ADVANCED PATH PLANNING FOR JET FIGHTER AIRCRAFTS UTILIZING DYNAMIC THREAT LEVEL DISCRIMINATION IN COMPLEX ENVIRONMENTS

KARMAŞIK ORTAMLARDA DİNAMİK TEHDİT SEVİYESİ AYRIMINA DAYALI JET SAVAŞ UÇAKLARI İÇİN GELİŞMİŞ YOL PLANLAMA

BENGİSU YÜCEL

PROF. DR. MEHMET ÖNDER EFE

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

Haziran 2026

ABSTRACT

ADVANCED PATH PLANNING FOR JET FIGHTER AIRCRAFTS UTILIZING DYNAMIC THREAT LEVEL DISCRIMINATION IN COMPLEX ENVIRONMENTS

Bengisu Yücel

Master of Science, Computer Engineering

Supervisor: Prof. Dr. Mehmet Önder EFE

Mayıs 2026, 115 pages

In today's battlefields, the widespread use of advanced Integrated Air Defense Systems (IADS) means that fighter jets have to fly through threat profiles that are very dynamic and complicated. Conventional path-planning techniques often find it challenging to reconcile near-real-time geometric optimality with stringent aircraft kinodynamic limitations, including minimum turn radii. To address these constraints, this thesis introduces a cohesive, neuro-adaptive tactical mission-planning framework, rigorously assessed using Digital Elevation Model (DEM) derived real terrain.

The proposed architecture smoothly combines layers for perception, decision-making, and action into a single operational pipeline. A Deep Neural Network Tactical Risk Estimator (DNN-TRE) serves as a surrogate model in the perception layer. The DNN-TRE gets up to a **3.4**$\times$ speedup in calculations by skipping physics-based geometric ray-tracing, while still keeping a strong spatial correlation of **0.9345** with the baseline risk topology, even when there is no prior knowledge in dynamic environments. The action layer uses one of three built-in kinodynamic planners to carry out the trajectory: the Kinematic Guidance

Navigation Planner (K-GNP), which makes sure that the minimum turn radius is possible; the Tactical Guidance Navigation Planner (T-GnP), which lowers risk even more by using threat-aware and smoothness-based costs; and a highly reactive Dueling Deep Q-Network (DQN) based RL-Pilot.

To make sure that the evaluation was thorough, all of the proposed planners and classical baselines (A^*, Dijkstra, RRT*, and PSO) were put into a single Feasibility Engine. The system was tested using Monte Carlo simulations that were affected by random wind changes in four different scenarios that got more and more complicated. The experimental results show that the suggested neuro-adaptive framework guarantees a **100% (30/30)** mission success rate in all settings, even in highly dynamic threat corridors (S4) where traditional stochastic planners like RRT* had terrible failure rates (dropping to 16.6% success). The RL-Pilot also had great kinodynamic stability, keeping tracking errors as low as **21.53 meters** and always getting rid of control saturation. Ultimately, this thesis connects the worlds of artificial intelligence and aerodynamics to create a very strong, real-time tactical navigation system for modern combat aircraft.

Keywords: Kinodynamic Path Planning, Fixed-Wing Aircraft, Neuro-Adaptive Routing, Deep Neural Networks, Reinforcement Learning, Dynamic Threats, Tactical Trajectory Optimization.

ÖZET

KARMAŞIK ORTAMLARDA DİNAMİK TEHDİT SEVİYESİ AYRIMINA DAYALI JET SAVAŞ UÇAKLARI İÇİN GELİŞMİŞ YOL PLANLAMA

Bengisu Yücel

Yüksek Lisans, Bilgisayar Mühendisliği

Danışman: Prof. Dr. Mehmet Önder EFE

Mayıs 2026, 115 sayfa

Modern muharebe ortamlarında, gelişmiş Entegre Hava Savunma Sistemlerinin (IADS) yaygınlaşması, savaş uçaklarının oldukça dinamik ve karmaşık tehdit profilleri arasında güvenle uçabilmesini zorunlu kılmaktadır. Geleneksel yol planlama yöntemleri, gerçek zamanlı geometrik optimallik ile uçakların minimum dönüş yarıçapı gibi katı kinodinamik kısıtlarını dengelemekte sıklıkla yetersiz kalmaktadır. Bu sınırlamaları aşmak amacıyla bu tez kapsamında, Sayısal Yükseklik Modeli (DEM) tabanlı gerçek araziler üzerinde sistematik olarak test edilen, birleşik ve nöro-adaptif bir taktiksel görev planlama mimarisi önerilmiştir.

Önerilen mimari; algı, karar ve eylem katmanlarını kesintisiz bir operasyonel işlem hattında birleştirmektedir. Algı katmanında, Derin Sinir Ağı tabanlı bir Taktiksel Risk Tahmincisi (DNN-TRE), bir vekil model olarak görev yapmaktadır. DNN-TRE, hesaplama maliyeti yüksek fizik tabanlı geometrik ışın izleme yöntemlerini devre dışı bırakarak taktiksel risk haritalamada **3.4 kata** kadar hızlanma sağlamakta ve eğitim setinde bulunmayan hareketli tehdit ortamlarında dahi temel risk topolojisi ile **0.9345** oranında güçlü bir uzamsal korelasyon sürdürmektedir. Eylem katmanı ise yörüngeyi üç yerleşik kinodinamik

planlayıcıdan biriyle yürütmektedir: Minimum dönüş yarıçapını zorunlu kılan Kinematik Güdüm Navigasyon Planlayıcısı (K-GNP), pürüzsüzlük ve tehdit farkındalığı ile riski daha da azaltan hız-uyarlamalı Taktiksel Güdüm Navigasyon Planlayıcısı (T-GnP) ve yüksek reaktif kapasiteye sahip Dueling Deep Q-Network (DQN) tabanlı RL-Pilot.

Adil bir değerlendirme sağlamak için, önerilen tüm planlayıcılar ve klasik referans algoritmalar (A^*, Dijkstra, RRT* ve PSO) ortak bir Fizibilite Motoruna entegre edilmiştir. Geliştirilen sistem, stokastik rüzgar bozucularını barındıran ve zorluk derecesi giderek artan dört farklı senaryoda Monte Carlo simülasyonları ile test edilmiştir. Deneysel sonuçlar, önerilen nöro-adaptif mimarinin, RRT* gibi klasik stokastik planlayıcıların başarı oranının %16.6'ya kadar çakıldığı zorlu hareketli tehdit senaryolarında (S4) bile **%100 (30/30)** görev başarısı sağladığını kanıtlamaktadır. Ayrıca, RL-Pilot modülü izleme hatasını (tracking error) **21.53 metreye** kadar düşürerek rüzgarlı ortamlarda kontrol doygunluğunu (saturation) ortadan kaldırmış ve mükemmel bir kinodinamik stabilite sunmuştur. Sonuç olarak bu çalışma, aerodinamik fizibilite ile yapay zeka arasındaki boşluğu doldurarak modern savaş uçakları için gürbüz ve gerçek zamanlı bir taktiksel navigasyon çözümü sunmaktadır.

Anahtar Kelimeler: Kinodinamik Yol Planlama, Sabit Kanatlı Uçak, Nöro-Adaptif Rotalama, Derin Sinir Ağları, Pekiştirmeli Öğrenme, Dinamik Tehditler, Taktiksel Yörünge Optimizasyonu.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my parents, Naringül and Hakan YÜCEL, for their unwavering support and belief in me throughout this journey. Their tireless encouragement, not only during the writing of this thesis but throughout my entire life, has been the foundation of my success.

CONTENTS

Page

TABLES

FIGURES

ABBREVIATIONS

AGL	:	**A**bove **G**round **L**evel
APF	:	**A**rtificial **P**otential **F**ield
CGF	:	**C**omputer-**G**enerated **F**orces
CTE	:	**C**ross-**T**rack **E**rror
DDQN	:	**D**ouble **D**eep **Q**-**N**etwork
DEM	:	**D**igital **E**levation **M**odel
DNN	:	**D**eep **N**eural **N**etwork
DNN-TRE	:	**D**eep **N**eural **N**etwork-**T**actical **R**isk **E**stimator
DQN	:	**D**eep **Q**-**N**etwork
DRL	:	**D**eep **R**einforcement **L**earning
EW	:	**E**lectronic **W**arfare
GLOS	:	**G**eometric **L**ine-**o**f-**S**ight
HUD	:	**H**eads-**U**p **D**isplay
IADS	:	**I**ntegrated **A**ir **D**efense **S**ystem
K-GNP	:	**K**inematic **G**uidance **N**avigation **P**lanner
MAE	:	**M**ean **A**bsolute **E**rror
MCDA	:	**M**ulti-**C**riteria **D**ecision **A**nalysis
MLP	:	**M**ulti-**L**ayer **P**erceptron
MSL	:	**M**ean **S**ea **L**evel
PDA	:	**P**erception-**D**ecision-**A**ction
PDF	:	**P**ortable **D**ocument **F**ormat
PNG	:	**P**ortable **N**etwork **G**raphics
PPO	:	**P**roximal **P**olicy **O**ptimization
PSO	:	**P**article **S**warm **O**ptimization
QoS	:	**Q**uality-**o**f-**S**ervice
RL	:	**R**einforcement **L**earning

RMSE	:	**R**oot **M**ean **S**quare **E**rror
RRT	:	**R**apidly-**e**xploring **R**andom **T**ree
RRT*	:	**R**apidly-**e**xploring **R**andom **T**ree **S**tar
SAM	:	**S**urface-**t**o-**A**ir **M**issile
T-GnP	:	**T**actical **G**uidance **N**avigation **P**lanner
UAS	:	**U**nmanned **A**erial **S**ystem
UAV	:	**U**nmanned **A**erial **V**ehicle
UTM	:	**U**niversal **T**ransverse **M**ercator

1. INTRODUCTION

In todays world, the air missions done with mission planning that's why the route is planned at the pre-flight phase. Also, in simulators world the waypoints and the mission is predefined with threats. However, in real world folowing the waypoints is not sufficient. In practice, the aircraft must follow the route within its physical limits while face the real world noise such as wind disturbances or unknown last minute suprises. Aircraft-specific limitations, such as minimum turn radius, bank angle restrictions, and heading angle following are not checked. Directly using the navigation algorithms maybe give the great results in mathematical model in a simulated environment it might be impossible to follow for a real fighter jet. This paper states the incoherence between the kinematic constraints and geometric calculations and gives a solution planners for build with these two major works. By interpreting the tactical situation according to the defined environment and the intensity of threats based on environmental factors, the planner optimizes the route. This adaptable planning during the proposed flight, based on these systems, enables both pilot training and the rational route creation of virtual aircraft, referred to as computer-generated forces, resulting in a high-level decision-making process. This project combined kinematic constraints to make this system. It used threat-aware risk modeled on terrain data and cost. Learning the threat awareness risk model during the flight phase with an integrated neural network module shortens the time it takes to make a prediction. This is because the module gives quick results without losing important information by threat aware terrain model. To propose a novel study in this research, reduced-order fixed-wing aircraft and circular threats are modelled, done with terrain-aware and threat-aware risk estimation is calculated, kinematically feasible trajectory generation applied to the aircraft model, the risk estimation based on neural network is applied and evaluated an adaptive planner selection mechanisms finally to compare with traditional path optimization methods are compared with proposed planning methods those are .

At the core of this framework is the Kinematic Guidance Navigation Planner (K-GNP), which ensures that the generated trajectories are truly flyable by considering heading and

curvature constraints during the search process. This foundation is extended by the Tactical Guidance Navigation Planner (T-GnP), which introduces maneuver-aware costs and adaptive speed logic to minimize threat exposure. Unlike conventional methods that treat terrain, threats, and maneuverability as separate problems, this study integrates them into a unified process. To handle different mission contexts, a reinforcement learning-based advisor selects the most suitable planning strategy, supported by a Deep Neural Network-based Tactical Risk Estimator (DNN-TRE), which computes threat levels much faster than geometric methods. Together, these components form a complete perception-decision-action architecture, validated through various simulations that measure success rates and tracking accuracy. Thesis paper begins with the theoretical background in Chapter 1, Chapter 2 provides a detailed look at the methodology, explaining the K-GNP and T-GnP algorithms, the neural risk module, and the advisor mechanism. The performance of the integrated system is analyzed in Chapter 3 through multiple benchmark scenarios of increasing complexity. Finally, Chapter 4 summarizes the research and describes the future work.

2. BACKGROUND OVERVIEW

The project background states the fundamental of the project framework. The formulation of the cost functions, aircraft and threat model basics, terrain model information, for artificial intelligence parts the basic information about the trained model and overall framework of decision making methodologies, explaining and reducing the question marks on technical details during the methodology and results parts. The goal is to show how these parts work together in a realistic tactical planning environment.

2.1. Threat Modelling, Aircraft Kinodynamics and Control Constraints

In this study, aircraft motion is modeled using a 3 DoF with fixed-wing point-mass representation. When compared to six-degree-of-freedom models it reduced the computation

cost with balance between physical accuracy and compuational efficiency. The main purpose of this paper tried to find optimal trajectory with given risky terrain the aircraft model evulation is planned for future work.

The aircraft state is defined as

$$\mathbf{x} = [x,\ y,\ h,\ \psi]^T \tag{1}$$

where x and y denote horizontal position, h represents altitude, and ψ corresponds to the heading angle. The control input vector is expressed as

$$\mathbf{u} = [T,\ \phi,\ \gamma]^T \tag{2}$$

with T representing thrust-related control, ϕ the bank angle, and γ the flight-path angle. Assuming coordinated-turn motion, the kinematic relations can be written as

$$\dot{\psi} = \frac{g \tan \phi}{V}, \quad \dot{h} = V \sin \gamma, \quad \dot{x} = V \cos \gamma \cos \psi, \quad \dot{y} = V \cos \gamma \sin \psi \tag{3}$$

where V is the airspeed and g is the gravitational acceleration. The longitudinal speed dynamics are approximated as

$$\dot{V} = \frac{T - D}{m} - g \sin \gamma \tag{4}$$

The A/C model fits to necessary behvaior to defined trajectory feasibility and evaluate performance value in scenarios. In the pre-defined flight, the load factor changes upon the bank angle with respect to,

$$n = \frac{1}{\cos \phi} \tag{5}$$

This means that a larger bank angle improves maneuverability, but it also increases the aerodynamic load. As a result, the smallest possible turning radius becomes a key limiting factor:

$$R_{\min} = \frac{V^2}{g \tan \phi_{\max}} \tag{6}$$

That limitations states the bounded curvature behavior of A/C model. In real world, it bannes the sharpness of the aircraft's turns and effects the flight trajectory. The generated trajectory, follows the guidance plan, which ψ_{des} denote the desired heading toward a A/C heading point.

The corresponding tracking error is defined as $e_\psi = \psi_{\text{des}} - \psi$. Similarly, altitude tracking is governed by $e_h = h_{\text{des}} - h$. These errors are used to generate control commands. The algorithms might flight with instant turns without looking dynamic conditions. To provide that feasible flight that check minimum turn radii such as Dubins-like arcs there will be discontinuities for smoothing using bounded curvature.

2.2. Operational Environment and Threat Modeling

The terrain is shown using Digital Elevation Model (DEM) To ensure safe flight, the aircraft must maintain a minimum clearance above terrain:

$$h(x, y) \geq h_{\text{terrain}}(x, y) + h_{\text{margin}} \tag{7}$$

In addition to terrain values, threats are modeled as spatially distributed cost fields. The Static threats are represented as,

$$J_{\text{threat}} = J_{\text{threat}}(x, y) \tag{8}$$

while dynamic threats introduce time dependency:

$$J_{\text{threat}} = J_{\text{threat}}(x, y, t) \tag{9}$$

The fields are differences with constraint types (hard and soft). The hard risk zones called as prohibited regions. For moving threats, the planner algorithm look instantenous position, motion pattern and its physical constraints. Threat models parameters are risk zones defined by their spatial center with radii, threat type and threat danger coefficient. Each threat is represented by a tuple consisting of (x_j, y_j), a hard-kill radius r_j, a threat type label, and a

scalar threat level ℓ_j. Four threat classes is used by scenarios. *SAM*, *SAM_Big*, *Radar*, and *EW*. SAM-type threats states the baseline high-risk category, SAM_Big defines as to a larger and more dense air-defense zone, While radar threats create wider but less risky areas of effect, electronic warfare threats are modeled with lower tactical weight. The scalar threat level ℓ_j further scales the threat cost and is varied across scenarios in order to represent changing tactical severity.

For static threats, the risk is modeled as a radial cost area outside the hard radius and becomes impossible inside the hard zone. For dynamic threats, two terms are activated: the predictive directional cost, aligned with the threat's motion vector, and the patrol corridor cost, distributed along the motion path. Threats are composed of three components: instantaneous radial risk, forward-looking predictive exposure, and dynamic movement corridor penalty.

2.3. Tactical Path Planning Methodologies

The path planning methodology could be consider as a multi-objective optimization process. A demanded trajectory τ is evaluated using a composite cost function:

$$J(\tau) = J_{\text{path}} + \lambda_1 J_{\text{risk}} + \lambda_2 J_{\text{terrain}} + \lambda_3 J_{\text{maneuver}} \tag{10}$$

Thes path efficiency, threat motion, collision avoidance with terrain, and maneuver feasibility.

2.4. Neuro-Adaptive Framework Architecture

In the Neuro-Adaptive Framework which is thesis another contribution, the architecture provides a full covered tactical navigation system completes with layered perception-decision- action process. To solve this problem, a proxy model is used to estimate the risk function:

$$\hat{J}\text{risk} = f\theta(\mathbf{z}) \tag{11}$$

where $\mathbf{z}$ represents a feature vector describing the local environment, and f_θ is a neural network. This approximation allows the system to process the structure of the threat field by reducing computation time. At the decision level, planner selection is formulated as a policy:

$$\pi_{\text{advisor}}(s) \rightarrow \text{RL-Pilot},\ \text{K-GNP},\ \text{T-GnP} \tag{12}$$

where s encodes the mission state. Based on this information, the system selects the most appropriate planning strategy for the given scenario. To evaluate the efficiency of the proposed framework. Path length is defined as,

$$L = \sum_{i=1}^{N-1} |\mathbf{p}i + 1 - \mathbf{p}i| \tag{13}$$

The cumulative threat exposure along the trajectory is given by

$$R_{\text{exp}} = \int J_{\text{threat}}(x(t), y(t), t), dt \tag{14}$$

Furthermore, the metrics are also, success rate, flight duration, minimum terrain clearance, tracking error, the number of violations, and computation time.

3. RELATED WORK

For a long time, researchers have used deterministic and sampling-based techniques to solve path planning problems for fixed-wing aircraft and autonomous systems. This section reviews various trajectory optimization and planning algorithms by grouping the relevant studies based on their approach. Generally, the existing literature can be divided into four main categories: deterministic and exact planning, sampling-based geometric methods, heuristic and learning-based models, and research focusing on flight dynamics and tactical constraints

3.1. Deterministic and Exact Planning Methods

3.1.1. Grid-Based Search Algorithms (A*, D* and Variants)

Grid based algorithms are very fast and efficient on regular grid maps generally A* and Dijkstra have been used in many years. To increase the performance of this algorithms on many applications and reduce costs, researches benefit adaptive neighborhoods, heuristics and post processing. An enhanced neighborhood A* that modifies the search radius and step size based on obstacle density suggested from Xu et. Al. [1]. This paper demonstrates, The purposed methodology used bi directional search utilize Airtifical potential fields with smoothing approaches. While MSF MTPO achieve minimum costs on complex 3D environments, it has limited because of grid resolution and require post processing to validate flight feasibility. Yin et al. [2] purpose A* and DWA fusion method for cars. It used for collapse avoidance however fine tuning on DWA parameters is significant for sharp turns. A study on multi-UAV systems by [3] demonstrates that traditional A* produces shorter and smoother trajectories, whereas modified variety improve tolerance in saturated conditions. In complex scenarios, , Chen et al.[4] find that metaheuristic methods, such as cultural algorithms, can outperform grid planners like D* in real-time replanning and path length efficiency. This indicates that pure grid search methods are ineffective in high-dimensional or rapidly change for air platforms. The significant other research by Lei et al. [5] and Jun et al. [6] focused on basic method of avoidance of obstacles using A* or dynamic programming. [7] and [8] included low profile criteria, such as radar cross-section and access limitations, into the A* cost function for current tactical scenarios.

The applying these algorithms to unmanned surface vehicles is working by Zhang et al.[9] Focused on coverage planning, Lei ([10]) has successfully demonstrated that the mathematical proof of global optimality—these A*-based models consistently identify the shortest or safest path on a map.

The high computational cost in shifting circumstances is a major disadvantage. New obstacles usually need recomputation on the grid matrix. [11] Chen et al. introduced,

Focused D method its helps to reduce computation time because of focusing focal area that near the defined obstacles or threats. Other method by Chen et. al., [12] states that Dijkstra models worked for static and dynamic obstacles simulatenously to find a path.

3.2. MILP and Nonlinear Mathematical Optimization

MILP established as a efficient technique on collision avoidance. Richards et al. [13] demonstrated that linear inequalities combined with discrete variables can effectively used for guidance on trajectory for aircrafts. Researchers advanced this technology by creating ”fast-dynamic” MILP formulations specifically designed for UAV formations, allowing flexible control over changing challenges during flight. Further studies by Erokhin et al. [14] and Jafarimoghaddam Soler [15] grew these methodologies including path length versus fuel efficiency, as part with the complex control problems related to vertical flying trajectories.

Turker et al. [16] demonstrated that Simulated Annealing can efficiently Turker et al. [16] demonstrated that Simulated Annealing can efficiently identify near-optimal multi-target trajectories, and they additionally included tactical escape logic into the planner by establishing a guard band surrounding radar coverage areas and redirecting vehicles through secure virtual waypoints. This combination gives solution for quick and secure 2D tactical navigation. In settings with many moving agents, Ran et al. [17] introduced a hybrid architecture that integrates trajectory prediction with a Priority-Aware Dynamic Window Approach, facilitating collision-free navigation without the need for centralized coordination and Hoang et al. [18] utilized Teaching-Learning-Based Optimization for three-dimensional multicopter path planning, employing population diversity to evade local optima in complicated terrain. Real-time responsiveness necessitates a distinct focus. Bashir et al. [19] investigated local reactive techniques for obstacle evasion, whereas Peng et al. [20] proposed a rolling three-dimensional path planning methodology influenced by fluid dynamics to produce smoother, more continuous paths. Niu et al. [21] subsequently integrated a fluid-dynamical system with an Artificial Neural Network to attain actual real-time performance in dynamic situations. When the mission involves not one vehicle

but a coordinated swarm, the problem changes fundamentally. Distributed multi-UAV systems must simultaneously avoid inter-agent collisions, maintain communication, and synchronize decisions — constraints that require planner architectures quite distinct from their single-vehicle counterparts. Fan et al. [22] investigated swarm path planning under diverse environmental stressors, while Duan et al. [23] developed a distributed cooperative model designed to remain effective in rapidly evolving, information-rich settings. Local replanning has remained a persistent concern throughout this body of work. Xiao [24] proposed a local path replanning technique for UCAVs operating in uncertain terrain on dynamic threat environments. Probabilistic threat modeling has added further rigor to this challenge: Ji et al. [25] demonstrated how formal mathematical frameworks can handle uncertainty arising from real-world threat sources in aircraft trajectory planning. The stealth dimension of this problem — designing routes that minimize radar detectability — was addressed early by Ling and Xiao [26], whose penetration path design methods laid groundwork that later studies continued to refine.

3.3. Sampling-Based and Geometric Approaches

3.3.1. Incremental Sampling Algorithms (RRT and Advanced Variants)

With considering the multi agent coordination on path trajectory gives another constraints, [27] proposed a new approach for multi-UAV system with integration of Grey Wolf Optimization (GWO) with the RRT algorithm. Within this search, GWO focus the swarm's global coordination, which ensures that the drones reduce total path length and avoid specified threat areas; simultaneously, RRT is utilized for local path verification. Another search by [28] uses the fusion of the APF-IRRT (Artificial Potential Field - Improved RRT*) at single UAVs regardless of complex disaster scenarios.

This method is effective because it dynamically reduces the search step size as it gets closer to obstacles and guides the search direction toward the target.By the integration using APF logic, search nodes are drawn towards the objective by a ”attractive force” and driven from threats by a ”repulsive force.” When this is combined with the Dynamic Window Approach

(DWA), the system can handle both fixed and moving obstacles. This creates a hybrid architecture that links high-level global planning with fast local avoidance. For suitable to flight cubic spline interpolation is used depending on the UAV's maximum turning radius. During the flight, the Dynamic Window Approach (DWA) constantly monitors the aircraft's speed and direction. This allows for real-time adjustments to the control inputs. Because these algorithms, sampling-based methods, like RRT and its variations, do not rely on fixed grids, they can explore complex 3D environments much faster than many traditional techniques described in [27]. Since, these methods are naturally random, that doing the same thing twice can lead to two quite different routes, making the execution time and path quality hard to predict [28, 29].

