can plan routes in a faster time than the single-aircraft algorithm.

In Figure 5.7, we tested Scenario II. Analyzing the two-dimensional representation reveals that the generated paths for both aircraft strive to find the optimal route by navigating close to the edges of the threats. Figure 5.7(b) provides a three-dimensional representation, showing a smooth increase in altitude. In this scenario, we calculated the generated paths for both aircraft to be 282.61 m and 270.02 m, respectively, and this process occurred in 26.5 ms.

In Figure 5.8, we conducted Scenario III testing. This scenario reveals that as the concentration of threats increases in a specific area, the route planning time also increases. Specifically, the algorithm generated 277.82 m and 250.13 m paths for each aircraft, respectively, in 46.1 ms. In this scenario, the increase in threat density corresponds to the



(a) Three Threat(2D)

(b) Three Threat(3D)

Figure 5.8 Multi-Aircraft Path Planning with Three Threat

rise in the number of samples, and we can understand that the escalation in route planning time is primarily attributed to this factor.

Table 5.2 Scenarios for multi-aircraft path planning

Scenario	$T_{init}[]$	$T_{goal}[]$	**Threat I**	**Threat II**	**Threat III**	**Execution Time (ms)**
Scenario I	(0 0 40), (200 0 30)	(0 200 35), (200 200 35)	(100, 100, 0, 80)	-	-	15.4
Scenario II	(0 0 40), (200 0 30)	(0 200 35), (200 200 35)	-	(0, 100, 0, 70)	(230, 100, 0, 75)	26.5
Scenario III	(0 0 40), (200 0 30)	(0 200 35), (200 200 35)	(100, 100, 0, 80)	(0, 100, 0, 70)	(230, 100, 0, 75)	46.1

In conclusion, Figure 5.9 illustrates the results obtained when employing the narrowed region method and when omitting it. Figure 5.9(a) shows results identical to those in Scenario III, generated in 142.4 ms. Although the duration exhibited slight variability due to temporary sampling congestion at the second aircraft's current position, it did not induce alterations in the route lengths of both aircraft. In Figure 5.9(b), route planning was conducted in a larger static sampling area, resulting in a significant increase in route lengths compared to (a). Naturally, the route planning time also increased, and the routes were obtained in 187.3 ms.

Upon examining the results of multi-aircraft path planning, it is evident that this method excels in mission scenarios with a high density of threats. When employing the Narrowed-BiRRT method, we observed an increased number of samplings. Although this slightly extended the route planning duration for some scenarios, utilizing our developed method proves advantageous due to generating shorter and more optimal routes. Notably, in multi-aircraft missions without threats, the Narrowed-BiRRT exhibits suboptimal

(a) Enabled narrowed regions

(b) Disabled narrowed regions

Figure 5.9 Multi-Aircraft Path Planning Comparison

performance. In Figure 5.1, when we tested a scenario with no threats for two aircraft, we observed non-linear paths generated, attributed to the extension of $Tree_{goal}$ trees towards all $Tree_{init}$ nodes. In the multi-aircraft scenario, this phenomenon manifested because it is not achievable in a single-aircraft scenario. Using Scenario III, we initially conducted separate route planning for both aircraft and, subsequently, individually performed single-aircraft route planning for each aircraft. In the multi-aircraft test, results were obtained in a total of 130 ms, producing routes of 259 m for AC-I and 288 m for AC-II. Route planning was initially performed for AC-I, followed by AC-II, with consistent threats. For AC-I, a route of 297 m was generated in 80 ms, while for AC-II, a 252 m route was produced in 72 ms. Evaluating these results, it is evident that the Narrowed-BiRRT demonstrates shorter planning times overall in multi-aircraft scenarios compared to single-aircraft route planning. The route length for AC-I in the multi-aircraft scenario is shorter than in the single-aircraft scenario, whereas the opposite holds for AC-II. This observation suggests that the Narrowed-BiRRT may generate more optimal routes for some aircraft and longer but still safe routes for others, depending on the specific scenario. Finally, throughout all scenarios, the extension of $Tree_{init}$ and $Tree_{goal}$ was conducted with a maximum step size of 15 m. This value should vary based on the average speed of the aircraft. In a simulated scenario with constant speeds of 450 and 400 knots for the aircraft, we adjusted the step size to 100 m. The step size may vary according to the aircraft's instantaneous speed and maneuvering capabilities and should be adjusted accordingly in a realistic scenario.