Another problem is the quality of the first routes. This is because global sampling methods sometimes place points randomly across a given area, which can make the paths look sharp and irregular. To solve these issues, RRT* employ a tree-based structure that facilitates dynamic replanning referenced on Diao's paper.[28]. Another work by Wu et. al., [29], suggests a hybrid approach that combines global planning with real-time local avoidance. Combine with the Informed-RRT* method with Artificial Potential Fields (APF), it guides the search toward the goal and reduce the number of redundant waypoints and create a flyable final path, cubic spline interpolation to ensure the route within the UAV's maximum turning radius.

The introduction of RRT* was a big step forward because it eliminated the problems with randomness in traditional RRT by giving it asymptotic optimality, even in very dangerous areas [30]. In faster search on complicated areas, [31] created a bidirectional method that works from both the starting point and the goal point at the same time. This reasoning was later modified for fixed-wing aircraft flying at very low altitudes using specific search algorithms [32]. Standard RRT often does not work well when there are moving obstacles, so novel models like ADRRT* and HPO-RRT* were suggested for dynamic tree restoration [33, 34]. More recent research has gone much farther, including immediate tree updates and applying dynamic sensors for real-time avoidance [35, 36]. These methods can

also be utilized for coordinating several UAVs or swarms, where multi-goal and conflict search-based RRTs are employed to handle communication and task sharing. [37, 38].

Chen et. al. suggests ADRRT* [33], it is the ability to alter only the nodes that are affected in microseconds rather than having to recalculate the whole path when something gets in the way. But this efficiency comes at the expense of natural randomness. The method uses random sampling, thus it makes a distinct path every time it runs for the same task.

3.3.2. Geometric Models, Curve Fitting and Flight Kinematics

Using the Voronoi Diagrams to set limits that make sure planes stay in the middle of obstacles [39, 40]. Mathematical model based on geometry instead of only algorithmic searches is one way to push unmanned aerial vehicles (UAVs) to their aerodynamic limitations. Visibility graphs used for searching in three-dimensional cities by connecting building corners and other obstacles [41]. Moreover, research has investigated limited shortest path networks and analytical linear optimization for addressing complext points in specialized military operations referenced on [42–44].

Methods that use tangents and the Traveling Salesman Problem (TSP) help ensure that the path closely follows the edges of real-world obstacles by produce flyable routes [45, 46]. To enhance this, techniques such as Improved Bézier curves or Dubins polygon modeling are employed to integrate the UAV's turning radius and bank angle constraints directly into the trajectory [47, 48]. Some models can even work with urban risk maps with millimeter-level accuracy [49].

However, there is a significant practical challenge for Dynamics updated like a new threat appearance in that case the entire Voronoi or Visibility network must be rebuilt. This takes a lot more computing power than simple search algorithms like A* or RRT, which can make real-time response slower. To fix this, hybrid frameworks like the one in [50] use Voronoi roadmaps along with Dijkstra and Particle Swarm Optimization (PSO). The Voronoi diagram acts as a safe ”skeleton” in this design. In [50] Shin et. al., integrating

Voronoi diagrams, Dijkstra's algorithm, and particle swarm optimization (PSO). First, a Voronoi-based roadmap and graph search create a preliminary path that is clear of obstacles. After that, PSO improves this path, and then a three-dimensional smoothing phase makes the trajectory more continuous. Voronoi structures made a safe corridor, while the trajectory is getting smoother. On the other hand, this technique fit in static situations and does not strictly enforce kinematic limits, like a minimum turning radius. Yin et. al. [2] combines a kinematically constrained A* method with B-spline curve fitting to generate pathways that are smooth and continuous. The B-spline form naturally guarantees higher-order continuity so it easier to follow by reducing sudden changes in that direction.

Despite these advantages, spline-based methodologies do not invariably guarantee obstacle avoidance or strict adherence to curvature constraints, particularly within intricate environments.

Polynomial curve fitting has also been used to improve planners that use sampling. The initial path is made using an improved Informed-RRT* algorithm in [29]. Then, polynomial fitting is used to smooth it out. This method improves the trajectory's smoothness, which is beneficial for control algorithms like the Dynamic Window Approach (DWA). However, using polynomials can cause oscillations and violate obstacle avoidance rules if not carefully controlled.

Hybrid global-local frameworks, which integrate discrete optimization with geometric reasoning, have also been the subject of research. Debnath et al. (2024) employed a genetic algorithm for global mission-level planning, while a geometric obstacle avoidance module, QuickNav, was utilized to implement real-time local adjustments. This layered method does a good job of keeping combinatorial optimization and continuous geometric reasoning apart. However, the method does not directly simulate the dynamics of aircraft, which means it can only be used on high-speed fixed-wing platforms. More recent work has tried to add geometric primitives that better fit kinematic constraints. In [1], smoothing based on tangent circles is used to cut down on inflection points in paths made by A*. The method simulates Dubins-like motion by adding circular arcs between straight

segments. This approach makes the paths more consistent with the limits of minimum turning radius. While this method improves smoothness and ease of implementation, it's still a post-processing step and doesn't guarantee the best possible solution when considering movement restrictions. In all of the studies that were looked at, there is a common architectural pattern: (i) a global planner makes a discrete or sampling-based path, (ii) a geometric or curve-fitting method is used to smooth the path, and (iii) a local planner takes care of avoiding moving obstacles. This pipeline works well in a lot of situations, but it usually only considers kinematic feasibility after the fact, not as a main constraint when planning. Because of this, most current methods don't make sure that the trajectories they create can be flown under strict aircraft dynamics, especially for high-speed fixed-wing platforms. Also, dynamic threat environments are often simplified or modeled in a limited way, which makes the planning problem less realistic. Conversely, this thesis mitigates these limitations by integrating kinematic constraints directly into the planning process and assessing performance under dynamic threat conditions. The proposed framework aims to combine geometric optimization with physically possible paths in real-world situations. This is achieved by merging planning that considers feasibility with cost modeling that accounts for risk. The literature reviewed indicates that grid-based methods ([7]) provide complete safety and target assurance but are slow; sampling-based methods (RRT) ([34]) are rapid yet yield imprecise routes; and geometric models ([47]) are smooth but constrained by static limitations. This situation makes it clear that future studies on UAV path planning will have to use either DRL or hybrid methods. The current thesis is based on two theoretical frameworks: reduced-order aircraft simulation and coordinated-turn flight mechanics, which serve as the foundation for the fixed-wing execution model [51].

3.4. Heuristic and Learning-Based Models

3.4.1. Metaheuristic and Swarm Intelligence-Based Path Planning

Path planning for Unmanned Aerial Vehicles (UAVs) in complex three-dimensional environments represents a fundamentally high-dimensional and multi-constrained

optimization problem. Dynamic threats, shifting terrain, and stringent kinematic constraints significantly complicate the search space, rendering the problem NP-Hard and challenging for real-time applications [3, 52, 53]. Conventional deterministic and graph-based methodologies often face scalability issues and fail to produce physically feasible trajectories under realistic constraints. There are a lot of problems with traditional methods. Khatib et al. [54] employed Artificial Potential Field (APF) techniques to represent the UAV as a particle influenced by attractive and repulsive forces. APF works well on computers, but it often gets stuck in local minima, especially in tight or messy spaces. This means it can't find solutions that are both useful and complete [55, 56]. Rapidly-exploring Random Tree (RRT) methods search the space by taking random samples, and they work well in areas with a lot of dimensions [31]. But they make paths that aren't smooth and have extra nodes that need to be smoothed out, which could break kinematic rules [30, 32]. Graph-based algorithms, such as A*, also see their computational costs rise exponentially as resolution increases. This makes them not very scalable in large 3D environments [9]. These methodologies also struggle with incorporating continuous kinematic constraints, often resulting in mathematically optimal but physically infeasible trajectories [48]. To mitigate these limitations, researchers have progressively utilized metaheuristic algorithms and swarm intelligence frameworks, which produce near-optimal solutions within a pragmatic computational duration [57, 58]. In [58], a multimodal adaptive Pigeon-Inspired Optimization (PIO) method that is triggered by events was developed. The best thing about swarm-based and evolutionary optimizers is that they can optimize things all over the world. But they need a lot more computing power than local model-based methods, which makes them better for global planning that doesn't need to be done right away. These methods, which are based on how people and animals act in nature and society, are great for optimization problems that are nonlinear, non-convex, and have a lot of dimensions [59, 60]. Particle Swarm Optimization (PSO) is one of the most popular methods because it is easy to use, works well, and gets results quickly [61, 62]. In PSO, particles move around based on their own experiences (p_{best}) and the knowledge of the whole group (g_{best}). This is called a population-based search method. This lets you quickly look through and use the search space. However, conventional PSO encounters premature convergence and

stagnation, particularly in complex environments marked by constricted passages and highly non-convex cost landscapes [63, 64]. To address these limitations, several enhanced PSO variants have been proposed. For example, spherical vector-based PSO improves global search [63], and adaptive evolutionary PSO makes convergence more stable [64]. When hybrid methods use chaos theory or local planners, the quality of the solutions and the variety of the solutions are even better.

The Co-evolutionary Multi-group Particle Swarm Optimization (CMPSO) algorithm [52] represents a substantial advancement in this domain. CMPSO employs an adaptive mutation strategy contingent upon the activity level of particles, quantified by their velocity. The population is divided into groups based on how active they are, and each group has its own way of mutating. CMPSO makes the population much more diverse by selectively mutating particles that aren't moving using global best information (G_{best}), randomly chosen individuals, and adaptive mutation factors. This mechanism helps with exploration in the early stages of optimization and keeps particles from getting stuck in local optima.

Even with these changes, metaheuristic algorithms still have some problems that can't be fixed. Many methods improve optimization and make it more flexible, but they don't always make sure that kinematic and dynamic constraints are followed when planning. Instead, people often use post-processing techniques like smoothing or curve fitting [65, 66] to make sure that something can be done. Consequently, a significant disparity persists between numerically optimal solutions and physically achievable trajectories, particularly for high-speed fixed-wing UAVs constrained by stringent flight dynamics [48, 53]. This observation highlights a critical research challenge: existing methodologies treat kinematic feasibility as a secondary consideration instead of integrating it directly into the optimization framework. As a result, there is a clear need for advanced planning methods that naturally take into account physical limits, making it possible to create truly flyable and operationally resilient paths in complex and changing environments.

Biomimetic inspirations are important in recent research. For instance, [18] talks about the Denavit-Hartenberg parameterization and the Navigation variable-based MOPSO

(NMOPSO) method. We modeled the flight path as a flexible kinematic chain made up of robotic manipulators, and the navigation variables directly limited the search process. Furthermore, [67] clarifies Fitness-Distance Balance (FDB) and Dynamic Reorganization, utilizing a topology in the MSCPSO model that relies on FDB and a non-uniform mutation structure in which the leader particles are continuously reorganized with adaptive nonlinear inertial weights.

In connection with Guan's paper, MSCPSO employs the FDB strategy for the determination of learning objectives. For each candidate particle, FDB computes a complete score (FDBscore) by normalizing and weighting the particle's fitness value in relation to its Euclidean distance from G_{best}. The particle with the highest score is selected as FDB_{best}. The distance factor is given more weight in the first few versions to encourage broad exploration and increase diversity. As iterations go on, the fitness weighting goes up. This makes FDB_{best} closer to G_{best} and makes it easier to quickly find the global optimum (exploitation). Then, with a certain chance (p_f), followers learn from FDB_{best} instead of G_{best}.

Non-uniform mutation strategy is used to avoid the problem of leader particles getting trapped into local optima. With the use of the mutation technique, there is movement of the search process in small localized steps based on the current positions of the leaders. There is a gradual reduction in the mutation probability (pm) in a linear manner with each successive iteration. As such, there is enhanced exploration at the start of the process and gradual exploitation after that. Unlike in the normal PSO, all swarm particles have common inertia weightings. However, in the MSCPSO technique, inertia weightings depend on the function of the particles, with high inertia assigned to leaders for explorative purposes, while the followers have low inertia to conduct exploitation.

Particles in the population are organized into leaders. Should the average distance of the particles within a neighborhood relative to their Lbest be less than $D_{threshold}$ value, then there would be indication of excess clustering. Thus, shuffling of neighborhoods will occur to promote diversity. There is adjustment of leader and follower ratio in such a way that there

are more leaders than followers in the early iterations, but as time goes, more followers are generated.

The algorithm finds the particle that is doing the worst and replaces it with a new one during the last 40% of the iterations. The best positions of two randomly chosen elite particles from earlier steps are used to make this new particle. This process helps narrow down the population so that too much diversity doesn't slow down the overall convergence.Researchers in the field of hybrid swarm intelligence have combined different strategies to make things work better. For example, Gupta et al. [68] created the HCPSOA algorithm by combining the local search features of the Coyote Optimization Algorithm (COA) with the global search features of the Particle Swarm Optimization (PSO) algorithm. In a similar way, another study [69] combined MSCSO with Lévy flight operators, using a Metropolis acceptance criterion and a Simulated Annealing (SA) mechanism to improve the exploitation phase.A lot of these ideas come from biological metaphors, especially how fish school and birds flock together. PSO, which is still one of the most popular ways to plan routes for UAVs and Autonomous Underwater Vehicles (AUVs), works by sharing personal and social experiences. Each particle is a possible solution, and it finds its path by balancing its own personal best (p_{best}) with the global best (g_{best}) that the whole group has found.When these algorithms are used in underwater settings, physical factors like ocean currents play a big role. As pointed out by [70], AUVs' hardware limits mean that unpredictable currents and eddies can waste energy and push the vehicle off course. The Navier-Stokes equations are used to model ocean currents in three dimensions to make simulations more realistic. This mathematical method shows how the fluid's speed, pressure field, and viscous effects change from one layer to the next. In this case, the modeled currents have a direct effect on the path planning process. Velocity vectors are used as penalty parameters in the cost functions to cut down on energy loss.When these currents change over time, standard algorithms often have trouble. Dynamic adaptive mechanisms are used to keep a balance between exploration and exploitation, even when the environment changes. Studies show that Navier-Stokes-based modeling works well even when there are a lot of problems and strong flows [70]. Future endeavors in this domain will probably concentrate on enhancing the resilience of these routes to abrupt turbulence

through the incorporation of real-time current prediction.Other biological inspirations have also been looked into besides PSO and GWO. Genetic Algorithms (GA) have been utilized to assist aircraft in safely navigating through dynamic threat zones [71]. The Artificial Immune Algorithm (AIA) is another interesting method. It lets a system ”remember” threats and stay away from them based on past experiences (Liu 2020). In complex reconnaissance missions with multiple objectives, advanced threat models have been developed to assist UAVs in managing conflicting criteria during flight [72].

3.4.2. Social and Biological Behavioral Models (GWO, Sand Cat, Nutcracker vb.)

There are many methodologies on trajectory optimization and path planning. The methodologies can be divide into three subcategories. Firstly, The Swarm intelligence and bio-inspired methodologies. Secondly, Evolutionary and Multi-Objective methodologies. Final is Physics-Inspired and Hybrid methodologies. About the first subcategory, the Swarm intelligence and bio-inspired methodologies; [59] developed an algorithm called the Honey Badger Algorithm (HBA), this metaheuristic method mimics the intelligent foraging behavior of the honey badger. The methodology is divided into a digging phase (using smell to locate and dig for prey) and a honey phase (following a honeyguide bird directly to a beehive). It utilizes an inverse square law to calculate smell intensity and employs dynamic randomization control factors to smoothly transition between broad search exploration and localized exploitation. HBA uses an inverse square law to estimate the intensity of smell and employs dynamic randomization control to effect a smooth and seamless change between broad search exploration and more focused search exploitation. [73] presents an algorithm Multi-Population Evolutionary Slime Mould Algorithm (MESMA). A nature-inspired metaheuristic optimization method that mimics the foraging and oscillatory contraction behavior of Physarum polycephalum and was introduced in 2020. It determines optimal pathways by mimicking the behavior of slime molds and their feedback mechanisms of positive and negative constructs in the connecting food sources. Because of this, it is very applicable to complicated multi-modal engineering optimization problem. In order to enhance the original Slime Mould Algorithm (SMA), MESMA uses a multi-population

evolutionary strategy and a sine-cosine function-based movement approach, which helps to prevent local optima trapping. MESMA employs a logistic chaos map initialization to improve the diversity of its first population. A lens imaging inverse learning strategy helps to optimize the search target by refining the target. Another metaheuristic algorithm is presented by [74] called as, Improved Nutcracker Optimization Algorithm. The method is based on the caching and foraging behavior of nutcrackers. It employs a cache-search and recovery strategy along with a sine-cosine strategy. These strategies combine with a logistic chaos map initialization for population diversity. Also, the method employs cubic spline interpolation to smooth the generated points which is used for flight along generated points by creating smooth, flyable curves constructed by linking the points via cubic polynomials dispersed along the generated points. [75], proposed the method Enhanced Grey Wolf Optimization (E-GWO), tailored for multi-UAV trajectory planning and execution within dynamically changing threat environments. Uses greedy initialization for the generation of quality first trajectories and incorporates K-means cluster analysis for the population division relative to multi-indicator fitness constraints (fuel consumption, altitude, collision, risk, time, etc.). Includes an obstacle real-time decision-making mechanism for phased avoidance to go around or penetrate obstacles. [18] presents a method called as Modified Teaching-Learning-Based Optimization (MS-TLBO), This algorithm simulates classroom learning through a Teacher phase (students learning from the most knowledgeable individual) and a Learner phase (peer-to-peer interactions). The modified version enhances local search capabilities by employing a chaotic-sequence mutation strategy to explore the neighborhood of the current best solution. The next subcategory is evolutionary and multi-objective methodologies, [76], highlights a methodology, Hierarchical Response System (HRS) for Dynamic Multiobjective Optimization. This methodology quantifies the severity of environmental changes to select the most efficient response strategy dynamically. It utilizes Support Vector Machine (SVM)-based Transfer Learning (TL) to initialize populations for medium changes, re-initialization for severe, dramatic shifts, and Pareto set refinement for slight changes. It also applies local search strategies to identify high-quality Transfer Reference Points (TRPs). [57], working on artificial objective evolutionary multi-objective optimization methods functions (EMOs) to tackle dynamic

environments, this methodology transforms single-objective problems into multi-objective tasks by adding artificial objective functions. These artificial objectives (e.g., time-based, random, inverse, or distance to the nearest neighbor) act to continuously maintain population diversity over time without relying solely on high mutation rates. [77], Accurate Path Planning Method for Swarms (APPMS), presents a method for emergency situations to enhance in complicated dynamic environments, for instance, disaster scenarios. This hybrid evolutionary method tackles constrained 3D path planning through a combination of global and local search mechanisms. It uses pheromone updating (borrowed from Ant Colony Optimization) for global exploration, neighborhood search by random disturbance for local refinement, and a knee point-based selection method to identify the final optimal path from a Pareto front. [66], DE3D-NURBS Path-Planner: This framework integrates Non-Uniform Rational B-Splines (NURBS) with an adaptive Differential Evolution algorithm called L-SHADE-COP. L-SHADE-COP uses Success-History Based Parameter Adaptation to dynamically update crossover and mutation probabilities based on historical memory, while the NURBS method guarantees that the resulting 3D trajectories satisfy kinematic curvature and pitch constraints.The third subcategory is, Physics-Inspired and Hybrid Methodologies. Optical Microscope Algorithm with Precision Focusing (PMOMA) is presented by [78]. Modeled after the magnification process of a microscope, this physics-inspired algorithm introduces a multi-fusion strategy combining random initialization, reverse learning, and SPM chaotic mapping to build high-quality initial populations. It integrates a precise focusing strategy that mimics coarse and fine focal adjustments using the Cauchy inverse cumulative distribution to push search agents toward or away from focal points. A migration strategy is also used to help agents jump out of local optima. [79], Hybrid GA-APC Algorithm: Used for deploying UAV swarms, this hybrid algorithm links Affinity Propagation Clustering (APC) with a Genetic Algorithm (GA). APC intelligently groups UAVs without requiring a predetermined cluster count, while the GA optimizes the real-time flight paths. The methodology also features the Okumura-Hata model for robust real-world path-loss modeling. The study is referred that mathematic model used to simulate realistic communication links between Unmanned Aerial Vehicle (UAV) swarms and user devices during disaster response scenarios. Because these UAVs act as temporary

mobile aerial base stations when traditional infrastructure fails, it is crucial to accurately predict signal strength across different landscapes. The systematic review is handled by [80]. The meta-analysis on meta-heuristic, analyzing the optimization of UAV path planning, reveals that hybrid methodologies are currently the most dominant choice in the research community, significantly outperforming standalone swarm-based, evolutionary, or physics-based algorithms. These hybrid models extensively rely on nonlinear mathematical modeling (due to complex environmental variables like wind speed and diverse landscapes) and predominantly focus on simultaneous optimization of path length, CPU time, energy consumption, and collision avoidance. Consequently, meta-heuristic programs have become the industry standard for finding a ”suboptimal solution within an acceptable time” in 3D UAV path planning problems. However, in fully dynamic combat scenarios where the environment changes very rapidly, these applications must search for a solution from scratch in each iteration, which strains real-time processing constraints. This bottleneck has paved the way for the rise of Deep Reinforcement Learning (DRL) models today.

3.4.3. Artificial Intelligence and Deep Learning (DL) Models

In the literature, traditional and meta-heuristic algorithms demonstrate superior performance in static or partially dynamic environments; however, they encounter difficulties in satisfying real-time processing requirements in highly dynamic, uncertain, and adversarial combat situations. Recent research has focused a lot on Deep Reinforcement Learning (DRL) algorithms to get around this problem. The DNN-TRE module uses the surrogate modeling approach that Forrester et al. [81] talk about to avoid costly physics. Its neural architecture and training stability are based on standard deep learning ideas (Goodfellow et al. [82] [Y]).These methods let autonomous agents make decisions in less than a second by trying things out and getting the most rewards over time. Methodologically, DRL studies in UAV path planning can be analyzed through two primary branches: Models Based on Value and Policy Gradient Methods. Value-Based Models (DQN, DDQN, D3QN) Value-based DRL algorithms, which come from Q-Learning, figure out how much taking a certain action in a certain state is worth. The Deep Q-Network (DQN), which uses deep neural networks to

estimate functions, is a leader in this area. According to [83] and [84], classical DQN can successfully guide UAVs and multi-robot systems to their targets in changing environments by only using raw sensory data. But classical DQN has a bias toward overestimating. To address this issue, researchers have implemented advanced architectures. To make it easier to avoid local obstacles, [85] suggested using a Double DQN (DDQN) with the Dynamic Window Approach (DWA). For stealth UAVs confronting multi-radar threats, [86] and [87] employed DDQN to enhance penetration trajectories by dynamically adjusting the radar cross-section (RCS) penalties. Moreover, [88] and [89] advanced the field by employing the Dueling Double DQN (D3QN) framework with prioritized experience replay, significantly improving learning stability and energy-efficient path generation in ultra-low altitude missions. The best thing about value-based methods is that they are *sample efficient* and can learn the best deterministic policies in spaces with a limited number of actions. Once the offline training is done, they let you make decisions (inferences) very quickly online. The major drawback of DQN variants is their mathematical limitation to discrete action spaces. When you break up the continuous flight dynamics of a fixed-wing UAV (like yaw, pitch, and roll angles) into grids, you lose accuracy and get jagged trajectories. To address the limitation of discrete action spaces in value-based methods, Actor-Critic architectures that directly optimize the policy (action distribution) in continuous spaces have gained widespread adoption. For high-performance aircraft with 6-Degrees-of-Freedom (6-DoF), [90] employed the Deep Deterministic Policy Gradient (DDPG) controller, facilitating autonomous aggressive maneuvers without the need to discretize control inputs. In the same way, [91] used continuous reinforcement learning to improve the energy-efficient online path planning of several drones. In highly dynamic environments characterized by wind disturbances and mobile obstacles, [92] effectively integrated the Proximal Policy Optimization (PPO) algorithm with Fluid Field mechanics, thereby providing robust routing planning. Multi-Agent Reinforcement Learning (MARL) frameworks like MADDPG [93] and Stackelberg Game approaches [94] have been used to solve collision avoidance and task allocation at the same time for multi-UAV cooperation and target assignment. Furthermore, [95] underscored the adaptive features of policy gradient methods in trajectory design under adverse weather conditions. In adversarial threat environments, DRL agents necessitate

highly precise threat assessment models as reward functions. The mathematical basis for these reward structures comes from the work of [96] on hidden Markov models and [97, 98] on three-way decision methods. The best thing about Policy Gradient methods is that they naturally support continuous action spaces. This makes it possible to make trajectories that are 100% flyable, smooth, and kinematically possible. Additionally, MARL extensions (such as MADDPG) naturally manage tasks that require cooperation between multiple agents. Their biggest problem is that they are very unstable during training and very sensitive to hyper-parameters. They have low sample efficiency, which means they need millions of training episodes in simulators (like Gazebo/AirSim) before they can be used in the real world. According to [99], the biggest problem with DRL-based UAV navigation is still closing the ”Sim-to-Real gap.” The advisor layer does not use a fully learned end-to-end policy. Instead, it uses an explainable, weighted multi-criteria fusion logic based on the ideas behind Multi-Criteria Decision Analysis (MCDA) (Keeney & Raiffa [100]).

3.4.3.1. UAV Obstacle Avoidance with Q-Learning and Hybrid Architectures In the literature, UAV safety in obstacle-filled situations is a major issue. Due to their model-free nature, Deep Q-Network (DQN) and standard Q-Learning methods are commonly utilized to solve this problem. Research by [101] and [102] shows that improved DQN methods can lead autonomous UAVs in 3D obstacle-cluttered settings without a pre-existing map at [103] and [104], deep reinforcement learning was shown to be successful at avoiding obstacles for autonomous vehicles and robots. [105] expanded this to complex dynamic settings. Q-Learning-based obstacle avoidance is optimal because the agent learns from its collision mistakes, generating a robust reflex mechanism. The discrete action space problem and sluggish convergence speed are its key drawbacks. Researchers recently hybridized Q-Learning with other algorithms to solve these drawbacks. A study by Kong et al. (2023) integrated DQN with APF, using APF for local repulsions and DQN for global routing. In contrast, [106] suggested a PSO-Enhanced DQN for smooth UAV movement in continuous space. A reverse hybrid technique [107] utilized reinforcement learning to alter parameters of a multi-strategy cuckoo search algorithm, rather than directly planning paths. Multi-strategy

fusions using A* were used in [1] to decrease inflection points. The hybrid approaches blend conventional mathematics with current AI.

3.5. Tactical Constraints and Flight Dynamics

3.5.1. Target Assessment Models and Radar-Evasive Path Planning

Tactical UAVs and fourth-generation fighter aircraft must survive radars and dynamic threats, not just obstacles. Thus, Target Assessment Models and radar-evasive routing are popular. [108] and [109] compared DRL-based tactical UAV path optimization under radar threats. They showed AI can dynamically adjust to radar cross-sections. Two studies [110] and [53] analyzed 3D path planning challenges for stealth aircraft, generating routes that reduce detection probability. Before avoiding a threat, the UAV must appropriately assess it. In 2025, [111] Wu suggested a data-fusion technique to evaluate the threat level of small objects at low altitudes. Using sensor data fusion in threat assessment significantly enhances drone survival. Environmental factors present natural obstacles beyond hostile threats. [112] Jayaweera et. al. ,explored UAV path planning in windy conditions, emphasizing the importance of meteorological limitations in optimization. Using restricted multi-objective evolutionary algorithms, [113] improved UAV safety in smart city scenarios. The algorithm minimises total flight distance (calculated using Euclidean distance between waypoints) and risk cost, which is a weighted combination of static terrain risk (proximity to buildings and topography) and dynamic pedestrian risk (the potential impact of a crash on dynamic ground crowds, based on real-time pedestrian density data). The produced pathways must meet five tight requirements to assure flight feasibility: minimum and maximum safe altitudes, maximum yaw angle, ascending slope, and distance between waypoints. The suggested solution relies on M2M-DW, a decomposition-based algorithm for constrained optimization problems. It divides the objective space into subregions and assigns feasible and infeasible weight vectors to subpopulations. Feasible weight vectors face strict constraint penalties to optimize distance and safety, while infeasible weight vectors face milder penalties to retain solutions near the feasible boundary to improve

global search and prevent local optima. To apply theoretical models to modern aviation, [114] highlighted real-time flight efficiency, while [115] described the crucial difference between simulation and real-world aerodynamics in digital flight control systems on F-16XL. According to [95] and [89], intelligent learning approaches are the best way to address several constraints simultaneously. The literature suggests that DRL models are the ultimate paradigm change in UAV path planning. Traditional algorithms ([7]) compute the path "at the moment," but DRL models shift the computational burden to offline training, serving as a learned reflex mechanism during online execution. The literature shows a modern trajectory planning split. Optimization methods (xPTR, DE-MPC) provide rigorous, physically feasible global pathways but are computationally intensive. Heuristic/reactive approaches (A*, DWA) and online fuzzy systems adapt to uncertain terrain faster and in real time. The integration of Deep Reinforcement Learning (PPO, DDPG) allows aircraft to learn stealth and agile control strategies through interaction with simulated environments, bridging the gap between classical nonlinear control and AI. Researchers have integrated complicated physical aircraft performance models and 6-DoF differential equations directly into path planning algorithms to bridge the "Sim-to-Real gap". Research on integrating physical performance with stealth dynamics is extensive. Early efforts, including [116], coupled optimal terrain and threat-based trajectory planning with aircraft restrictions. On this foundation, [117] introduced a comprehensive Physical Aircraft Performance Model for multi-criteria trajectory optimization. This method assures that produced pathways respect UAV energy states, fuel consumption, and load factors. Numerical optimization is preferred because it generates dynamically viable trajectories that meet operational restrictions. In [**?**], an optimization framework is used to solve the approach and landing issue for a 6-DoF aircraft. The Extrapolated Penalized Trust-Region (xPTR) technique, a type of Sequential Convex Programming (SCP), is proposed to address the nonconvexity of 6-DoF dynamics [**?**]. The xPTR algorithm uses Nesterov's acceleration-inspired extrapolation step to accelerate algorithmic convergence and satisfy continuous-time constraints like runway alignment and obstacle avoidance. For high-fidelity models, [116] divides the terrain/threat-based trajectory problem into two parts. A Differential Evolution (DE) algorithm is used to construct globally optimal trajectories, minimizing time, fuel, and altitude. They use a multi-input,

multi-output nonlinear model predictive controller (MPC) with a neurofuzzy predictor to execute these trajectories in real time and continually correct future tracking mistakes due to complicated aerodynamic interaction. Modern stealth aircraft combat scenarios make kinematic limitations even more important. To model stealth aircraft penetration trajectories in 3D complicated dynamic situations, [118] rigorously adhered to the ”Radar Valley Radius” and maximum ”Turning Maneuver” restrictions. The advantage of using physical performance models [117, 118] is their absolute aerodynamic feasibility. To avoid stalling and structural damage, the UAV will never be ordered to do an impossible maneuver like a 90-degree quick turn. These models require precise aerodynamic derivatives (lift/drag coefficients), which might be difficult to predict. Additionally, solving these physics equations for every node growth significantly increases processing time.

3.5.2. 6-DoF Kinematics and Dynamic Obstacle Avoidance

For critical flight phases, lower-fidelity models (like 3-DoF) are insufficient. [119] demonstrated this by optimizing 6-DoF aircraft landing trajectories with precise runway alignment. By considering pitch, roll, and yaw dynamics simultaneously, the model generates highly realistic approach paths. To handle dynamic threats while respecting flight dynamics, [120] developed a hybrid architecture that merges global aircraft route planning with local dynamic obstacle avoidance. Taking this a step further into the AI domain, [90] developed a deep reinforcement learning (DRL) control approach specifically tailored for high-performance aircraft operating under 6-DoF constraints. The combination of DRL with 6-DoF models [90] provides *real-time execution of highly non-linear maneuvers*. Once the neural network is trained with the 6-DoF physics engine, it can output flyable commands in milliseconds to avoid local dynamic obstacles [120]. The main weakness of 6-DoF planning is the *extreme mathematical complexity*. Solving non-linear differential equations online is computationally heavy, and training a DRL agent in a 6-DoF simulator suffers from severe instability and requires millions of episodes to converge. As a result, integrating physical constraints and 6-DoF models represents the ultimate maturity level of UAV path planning. While [116] and [119] proved that analytical physics models guarantee realism, [90] and

[120] showed that hybridizing these physics engines with AI and local avoidance algorithms is the only way to achieve real-time survivability in modern adversarial environments. In the literature, most terrain following (TF) and terrain avoidance (TA) path planners generate the entire 3D flight path directly before the flight (offline). However, in completely unknown rough terrain environments, offline planning is not feasible. To address this, [121] developed an efficient online algorithm for aircraft velocity and normal acceleration planning. Their proposed algorithm does not directly create a geometric flight path like traditional methods do. Instead, it uses real-time data from the vehicle's onboard sensors and quickly sends out the right Guidance Commands (GCs) at every point along the way. To make sure these commands could be carried out in real life, [121] added a dynamic model that takes into account the lags in the vehicle dynamics. As a result, the flight path gradually takes shape when the GCs are applied to the aircraft's dynamics, giving it a very autonomous ability to follow the terrain. The best part of this method is that it can change in real time without needing maps ahead of time. It cuts down on the pilot's workload a lot by directly calculating the necessary normal acceleration and velocity commands in real time. This stops aerodynamic stalls during steep mountain climbs. It makes it easy to go from path planning to low-level flight control systems. The biggest problem with this reactive strategy, on the other hand, is that it relies too much on the accuracy of the onboard sensors. If the forward-looking sensors stop working or have a delay (sensor noise), the plane doesn't have a global safety path to fall back on. Also, because the path is built up over time, it is not possible to guarantee global optimality (for example, finding the shortest route over the mountains). The method by [121] is different from the physical performance model by [117] and the 6-DoF approaches by [119] because it is a more reactive, sensor-driven control mechanism instead of just a math optimization problem. It serves as an online survival reflex instead of an offline routing tool. Alternative methods put more emphasis on online decision-making and continuous physical simulation than on heavy pre-flight numerical optimization. The Compromised Aircraft Performance Model with Limited Accuracy (COALA) is introduced in [117] to optimize Free Route trajectories based on multiple target functions, such as time, fuel burn, and emissions. COALA doesn't use simple geometric grids to guess at flight paths. Instead, it uses a Proportional-Integral-Derivative (PID) controller that constantly

changes the aircraft's lift coefficient to change the angle of attack based on real 4D weather data. The literature emphasizes a notable divergence in contemporary trajectory planning. Optimization methods (xPTR, DE-MPC) give you strict, physically possible global routes, but they take a lot of computing power. On the other hand, heuristic and reactive methods (A*, DWA) and online fuzzy systems can quickly adapt to changing terrain in real time. Additionally, the incorporation of Deep Reinforcement Learning (PPO, DDPG) represents a significant paradigm shift, allowing aircraft to acquire optimal stealth and highly agile control strategies via interaction with simulated environments, thereby connecting classical nonlinear control with artificial intelligence. Global-local fusion methods have become the norm because they can get around the problems with single algorithms. For instance, [122] suggested a strong hybrid architecture that uses heuristic Particle Swarm Optimization (PSO) for global routing and the Dynamic Window Approach (DWA) for strong local dynamic obstacle avoidance. Also, for all of these advanced path planning algorithms to work in the real world, they need to follow the laws of flight in the real world. The basic physics, 6-DoF differential equations, and control systems used in modern trajectory models are based on classic aerospace literature, like the detailed formulations for aircraft dynamics and control by Steven et al. [51].

3.6. Deep Neural Networks for Speeding Up Computation

Mnih et al. [123] states for deep neural networks in autonomous control by showing that neural architectures can accurately approximate complex value functions in high-dimensional spaces. Their Deep Q-Network (DQN) brought about a major computational asymmetry: the huge amount of processing power needed to figure out how the environment changes is only available during offline training. So, once the best policy is mapped into the network's weights, real-time online execution becomes a very optimized, deterministic forward pass.

This thesis proposes a tactical mission-planning framework that avoids the severe computational bottlenecks of traditional geometric line-of-sight (G-LOS) ray-tracing. The

suggested Deep Neural Network Tactical Risk Estimator (DNN-TRE) works as a stand-in regression model, moving the difficult work of physics-based spatial analysis to offline training. When a flight is happening, tactical risk estimation turns into a quick, $O(1)$ forward pass. This change in thinking gets rid of the latency that comes with classical methods, speeding things up by 3.4 times. It also frees up important onboard computing resources, letting the action layer planners focus only on optimizing kinodynamic trajectories.

4. METHODOLOGY

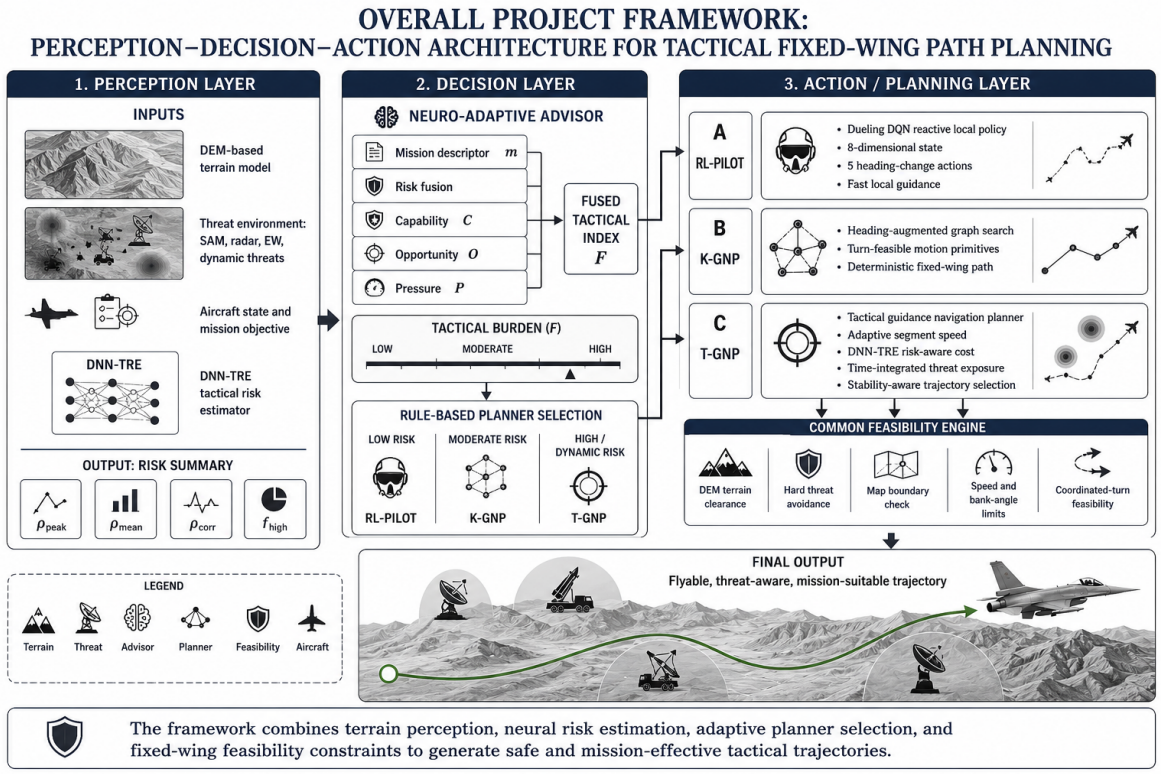

Figure 4.1 The architecture of the project

The overall architecture of the proposed search is demonstrated in Figure 4.1. The methodology not only developed new path planning algorithms but also develop treating perception, decision making and trajectory generation into one end-to-end system. The system has four main layers. First, Three degree-of-freedom point mass model aircraft and environment terrain model defines the physical and operational constraints with evaluation

metrics. Second, two kinodynamically-aware planners, namely K-GNP and T-GnP, are developed for trajectory generation. Third, a Deep Neural Network-based Tactical Risk Estimator (DNN-TRE) provides fast approximations of the spatial threat area. The last but not least, a Reinforcement Learning-based Advisor operates at the mission level to select the most appropriate planning strategy those are Kinematic, Tactical guidance planner or RL pilot.

4.1. Integration of Benchmark Planners and Evaluation Metrics

To evaluate the proposed study in a fair and systematic manner, a standardized benchmarking methodology was employed. The classical planning algorithms considered in this study (A*, Dijkstra, RRT*, and PSO) are not main components of the proposed architecture. Rather, they used as baseline methods to compare with respect to evaluation environment. Consistency is a fundamental design principle in the benchmarking process. All planners work under the same conditions, including terrain representation, threat modeling, and aircraft restrictions.

All planners are running on defined terrain maps and across the threat distributions across scenarios S1–S4. Since classical planners produce discrete waypoints rather than dynamically feasible trajectories, their outputs are processed through the same flight guidance controller used for the proposed planners. This allows a realistic evaluation of trajectory execution. All planners produce trajectory with consider the wind disturbances are applied to all planners during Monte Carlo simulations. Evaluation Metrics: Within the one comparison environment, all planners were evaluated using joint set of mission-level and implementation-level performance metrics. These criteria measure the geometric path quality and also whether the generated trajectories are physically executable when run through the same feasibility-focused flight simulation pipeline. In this sense, the shared feasibility engine provides the common validity layer that makes cross-planner comparison meaningful under fixed-wing flight constraints.

- **Success Rate:** Without violating terrain, threat, or dynamic feasibility constraints the percentage of Monte Carlo runs in which the aircraft successfully reaches the goal region.
- **Path Length (km):** Total length of the planned route. This metric reflects geometric mission efficiency and is also relevant to overall operational cost.
- **Flight Time (s):** Total simulated mission duration necessary to reach the goal under the planned trajectory. This metric captures the time penalty induced by maneuvering and execution Dynamics rather than a simple distance-over-speed estimate.
- **Risk Exposure (M):** Integrated cumulative threat exposure along the executed trajectory. This metric quantifies the aircraft entered unsafe regions and how much tactical risk it increasing by adding at flight phase.
- **Minimum AGL (m):** Minimum above-ground-level terrain clearance observed during execution. This metric indicates terrain safety and provides a direct measure of collision risk with the ground terrain.
- **Violation Count:** The number of strict restriction violations observed during the operation, such as getting in the protected threat zones.
- **Tracking Error (m):** Mean lateral deviation between the planned path and the executed flight trajectory. This metric is particularly important for distinguishing geometrically valid paths from practically flyable ones.
- **Computation Time (s):** Planner required runtime to generate the trajectory. This metric reflects suitability for real-time or near-real-time tactical applications.
- **Saturation Ratio (%):** The proportion of flight time during which the aircraft operates near its dynamic maneuverability limits, such as aggressive bank angle or roll requirements. This metric is particularly important for assessing operational stress and practical flight capability.

In addition to these principal metrics, supplementary indicators such as maximum tracking deviation, maximum observed bank angle, and minimum observed speed were also recorded in selected analyses to provide a more detailed picture of execution quality and dynamic feasibility.

4.2. Aircraft and Environment Modelling

On the thesis, fort he reduced order fixed wing aircraft model and its dynamics such as coordinated-turn, speed values are referenced from classical flight simulation principles described by Stevens et al. [51] and Beard & McLain [124]

The proposed planner depends on a reduced-order, three-degree-of-freedom point-mass representation that captures the kinematic motion behavior of a aircraft while avoiding the computational cost because of the six-degree-of-freedom simulations. This starting is fit for large-scale path planning problems where repeated feasibility evaluations are required. The system state is formulazed as

$$\mathbf{x}(t) = \left[x,\ y,\ h,\ \psi,\ V,\ \phi,\ \gamma\right]^{\top}, \tag{15}$$

where (x, y) denote the horizontal inertial coordinates, h represents altitude, ψ is the heading angle, V is the true airspeed, ϕ is the bank angle, and γ is the flight-path angle.

The governing equations of motion are propagated at each guidance step Δt as

$$\dot{x} = V\cos\gamma\cos\psi + w_x, \tag{16}$$

$$\dot{y} = V\cos\gamma\sin\psi + w_y, \tag{17}$$

$$\dot{h} = V\sin\gamma, \tag{18}$$

$$\dot{V} = \frac{T - D}{m} - g\sin\gamma, \tag{19}$$

$$\dot{\psi} = \frac{g\tan\phi}{V}, \tag{20}$$

$$\dot{\gamma} = \frac{L\cos\phi - mg\cos\gamma}{mV}, \tag{21}$$

where m denotes the aircraft mass, g is the gravitational acceleration, w_x and w_y represent horizontal wind disturbances, T is thrust, $D = c_D V^2$ models aerodynamic drag, and L corresponds to lift. Thrust is parameterized using a normalized throttle command as

$$T = \delta_T \cdot T_{\max}, \qquad T_{\max} = 2\,mg, \tag{22}$$

where $\delta_T \in [0, 1]$.

A key feasibility constraint changes from the coordinated-turn relationship, which directly relatives the bank angle and velocity to the achievable turning curvature. For level flight conditions, the instantaneous turn radius is given by

$$R = \frac{V^2}{g\tan\phi}, \tag{23}$$

and enforcing a maximum allowable bank angle $\phi_{\max}$ yields the minimum feasible turning radius

$$R_{\min} = \frac{V^2}{g\tan\phi_{\max}}. \tag{24}$$

This constraint is demonstrates the during graph development, ensuring that all planning trajectories still same form for physically realizable for a fixed-wing platform.

In addition to curvature constraints, realistic flight behavior is maintained through an operational speed envelope. A lower speed boundary $V_{\min} = 120$ m/s is introduced for particularly during aggressive maneuvering or terrain-following segments. A nominal cruise speed of $V_{\text{cruise}} = 220$ m/s is used as the flight speed that operating on scenarios, while an upper limit of $V_{\max} = 280$ m/s is enforced to avoid unrealistic high-speed situations and provide strecth of the turning radius. These limits are continously used during both trajectory generation and flight simulation.

A raster-based Digital Elevation Model (DEM) is used to model how the environment interacts with itself. It is shown as a continuous surface $\mathcal{T} : \mathbb{R}^2 \to \mathbb{R}$ through bilinear interpolation. An above-ground-level (AGL) clearance constraint is used to make sure that terrain avoidance happens.

$$h(t) \geq \mathcal{T}\big(x(t), y(t)\big) + h_{\text{AGL}}, \quad \forall\, t \in [0, T_f], \tag{25}$$

which guarantees that all feasible trajectories maintain a minimum vertical separation from the terrain surface throughout the flight horizon.

4.3. Threat Modelling

Static and dynamic point threats are modelled as disc-shaped exclusion zones zones in the horizontal plane. A threat j is characterised by its centre (x_j, y_j), hard-kill radius r_j, threat-level coefficient $\ell_j \geq 0$, and, for dynamic threats, a velocity vector $(\dot{x}_j, \dot{y}_j)$.

The geometric line-of-sight (GLOS) risk experienced by the aircraft at position (x, y, h) due to threat j is

$$\rho_j(x, y, h) = \begin{cases} \ell_j \cdot \exp\left(-3\dfrac{d_j}{d_{\max}}\right) & \text{if not occluded,} \\ 0 & \text{otherwise,} \end{cases} \tag{26}$$

where $d_j = \sqrt{(x - x_j)^2 + (y - y_j)^2}$ is the horizontal range and $d_{\max}$ is the maximum effective engagement range. Terrain occlusion is evaluated by ray-marching between the threat and the aircraft along $N_{\text{ray}} = 15$ equally spaced sample points and checking whether any sample falls below the DEM surface.

The tactical risk at a point is

$$\rho(x, y, h) = \sum_j \rho_j(x, y, h). \tag{27}$$

4.4. Kinematic Guidance Navigation Planner (K-GNP)

4.4.1. Graph Structure and State Discretisation

K-GNP extends the classical Dijkstra search framework to operate on a heading-augmented state space. Each graph node is a tuple

$$\mathbf{s} = (i,\ j,\ k_\psi), \tag{28}$$

where i and j are the discrete horizontal grid indices with resolution Δ_{xy} (m) and k_ψ indexes one of N_ψ uniformly distributed heading bins

$$\psi_k = \frac{2\pi k}{N_\psi}, \quad k = 0, 1, \ldots, N_\psi - 1. \tag{29}$$

Altitude is treated as a continuously commanded variable governed by the vertical guidance law described in Section 4.6.1..

4.4.2. Motion Primitives and Curvature Enforcement

From each node $\mathbf{s} = (i, j, k_\psi)$ the planner expands a set of motion primitives $\mathcal{P}$. Each primitive $p \in \mathcal{P}$ corresponds to a heading change $\Delta\psi_p \in \{-\psi_{\max}, \ldots, 0, \ldots, +\psi_{\max}\}$ applied over a fixed arc length L_p. The resulting horizontal displacement is

$$\Delta x_p = L_p \cos\Big(\psi_k + \tfrac{\Delta\psi_p}{2}\Big), \tag{30}$$

$$\Delta y_p = L_p \sin\Big(\psi_k + \tfrac{\Delta\psi_p}{2}\Big). \tag{31}$$

The minimum curvature radius applied by each initial is

$$R_p = \frac{L_p}{|\Delta\psi_p| + \varepsilon}, \tag{32}$$

with $\varepsilon = 10^{-6}$ for numerical stability. A primitive is kinematically admissible only if $R_p \geq R_{\min}$; Otherwise, it is pruned from the expansion set.

4.4.3. Edge Cost Function

The edge cost associated with initial p from node $\mathbf{s}$ to successor $\mathbf{s}'$ is the weighted sum

$$c_p = \underbrace{L_p}_{\text{distance}} + w_\rho\, \bar{\rho}_p + w_{\mathrm{AGL}} \mathbb{1}\big[h_p < h_{\mathrm{AGL}}\big] \cdot \big(h_{\mathrm{AGL}} - h_p\big), \tag{33}$$

where $\bar{\rho}_p$ is the mean GLOS risk sampled along the primitive at the planning altitude, w_ρ is the threat-weight parameter, and the last term penalises terrain constraint violations.

4.4.4. Search Algorithm

K-GNP employs Dijkstra's algorithm on the heading-augmented graph. The open set is a binary min-heap keyed on the accumulated cost-to-come $g(\mathbf{s})$. The goal condition is satisfied when any expanded node falls within the goal tolerance r_{goal} of the target coordinates.

4.5. Tactical Guidance Navigation Planner (T-GnP)

The heading-aware state expansion structure of the proposed T-GnP planner is consistent with state-lattice planning principles for differentially constrained vehicles as outlined by LaValle [125] and Pivtoraiko et al. [126]. T-GnP extends K-GNP with three additional tactical capabilities: turn-aware cost shaping, adaptive speed-aware planning, and an optional neural risk integration layer.