5.3. Evaluations

In the context of the thesis, we applied the presented methods to single and multi-aircraft missions and shared the results. In the following section, we evaluate the outcomes by considering various parameters.

5.3.1. Completeness

Aircraft systems must adhere to stringent standards such as RTCA DO-178B and DO-178C and require certification [53–56]. The present method shall demonstrate operability in any environment where a solution is possible. Evaluating the results in Sections 5.1 and 5.2 reveals that the method successfully identifies a solution when one is feasible. When Scenario III is executed 100 times without interruption consecutively, the Narrowed-BiRRT consistently generates routes for AC-I and AC-II on each iteration. Across the 100 attempts, an average of 81 ms is observed for AC-I, with an average route length of 298 m, while AC-II demonstrates an average route length of 264 m over the same 100 iterations. However, dynamic combat environments may necessitate navigating through regions with changing threats. In such scenarios, conducting a risk assessment enables the selection of routes through less perilous threat zones. As our method lacks a risk assessment, the generated routes consistently avoid threat zones. In essence, our method succeeds when capable of producing a collision-free route.

5.3.2. Efficiency

Combat environments, being dynamic, require quick decision-making. In missions with multiple aircraft, the mission commander needs to manage operations and command other pilots. Swift decision-making is crucial, as sudden threats invalidate the previous mission plan. Therefore, our developed method should rapidly provide a solution. When tested in an environment with a sufficiently large space and aircraft-threat distances exceeding 5 km, we observed that our approach generates routes in less than 1 second. However, this time will

vary depending on the number of aircraft and threats. As the number of aircraft increases, generating all routes will require more time.

5.3.3. Smoothness

Combat aircraft, characterized by their high speeds and remarkable maneuverability, can execute sharp maneuvers at elevated angles, but running perfect 90-degree turns is impractical. Therefore, the generated routes must be conducive to the aircraft's capabilities. Our proposed approach addressed instantaneous altitude changes by incorporating altitude optimization. However, optimization may be necessary for movements along the x and y axes. In some scenarios, we encountered situations where the turn angle between two legs in the generated routes exceeded 90 degrees in 2D. While following mission plans, aircraft may sometimes need to pass directly over waypoints and, at other times, slightly closer to create a wider arc. The recommended method suggests that, on occasion, aircraft may need to pass slightly more relative to waypoints for better adherence to the planned route.

5.3.4. Optimality

In path planning studies, we can obtain different routes, emphasizing either route length or planning time depending on the nature of the study. For military aircraft, executing the mission within the planned timeframe is crucial. Hence, we must find the shortest, safest route to the mission point. The results of single and multi-aircraft path planning studies reveal that the proposed methods generate nearly the fastest routes for each aircraft. When testing Narrowed-BiRRT with Scenario III, we observed that the generated route for AC-I was, on average, within 29 meters of the nearest threat, and at points closest to the threat, the distance between the path and the threat dropped to 1.6 meters. We also noted that, for AC-II, the generated route's average proximity to the nearest threat was approximately 25 meters, and the distance to the closest waypoint to the threat decreased to 1.2 meters. Notably, sampling, particularly in narrowed regions, proves effective. Analyzing Figure 5.5

and Figure 5.9 demonstrates that the use of the narrowed regions technique leads to shorter routes according to traditional RRT, RRT* and RRT-Connect.

5.3.5. Robustness

Robustness is crucial for completing a mission in military aircraft, emphasizing the importance of keeping mission plans secret and remaining unpredictable to enemy elements. Our developed method demonstrates the ability to generate different routes for the same scenario with each execution. When Narrowed-BiRRT was tested with Scenario III, the generation of routes of varying lengths, accompanied by a different number of waypoints in each test result, highlights the robustness of our method. It is essential to note that Narrowed-BiRRT consistently maintains robust performance, persistently producing complete, optimal, efficient, and smooth routes. This showcases the applicability of our developed method to critical systems.

5.3.6. Adaptability

Combat environments, which feature multiple elements, are dynamic and undergo continuous variability. Military systems necessitate solutions capable of adapting to different environmental conditions and constraints. We tested our developed method with various scenarios. The same algorithm showcased its capability to generate optimal safe routes in single and multi-aircraft missions rapidly. Furthermore, the method succeeded when we altered the number of aircraft, targets, and environmental threats. This underscores the adaptability of our developed method to diverse conditions.