4.5.1. Turn-Aware Cost Shaping

Large heading changes increase both the geometric path length and the time the aircraft is exposed to threats. T-GNP augments the edge cost with a heading-change penalty proportional to the turn severity:

$$c_p^{\text{T-GnP}} = c_p + w_\psi \left|\Delta\psi_p\right|^{\alpha_\psi}, \tag{34}$$

where $w_\psi > 0$ is the turn-penalty weight and $\alpha_\psi \in (1, 2]$ controls the degree of penalisation for large turns. This encourages the planner to prefer smooth heading changes unless a sharp manoeuvre is tactically necessary.

4.5.2. Adaptive Speed-Aware Planning

T-GnP models a speed-dependent edge traversal time to account for the fact that a slower aircraft spends more time under threat exposure. Given a curvature-adjusted target speed V_p^*, the time to traverse primitive p is

$$\tau_p = \frac{L_p}{V_p^*}, \tag{35}$$

and the time-integrated risk contribution is

$$\tilde{\rho}_p = \bar{\rho}_p \cdot \tau_p. \tag{36}$$

The curvature-adjusted speed follows the coordinated-turn speed limit derived from Equation (23):

$$V_p^* = \min\Big(V_{\text{cruise}},\ \eta\sqrt{R_p\, g\, \tan\phi_{\max}}\Big), \tag{37}$$

where $\eta \in (0, 1]$ is a safety margin factor.

4.5.3. Neural Risk Integration

When the DNN-TRE module (Section 4.8.) is enabled, each edge cost is augmented with a neural risk estimate:

$$c_p^{\text{neural}} = c_p^{\text{T-GnP}} + w_{\text{nn}}\,\hat{\rho}_{\text{nn}}\Big(\tfrac{x_p+x_p'}{2},\ \tfrac{y_p+y_p'}{2}\Big), \tag{38}$$

where $\hat{\rho}_{\text{nn}}$ is the DNN-TRE output at the mid-point of the primitive, and w_{nn} is the neural risk weight.

4.5.4. Stability and Composite Cost

The full T-GnP composite cost integrating speed weight w_V and stability weight w_S is

$$c_p^{\star} = c_p^{\text{T-GnP}} + w_V\,\tilde{\rho}_p + w_S\,\mathcal{S}_p + w_{\text{nn}}\,\hat{\rho}_{\text{nn}}, \tag{39}$$

where $\mathcal{S}_p$ is a stability score that penalises rapid bank-angle reversals between successive primitives:

$$\mathcal{S}_p = \max\bigl(0,\ |\Delta\phi_p - \Delta\phi_{p-1}| - \Delta\phi_{\text{tol}}\bigr). \tag{40}$$

4.6. Flight Guidance and Simulation Layer

4.6.1. Lateral and Vertical Guidance Laws

The guidance system tracks the planned waypoint sequence using a lookahead-point strategy. Given the current aircraft position $\mathbf{p} = (x, y, h)^{\top}$ and the lookahead target waypoint $\mathbf{w} = (x_w, y_w, h_w)^{\top}$, the desired heading is

$$\psi_{\text{des}} = \text{atan2}(y_w - y, x_w - x) \tag{41}$$

The lateral heading error $e_\psi = \psi_{\text{des}} - \psi$ (wrapped to $(-\pi, \pi]$) drives a proportional turn-rate command

$$\dot{\psi}_{\text{cmd}} = k_\psi\, e_\psi, \tag{42}$$

which is converted to a bank-angle command via the coordinated-turn relation:

$$\phi_{\text{cmd}} = \arctan\left(\frac{\dot{\psi}_{\text{cmd}}\, V}{g}\right). \tag{43}$$

The vertical guidance commands a flight-path angle proportional to altitude error:

$$\gamma_{\text{cmd}} = \text{clip}\Big(k_h\,(h_w - h),\ -\gamma_{\max},\ \gamma_{\max}\Big). \tag{44}$$

4.6.2. Curvature-Based Speed Management

To ensure fair comparison across planners that produce paths of different waypoint densities, the simulation uses a *distance-based* lookahead for curvature estimation. The cumulative arc length along the planned path is

$$s_k = \sum_{i=1}^{k} \|\mathbf{p}_i - \mathbf{p}_{i-1}\|_2, \tag{45}$$

and three distance-spaced samples at s_{base}, $s_{\text{base}} + 0.8\,d_{\text{LA}}$, and $s_{\text{base}} + 1.6\,d_{\text{LA}}$ (where d_{LA} is the lookahead distance) are used to estimate the local turn angle θ and radius $\hat{R}$ of the upcoming path segment. The resulting safe speed command is

$$V^* = \text{clip}\Big(\eta\sqrt{\hat{R}\,g\,\tan\phi_{\max}},\ V_{\min},\ V_{\text{cruise}}\Big). \tag{46}$$

The throttle command that realises V^* is computed from a first-order speed-tracking law with time constant τ_V:

$$\dot{V}_{\text{des}} = \frac{V^* - V}{\tau_V}, \tag{47}$$

$$\delta_T = \text{clip}\left(\frac{m\,\dot{V}_{\text{des}} + c_D V^2}{T_{\max}},\ 0,\ 1\right). \tag{48}$$

4.7. Dueling DQN Architecture of the RL-Pilot

When the advisor selects the RL-Pilot mode, the reactive local flight policy is driven by a Dueling Double Deep Q-Network (DQN). This structure was adopted to improve action-value estimation stability in a tactical flight environment where heading corrections must be evaluated under coupled terrain, threat, and path-following effects. The underlying design follows the exploration-exploitation principles of reinforcement learning described by Sutton and Barto [123] and extends them through a dueling value-advantage decomposition.

The state-action value function is written as

$$Q(s,a;\theta) = V(s;\theta_V) + \left(A(s,a;\theta_A) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s,a';\theta_A) \right) \tag{49}$$

where $V(s)$ denotes the state-value stream and $A(s,a)$ denotes the action-advantage stream. This decomposition allows the network to separately estimate the global quality of the current tactical state and the relative benefit of individual heading actions.

The RL-Pilot operates on an 8-dimensional state vector composed of normalized goal-direction geometry and directional local risk sensing. The action space consists of five discrete heading-change commands,

$$\mathcal{A} = \{-30^\circ, -15^\circ, 0^\circ, +15^\circ, +30^\circ\}, \tag{50}$$

which correspond to aggressive left, mild left, straight, mild right, and aggressive right turn actions.

The policy network contains a shared feature backbone with two fully connected layers of size 128, followed by two separate streams of size 64 for state-value and action-advantage estimation. To stabilize learning, the implementation employs experience replay, a target network, Smooth L1 loss, and gradient clipping. The optimizer is AdamW with learning rate 10^{-3} and weight decay 10^{-4}. The replay buffer size is 50,000, the batch size is 128, the discount factor is $\gamma = 0.99$, and the gradient norm is clipped at 2.0. The target network is

synchronized internally every 200 gradient steps, while an additional hard synchronization is applied every 20 training episodes.

4.7.1. RL-Pilot Training Configuration

The RL-Pilot training process was designed as a mixed-scenario reinforcement learning pipeline rather than a single-scenario control task. At each episode, one of the benchmark scenarios is sampled according to a fixed scenario-probability vector,

$$p_{scen} = [0.10, 0.30, 0.35, 0.25] \tag{51}$$

corresponding to S1 Base, S2 Dense, S3 Long, and S4_DynamicThreat, respectively. This distribution intentionally biases training toward the denser and longer-range cases while still retaining exposure to the nominal baseline scenario.

The training script is parameterized through environment-configurable settings. In the default implementation, the number of episodes is 4000 and the maximum rollout horizon is 360 steps per episode. A safe-altitude parameter of 1000 m is used during training, and progress statistics are reported every 250 episodes. The policy is trained using an ϵ-greedy exploration strategy with an initial exploration level of 1.0 and a minimum exploration level of 0.05. In addition to the internal decay behavior of the agent, the outer training loop applies an episodic exploration reduction every 15 episodes according to

$$\epsilon_{max} = \max(\epsilon_{min}, 0.985\epsilon), \tag{52}$$

where $\epsilon_{min} = 0.05$.

To improve robustness and reduce geometric overfitting, each sampled scenario is perturbed during training. Threat centers are shifted independently within ± 800 m in both horizontal directions, and the initial aircraft position is perturbed within ± 250 m around the nominal

start location. In this way, the agent is exposed not to a single deterministic geometry, but to a family of nearby tactical variations.

The RL-Pilot does not rely solely on direct goal pursuit. Instead, it is trained with a guide-aligned behavioral prior constructed from a cached global reference path. For each scenario, a global guide route is first generated and then converted into a moving local navigation target. The RL agent is therefore encouraged to combine reactive maneuvering with corridor-following behavior rather than acting as a purely myopic local controller.

The reward function reflects this design. Let d_t denote the Euclidean distance to the final goal at time step t, and let d_t^{nav} denote the distance to the active guide target. The step reward is formulated as

$$\begin{aligned} r_t = {} & 0.10(d_{t-1} - d_t) + 0.14(d_{t-1}^{nav} - d_t^{nav}) - 0.12 \\ & - 4.8\rho_t^{center} - 1.8\rho_t^{side} - |\epsilon_{\psi,t}| - 0.6\min(n_t^{rev}, 3) \\ & - 350[\text{collision}] + 700[\text{success}], \end{aligned} \tag{53}$$

where ρ_t^{center} and ρ_t^{side} are the forward and lateral local risk estimates, $\epsilon_{\psi,t}$ is the normalized heading error, and n_t^{rev} is the revisit count of the current discretized spatial cell. This reward structure encourages simultaneous progress toward the final mission goal and the guide corridor, while penalizing excessive threat exposure, poor heading alignment, oscillatory revisits, and collisions.

Overall, the RL-Pilot training methodology used in this thesis combines mixed-scenario curriculum learning, local geometry perturbation, guide-aligned reward shaping, and stabilized Dueling Double DQN optimization. This configuration was adopted to produce a reactive tactical planner that remains effective across multiple benchmark geometries rather than over-specializing to any single scenario.

The final RL-Pilot benchmark values discussed in this thesis are taken from the local archived evaluation pipeline, while intermediate Colab-based benchmark summaries are provided only as supplementary material in the appendix.

4.8. Deep Neural Network Tactical Risk Estimator (DNN-TRE)

DNN TRE risk estimation is accesible for all planners; it's integrated into the feasibility engine module on GUI. So the PSO, A*, RRT*, Dijkstra, K-GNP, T-GNP and RL pilot could be operate using a common segment cost. For RL-Pilot and T-GnP, DNN can be used in a more specific and meaningful planner-level manner. The results are shown for this purpose.

4.8.1. Motivation and Input Feature Set

The geometric line-of-sight (G-LOS) based tactical risk computation used in this study is informative, but it is also computationally demanding. The main reason is that each risk query requires repeated terrain-occlusion checks over the DEM. When this process is performed densely over a mission area, the overall cost becomes significant.

To reduce this burden, a surrogate model called the Deep Neural Network Tactical Risk Estimator (DNN-TRE) is introduced. The role of this module is to approximate the tactical risk field from a compact set of terrain- and threat-related features, without performing full geometric computation at inference time.

According to the final dataset configuration, each sample is represented by the following four-dimensional feature vector:

$$\mathbf{f} = \left[d_{\mathrm{norm}},\ h_{\mathrm{rel,norm}},\ s,\ \ell \right]^{\top}, \tag{54}$$

where d_{norm} denotes the normalized distance to the nearest threat, $h_{\mathrm{rel,norm}}$ is the normalized relative altitude term, s represents the local terrain slope, and ℓ is the threat-level coefficient. The regression target is a scalar tactical risk value denoted by

$$y = \rho_{\mathrm{risk}}. \tag{55}$$

4.8.2. Network Architecture

The DNN-TRE is implemented as a fully connected feedforward neural network. In the final deployed version, the hidden-layer dimensions are

$$(128,\ 128,\ 64).$$

The network maps the four-dimensional input vector to a single scalar risk estimate through ReLU-activated hidden layers, followed by a Softplus output layer:

$$\mathbb{R}^4 \to \mathbb{R}^{128} \xrightarrow{\text{ReLU}} \mathbb{R}^{128} \xrightarrow{\text{ReLU}} \mathbb{R}^{64} \xrightarrow{\text{ReLU}} \mathbb{R}^{1} \xrightarrow{\text{Softplus}} . \tag{56}$$

A Softplus activation is used at the output stage in order to keep the predicted risk non-negative:

$$\hat{\rho} = \ln(1 + e^{z}), \tag{57}$$

where z denotes the scalar pre-activation output of the last linear layer.

The raw tactical risk labels are not used directly during training. Instead, a transformed target representation is adopted to improve numerical stability and learning behavior. Based on the deployed model configuration, the target is compressed using a logarithmic transformation with scaling:

$$\tilde{y} = \ln\left(1 + \frac{y}{\lambda}\right), \tag{58}$$

where the scale factor is

$$\lambda = 1.3361.$$

During inference, the inverse transformation is applied to recover the estimated tactical risk in the original scale:

$$\hat{y} = \lambda \left(e^{\hat{\rho}} - 1\right). \tag{59}$$

Input features are standardized using the training-set mean and standard deviation values stored in the model metadata:

$$\tilde{\mathbf{f}} = \frac{\mathbf{f} - \boldsymbol{\mu}_{\mathbf{f}}}{\boldsymbol{\sigma}_{\mathbf{f}}}, \tag{60}$$

This preprocessing step helps stabilize optimization and ensures that the same scaling is preserved during deployment.

The DNN-TRE is formulated as a regression model rather than a classification model. Its purpose is to estimate the continuous tactical risk magnitude produced by the geometric baseline, while avoiding the computational expense of repeated ray-tracing operations.

In the final implementation, the network is optimized using a robust regression objective consistent with Smooth L1 behavior. The best recorded validation loss is

$$\mathcal{L}^{\star}_{\mathrm{val}} = 0.06122. \tag{61}$$

This result indicates that the network is able to capture the dominant structure of the tactical risk field with sufficiently low approximation error for practical inference-time use.

The last training run for the DNN-TRE used a batch size of 256 and a maximum training length of 320 epochs. We used a patience parameter of 40 epochs to enable early halting. This helped prevent overfitting and kept the most stable checkpoint.

The training metadata shows that the best model was found at epoch 57. There was no significant improvement in validation after that point. This is why the deployed model is based on the best-validation checkpoint instead of the last epoch of the planned training period.

This choice was made to make the model more broad and to keep the state that gave the best approximation performance.

The training labels used by the DNN-TRE were generated from the geometric G-LOS tactical risk formulation over three benchmark scenarios:

$$\{\text{S1\_Base},\ \text{S2\_Dense},\ \text{S3\_Long}\}.$$

The final dataset contains 24,000 labeled samples. Sampling was performed using a threat-focused strategy with focus probability

$$p_{\text{focus}} = 0.9,$$

meaning that most samples were drawn from regions close to threats rather than from uniformly safe areas. This increases the representation of tactically informative regions, especially where the local risk gradient is stronger.

An additional point should be noted here. The S4_DynamicThreat scenario was not included in the DNN-TRE training set. Therefore, the results later reported for S4_DynamicThreat do not represent in-distribution performance. Instead, they should be interpreted as a generalization-oriented evaluation under a previously unseen tactical configuration.

The exact geometric line-of-sight (G-LOS) tactical risk computation is computationally expensive because each query requires repeated terrain-occlusion checks against the DEM. To reduce this burden, the Deep Neural Network Tactical Risk Estimator (DNN-TRE) was developed as a surrogate regression model that approximates the tactical risk field from a compact set of terrain- and threat-related features.

According to the final dataset metadata, each DNN-TRE sample is represented by a four-dimensional feature vector,

$$\mathbf{f} = \left[d_{\text{norm}},\ h_{\text{rel,norm}},\ s,\ \ell\right]^{\top}, \tag{62}$$

where d_{norm} is the normalized distance to the nearest threat, $h_{\text{rel,norm}}$ is the normalized relative altitude term, s is the local terrain slope, and ℓ is the threat-level coefficient. The regression target is a scalar tactical risk label denoted by $y = \rho_{\text{risk}}$.

4.9. Neuro-Adaptive Mission Planning Framework

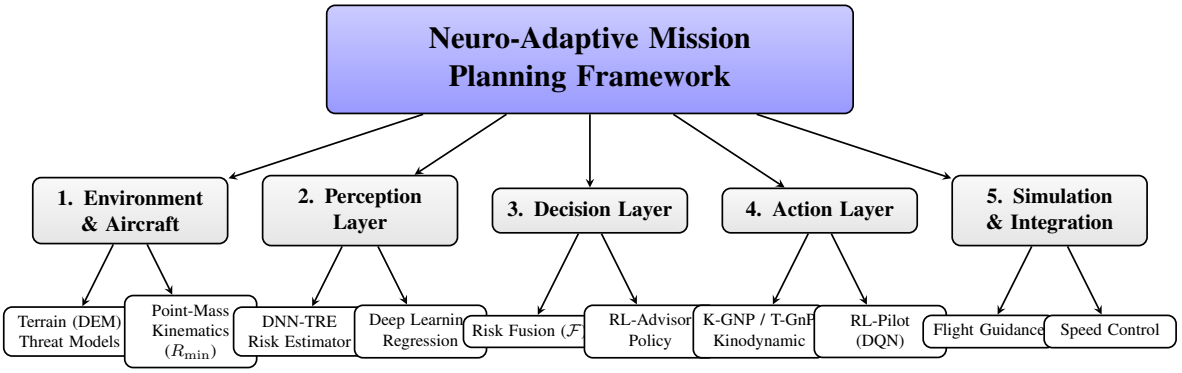

Figure 4.2 Taxonomy of the proposed neuro-adaptive framework, illustrating the core layers and subcomponents developed in this thesis.

4.9.1. Unified Framework Integration

1. **Perception Layer.** The DNN-TRE is queried to produce a fast approximate risk map over a grid of sample points. The resulting statistics $(\rho_{peak}, \overline{\rho}, \rho_{corr}, f_{high})$ are passed to the decision layer.

2. **Decision Layer.** The RL Advisor constructs the mission descriptor m (Equation 63), computes the fused risk index $\mathcal{F}$ (Equation 70), and selects the planning mode π^* via Algorithm 1.

3. **Action Layer.** The selected planner (K-GNP, T-GnP, or RL-Pilot) generates a waypoint sequence, which is post-processed by the trajectory processor to enforce $R \geq R_{min}$. The simulation guidance layer then executes the path under Monte Carlo disturbances.

Table 4.1 Key training parameters of the RL-Pilot module.

Parameter	**Value**
State dimension	8
Action dimension	5
Action set	$\{-30^\circ, -15^\circ, 0^\circ, +15^\circ, +30^\circ\}$
Default training episodes	4000
Maximum steps per episode	360
Scenario weights	S1: 0.10, S2: 0.30, S3: 0.35, S4: 0.25
Replay buffer size	50,000
Batch size	128
Discount factor (γ)	0.99
Optimizer	AdamW
Learning rate	10^{-3}
Weight decay	10^{-4}
Loss function	Smooth L1 loss
Initial exploration ϵ_0	1.0
Minimum exploration ϵ_{min}	0.05
Outer-loop epsilon decay	0.985 every 15 episodes
Target sync interval	200 gradient steps + every 20 episodes
Safe altitude in training	1000 m
Threat perturbation	± 800 m
Start perturbation	$+250$ m
Logging interval	250 episodes

4.9.2. Mission State Representation

RL Advisor operates at the mission level by receiving an encoded mission description vector m, which is based on geometry and statistics calculated using the outputs of the DNN-TRE algorithm:

$$m = [d_{mission}, \sigma_T, n_{dyn}, u, T_{budget}, \rho_{peak}, \overline{\rho}, \rho_{corr}]^\top, \tag{63}$$

where $d_{mission}$ is the Euclidean start-to-goal distance, $\sigma_T = \sum_j l_j$ is the aggregate threat score, n_{dyn} is the count of dynamic threats, $u \in \{0, 1\}$ is the binary urgency flag (1 if $T_{budget} \leq 3$ s), ρ_{peak} is the maximum DNN-TRE output over the mission area, $\overline{\rho}$ is the mean DNN-TRE output, and ρ_{corr} is the average DNN-TRE output along the straight-line corridor from start to goal.

4.9.3. Multi-Attribute Risk Fusion

The advisor fuses the descriptor into a scalar tactical risk index $\mathcal{F}$ using three interpretable sub-scores.

The capability score measures local threat severity by combining peak risk, mean risk, and the overall threat density in the scenario:

$$\mathcal{C} = 0.58\hat{\rho}_{peak} + 0.28\hat{\rho}_{mean} + 0.14\tau. \tag{64}$$

where the normalised values are

$$\hat{\rho}_{peak} = \frac{\rho_{peak}}{\rho_{peak} + 1.1}; \quad \hat{\rho}_{mean} = \frac{\overline{\rho}}{\overline{\rho} + 0.35}; \quad \tau = \min\left(\frac{n_{threat}}{5}, 1\right). \tag{65}$$

The opportunity score measures the tactical accessibility of the direct corridor and the density of high-risk zones in the mission area:

$$\xi = \frac{\rho_{corr}}{\max(\rho_{peak}, 10^{-6})}, \tag{66}$$

$$\hat{\rho}_{corr} = \frac{\rho_{corr}}{\rho_{corr} + 0.28}, \tag{67}$$

$$\mathcal{O} = 0.50\xi + 0.30 f_{high} + 0.20\hat{\rho}_{corr}. \tag{68}$$

where $f_{high} \in [0, 1]$ is the fraction of sampled map cells with risk exceeding a high-risk threshold.

The pressure score encodes operational urgency, dynamic-threat activity, mission scale, and threat density:

$$\mathcal{P} = 0.38\hat{n}_{dyn} + 0.22\hat{u} + 0.25\hat{d}_{mission} + 0.15\tau, \tag{69}$$

where

$$\hat{n}_{dyn} = \min\left(\frac{n_{dyn}}{3}, 1\right); \quad \hat{d}_{mission} = \min\left(\frac{d_{mission}}{120000}, 1\right); \quad \hat{u} = \begin{cases} 1, & u = \text{HIGH}, \\ 0, & u = \text{LOW}. \end{cases} \tag{70}$$

The three sub-scores are combined into the fused tactical risk index ($\mathcal{F}$) using a weighted linear combination:

$$\mathcal{F} = \alpha\mathcal{C} + \beta\mathcal{O} + \gamma\mathcal{P}, \tag{71}$$

where the weighting coefficients represent the following tactical priorities:

- Alpha (α) coefficient: This represents the weight of the Capability ($\mathcal{C}$) metric, denoting the local severity of the threat field and the density of threat sources.
- Beta (β) coefficient: This represents the weight of the Opportunity ($\mathcal{O}$) metric, denoting the degree of exposure along the current mission corridor.
- Gamma (γ) coefficient: This represents the weight of the Pressure ($\mathcal{P}$) metric, denoting the operational burden exerted by dynamic threats, urgency, mission distance, and threat density.

In this thesis, empirical weights are set to $\alpha = 0.34$, $\beta = 0.41$, and $\gamma = 0.25$, yielding the final tactical index:

$$\mathcal{F} = 0.34\mathcal{C} + 0.41\mathcal{O} + 0.25\mathcal{P}. \tag{72}$$

Based on the fused tactical risk score ($\mathcal{F}$) and secondary mission indicators, the system selects the most suitable planning mode. It engages the RL-Pilot mode for low-risk, short-range, static-threat conditions; switches to the more robust T-GnP planner under high-risk or dynamic-threat conditions; and uses K-GNP for intermediate tactical complexity.

Algorithm 1 summarises the rule-based selection policy. The policy maps the fused risk index and secondary indicators to one of three planning modes: RL-Pilot (fast reactive policy), K-GNP (kinematic guidance planner), or T-GnP (tactical kinodynamic planner).

Algorithm 1 RL Advisor Planner Selection Policy

mission descriptor $\mathbf{m}$, fused risk score $\mathcal{F}$, corridor ratio ξ, normalized corridor risk $\hat{\rho}_{\text{corr}}$, high-risk area fraction f_{high}, dynamic threat count n_{dyn}, total threat count n_{threat}, mission distance d_{mission} selected planner π^* $n_{\text{dyn}} \geq 2$ **and** $(\xi \geq 0.38$ **or** $\hat{\rho}_{\text{corr}} \geq 0.42)$

$\pi^* \leftarrow$ T-GNP ▷ Dynamic high-exposure corridor $n_{\text{dyn}} = 0$ **and** $n_{\text{threat}} \leq 3$ **and** $d_{\text{mission}} < 45000$ **and** $\mathcal{F} < 0.27$ **and** $\hat{\rho}_{\text{corr}} < 0.22$ **and** $f_{\text{high}} < 0.10$

$\pi^* \leftarrow$ RL-PILOT ▷ Low-risk, short-range, static-threat setting $\mathcal{F} \geq 0.56$ **or** $\xi \geq 0.50$ **or** $\hat{\rho}_{\text{corr}} \geq 0.46$ **or** $f_{\text{high}} \geq 0.28$ **or** $(n_{\text{dyn}} \geq 1$ **and** $\hat{\rho}_{\text{corr}} \geq 0.34)$

$\pi^* \leftarrow$ T-GNP ▷ High-risk or dynamically constrained case

$\pi^* \leftarrow$ K-GNP ▷ Intermediate tactical complexity

π^*

4.10. Experimental Setup and Benchmark Scenarios

To evaluate the proposed kinodynamic planning framework in a consistent and meaningful way, a structured experimental setup was designed. Instead of relying on a single test case, multiple benchmark scenarios were used in order to capture different operational conditions. In this context, the evaluation focuses not only on path generation quality but also on execution behavior, safety, and the ability of the system to adapt to varying levels of complexity.

The overall design of the experiments follows a progressive difficulty strategy. The scenarios are constructed such that each one introduces additional challenges compared to the previous case. These challenges include increased threat density, more restrictive maneuvering corridors, longer mission ranges, and the presence of dynamic threats. As a result, different aspects of the framework can be observed more clearly, including perception accuracy, planner robustness, and adaptive decision-making behavior.

Rather than treating each scenario independently, they are considered as part of a unified benchmark suite. This allows the system to be evaluated under both nominal and highly

demanding conditions, which is particularly important for tactical applications where environmental uncertainty plays a significant role.

The first scenario, S1_Base, represents an open-terrain environment with relatively low threat density. In this case, the available maneuvering space is wide, and the primary objective is to assess basic flyability and nominal planner behavior. Since the constraints are mild, this scenario provides a reference point for comparing classical and proposed methods under standard conditions.

In contrast, S2_Dense introduces a much more restrictive environment. Here, threats are placed in clustered configurations, often forming narrow corridors that limit maneuverability. Under these conditions, the planner must satisfy the minimum turn radius constraint while simultaneously avoiding high-risk regions. This makes it possible to observe the difference between purely geometric planning and kinodynamically aware approaches, particularly in terms of feasibility and safety.

The third scenario, S3_Long, focuses on long-range mission execution. In this case, the operational distance is significantly increased, while the threat distribution remains relatively sparse but strategically positioned. The emphasis shifts from local maneuvering to maintaining efficient and trackable trajectories over extended distances. This scenario is therefore useful for evaluating path efficiency, computational stability, and long-duration guidance performance.

Finally, S4_DynamicThreat represents the most complex environment in the benchmark set. In addition to static threats, moving radar and missile systems are introduced, following predefined motion patterns such as patrol-line and patrol-circle trajectories. These dynamic elements continuously alter the structure of the threat field, forcing the planner to adapt in real time. As a result, this scenario serves as a stress test for the full perception–decision–action pipeline, particularly in terms of robustness and adaptive routing capability. The characteristics of all scenarios are summarized in Table 4.2.Taken together, these environments provide a comprehensive evaluation framework that captures a wide range of operational conditions, from relatively simple planning tasks to highly dynamic

Table 4.2 Summary of the benchmark scenarios used in the thesis.

Component	Parameter	Value
Aircraft model	Cruise speed V_{cruise}	220 m/s
	Max bank angle ϕ_{max}	60°
	Min turn radius R_{min}	1500 m
Simulation	Guidance time step Δt	0.25 s
	Lookahead distance d_{LA}	1000 m
	Speed time constant τ_V	1.8 s
DNN-TRE	Hidden dims	(128, 128, 64)
	Batch size	256
	Requested epochs	320
	Early-stop patience	40
	Best epoch	57
	Best validation loss	0.06122
	Training scenarios	S1, S2, S3
	Dataset size	24,000
RL-Pilot	State dimension	8
	Action dimension	5
	Default training episodes	4000
	Max steps per episode	360
	Replay buffer size	50,000
	Batch size	128
	Discount factor γ	0.99
	Learning rate	10^{-3}
	Final ε_{min}	0.05

and constrained tactical situations. This multi-scenario design makes it possible to assess not only the overall performance of the proposed method, but also its ability to generalize across different mission profiles. subcaption subfigure

5. EXPERIMENTAL RESULTS

This chapter presents the experimental findings obtained from the benchmark scenarios and supplementary framework evaluations. The results are organized in five parts. First, the performance of the planning algorithms is analyzed on a scenario-by-scenario basis. Second,

the run-to-run stability of stochastic planners is examined in order to reveal variability across Monte Carlo trials. Third, the tactical risk mapping performance of the proposed DNN-TRE module is evaluated against the geometric baseline. Fourth, the final benchmark performance of the RL-Pilot is summarized across all scenarios. Finally, the behavior of the neuro-adaptive framework is assessed in terms of planner selection and adaptive routing.

5.1. Benchmark Results of the Planning Algorithms

5.1.1. S1 Base Results

This subsection looks at how well the planners did in the open-terrain baseline scenario. Because S1_Base isn't very complicated tactically, it can be used as a general guide for seeing how things normally go. This situation lets us test basic qualities like how smooth the trajectory is, how well it flies, and how stable the execution is, all without the stress of being in a very dangerous situation.Figure 5.1 shows that practically all planners attain the aim successfully in this environment. Classical algorithms like A-Star and Dijkstra work well and find paths that don't cross one other. However, their answers are mostly based on geometric shortest-path calculations. These pathways may not always have the kinematic realism needed for high-speed fixed-wing flying because they don't take flight restrictions into account.The proposed K-GNP and T-GnP planners, on the other hand, include the minimal turn radius ($R_{\min}$) directly in the planning loop. This integration results in trajectories that are far smoother and more congruent with fixed-wing motion. In a practical sense, these routes are more "flight-ready" because they need less changes while being carried out and better show how people actually move around.The numbers in Table 5.1 show that the RL-Pilot does the best overall execution in this baseline situation. It had a success rate of $29/30$ and the shortest average path length (38.15 km), the shortest mission duration (158.94 s), and the lowest mean tracking error (25.85 m). Deterministic planners like A-Star, K-GNP, and T-GnP had 100% success rates ($30/30$), which showed that S1_Base is not tactically limiting. However, stochastic planners like RRT-Star and PSO were not as reliable. The success rates

for these methods were $23/30$ and $24/30$, respectively, while the tracking errors were much more.

In terms of safety, all deterministic planners and the RL-Pilot provide basically zero average threat exposure and no threat violation in their successful runs, which suggests that open-terrain baseline is better suited for measuring route performance and smoothness than survival in an extremely risky situation. In this particular case, the Neuro-Adaptive algorithm has selected the T-GnP strategy mode, thus having its trajectory identical to the T-GnP one and not being included into the comparison map.

Overall, S1_Base findings form the basis for this thesis by presenting the benchmark for the proposed planners: under moderate circumstances, they can deliver safe, efficient, and dynamically executable routes along with low tracking errors and very high mission reliability.

As Table 5.1 demonstrates, the open-terrain baseline does not penalize much deterministic algorithms, as A-Star, Dijkstra, K-GNP, T-GnP, and Neuro-Adaptive have provided perfectly executed missions. However, the RL-Pilot provides the highest effectiveness: being the fastest with the lowest tracking error and almost perfect mission reliability, it also delivers the shortest possible trajectory of the proposed algorithms.

Table 5.1 Benchmark summary for S1_Base.

Algorithm	**Success**	**Path (km)**	**Time (s)**	**Risk (M)**	**AGL (m)**	**Viol.**	**Track (m)**	**Calc. (s)**
A-Star	30/30	39.64	163.30	0.00	656	0	31.63	0.00
Dijkstra	30/30	39.92	167.44	0.00	677	0	61.56	0.00
RRT-Star	23/30	42.65	181.51	0.04	702	0	357.95	2.01
PSO	24/30	55.48	235.30	0.03	686	0	873.09	2.00
K-GNP	30/30	39.57	166.28	0.00	718	0	40.11	0.00
T-GnP	30/30	41.64	166.69	0.00	730	0	69.79	0.00
RL-Pilot	29/30	**38.15**	**158.94**	0.00	705	0	**25.85**	0.34
Neuro-Adaptive	30/30	41.64	166.59	0.00	728	0	70.11	0.00

Figure 5.1 further confirms that the proposed planners generate smoother and more dynamically compatible trajectories than the stochastic baselines.

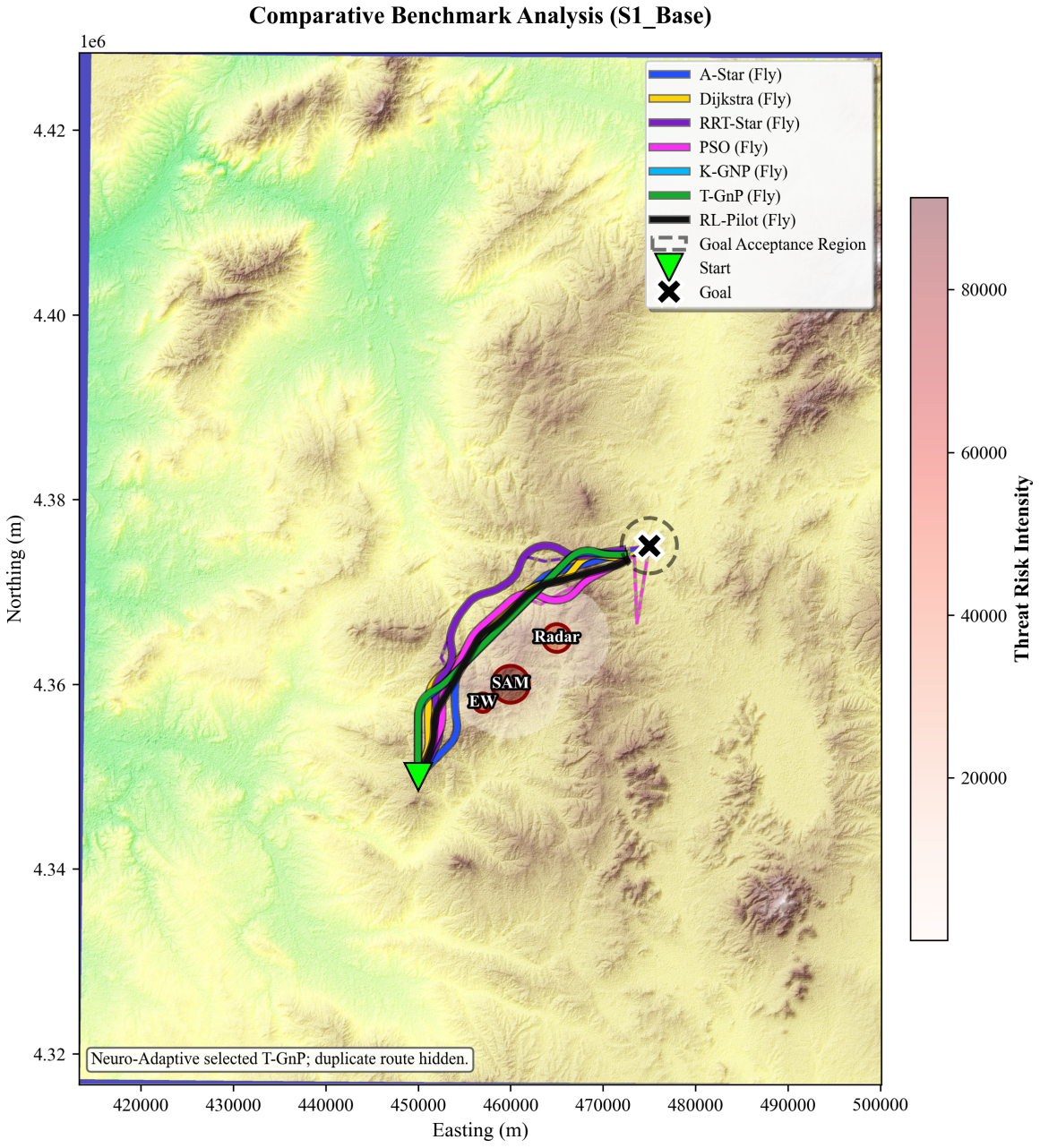

Figure 5.1 Comparative flight trajectories of the evaluated algorithms in the S1_Base scenario. The Neuro-Adaptive route is not separately shown because it selected the T-GnP mode and produced an overlapping trajectory.

5.1.2. S2 Dense Results

This sub-section highlights the results for the benchmark on the dense threat scenario. In contrast to the base line scenario, the dense threat scenario is highly restrictive: the path planners must be able to move around tight passageways yet remain feasible and clear of obstacles. As a result, the use of S2_Dense serves to highlight the differences between the two.

As shown in Figure 5.2, In case the radar and SAM systems cluster into a small area, a restricted space channeling towards the target will emerge. For this configuration, the deterministic planners, such as A-Star, Dijkstra, K-GNP, T-GnP, and Neuro-Adaptive, achieved the same results by succeeding 30 out of 30 attempts. The RL-Pilot performed just as good, achieving another impressive result with $30/30$ accuracy, which shows its stability against growing tactical difficulties. Meanwhile, the stochastics failed in providing consistent performance: RRT-Star scored 23 out of 30, while PSO succeeded 17 out of 30.

These statistics can be further illustrated with figures presented in Table 5.2. Among all other methods, the RL-Pilot was able to create an optimal flight plan based on minimum path length equal to 42.60 km, fastest flight time at 178.88 seconds, and minimum tracking error measured in 25.49 meters. Both A-Star and Dijkstra have achieved maximum performance and reached the target, but they had slightly larger path lengths with a higher degree of tracking error, especially in the case of Dijkstra. All three planning algorithms gave similar path lengths in their solutions. However, what really differentiates the three is seen in the behavior of their respective trajectories during execution. Specifically, T-GnP and Neuro-Adaptive planning systems were able to maintain consistent flight trajectories without any violations. Thus, it proves that the proposed planner is resilient even in cases where the motion requirements are very tight. Moreover, it should be mentioned that Neuro-Adaptive path is not shown separately in Fig. 5.2since both paths are exactly the same for this case.

At the same time, the stochastic planning algorithms failed in terms of consistency and trajectory execution. The tracking error for RRT-Star and PSO amounted to $377.61\ m$ and

$600.64\ m$, respectively. In addition, the latter also had some non-zero risk exposure values. Consequently, this means that the two can find feasible paths from time to time but have worse stability and resilience in narrow and risky areas.

Thus, it can be said that dense-threat routing is more than just geometric path-planning problem. Maneuver feasibility and execution dynamics should also be considered. Therefore, RL-Pilot proved to be very efficient numerically, while kinodynamic planners provided consistent physical trajectories.

Table 5.2 Benchmark summary for S2_Dense.

Algorithm	**Success**	**Path (km)**	**Time (s)**	**Risk (M)**	**AGL (m)**	**Viol.**	**Track (m)**	**Calc. (s)**
A-Star	30/30	44.09	182.41	0.00	783	0	27.72	0.00
Dijkstra	30/30	44.45	186.48	0.00	821	0	53.07	0.00
RRT-Star	23/30	48.76	210.98	0.08	710	0	377.61	3.01
PSO	17/30	53.17	227.93	0.07	711	0	600.64	3.01
K-GNP	30/30	44.52	188.28	0.00	746	0	37.78	0.00
T-GnP	30/30	44.52	187.97	0.00	744	0	38.60	0.00
RL-Pilot	30/30	**42.60**	**178.88**	0.00	812	0	**25.49**	0.47
Neuro-Adaptive	30/30	44.52	187.72	0.00	744	0	35.44	0.00

Table 5.2 further confirms that dense-threat routing places stronger constraints on stochastic planners compared to the proposed execution-aware methods. While deterministic and kinodynamic planners maintain full success, the RL-Pilot provides the best overall balance between efficiency and tracking accuracy. Figure 5.2 also illustrates that feasible solutions are confined to a narrow corridor shaped by overlapping radar and SAM coverage.

5.1.3. S3 Long Results

In this subsection, the results for the long range mission scenario are discussed. As the focus is not on tight control, but on executing reliably over a longer distance, S3 Long will be primarily used to test the ability of the planners to maintain behavior stability and safety throughout the mission.

From Figure 5.3, we see that all of the deterministic planners were able to successfully execute their missions and maintain safe separation from any threats in their trajectory, for

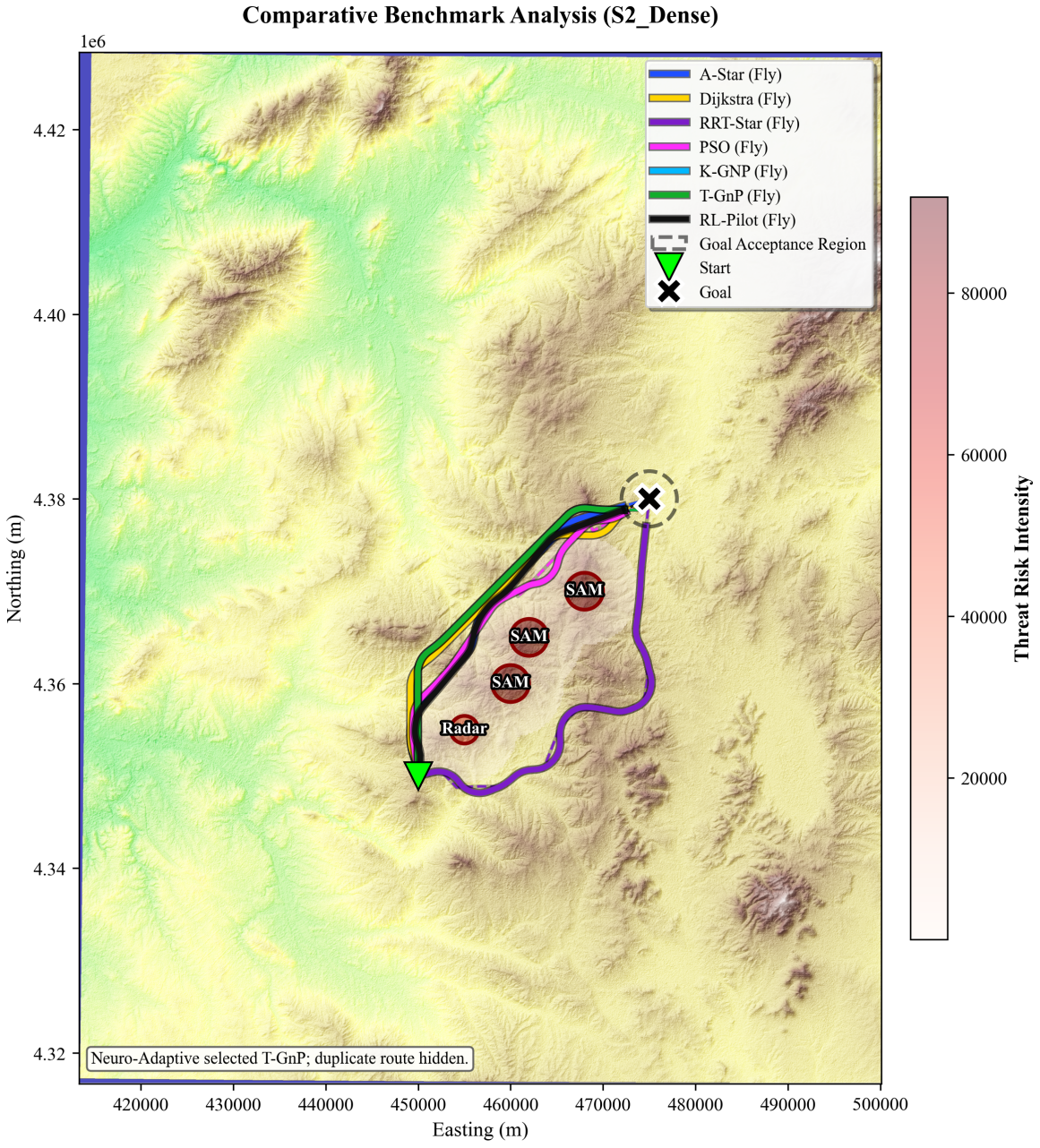

Figure 5.2 Comparative flight trajectories of the evaluated algorithms in the S2_Dense scenario. The dense SAM and radar layout forces the planners to operate within a narrow tactical corridor. The Neuro-Adaptive route is omitted because it selected the T-GnP mode and produced an overlapping trajectory.

both the middle and end parts of the mission path. For these tests, all A-Star, Dijkstra, K-GNP, T-GnP, RL-Pilot, and Neuro-Adaptive managed to achieve a $30/30$ success rate. With respect to the stochastic planners, the results of RRT-Star and PSO were respectively $27/30$ and $28/30$.

According to Table 5.3, the RL-Pilot algorithm had the best execution performance, showing not only the shortest path length ($50.45\ km$) and mission duration ($216.61\ s$), but also having the least tracking error ($24.70\ m$). Furthermore, it achieved these results while being fully safe, i.e., having 0% mission risk and zero violations.

The deterministic planners exhibit good reliability. Mission success is achieved by both K-GNP, T-GnP, and Neuro-Adaptive planners without any average risk value, with relatively low tracking errors as well. The completion time is about the same for all the three planners: K-GNP takes 225.26 s, while T-GnP – 225.55 s, and Neuro-Adaptive – 227.50 s. The tracking error remains relatively the same; therefore, one can assume that those three planners provide feasible trajectories which are suitable for a fixed-wing aircraft. On the contrary, A-Star and Dijkstra algorithms finish all missions successfully, yet with higher values of tracking error (36.76 m and 44.75 m correspondingly).

Stochastic planners are still worse in all metrics except path validity. Both RRT-Star and PSO planners demonstrate longer mission route distances (55.71 km and 57.48 km), higher computation times ((242.56 s and 251.21 s) correspondingly), and greater tracking error (364.12 meters for RRT-Star, and 359.84 meters for PSO respectively). Besides, both planners display some average risk values (0.12 million units for RRT-Star, and 0.04 million units for PSO), which means that sometimes they work closer to threats zones.

In this scenario, the Neuro-Adaptive framework again selects the T-GnP mode. As a result, its trajectory overlaps with the T-GnP solution and is omitted from Figure 5.3 for clarity. This behavior is consistent with the selection logic of the framework, which favors a robust kinodynamic planner for longer and more structured missions.

Overall, the results in S3_Long indicate that long-range planning requires not only successful avoidance of threats, but also stable execution over time. In this context, the RL-Pilot achieves the best numerical performance, while K-GNP, T-GnP, and Neuro-Adaptive maintain highly reliable and physically feasible solutions across all runs.

Table 5.3 Benchmark summary for S3_Long.

Algorithm	**Success**	**Path (km)**	**Time (s)**	**Risk (M)**	**AGL (m)**	**Viol.**	**Track (m)**	**Calc. (s)**
A-Star	30/30	52.83	220.61	0.00	750	0	36.76	0.00
Dijkstra	30/30	52.98	223.64	0.00	791	0	44.75	0.00
RRT-Star	27/30	55.71	242.56	0.12	699	0	364.12	5.02
PSO	28/30	57.48	251.21	0.04	695	0	359.84	5.00
K-GNP	30/30	53.00	225.26	0.00	786	0	33.55	0.00
T-GnP	30/30	53.00	227.50	0.00	787	0	32.30	0.00
RL-Pilot	30/30	**50.45**	**216.61**	0.00	748	0	**24.70**	0.56
Neuro-Adaptive	30/30	53.00	225.55	0.00	786	0	33.93	0.00

Table 5.3 shows that deterministic planners and the RL-Pilot maintain full success across all runs, while the stochastic methods remain less consistent. Among all approaches, the RL-Pilot achieves the shortest path, lowest mission time, and smallest tracking error, indicating the strongest overall execution performance in the S3_Long scenario.

5.1.4. S4_DynamicThreat Results

This scenario 4 is set to dynamic threat scenarion so the results achieved in the dynamic-threat scenario, which is regarded as the most challenging case within the benchmark set. The threats are motion based so the issue becomes time-sensitive, requiring planners to adjust not only to spatial limitations but also to changing threat scenarios.