6. CONCLUSION

In this thesis, we have delved into the intricate domain of dynamic mission assignment and route planning, addressing the unique challenges posed by single and multiple aircraft scenarios. The initial emphasis was placed on the critical role of mission planning for combat aircraft, highlighting the necessity for adaptive strategies that can promptly formulate new mission plans and routes in response to evolving threats during flight. Our comprehensive literature review scrutinized various path-planning methodologies, elucidating the prevalence of grid-based and sampling-based approaches. Notably, our focus leaned towards sampling-based techniques, discerning their heightened relevance and applicability within the scope of our study. Evaluating analogous works in the literature allowed us to glean insights into their methodologies, strengths, and limitations.

The core of our contribution lies in introducing the Narrowed-BiRRT method, a dynamic algorithm designed to cater to single and multi-aircraft missions. We meticulously defined the mathematical representation of the problem environment, incorporating the dynamic elements of multiple aircraft, their current positions, target locations, and identified threats. Our dynamic target assignment method, integral to the route planning process, was expounded upon, showcasing its pivotal role in simultaneously generating routes for each aircraft and constructing a real-time environment roadmap. Noteworthy is the adaptability embedded within our methodology, enabling swift task reassignment among aircraft leveraging the existing roadmap.

After task assignment, we elucidated our Narrowed Regions-based sampling method, a sophisticated technique that optimizes the RRT algorithm's sampling step. By judiciously considering the instantaneous speed of each aircraft, this method facilitates the generation of optimal, collision-free paths by confining the exploration to a narrower space, thereby significantly accelerating the planning process.

Recognizing that combat aircraft must adhere to feasible routes that account for their inherent operational constraints, we introduced our altitude optimization method. This feature ensures

that the generated routes facilitate smooth trajectory tracking by preventing abrupt changes in altitude, a crucial factor in enhancing the overall robustness and viability of the flight plans.

The culmination of our efforts involved rigorous testing across diverse scenarios. Initial assessments centered on single-aircraft mission scenarios, where the algorithm's performance was systematically evaluated under increasing threat scenarios. Subsequent examinations involving two aircraft, two targets, and varying threat levels demonstrated the algorithm's prowess, with execution times measured at 15.4, 26.5, and 46.1 milliseconds, respectively. The algorithm exhibited a consistent convergence rate of 100% across all tested scenarios, underscoring its reliability. Further scrutiny of the generated routes revealed not only their consistency but also their optimality.

Our methodology demonstrated its practical utility in a real-world scenario involving two aircraft and three threats by achieving an average route generation time of 1.4 seconds. The resulting routes exhibited average lengths of 6622.37 meters and 4420.39 meters for the respective aircraft, further attesting to the adaptability and efficiency of our proposed approach.

In conclusion, our meticulously developed methods have consistently proven their mettle, yielding routes that are complete, more efficient and optimal, smooth, robust, and adaptable to evolving scenarios. The methodology presented is a compelling solution applicable to real-life challenges in single- and multi-aircraft missions. As we navigate the complexities of modern aviation, the insights gained from this research contribute to the burgeoning field of autonomous flight and pave the way for innovative advancements in combat mission planning.

REFERENCES

[1] Robert L Shaw. Fighter combat. *Tactics and Maneuvering; Naval Institute Press: Annapolis, MD, USA*, **1985**.

[2] Nils-Hassan Quttineh and Torbjörn Larsson. Military aircraft mission planning: efficient model-based metaheuristic approaches. *Optimization Letters*, 9:1625–1639, **2015**.

[3] Janice H Laurence. Military leadership and the complexity of combat and culture. *Military Psychology*, 23(5):489–501, **2011**.

[4] Don Harris. *Decision making in aviation*. Routledge, **2017**.

[5] Robert E Enck. The ooda loop. *Home Health Care Management & Practice*, 24(3):123–124, **2012**.

[6] Jack H Sheehan, Paul H Deitz, Britt E Bray, Bruce A Harris, and Alexander BH Wong. The military missions and means framework. In *Proceedings of the Interservice/Industry Training and Simulation and Education Conference*, pages 655–663. **2003**.

[7] Mike C Bartholomew-Biggs, Steven C Parkhurst, and Simon P Wilson. Using direct to solve an aircraft routing problem. *Computational Optimization and Applications*, 21:311–323, **2002**.

[8] Nils-Hassan Quttineh, Torbjörn Larsson, Kristian Lundberg, and Kaj Holmberg. Military aircraft mission planning: a generalized vehicle routing model with synchronization and precedence. *EURO Journal on Transportation and Logistics*, 2(1-2):109–127, **2013**.