As illustrated in Figure 5.4, the dynamic IADS configuration significantly reshapes the feasible route between the start and goal locations. In such a setting, purely geometric feasibility is no longer sufficient. Instead, the planner must maintain temporal consistency with the motion of the threats. Under these conditions, A-Star, Dijkstra, K-GNP, T-GnP, and Neuro-Adaptive all achieved a success rate of $30/30$, while the RL-Pilot achieved $27/30$. The stochastic baselines showed a much stronger degradation, with RRT-Star succeeding in

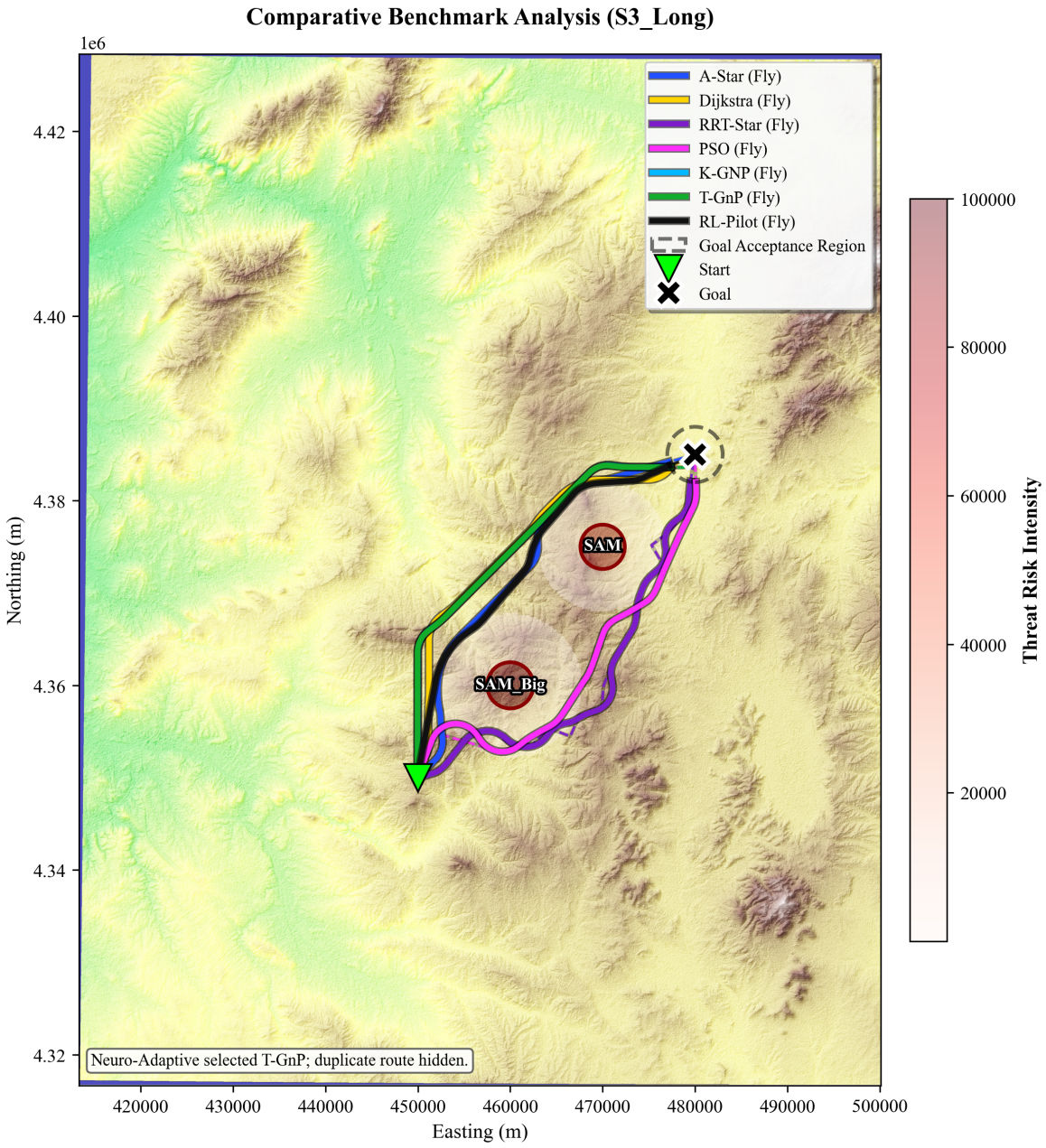

Figure 5.3 Comparative flight trajectories of the evaluated algorithms in the S3_Long scenario. The figure highlights the extended mission corridor and the need for stable long-range execution around strategically positioned threats. The Neuro-Adaptive route is omitted because it selected the T-GnP mode and produced an overlapping trajectory.

only $5/30$ runs and PSO in $13/30$ runs. This drop indicates that planners relying on sampling or stochastic exploration struggle when the environment changes over time.

The quantitative results in Table 5.4 show that the RL-Pilot achieves the strongest overall execution performance. It produces the shortest average path length (99.05 km), the lowest mission time (436.81 s), and the smallest tracking error (21.53 m), while maintaining a very low average risk exposure of 0.01 M. These results suggest that the learned policy is able to react effectively to moving threats while maintaining stable flight behavior.

At the same time, the deterministic planners maintain full mission reliability. K-GNP, T-GnP, and Neuro-Adaptive all achieve $30/30$ success with zero average risk and no violations. Among these, K-GNP yields the smallest tracking error (25.00 m), while Neuro-Adaptive completes the mission in 452.48 s after selecting the T-GnP mode. Although the T-GnP-based trajectories are slightly longer than those of A-Star and Dijkstra, this behavior can be interpreted as a more conservative response to dynamic exposure regions rather than a limitation of the method.

The difference becomes more apparent when considering the stochastic planners. RRT-Star and PSO produce longer and less stable trajectories, with tracking errors of 363.82 m and 123.37 m, respectively. In addition, their non-zero risk exposure values (0.11 M and 0.18 M) indicate that these planners occasionally operate closer to threat regions. Their computation times, around 6 s per run are considerably higher than those of the deterministic approaches, which further diminishes their practicality in dynamic settings.

The conduct of the Neuro-Adaptive system aligns with the decision logic. The DNN-TRE estimations indicate that the situation is marked by considerable exposure and changing pressure ($\rho_{\text{peak}} = 1.15$, $\bar{\rho} = 0.52$, $\rho_{\text{corr}} = 0.54$, $f_{\text{high}} = 0.33$), resulting in capability, opportunity, and pressure scores of 0.60, 0.47, and 0.73, respectively, and a fused score of 0.58.In response to this condition, the framework selects the T-GnP mode, which aligns with the increased tactical complexity. The Neuro-Adaptive trajectory is omitted from Figure 5.4, as it points of intersection with the T-GnP solution.

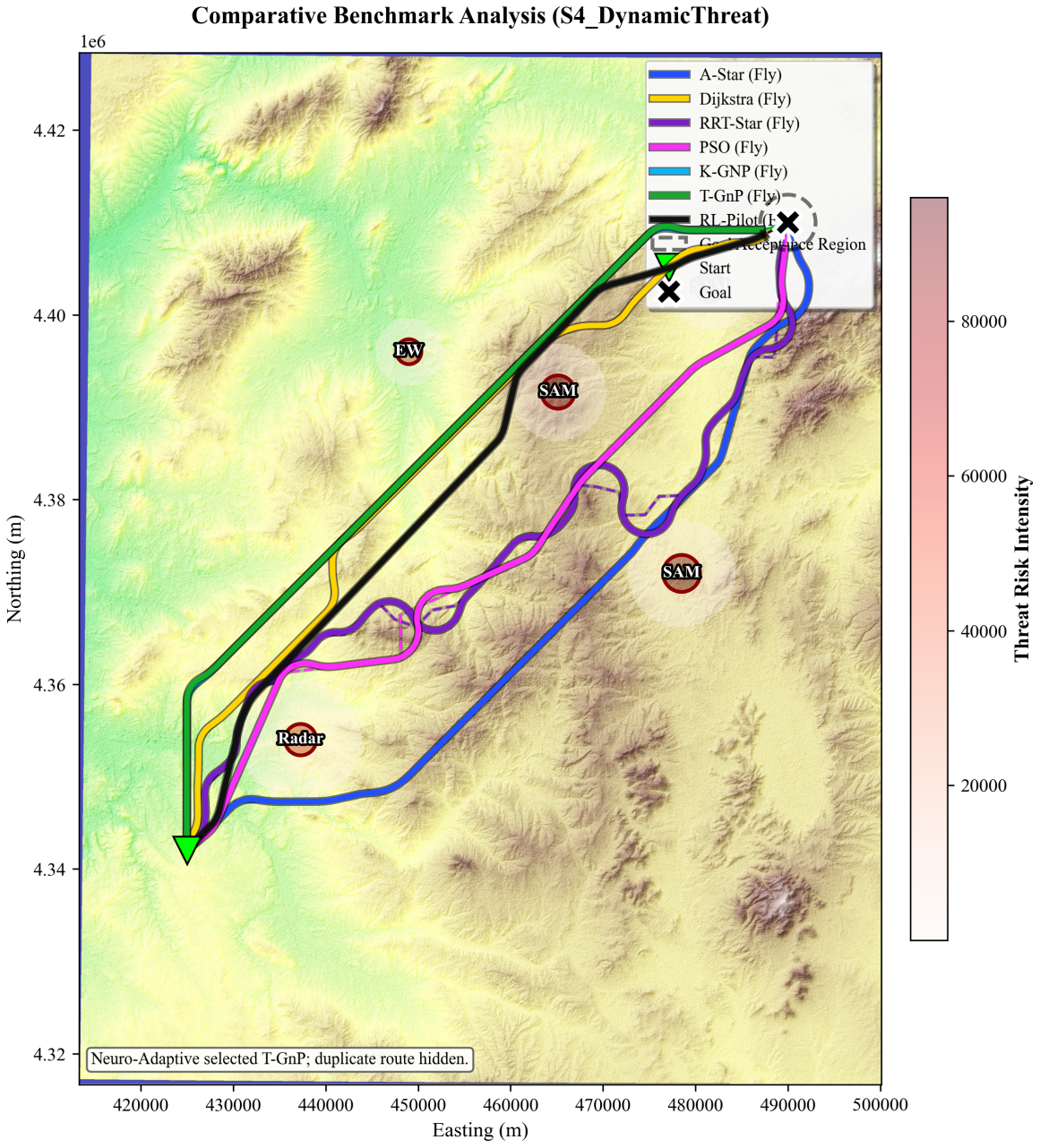

Figure 5.4 Comparative flight trajectories of the evaluated algorithms in the S4_DynamicThreat scenario. The moving threat structure creates dynamically changing tactical corridors. The Neuro-Adaptive route is omitted because it selected the T-GnP mode and produced an overlapping trajectory.

Overall, the results in S4_DynamicThreat show that dynamic routing cannot be treated as a static path planning problem. Instead, it requires coordinated perception, decision-making, and trajectory generation. In this scenario, the RL-Pilot provides the best route efficiency and tracking performance, while K-GNP, T-GnP, and Neuro-Adaptive maintain full mission reliability.

Table 5.4 Benchmark summary for S4_DynamicThreat.

Algorithm	**Success**	**Path (km)**	**Time (s)**	**Risk (M)**	**AGL (m)**	**Viol.**	**Track (m)**	**Calc. (s)**
A-Star	30/30	105.95	460.15	0.00	585	0	29.30	0.00
Dijkstra	30/30	102.37	448.99	0.00	691	0	41.61	0.00
RRT-Star	5/30	111.83	498.60	0.11	717	0	363.82	6.01
PSO	13/30	104.44	461.71	0.18	754	0	123.37	6.01
K-GNP	30/30	102.79	453.82	0.00	731	0	25.00	0.00
T-GnP	30/30	109.18	452.97	0.00	731	0	41.55	0.00
RL-Pilot	27/30	**99.05**	**436.81**	0.01	706	0	**21.53**	3.10
Neuro-Adaptive	30/30	109.18	452.48	0.00	731	0	36.65	0.00

Table 5.4 confirms that the dynamic-threat scenario is the most demanding case in the study. While deterministic and kinodynamic planners maintain full reliability, the stochastic baselines show a significant drop in performance. Although the RL-Pilot does not achieve perfect success, it still provides the best overall efficiency and tracking accuracy, while the Neuro-Adaptive framework correctly switches to T-GnP under high dynamic pressure.

5.2. Run-to-Run Stability of Stochastic Planners

Monte carlo runs are used on stochastic planners such as RRT-Star, PSO, and RL-Pilot to demonstrate the variability and get average benchmark results. Analysis is essential, as average performance alone does not adequately represent the consistency of a planner's behavior across repeated trials. The results of the study demonstrate that stochastic planners exhibit heightened sensitivity with increasing scenario complexity. In more limited or dynamic contexts, sampling-based and swarm-based approaches exhibit less steady convergence, but the RL-Pilot demonstrates a more consistent performance profile. The distinction is especially visible in S2_Dense and S4_DynamicThreat, where the outcome

distribution strongly separates the obtained policy from the traditional stochastic baselines. On the figure 5.5 demonstration of the stochastic run outcomes. The black line shows the cumulative success curve. This figure helps to understand that all the stochastic runs are determined and consistent to reach the goal.

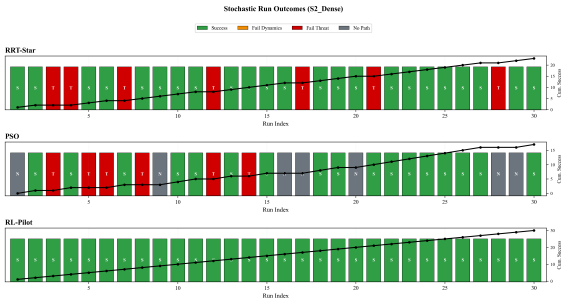

Figure 5.5 Stochastic Run Outcomes in S2

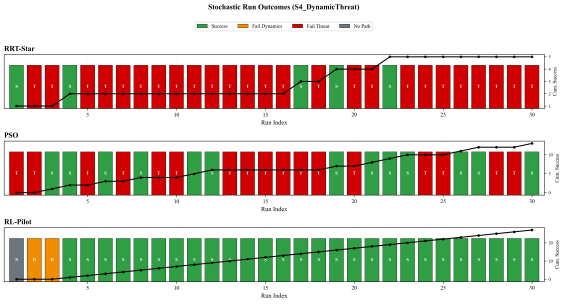

Figure 5.6 Stochastic Run Outcomes in S4.

Figure 5.7 Run-to-run outcome distribution of the stochastic planners in the S2 Dense and S4 Dynamic Threat scenario.

5.3. DNN-TRE Tactical Risk Mapping Results

This subsection evaluates the DNN-TRE module as a standalone risk estimation component. The main objective is to determine whether the learned model can preserve the overall spatial structure of the tactical risk field while significantly reducing computational cost compared to the geometric baseline.

The quantitative comparison with the geometric G-LOS baseline is summarized in Table 5.5.

As summarized in Table 5.5, the DNN-TRE consistently reproduces the main structure of the risk field while requiring significantly less computation time. In S1_Base, the model achieves a 2.3$\times$ speedup with very low error and near-perfect spatial correlation. A similar trend is observed in S2_Dense, where the model maintains strong correlation despite the increased complexity of the threat layout.

In the S3_Long scenario, the model preserves the large-scale spatial organization of the risk field, indicating that it captures the underlying terrain-dependent structure rather than memorizing individual patterns. The most challenging case is S4_DynamicThreat, which

Table 5.5 Comprehensive performance comparison of the DNN-TRE tactical risk mapping against the geometric baseline (G-LOS) across all scenarios.

Metric	**S1_Base**	**S2_Dense**	**S3_Long**	**S4_DynamicThreat**
Computational Performance				
Baseline Time (s)	116.86	137.39	74.71	164.94
DNN-TRE Time (s)	50.18	45.39	44.99	48.00
Speedup Ratio	**2.3**×	**3.0**×	**1.7**×	**3.4**×
Approximation Accuracy				
Mean Absolute Error (MAE)	0.0102	0.0202	0.0157	0.0640
Root Mean Square Error (RMSE)	0.0255	0.0530	0.0531	0.1300
Spatial Correlation	**0.9868**	**0.9729**	**0.9555**	**0.9345**

was not included during training. Even in this setting, the model achieves a 3.4× speedup and maintains a high correlation value of 0.9345, despite a moderate increase in approximation error.

Overall, these results suggest that the DNN-TRE provides a practical balance between computational efficiency and spatial accuracy, making it suitable for use in the perception layer of the proposed framework.

This subsection summarizes the final benchmark performance of the RL-Pilot within the local evaluation framework. Unlike the methodology section, which focuses on model design and training, the discussion here is limited to the observed scenario-level performance.

Across the benchmark suite, the RL-Pilot demonstrates strong performance, particularly in the first three scenarios. It achieves 11/12 success in S1_Base and perfect success (12/12) in both S2_Dense and S3_Long. In these cases, the planner produces relatively short trajectories with low tracking errors of 27.30 m, 34.10 m, and 25.83 m, respectively. This indicates that the learned policy generalizes well across different mission structures.

The most challenging case remains S4_DynamicThreat, where the success rate decreases to 9/12. Despite this reduction, the planner still maintains a short path length of 99.05 km and

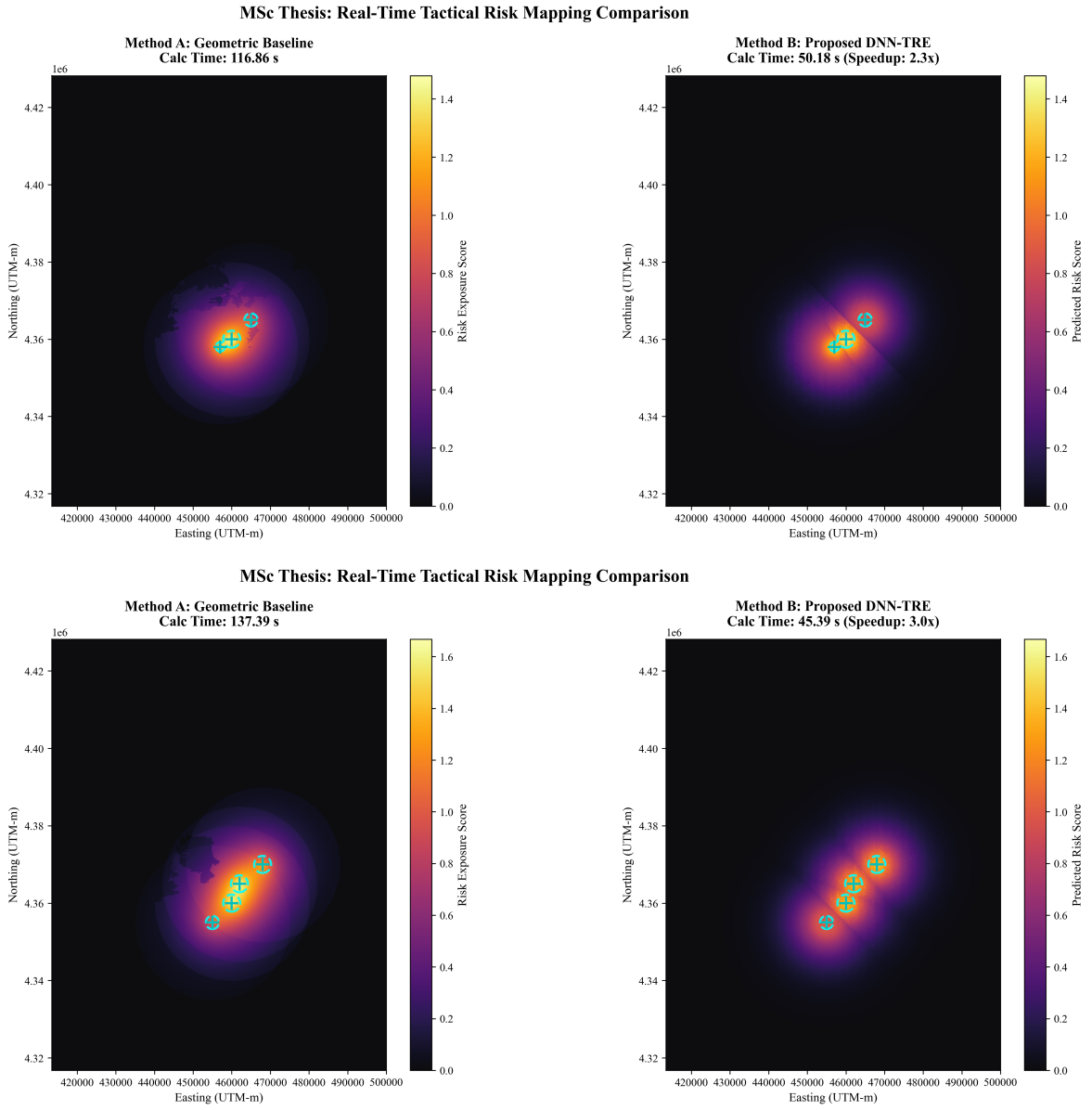

Figure 5.8 Visual comparison of tactical risk fields: G-LOS vs. DNN-TRE for S1 and S2 scenarios.

a low tracking error of 24.32 m. This suggests that the primary limitation lies in maintaining full reliability rather than in trajectory quality itself.

Overall, the RL-Pilot behaves as a robust reactive controller rather than a purely scenario-specific solution. Its low tracking errors indicate compatibility with the downstream guidance layer, while its path efficiency shows that it does not behave overly conservatively. However, in highly dynamic environments, its reliability remains slightly below that of

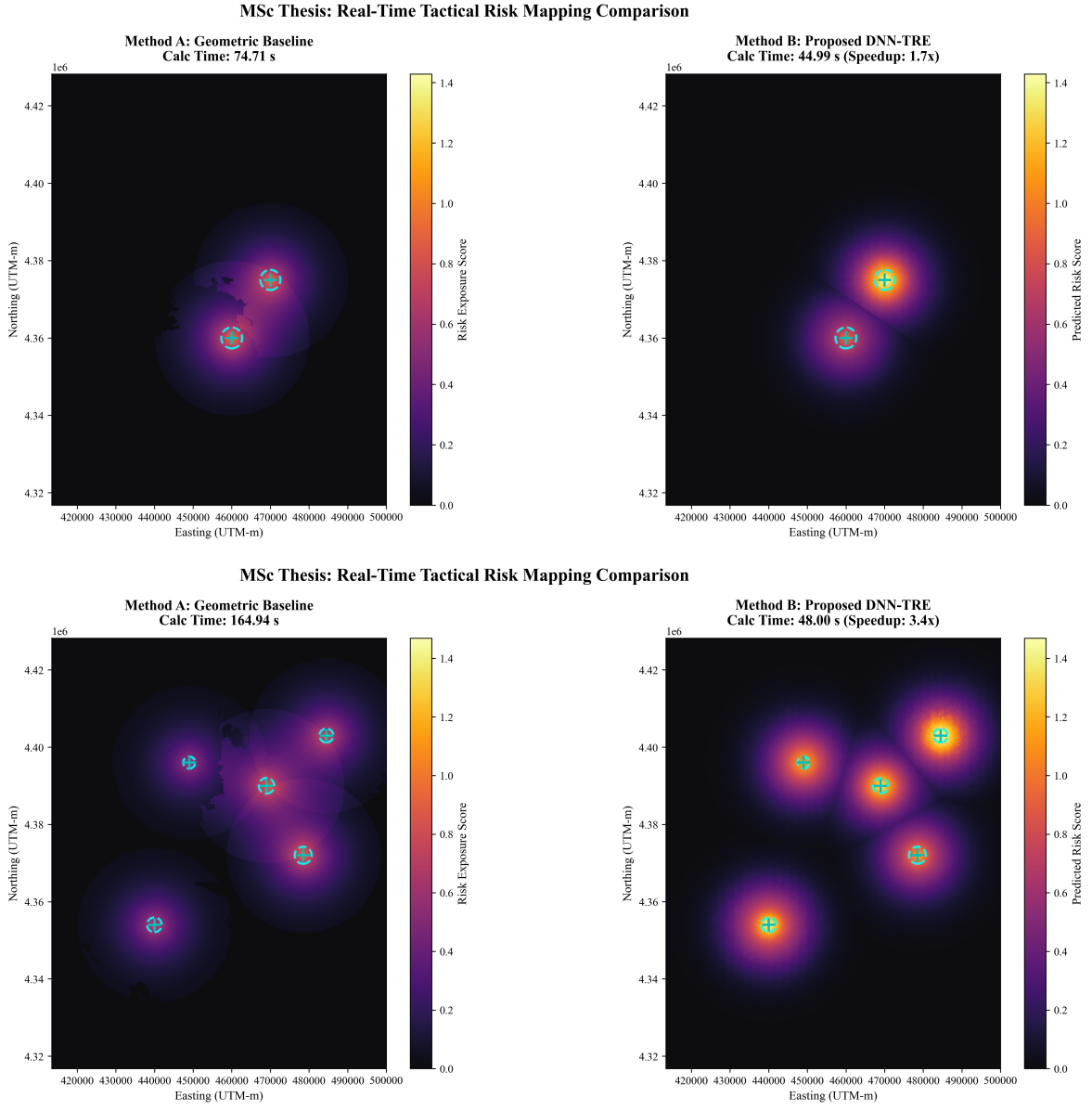

Figure 5.9 Visual comparison of tactical risk fields: G-LOS vs. DNN-TRE for S3 and S4 scenarios.

deterministic kinodynamic planners.Also, the DNN-TRE-based risk estimation is accessible to all planners through its integration into the shared feasibility engine. In this way, both DNN-enabled and non-DNN planner configurations can be evaluated under the same benchmark setting. It can be managed by GUI; however, the results show that DNN-TRE is not give changeable great performance onT-GNP and K-GNP despite RL pilot.

5.4. Final Performance of the RL-Pilot

The local evaluation system's RL-Pilot test results are examined here.

The first three cases had high completion rates, low tracking mistakes, and efficient routes. The numbers show that it scored 11/12 for S1 Base and 12/12 for S2 Dense and S3 Long. Average tracking errors were 27.30, 34.10, and 25.83 m for its short, easy-to-track courses. This suggests that the reactive policy responded to conventional, dense, and extended mission contexts rather than memorizing one configuration.

As observed, the S4 DynamicThreat scenario was the hardest, with 9 out of 12 runs successful. The planner kept the average path length to 99.05 km with a mean tracking error of 24.32 m despite moving threats. S4's biggest concern was not trajectory quality. However, the decline in success indicates that threats are continually changing, making perfect reliability challenging. The fact that the RL-Pilot acquired a realistic, reactive flight approach instead of a scenario-specific trick is promising. Low tracking errors meant its planned pathways worked well with the simulation's downstream guidance layer. The agent's path lengths and completion timings show it was not reluctant. In the toughest cases, the RL-Pilot was more vulnerable to highly dynamic, unpredictable threats than the fully deterministic kinodynamic planners.