[9] Vijay K Shetty, Moises Sudit, and Rakesh Nagi. Priority-based assignment and routing of a fleet of unmanned combat aerial vehicles. *Computers & Operations Research*, 35(6):1813–1828, **2008**.

[10] Ian Moir. *Military avionics systems*. John Wiley & Sons, **2019**.

[11] Merrill Ivan Skolnik. Introduction to radar systems. *New York*, **1980**.

[12] John Griffith, Mykel Kochenderfer, and James Kuchar. Electro-optical system analysis for sense and avoid. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, page 7253. **2008**.

[13] James Powell. *Aircraft radio systems*. Pitman, **1981**.

[14] Maede Zolanvari, Raj Jain, and Tara Salman. Potential data link candidates for civilian unmanned aircraft systems: A survey. *IEEE Communications Surveys & Tutorials*, 22(1):292–319, **2019**.

[15] Daniel Gonzales. *Network-centric operations case study: air-to-air combat with and without Link 16*, volume 268. Rand Corporation, **2005**.

[16] Howie Choset, Kevin M Lynch, Seth Hutchinson, George A Kantor, and Wolfram Burgard. *Principles of robot motion: theory, algorithms, and implementations*. MIT press, **2005**.

[17] Lydia Kavraki and J-C Latombe. Randomized preprocessing of configuration for fast path planning. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 2138–2145. IEEE, **1994**.

[18] Steven LaValle. Rapidly-exploring random trees: A new tool for path planning. *Research Report 9811*, **1998**.

[19] Franz Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, **1991**.

[20] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, **1968**.

[21] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160, **1972**.

[22] Rong Zhou and Eric A Hansen. Breadth-first heuristic search. *Artificial Intelligence*, 170(4-5):385–408, **2006**.

[23] Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, 12(4):566–580, **1996**.

[24] David Hsu. *Randomized single-query motion planning in expansive spaces*. Stanford University, **2000**.

[25] David Hsu, J-C Latombe, and Rajeev Motwani. Path planning in expansive configuration spaces. In *Proceedings of international conference on robotics and automation*, volume 3, pages 2719–2726. IEEE, **1997**.

[26] Steven M LaValle and James J Kuffner Jr. Randomized kinodynamic planning. *The international journal of robotics research*, 20(5):378–400, **2001**.

[27] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, **2011**.

[28] James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pages 995–1001. IEEE, **2000**.

[29] Mert Akinc, Kostas E Bekris, Brain Y Chen, Andrew M Ladd, Erion Plaku, and Lydia E Kavraki. Probabilistic roadmaps of trees for parallel computation of multiple query roadmaps. In *Robotics Research. The Eleventh International Symposium: With 303 Figures*, pages 80–89. Springer, **2005**.

[30] Nancy M Amato, O Burchan Bayazit, Lucia K Dale, Christopher Jones, and Daniel Vallejo. Choosing good distance metrics and local planners for probabilistic roadmap methods. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, volume 1, pages 630–637. IEEE, **1998**.

[31] Valérie Boor, Mark H Overmars, and A Frank Van Der Stappen. The gaussian sampling strategy for probabilistic roadmap planners. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 2, pages 1018–1023. IEEE, **1999**.

[32] Robert Bohlin and Lydia E Kavraki. Path planning using lazy prm. In *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, volume 1, pages 521–528. IEEE, **2000**.

[33] Michael S Branicky, Steven M LaValle, Kari Olson, and Libo Yang. Quasi-randomized path planning. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, volume 2, pages 1481–1487. IEEE, **2001**.

[34] Roland Geraerts and Mark H Overmars. A comparative study of probabilistic roadmap planners. In *Algorithmic foundations of robotics V*, pages 43–57. Springer, **2004**.

[35] Steven M LaValle, Michael S Branicky, and Stephen R Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *The International Journal of Robotics Research*, 23(7-8):673–692, **2004**.

[36] Fei Yan, Yan Zhuang, and Jizhong Xiao. 3d prm based real-time path planning for uav in complex environment. In *2012 IEEE international conference on robotics and biomimetics (ROBIO)*, pages 1135–1140. IEEE, **2012**.

[37] Kai Cao, Qian Cheng, Song Gao, Yangquan Chen, and Chaobo Chen. Improved prm for path planning in narrow passages. In *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, pages 45–50. IEEE, **2019**.

[38] Zhefan Xu, Di Deng, and Kenji Shimada. Autonomous uav exploration of dynamic environments via incremental sampling and probabilistic roadmap. *IEEE Robotics and Automation Letters*, 6(2):2729–2736, **2021**.

[39] Sertac Karaman, Matthew R Walter, Alejandro Perez, Emilio Frazzoli, and Seth Teller. Anytime motion planning using the rrt. In *2011 IEEE international conference on robotics and automation*, pages 1478–1483. IEEE, **2011**.

[40] Jonathan D Gammell, Siddhartha S Srinivasa, and Timothy D Barfoot. Informed rrt: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *2014 IEEE/RSJ international conference on intelligent robots and systems*, pages 2997–3004. IEEE, **2014**.

[41] Oren Salzman and Dan Halperin. Asymptotically near-optimal rrt for fast, high-quality motion planning. *IEEE Transactions on Robotics*, 32(3):473–483, **2016**.

[42] Iram Noreen, Amna Khan, and Zulfiqar Habib. Optimal path planning using rrt* based approaches: a survey and future directions. *International Journal of Advanced Computer Science and Applications*, 7(11), **2016**.

[43] Wang Xinyu, Li Xiaojuan, Guan Yong, Song Jiadong, and Wang Rui. Bidirectional potential guided rrt* for motion planning. *IEEE Access*, 7:95046–95057, **2019**.

[44] Jiankun Wang, Baopu Li, and Max Q-H Meng. Kinematic constrained bi-directional rrt with efficient branch pruning for robot path planning. *Expert Systems with Applications*, 170:114541, **2021**.

[45] Bin Liao, Fangyi Wan, Yi Hua, Ruirui Ma, Shenrui Zhu, and Xinlin Qing. F-rrt*: An improved path planning algorithm with improved initial solution and convergence rate. *Expert Systems with Applications*, 184:115457, **2021**.

[46] Denggui Zhang, Yong Xu, and Xingting Yao. An improved path planning algorithm for unmanned aerial vehicle based on rrt-connect. In *2018 37th Chinese control conference (CCC)*, pages 4854–4858. IEEE, **2018**.

[47] Jin-Gu Kang, Dong-Woo Lim, Yong-Sik Choi, Woo-Jin Jang, and Jin-Woo Jung. Improved rrt-connect algorithm based on triangular inequality for robot path planning. *Sensors*, 21(2):333, **2021**.

[48] Jiagui Chen, Yun Zhao, and Xing Xu. Improved rrt-connect based path planning algorithm for mobile robots. *IEEE Access*, 9:145988–145999, **2021**. doi:10.1109/ACCESS.2021.3123622.

[49] Sebastian Klemm, Jan Oberländer, Andreas Hermann, Arne Roennau, Thomas Schamm, J Marius Zollner, and Rüdiger Dillmann. Rrt*-connect: Faster, asymptotically optimal motion planning. In *2015 IEEE international conference on robotics and biomimetics (ROBIO)*, pages 1670–1677. IEEE, **2015**.

[50] Reza Mashayekhi, Mohd Yamani Idna Idris, Mohammad Hossein Anisi, Ismail Ahmedy, and Ihsan Ali. Informed rrt*-connect: An asymptotically optimal single-query path planning method. *IEEE Access*, 8:19842–19852, **2020**.

[51] Xiao Zhang, Tong Zhu, Lei Du, Yueqi Hu, and Haoxue Liu. Local path planning of autonomous vehicle based on an improved heuristic bi-rrt algorithm in dynamic obstacle avoidance environment. *Sensors*, 22(20):7968, **2022**.

[52] Chelsea Lau and Katie Byl. Smooth rrt-connect: An extension of rrt-connect for practical use in robots. In *2015 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, pages 1–7. IEEE, **2015**.

[53] RTCA (Firm). SC 167. *Software considerations in airborne systems and equipment certification*. RTCA, Incorporated, **1992**.

[54] Leanna Rierson. *Developing safety-critical software: a practical guide for aviation software and DO-178C compliance*. CRC Press, **2017**.

[55] I RTCA. Software considerations in airborne systems and equipment certification; rtca. *Inc.: Washington, DC, USA*, **2011**.

[56] Won Keun Youn, Seung Bum Hong, Kyung Ryoon Oh, and Oh Sung Ahn. Software certification of safety-critical avionic systems: Do-178c and its impacts. *IEEE Aerospace and Electronic Systems Magazine*, 30(4):4–13, **2015**.