The fact that the RL-Pilot acquired a realistic, reactive flight approach instead of a scenario-specific trick is promising. Low tracking errors meant its planned pathways worked well with the simulation's downstream guidance layer. The agent's path lengths and completion timings show it was not reluctant. In the toughest cases, the RL-Pilot was more vulnerable to highly dynamic, unpredictable threats than the fully deterministic kinodynamic

planners. The lower success rate in harsh situations highlights the limitations of reactive control, not a system failure. When the tactical scenario gets too intricate, the RL Advisor transfers control to more experienced planners like the K-GNP or T-GnP.

The statistics support the RL-Pilot's role as a fast, flexible planning mode in the neuro-adaptive framework. It is not meant to replace deterministic planners. It is efficient and reasonable for low- and moderate-risk scenarios.

Table 5.6 Final benchmark performance of the RL-Pilot in the local evaluation environment across the four benchmark scenarios.

Scenario	Success	Path (km)	Time (s)	Risk (M)	Min AGL (m)	Track Mean (m)	Track Max (m)	Calc. (s)	Sat. Ratio
S1 Base	11/12	38.15	160.45	0.00019	704.43	27.30	119.75	0.39	0.0022
S2 Dense	12/12	42.60	180.81	0.00087	811.40	34.10	137.64	0.56	0.0094
S3 Long	12/12	50.45	216.60	0.00106	749.18	25.83	122.38	0.77	0.0004
S4 DynamicThreat	9/12	99.05	435.86	0.00955	706.30	24.32	107.94	3.28	0.0034

Table 5.6 shows that the RL-Pilot maintained strong scenario-level performance in the local evaluation environment. The planner remained particularly effective in S1 Base, S2 Dense, and S3 Long, while S4 DynamicThreat continued to represent the most demanding case in terms of reliable completion under dynamic threat motion.

5.5. Assessment of the Neuro-Adaptive Framework

The final part of the results focuses on the neuro-adaptive framework as an integrated perception–decision–action system. Instead of evaluating path quality alone, the emphasis here is on how the framework adapts its behavior under different tactical conditions.

By combining Capability ($\mathcal{C}$), Opportunity ($\mathcal{O}$), and Pressure ($\mathcal{P}$) scores, the RL Advisor converts raw environmental information into a compact decision variable. In relatively simple environments such as S1_Base, the resulting low fused risk value ($\mathcal{F} < 0.33$) leads to the selection of the RL-Pilot, which prioritizes efficiency. In more constrained or dynamic scenarios such as S2_Dense and S4_DynamicThreat, the framework switches to T-GnP, which provides more robust and constraint-aware behavior.

This behavior supports the main idea of the thesis: a single planning strategy is not sufficient across all scenarios. Instead, adaptive selection of planners based on mission conditions provides a more reliable and effective solution.

5.6. AdaptCore - GUI

This GUI has been developed to run comparative experiments quickly, in a controlled and repeatable manner, by selecting benchmark scenarios, planner types, and DNN-TRE-supported or unsupported operating modes through a single interface. AdaptCore, a graphical user interface (GUI), has been designed as shown in 5.10. The

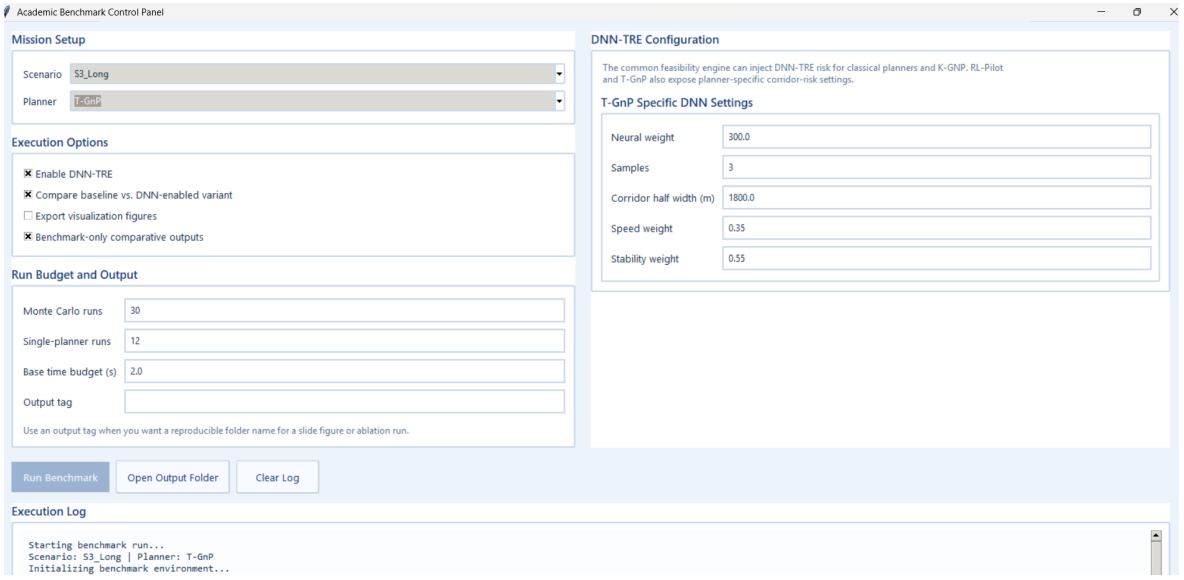

Figure 5.10 AdaptCore - GUI

GUI has DNN fine tuning settings control how strongly and over what spatial extent the DNN-TRE risk estimate influences planner decisions, thereby enabling ablation-style comparison between baseline and neural-risk-augmented planning modes. The Weight parameter determines how strongly the risk information from DNN-TRE will influence the planner's decision; if it is too low, the neural risk becomes almost ineffective, and if it is too high, the planner may act overly conservatively and create longer and more difficult routes. The Corridor Half Width parameter determines how wide a corridor, including the centerline and the surrounding areas to the right and left of a road segment, will be assessed for risk. If

it is too small, the threat structure in the immediate area may be overlooked, and if it is too large, the planner may consider an excessive area hazardous, leading to inefficient variations.

If the Samples parameter is too low, the risk area is roughly represented and local hazards may be ignored but if it is too high, the assessment is more precise but computationally expensive and planning may slow down.

6. CONCLUSION

The purpose of this thesis is to give a method for end to end path trajectory optimization by given multiple solutions and reduce the gap between the geometrically feasible paths and physically executable trajectories. The Kinematic Guidance Navigation Planner (K-GNP) is the first important contribution. It takes into consideration heading angle and minimum turning radius limits when planning. K-GNP makes sure that kinematic feasibility is maintained along the whole planned route, unlike other methods that make pathways without checking flyability. The results show that this algorithm always gives you solutions that are stable, continuous, and can fly for the tough conditions. The Tactical Guidance Navigation Planner (T-GnP) builds on this work by adding new effects relating to tactical exposure and maneuver cost to the K-GNP framework. Moreover, accounting for turn dynamics and speed-dependent behavior, T-GnP strikes a more balanced trade-off between safety and mission efficiency. Its benefits are most clear in situations that are crowded and changing, where planners that solely utilize mathematics likely to become less reliable as the environment gets more complicated. The Deep Neural Network Tactical Risk Estimator (DNN-TRE) is another important contribution. It uses a trained neural network model instead of a computationally expensive geometric risk evaluation. The findings indicate that DNN-TRE attains significant enhancements in speed while maintaining the fundamental elliptic threat-aware framework of the initial risk formulation. It is the planner system's preference that can be on or off at the planning phase. At the system level, the Neuro-Adaptive Planner integrates threat and terrain perception, mission-level decision-making, and trajectory generation within an end-to-end system. A reinforcement

learning advisor does not just use one fixed planner. Instead, it chooses amongst the RL-Pilot, K-GNP, and T-GnP modules based on the current mission environment. The results demonstrate that the adaptive selection behavior improves overall system performance. Another contribution of this work is its comprehensive evaluation methodology with Monte Carlo simulations. The framework is evaluated using a wide range of criteria, such as risk exposure, terrain clearance, tracking accuracy, dynamic saturation, and computing cost, rather than just path length or mission duration. This multi-criteria viewpoint gives us a better idea of how planners act in different situations. The results of this study indicate that tactical path planning for fixed-wing aircraft cannot be treated solely as a geometric issue. As the operational environment becomes increasingly limited or dynamically complicated, the significance of kinematic feasibility, flexibility, and execution stability substantially rises. The proposed structure completely meets these objectives and provides a feasible basis for developing real-time tactical mission planning systems.

6.1. Future Work

Despite the proposed framework shows a great enhancement for tactical path optimization, the current three degree of freedom model enhance to a extended to a full six-degree-of-freedom (6-DOF) formulation [119],incorporating advanced constraints such as angle-of-attack (AoA) limits, G-load factors, and fuel-aware optimization that allows more realistic modeling of inertial and actuator dynamics. For the threat models RCS effects can be included for the risk awareness with directonal sensing. With these enhancements allow the system to mor similar to real combat scenarios. Another futuristic work might be the using of representation of the high resolution terrain file. In the perspective of optimization, the incorporated mission-level objectives could be added that the fuel aware cost functions, timing constraints and sensor systems perspective might be considered. Including these factors would the planner to describe more complex operational trade-offs. Another proposed improvement is extending the planning with multi aircrafts. Multi-agent approaches could be used for coordinated behvaior between multiple aircraft. Wingman aircraft concepts also can be used for further applications. To sum up, the gap between the simulation and real-world

deployment remains a significant hardness. Hardware-in-the-loop (HIL) test and integration with real flight control systems tests in physical constraints with getting realistiv assessment of latency. For further applicaitons, DNN-TRE-based risk estimation is defined for accessible to all planners through its integration into the shared feasibility engine.

7. Appendix

This appendix provides additional benchmark materials that support the main results presented in Chapter 5.. These elements are not included in the main body in order to maintain clarity, but they provide further insight into intermediate evaluations and supplementary analyses.

7.1. Supplementary Benchmark Materials

The materials collected in this section include intermediate RL-Pilot evaluation results, comparisons between Colab-based and local experiments, additional stochastic outcome visualizations, and compact benchmark summary graphs.

7.2. Supplementary RL-Pilot Evaluation Tables

This section presents additional RL-Pilot benchmark tables that provide further transparency regarding the evaluation process. In particular, Table 7.1 shows the intermediate results obtained in the Colab environment, while Table 7.2 compares these values with the final local benchmark outputs used in the main analysis.

Table 7.1 Benchmark performance of the Colab-trained RL-Pilot model as observed in the Colab evaluation environment.

Scenario	Success	Path (km)	Time (s)	Risk (M)	Min AGL (m)	Track Mean (m)	Calc. (s)
S1_Base	11/12	38.68	160.18	0.00	705	27.84	0.51
S2_Dense	11/12	42.68	178.77	0.00	808	31.28	0.58
S3_Long	11/12	51.58	219.59	0.00	738	28.46	0.65
S4_DynamicThreat	9/12	99.77	439.92	0.01	724	21.97	3.89

Table 7.2 Comparison between the Colab benchmark summary and the final local benchmark artifacts for the RL-Pilot model.

Scenario	Source	Success	Path (km)	Time (s)	Risk (M)	Track Mean (m)	Calc. (s)	Remark
S1_Base	Colab	11/12	38.68	160.18	0.00	27.84	0.51	Console summary
S1_Base	Local	11/12	38.15	160.45	0.00019	27.30	0.39	Final CSV artifact
S2_Dense	Colab	11/12	42.68	178.77	0.00	31.28	0.58	Console summary
S2_Dense	Local	12/12	42.60	180.81	0.00087	34.10	0.56	Final CSV artifact
S3_Long	Colab	11/12	51.58	219.59	0.00	28.46	0.65	Console summary
S3_Long	Local	12/12	50.45	216.60	0.00106	25.83	0.77	Final CSV artifact
S4_DynamicThreat	Colab	9/12	99.77	439.92	0.01	21.97	3.89	Console summary
S4_DynamicThreat	Local	9/12	99.05	435.86	0.00955	24.32	3.28	Final CSV artifact

7.3. Supplementary Stochastic Outcome Graphics

This section presents additional stochastic run outcome graphics that complement the run-to-run stability discussion in the main text. In particular, Figure 7.1 provides the supplementary stochastic outcome distribution for the baseline scenario, while Figure 7.2 provides the corresponding supplementary result for the long-range scenario.

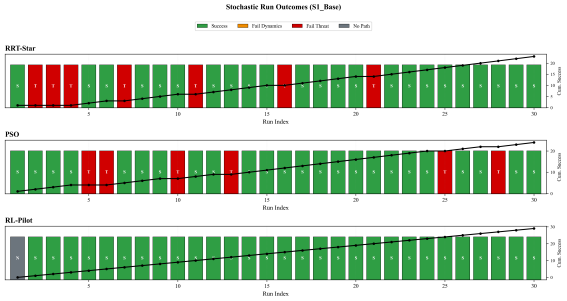

Figure 7.1 S1 Base scenario

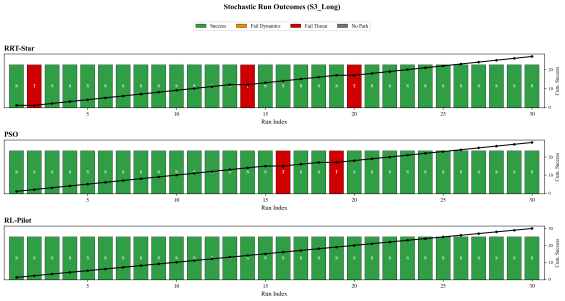

Figure 7.2 S3 Long scenario

Figure 7.3 Stochastic run outcomes for S1 and S3.

7.4. Additional Benchmark Summary Graphs

The benchmark summary graphs shown in Figures 7.3–7.6 are included here as supplementary visual summaries of the scenario-wise planner statistics. These plots are not required for the main argument of the thesis, since the quantitative comparison is already provided in the benchmark tables of Chapter 5.. However, they provide a compact graphical

overview of success rate, path length, tracking performance, bank-angle usage, and saturation behavior across the four scenarios.

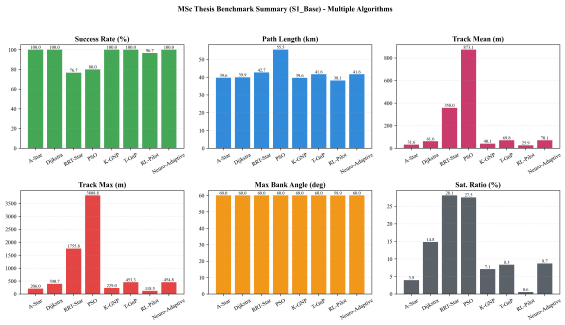

Figure 7.4 Benchmark summary graph for the S1

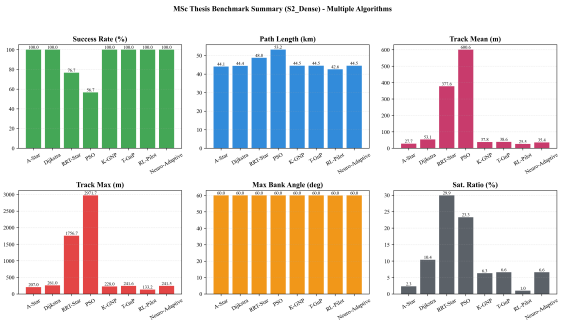

Figure 7.5 Benchmark summary graph for the S2

Figure 7.6 Supplementary benchmark summary graph for the S1 Base and S2 Dense scenarios.

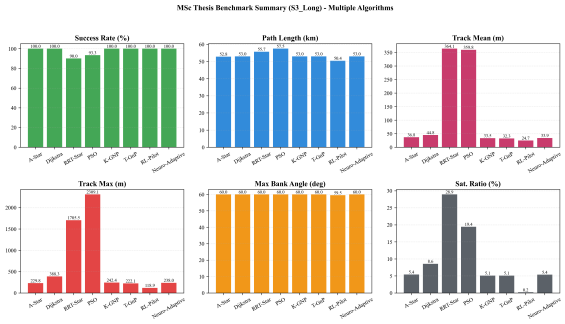

Figure 7.7 Benchmark summary graph for the S3

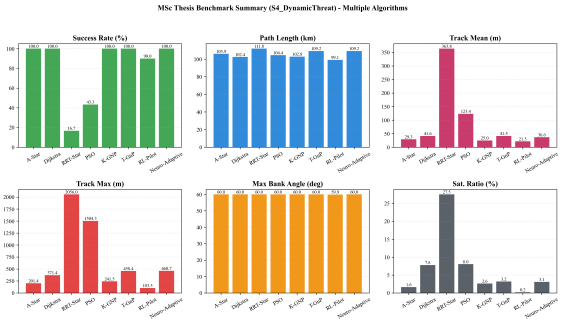

Figure 7.8 Benchmark summary graph for the S4

Figure 7.9 Benchmark summary graph for the S3 Long and S4 Dynamic Threat scenarios.

REFERENCES

[1] Longyan Xu, Mao Xi, Ren Gao, Ziheng Ye, and Zaihan He. Dynamic path planning of UAV with least inflection point based on adaptive neighborhood A* algorithm and multi-strategy fusion. *Scientific Reports*, 15, **2025**. doi:10.1038/s41598-025-92406-w.

[2] Xiong Yin, Pinglan Cai, Kangwen Zhao, Yu Zhang, Qian Zhou, and Daojin Yao. Dynamic Path Planning of AGV Based on Kinematical Constraint A* Algorithm and Following DWA Fusion Algorithms. *Sensors (Basel, Switzerland)*, 23, **2023**. doi:10.3390/s23084102.

[3] Ronald Sukwadi, Gregorius Airlangga, W. Basuki, Yoel Kristian, Radyan Rahmananta, Lai Ferry Sugianto, and Oskar Ika Adi Nugroho. Comparative Analysis of Path Planning Algorithms for Multi-UAV Systems in Dynamic and Cluttered Environments: A Focus on Efficiency, Smoothness, and Collision Avoidance. *International Journal of Robotics and Control Systems*, **2024**. doi:10.31763/ijrcs.v4i4.1555.

[4] Hao Chen, Hua Wang, and Leqi Jiang. Path planning of UAV based on cultural algorithm in dynamic environments. *2016 6th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, pages 130–134, **2016**. doi:10.1109/iceiec.2016.7589704.

[5] Yu Lei. Study on path planning for aircraft based on improved A˜* algorithm. *Systems engineering and electronics*, **2008**.

[6] Sun Li-Jun. Dynamic Programming Trajectory Planning Base on Penetration Gridding Model. *Modern Defence Technology*, **2013**.

[7] Zhe Zhang, Jian Wu, Jiyang Dai, and Cheng He. A Novel Real-Time Penetration Path Planning Algorithm for Stealth UAV in 3D Complex Dynamic

Environment. *IEEE Access*, 8:122757–122771, **2020**. doi:10.1109/access.2020.3007496.

[8] Zhe Zhang, Ju Jiang, Jian Wu, and Xiaozhou Zhu. Efficient and optimal penetration path planning for stealth unmanned aerial vehicle using minimal radar cross-section tactics and modified A-Star algorithm. *ISA transactions*, **2022**. doi:10.1016/j.isatra.2022.07.032.

[9] Huixia Zhang, Yadong Tao, and Wenliang Zhu. Global Path Planning of Unmanned Surface Vehicle Based on Improved A-Star Algorithm. *Sensors (Basel, Switzerland)*, 23, **2023**. doi:10.3390/s23146647.

[10] Tingjun Lei, C. Luo, G. Jan, and Z. Bi. Deep Learning-Based Complete Coverage Path Planning With Re-Joint and Obstacle Fusion Paradigm. *Frontiers in Robotics and AI*, 9, **2022**. doi:10.3389/frobt.2022.843816.

[11] Xia Chen, Dong Liu, and Ting Tang. Path Planning for UCAV in Dynamic and Uncertain Environments Based on Focused D* Algorithm. *2011 Fourth International Symposium on Computational Intelligence and Design*, 2:55–58, **2011**. doi:10.1109/iscid.2011.115.

[12] Xia Chen, Miaoyan Zhao, and Liyuan Yin. Dynamic Path Planning of the UAV Avoiding Static and Moving Obstacles. *Journal of Intelligent & Robotic Systems*, 99:909–931, **2020**. doi:10.1007/s10846-020-01151-x.

[13] Arthur Richards and J. How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, 3:1936–1941, **2002**. doi:10.1109/acc.2002.1023918.

[14] V. Erokhin, B. Lezhankin, and T. Portnova. Bi-criteria Aircraft Trajectory Optimization in Implementing the Area Navigation Concept. *International Journal of Aeronautical and Space Sciences*, 22:948–962, **2021**. doi:10.1007/s42405-021-00353-3.

[15] Amin Jafarimoghaddam and Manuel Soler. Multi-control commercial aircraft trajectory optimization in a vertical plane with state-inequality constraints via singular control theory. *Eur. J. Control*, 75:100930, **2023**. doi:10.1016/j.ejcon.2023.100930.

[16] Tolgahan Turker, O. Sahingoz, and G. Yilmaz. 2D path planning for UAVs in radar threatening environment using simulated annealing algorithm. *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 56–61, **2015**. doi:10.1109/icuas.2015.7152275.

[17] Fengrui Ran, Chengpu Yu, Erpei Xu, and Yunji Feng. Autonomous Uav Path Planning in Dynamic Environments: A Hybrid Framework of Trajectory Prediction and Priority-Aware DWA. *2025 IEEE 19th International Conference on Control & Automation (ICCA)*, pages 150–155, **2025**. doi:10.1109/icca65672.2025.11129782.

[18] V. Hoang and Manh Duong Phung. Enhanced Teaching-Learning-based Optimization for 3D Path Planning of Multicopter UAVs. *ArXiv*, abs/2205.15913, **2022**. doi:10.1007/978-3-030-99666-6_107.

[19] Nouman Bashir, S. Boudjit, G. Dauphin, and S. Zeadally. An obstacle avoidance approach for UAV path planning. *Simul. Model. Pract. Theory*, 129:102815, **2023**. doi:10.1016/j.simpat.2023.102815.

[20] Yao Peng, Wang Honglun, and Liu Chang. Three-dimensional rolling path planning via dynamic fluid disturbance. *Journal of Beijing University of Aeronautics and Astronautics*, 41:2280, **2015**. doi:10.13700/j.bh.1001-5965.2014.0773.

[21] Yanbiao Niu, Xue Yan, Yongzhen Wang, and Yanzhao Niu. 3D real-time dynamic path planning for UAV based on improved interfered fluid dynamical system and artificial neural network. *Adv. Eng. Informatics*, 59:102306, **2024**. doi:10.1016/j.aei.2023.102306.

[22] Xiangyu Fan, Hao Li, You Chen, and Danna Dong. A Path-Planning Method for UAV Swarm under Multiple Environmental Threats. *Drones*, **2024**. doi:10.3390/drones8050171.

[23] Pengfei Duan and Dawei Chen. Distributed Cooperative Path Planning for Multi-UAV in Information-Rich and Dynamic Environments. *Drones*, **2025**. doi:10.3390/drones9010038.

[24] Q. Xiao, X. Gao, X. Fu, and H. Wang. New local path replanning algorithm for unmanned combat air vehicle. In *2006 6th World Congress on Intelligent Control and Automation*, volume 1, pages 4033–4037. **2006**.

[25] W. Ji, C. Li, and Y. Zhang. Aircraft trajectory planning based on threat sources in uncertain environments. In *2024 4th International Signal Processing, Communications and Engineering Management Conference (ISPCEM)*, pages 693–697. **2024**.

[26] W. Lingxiao. Effective path planning method for low detectable aircraft. *Journal of Systems Engineering and Electronics*, 20(4):784–789, **2009**.

[27] Tanzeem Pasha, S. M, S. R., and B. D. Multi-UAV Path Planning using Grey Wolf Optimization and RRT Algorithm. *Journal of Soft Computing Paradigm*, **2025**. doi:10.36548/jscp.2025.2.002.

[28] Qifeng Diao, Jinfeng Zhang, Min Liu, and Jiaxuan Yang. A Disaster Relief UAV Path Planning Based on APF-IRRT* Fusion Algorithm. *Drones*, **2023**. doi:10.3390/drones7050323.

[29] Ting-Ting Wu, Zheng Zhang, Feng Jing, and Mei Gao. A Dynamic Path Planning Method for UAVs Based on Improved Informed-RRT* Fused Dynamic Windows. *Drones*, **2024**. doi:10.3390/drones8100539.

[30] Zhao Wei, Li Liu, Teng Long, and Zhu Wang. RRT*-based Threat-Avoidance Trajectory Planning for Aircrafts (IEEE/CSAA GNCC)*. *2018 IEEE CSAA*

Guidance, Navigation and Control Conference (CGNCC), pages 1–5, **2018**. doi:10.1109/gncc42960.2018.9018947.

[31] Zaid Tahir, A. Qureshi, Y. Ayaz, and Raheel Nawaz. Potentially Guided Bidirectionalized RRT* for Fast Optimal Path Planning in Cluttered Environments. *Robotics Auton. Syst.*, 108:13–27, **2018**. doi:10.1016/j.robot.2018.06.013.

[32] Haixiang Huang, Yaoxing Shang, Xianfei Liu, Xiaochao Liu, and P. Qi. An improved Bi-RRT*-based path planning algorithm with adaptive search strategy assignment mechanism for ultra-low-altitude penetration of fixed-wing aircraft. *Aerospace Science and Technology*, **2024**. doi:10.1016/j.ast.2024.109363.

[33] Yicheng Chen and Lingling Wang. Adaptively Dynamic RRT*-Connect: Path Planning for UAVs Against Dynamic Obstacles. *2022 7th International Conference on Automation, Control and Robotics Engineering (CACRE)*, pages 1–7, **2022**. doi:10.1109/cacre54574.2022.9834188.

[34] Yicong Guo, Xiaoxiong Liu, Qianlei Jia, Xuhang Liu, and Weiguo Zhang. HPO-RRT*: a sampling-based algorithm for UAV real-time path planning in a dynamic environment. *Complex & Intelligent Systems*, 9:7133–7153, **2023**. doi:10.1007/s40747-023-01115-2.

[35] Xiangyu Zhu, Yufeng Gao, Yanyan Li, and Bo Li. Fast Dynamic P-RRT*-Based UAV Path Planning and Trajectory Tracking Control Under Dense Obstacles. *Actuators*, **2025**. doi:10.3390/act14050211.

[36] Tao Jia, Shaohuan Han, Ping Wang, Wenyuan Zhang, and Yawu Chang. Dynamic obstacle avoidance path planning for UAV. *2020 3rd International Conference on Unmanned Systems (ICUS)*, pages 814–818, **2020**. doi:10.1109/icus50048.2020.9274865.

[37] Thu Hang Thi Khuat, D. Bui, Hoa Nguyen, Mien Trinh, Minh Nguyen, and Manh Duong Phung. Multi-Goal Rapidly Exploring Random Tree With

Safety and Dynamic Constraints for UAV Cooperative Path Planning. *IEEE Transactions on Vehicular Technology*, 74:13446–13457, **2025**. doi:10.1109/tvt.2025.3560658.

[38] Zhigang Wang, Huajun Gong, Mingtao Nie, and Xiaoxiong Liu. Research on Multi-UAV Cooperative Dynamic Path Planning Algorithm Based on Conflict Search. *Drones*, **2024**. doi:10.3390/drones8060274.

[39] Moon-Jung Kim, T. Kang, and Chang-Kyung Ryoo. Real-Time Path Planning for Unmanned Aerial Vehicles Based on Compensated Voronoi Diagram. *International Journal of Aeronautical and Space Sciences*, 26:235–244, **2024**. doi:10.1007/s42405-024-00771-z.

[40] Avneesh Sud, Erik Andersen, Sean Curtis, Ming Lin, and D. Manocha. Real-time Path Planning for Virtual Agents in Dynamic Environments. *2007 IEEE Virtual Reality Conference*, pages 91–98, **2007**. doi:10.1145/1401132.1401206.

[41] L. Blasi, E. D'Amato, M. Mattei, and I. Notaro. UAV Path Planning in 3-D Constrained Environments Based on Layered Essential Visibility Graphs. *IEEE Transactions on Aerospace and Electronic Systems*, 59:2359–2375, **2023**. doi:10.1109/taes.2022.3213230.

[42] J. Royset, W. Carlyle, R. Wood, and Network Methods. Routing Military Aircraft with a Constrained Shortest-Path Algorithm. *Military Operations Research*, 14, **2009**. doi:10.5711/morj.14.3.31.

[43] Jingyuan Shan. An Optimization Approach for Dynamic Aircraft Trajectory Planning Under Multiple Constraints. *2024 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 610–616, **2024**. doi:10.1109/rcar61438.2024.10671292.

[44] Shi Xiaoli, Wang Xinmin, Liu Yongcai, Wang Changqing, and Xu Cheng. Optimization of Fighter Aircraft Evasive Trajectories for Radar Threats

Avoidance. *2007 IEEE International Conference on Control and Automation*, pages 303–307, **2007**. doi:10.1109/icca.2007.4376368.

[45] Z. Yao. Tangent Based-Path Path Planning with Distinctive Topologies in a Dynamic Environment. *Preprints*, **2021**. doi:10.20944/preprints202108.0034.v1.

[46] G. Aiello, K. Valavanis, and A. Rizzo. Fixed-Wing UAV Energy Efficient 3D Path Planning in Cluttered Environments. *Journal of Intelligent & Robotic Systems*, 105, **2022**. doi:10.1007/s10846-022-01608-1.

[47] Zhihao Zhang, Xiaodong Liu, and Boyu Feng. Research on obstacle avoidance path planning of UAV in complex environments based on improved Bézier curve. *Scientific Reports*, 13, **2023**. doi:10.1038/s41598-023-43783-7.

[48] Parikshit Maini and P. Sujit. Path planning for a UAV with kinematic constraints in the presence of polygonal obstacles. *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 62–67, **2016**. doi:10.1109/icuas.2016.7502625.

[49] S. Primatesta, G. Guglieri, and A. Rizzo. A Risk-Aware Path Planning Strategy for UAVs in Urban Environments. *Journal of Intelligent & Robotic Systems*, 95:629–643, **2018**. doi:10.1007/s10846-018-0924-3.

[50] Jong-Jin Shin and H. Bang. UAV Path Planning under Dynamic Threats Using an Improved PSO Algorithm. *International Journal of Aerospace Engineering*, 2020:1–17, **2020**. doi:10.1155/2020/8820284.

[51] B. L. Stevens, F. L. Lewis, and E. N. Johnson. *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. John Wiley & Sons, Hoboken, NJ, USA, 3rd edition, **2016**.

[52] Gang Hu, Mao Cheng, Essam Houssein, and Heming Jia. CMPSO: A novel co-evolutionary multigroup particle swarm optimization for multi-mission

UAVs path planning. *Adv. Eng. Informatics*, 63:102923, **2025**. doi:10.1016/j.aei.2024.102923.

[53] Ping Yang, B. Xiao, Xin Chen, and Liang-Qi Guo. 3D path planning problem for fighter aircraft with multiple constraints. *The Journal of Engineering*, **2023**. doi:10.1049/tje2.12325.

[54] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, **1986**.

[55] Nakyeong Sung, Suhwan Kim, and Namsuk Cho. An Efficient Path Planning Algorithm Using a Potential Field for Ground Forces. *Comput.*, 11:12, **2023**. doi:10.3390/computation11010012.

[56] Longhao Liu, Le Ru, Wenfei Wang, Hailong Xi, Rui Zhu, Shiliang Li, and Zhenghao Zhang. UAV Path Planning in Threat Environment: A*-APF Algorithm for Spatio-Temporal Grid Optimization. *Drones*, **2025**. doi:10.3390/drones9090661.

[57] L. Bui, H. Abbass, and J. Branke. Multiobjective optimization for dynamic environments. *2005 IEEE Congress on Evolutionary Computation*, 3:2349–2356, **2005**. doi:10.1109/cec.2005.1554987.

[58] Zhe Zhang, Ju Jiang, K. Ling, and Wen-An Zhang. Real-Time Path Planning for Autonomous UAVs: An Event-Triggered Multimodal Adaptive Pigeon-Inspired Optimization Approach. *IEEE Transactions on Aerospace and Electronic Systems*, 61:10972–10981, **2025**. doi:10.1109/taes.2025.3550128.

[59] Fatma Hashim, Essam Houssein, Kashif Hussain, M. Mabrouk, and W. Al-Atabany. Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems. *Math. Comput. Simul.*, 192:84–110, **2022**. doi:10.1016/j.matcom.2021.08.013.

[60] Xinxing Hou, Yanwen Li, and Zhenzhong Liu. An Enhanced Polar Lights Optimization Algorithm with Symmetry Mechanisms for Global Optimization and Its Application to Wind Power Forecasting. *Symmetry*, **2025**. doi:10.3390/sym18010061.

[61] Zhenhua Yu, Zhijie Si, Xiaobo Li, Dan Wang, and H. Song. A Novel Hybrid Particle Swarm Optimization Algorithm for Path Planning of UAVs. *IEEE Internet of Things Journal*, 9:22547–22558, **2022**. doi:10.1109/jiot.2022.3182798.

[62] Ary Shared Rosas-Carrillo, Arturo Solís-Santomé, Carlos Silva-Sánchez, and O. Camacho-Nieto. UAV Path Planning Using an Adaptive Strategy for the Particle Swarm Optimization Algorithm. *Drones*, **2025**. doi:10.3390/drones9030170.

[63] Manh Duong Phung and Q. Ha. Safety-enhanced UAV Path Planning with Spherical Vector-based Particle Swarm Optimization. *Appl. Soft Comput.*, 107:107376, **2021**. doi:10.1016/j.asoc.2021.107376.

[64] Yanfei Liu, Hao Zhang, Hao Zheng, Qi Li, and Qi Tian. A spherical vector-based adaptive evolutionary particle swarm optimization for UAV path planning under threat conditions. *Scientific Reports*, 15, **2025**. doi:10.1038/s41598-025-85912-4.

[65] D. Debnath, F. Vanegas, Sebastien Boiteau, and Felipe Gonzalez. An Integrated Geometric Obstacle Avoidance and Genetic Algorithm TSP Model for UAV Path Planning. *Drones*, **2024**. doi:10.3390/drones8070302.

[66] Elias Freitas, Miri Weiss-Cohen, A. Neto, Frederico Guimarães, and Luciano Pimenta. DE3D-NURBS: A differential evolution-based 3D path-planner integrating kinematic constraints and obstacle avoidance. *Knowl. Based Syst.*, 300:112084, **2024**. doi:10.1016/j.knosys.2024.112084.

[67] Jun Guan, Shuanghui Ye, and Wenjun Yi. MSCPSO: A multi-strategy cooperative particle swarm optimization algorithm for UAV path planning. *Expert Systems with Applications*, **2026**. doi:10.1016/j.eswa.2025.131034.

[68] Himanshu Gupta and Om Prakash Verma. A novel hybrid Coyote-Particle Swarm Optimization Algorithm for three-dimensional constrained trajectory planning of Unmanned Aerial Vehicle. *Appl. Soft Comput.*, 147:110776, **2023**. doi:10.1016/j.asoc.2023.110776.

[69] Zhengsheng Zhan, Dangyue Lai, Canjian Huang, Zhixiang Zhang, Yongle Deng, and Jian Yang. MSCSO: A Modified Sand Cat Swarm Algorithm for 3D UAV Path Planning in Complex Environments with Multiple Threats. *Sensors (Basel, Switzerland)*, 25, **2025**. doi:10.3390/s25092730.

[70] Bing Sun and Ziang Lv. Multi-AUV Dynamic Cooperative Path Planning with Hybrid Particle Swarm and Dynamic Window Algorithm in Three-Dimensional Terrain and Ocean Current Environment. *Biomimetics*, 10, **2025**. doi:10.3390/biomimetics10080536.

[71] F. Qi, W. Zhenyu, and B. Zhaohui. Aircraft path planning based on GA in the moving threats environment. In *Proceedings of the 32nd Chinese Control Conference*, pages 5107–5110. **2013**.

[72] L. Swartzentruber, J. Foo, and E. Winer. Multi-objective UAV path planning with refined reconnaissance and threat formulations. In *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*. **2010**.

[73] Mingtian Da and Zhixin Sun. Path Planning Method for Uavs in Enemy-Controlled Threat Areas Based on Multi-Population Evolutionary Slime Mold Algorithm. *2025 44th Chinese Control Conference (CCC)*, pages 1821–1826, **2025**. doi:10.23919/ccc64809.2025.11178325.

[74] Chang Xiao, Huiliao Yang, and Bo Zhang. Multi-Unmanned Aerial Vehicle Path Planning Based on Improved Nutcracker Optimization Algorithm. *Drones*, **2025**. doi:10.3390/drones9020116.

[75] Zihan Zhou, Yanhong Guo, Yitao Wang, Jingfan Lyu, Haoran Gong, Xin Ye, and Yachao Li. Multi-UAV Trajectory Optimization Under Dynamic Threats: An Enhanced GWO Algorithm Integrating a Priori and Real-Time Data. *International Journal of Computational Intelligence Systems*, 18, **2025**. doi:10.1007/s44196-025-00863-y.

[76] Han Li, Zidong Wang, Cheng Lan, Peishu Wu, and Nianyin Zeng. A Novel Dynamic Multiobjective Optimization Algorithm With Hierarchical Response System. *IEEE Transactions on Computational Social Systems*, 11:2494–2512, **2024**. doi:10.1109/tcss.2023.3293331.

[77] Yuting Wan, Yanfei Zhong, and L. Zhang. An Accurate UAV 3-D Path Planning Method for Disaster Emergency Response Based on an Improved Multiobjective Swarm Intelligence Algorithm. *IEEE Transactions on Cybernetics*, 53:2658–2671, **2022**. doi:10.1109/tcyb.2022.3170580.

[78] Enhui Dai, Zhenxue He, Xiaojun Zhao, Xiaodan Zhang, Yijin Wang, and Xiang Wang. An optical microscope algorithm with precise focusing strategy and migration strategy with application in 3D path planning. *The Journal of Supercomputing*, 81, **2025**. doi:10.1007/s11227-025-07460-y.

[79] Mohammed Sani Adam, Nor Fadzilah Abdullah, Asma' Abu-Samah, Oluwatosin Ahmed Amodu, and R. Nordin. Advanced Path Planning for UAV Swarms in Smart City Disaster Scenarios Using Hybrid Metaheuristic Algorithms. *Drones*, **2025**. doi:10.3390/drones9010064.

[80] Maral Hooshyar and Yueh-Min Huang. Metaheuristic Algorithms in UAV Path-Planning Optimization: A Systematic Review (2018–2022). *Drones*, **2023**. doi:10.3390/drones7120687.

[81] Alexander Forrester, András Sóbester, and Andy Keane. *Engineering Design via Surrogate Modelling: A Practical Guide*. John Wiley & Sons, **2008**.

[82] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, **2016**.

[83] Chao Yan, Xiaojia Xiang, and Chang Wang. Towards Real-Time Path Planning through Deep Reinforcement Learning for a UAV in Dynamic Environments. *Journal of Intelligent & Robotic Systems*, 98:297–309, **2019**. doi:10.1007/s10846-019-01073-3.

[84] Yang Yang, Juntao Li, and Lingling Peng. Multi-robot path planning based on a deep reinforcement learning DQN algorithm. *CAAI Trans. Intell. Technol.*, 5:177–183, **2020**. doi:10.1049/trit.2020.0024.

[85] Joseph Abraham and Sreeja Kochuvila. Optimized Path Planning using Double Deep Q-Networks with Dynamic Window Approach. *2025 8th International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1489–1494, **2025**. doi:10.1109/iccmc65190.2025.11140738.

[86] Lei Bao, Zhengtao Guo, Xianzhong Gao, and Chaolong Li. Stealth UAV Path Planning Based on DDQN Against Multi-Radar Detection. *Aerospace*, **2025**. doi:10.3390/aerospace12090774.

[87] Zeyang Wang, Jun Huang, and M. Yi. A Stealth–Distance Dynamic Weight Deep Q-Network Algorithm for Three-Dimensional Path Planning of Unmanned Aerial Helicopter. *Aerospace*, **2023**. doi:10.3390/aerospace10080709.

[88] Mehmet Gök. Dynamic path planning via Dueling Double Deep Q-Network (D3QN) with prioritized experience replay. *Appl. Soft Comput.*, 158:111503, **2024**. doi:10.1016/j.asoc.2024.111503.

[89] Anping Yang, Jing Huan, Qi Wang, Hualong Yu, and Shang Gao. ST-D3QN: Advancing UAV Path Planning With an Enhanced Deep Reinforcement Learning Framework in Ultra-Low Altitudes. *IEEE Access*, 13:65285–65300, **2025**. doi:10.1109/access.2025.3559129.

[90] A. De Marco, Paolo Maria D'Onza, and Sabato Manfredi. A deep reinforcement learning control approach for high-performance aircraft. *Nonlinear Dynamics*, 111:17037–17077, **2023**. doi:10.1007/s11071-023-08725-y.

[91] Dooyoung Hong, Seonhoon Lee, Y. Cho, Donkyu Baek, Jaemin Kim, and N. Chang. Energy-Efficient Online Path Planning of Multiple Drones Using Reinforcement Learning. *IEEE Transactions on Vehicular Technology*, 70:9725–9740, **2021**. doi:10.1109/tvt.2021.3102589.

[92] Rongkun Xue, Jing Yang, Yuyang Jiang, Yiming Feng, and Zi Yang. Motion Dynamic RRT based Fluid Field - PPO for Dynamic TF/TA Routing Planning. *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 2743–2748, **2024**. doi:10.1109/iv55156.2024.10588621.

[93] Han Qie, Dian-Xi Shi, Tianlong Shen, Xinhai Xu, Yuan Li, and Liujing Wang. Joint Optimization of Multi-UAV Target Assignment and Path Planning Based on Multi-Agent Reinforcement Learning. *IEEE Access*, 7:146264–146272, **2019**. doi:10.1109/access.2019.2943253.

[94] Zhe Zhang, Ju Jiang, Keck Voon Ling, Xinhua Wang, and Wen-An Zhang. Cooperative Path Planning for Heterogeneous UAV Swarms: A Stackelberg Game Approach. *IEEE Transactions on Automation Science and Engineering*, 22:18531–18548, **2025**. doi:10.1109/tase.2025.3585946.

[95] Yan Yan and Wangbai Pan. Optimization of Aircraft Trajectory Design Based on Intelligent Learning Methods. *2024 International Conference on Information Technology, Comunication Ecosystem and Management (ITCEM)*, pages 23–27, **2024**. doi:10.1109/itcem65710.2024.00014.

[96] Ourania Theodosiadou, Despoina Chatzakou, T. Tsikrika, S. Vrochidis, and I. Kompatsiaris. Real-time threat assessment based on hidden Markov models. *Risk Analysis*, 43:2069–2081, **2023**. doi:10.1111/risa.14105.

[97] Yang Gao and Na Lyu. A New Multi-Target Three-Way Threat Assessment Method with Heterogeneous Information and Attribute Relevance. *Mathematics*, **2024**. doi:10.3390/math12050691.

[98] Humberto Baldessarini Pires, Lamartine Nogueira Frutuoso Guimarães, and Sergio Rebouças. A Multi-Target Threat Assessment Method Based on Objective Three-Way Decision. *IEEE Access*, 13:681–694, **2025**. doi:10.1109/access.2024.3523817.

[99] Lixiang Zhai, Husheng Wu, Linghong Lai, and Ziqian Gao. Intelligent Optimization Algorithms for Multi-UAV Path Planning: A Comprehensive Review. *IEEE Access*, 13:101106–101130, **2025**. doi:10.1109/access.2025.3577598.

[100] Ralph L Keeney and Howard Raiffa. *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*. Cambridge University Press, **1993**.

[101] Ghulam Farid, Muhammad Bilal, Lanyong Zhang, Ayman Alharbi, Ishaq Ahmed, and Muhammad Azhar. An Improved Deep Q-Learning Approach for Navigation of an Autonomous UAV Agent in 3D Obstacle-Cluttered Environment. *Drones*, **2025**. doi:10.3390/drones9080518.

[102] Yan Xu, Aiping Zhou, Hui Li, Jin Qian, Yixiao Xiao, and Shuai Liu. Optimized Path Planning and Safe Obstacle Avoidance for UAVs Based on Enhanced DQN. *2025 6th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*, pages 678–683, **2025**. doi:10.1109/icecai66283.2025.11171120.

[103] Yubin Qian, Song Feng, Wenhao Hu, and Wanqiu Wang. Obstacle avoidance planning of autonomous vehicles using deep reinforcement learning. *Advances in Mechanical Engineering*, 14, **2022**. doi:10.1177/16878132221139661.

[104] Huiyan Han, Jiaqi Wang, Liqun Kuang, Xie Han, and Hongxin Xue. Improved Robot Path Planning Method Based on Deep Reinforcement Learning. *Sensors (Basel, Switzerland)*, 23, **2023**. doi:10.3390/s23125622.

[105] Jiandong Liu, Wei Luo, Guoqing Zhang, and Ruihao Li. Unmanned Aerial Vehicle Path Planning in Complex Dynamic Environments Based on Deep Reinforcement Learning. *Machines*, **2025**. doi:10.3390/machines13020162.

[106] Le Han, Hui Zhang, and Nan An. A Continuous Space Path Planning Method for Unmanned Aerial Vehicle Based on Particle Swarm Optimization-Enhanced Deep Q-Network. *Drones*, **2025**. doi:10.3390/drones9020122.

[107] Xiaobing Yu and Wenguan Luo. Reinforcement learning-based multi-strategy cuckoo search algorithm for 3D UAV path planning. *Expert Syst. Appl.*, 223:119910, **2023**. doi:10.1016/j.eswa.2023.119910.

[108] M. Alpdemir. Tactical UAV path optimization under radar threat using deep reinforcement learning. *Neural Computing and Applications*, 34:5649–5664, **2022**. doi:10.1007/s00521-021-06702-3.

[109] R.U. Hameed, A. Maqsood, A. Hashmi, M. Saeed, and R. Riaz. Reinforcement learning-based radar-evasive path planning: a comparative analysis. *The Aeronautical Journal*, **2021**. doi:10.1017/aer.2021.85.

[110] Rong Yang, Yong, Zhangzhi Tao, and Rong Yang. A stealthy route planning algorithm for the fourth generation fighters. *2017 International Conference on Mechanical, System and Control Engineering (ICMSC)*, pages 323–327, **2017**. doi:10.1109/icmsc.2017.7959494.

[111] Wei Wu, Wenjie Jie, Angang Luo, Xing Liu, and Wei Luo. Data-Fusion-Based Algorithm for Assessing Threat Levels of Low-Altitude and Slow-Speed Small Targets. *Sensors (Basel, Switzerland)*, 25, **2025**. doi:10.3390/s25175510.

[112] H. Jayaweera and Samer Hanoun. Path Planning of Unmanned Aerial Vehicles (UAVs) in Windy Environments. *Drones*, **2022**. doi:10.3390/drones6050101.

[113] Junxi Gao, Yanglin Chen, Zhiyuan Cai, Caishi Huang, and Chaoda Peng. Enhancing UAV Safety in Smart City Environments: A Dynamic Constrained Multi-Objective Evolutionary Algorithm. *2025 IEEE/CIC International Conference on Communications in China (ICCC)*, pages 1–6, **2025**. doi:10.1109/iccc65529.2025.11148628.

[114] M. Venkateswarlu, G. Himaja, V. Reddy, and M. Vineeth. Real-Time Flight Path Optimization: Enhancing Efficiency in Modern Aviation. *2025 3rd International Conference on Inventive Computing and Informatics (ICICI)*, pages 1320–1326, **2025**. doi:10.1109/icici65870.2025.11069767.

[115] Susan Stachowiak and J. Bosworth. Flight Test Results for the F-16XL With a Digital Flight Control System. *NASA Technical Paper*, **2004**.

[116] R. Kamyar and E. Taheri. Aircraft Optimal Terrain/Threat-Based Trajectory Planning and Control. *Journal of Guidance Control and Dynamics*, 37:466–483, **2014**. doi:10.2514/1.61339.

[117] Rosenow Judith. A Physical Aircraft Performance Model for Multi-Criteria Trajectory Optimization. *International Journal of Astronautics and Aeronautical Engineering*, **2020**. doi:10.35840/2631-5009/7544.

[118] Xiaoqiang Lu, Jun Huang, Jingxin Guan, and Lei Song. Stealth Aircraft Penetration Trajectory Planning in 3D Complex Dynamic Based on Radar Valley Radius and Turning Maneuver. *Aerospace*, **2024**. doi:10.3390/aerospace11050402.

[119] Taewan Kim, Abhinav Kamath, Niyousha Rahimi, Behçet Açikmese, Mehran Mesbahi, and Jasper Corleis. Six-Degree-of-Freedom Aircraft Landing Trajectory Planning with Runway Alignment. *Journal of Guidance, Control, and Dynamics*, **2024**. doi:10.2514/1.g008521.

[120] Zhenyao Zhao, Biyuan Hu, Cunbao, Zhen Qian, A. Yao, and Jing Qu. Aircraft Route Planning and Local Dynamic Obstacle Avoidance. *2022 4th International Conference on Intelligent Control, Measurement and Signal Processing (ICMSP)*, pages 546–553, **2022**. doi:10.1109/icmsp55950.2022.9859034.

[121] Omid Kazemifar, Ali-Reza Babaei, and Mahdi Mortazavi. Online aircraft velocity and normal acceleration planning for rough terrain following. *The Aeronautical Journal*, 121(1244):1561–1577, **2017**. ISSN 0001-9240. doi:10.1017/aer.2017.27.

[122] W. Zhao, Z. Meng, and P. W. Chung. A heuristic particle swarm optimization with dynamic window approach for mobile robot path planning. *Applied Soft Computing*, **2022**.

[123] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, **2015**.

[124] Randal W Beard and Timothy W McLain. *Small Unmanned Aircraft: Theory and Practice*. Princeton University Press, **2012**.

[125] Steven M LaValle. *Planning Algorithms*. Cambridge University Press, **2006**.

[126] Mihail Pivtoraiko, Ross A Knepper, and Alonzo Kelly. Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, 26(3):308–333, **2009**.