aerospace

MDPI

Article

Prediction-Aware Hybrid A*–DWA Path Planning for UAV Obstacle Avoidance in Dynamic Environments

Yanxin Zhang, Tian Jing *, Meigen Huang, Tao Wang and Mengjun Ming

College of Systems Engineering, National University of Defense Technology, Changsha 410073, China
* Correspondence: jingtian16@nudt.edu.cn

Abstract

Traditional A* path planning fails in dynamic environments because it cannot anticipate moving obstacles and offers no local reactive avoidance. Most hybrid global–local planners such as A*–DWA mitigate this, but their dynamic-window layer typically forward-simulates the UAV against the obstacles' *current* positions, implicitly assuming obstacles are static over the prediction window; a recent 3D A*–DWA variant uses short-horizon constant-velocity prediction inside the DWA horizon, but it does not model prediction uncertainty or adaptive turning. This paper proposes a prediction–planning–avoidance framework that injects uncertainty-aware Kalman motion prediction into *both* planning layers. First, a Kalman filter augmented with an online turn-rate estimate predicts obstacle motion under an adaptive coordinated-turn model and builds a spatiotemporal risk field whose covariance grows with the prediction horizon, quantifying prediction uncertainty to produce a forward-looking cost map. Second, a risk-adaptive heuristic couples the local risk level into the A* evaluation function through a sigmoid-type weight. Third, the Dynamic Window Approach (DWA) is made *prediction-aware*: Its forward simulation propagates the predicted obstacle motion instead of treating obstacles as static, and risk-triggered mode switching passes risk information between the layers; redundant-waypoint removal and cubic spline interpolation further smooth the path. In numerical simulations with 12 static and 10 dynamic obstacles under UAV acceleration constraints, the complete framework attains an 87% success rate across 100 randomized trials, versus 0% for traditional A*, 35% for IA*-only, and 70% for a plain A*–DWA baseline (McNemar test, $p = 0.0005$). Ablation studies show that prediction-aware DWA adds a significant 10 percentage points over the non-predictive framework, the risk-adaptive heuristic contributes a further significant gain, and the Kalman risk field provides a smaller, borderline complementary benefit. Additional studies confirm that the result generalizes beyond this scene: The framework's advantage over a plain A*–DWA baseline grows with obstacle density (from 3 to 31 percentage points as the number of dynamic obstacles rises from 5 to 20), and prediction-aware DWA is decisive in narrow-passage, bottleneck, and dead-end environments (83–86% versus 45–59% for a static-obstacle DWA). In constrained geometry, the result is sharper still: A non-predictive reactive DWA oscillates in narrow openings and even underperforms plain A*, so prediction-aware DWA is the minimal intervention that removes a failure mode the reactive layer itself introduces. A full 6-DOF quadrotor validation in two three-dimensional scenarios over 100 paired trials preserves this advantage: Prediction-aware DWA attains 90% and 94%, versus 72% and 77% for a static-obstacle DWA.

Keywords: UAV; path planning; A* algorithm; dynamic obstacle avoidance; adaptive heuristic; prediction uncertainty; hybrid algorithm; Kalman filter

Check for updates

Academic Editor: Gokhan Inalhan

Received: 20 October 2026
Revised: 14 November 2026
Accepted: 20 November 2026
Published: 23 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

UAV technology has developed rapidly in recent years and has been widely applied in fields such as logistics delivery, environmental monitoring, and search and rescue. Path planning constitutes one of the core capabilities underpinning autonomous navigation. As mission environments grow increasingly complex, UAVs must not only avoid static obstacles such as buildings but also respond to dynamic threats posed by other aircraft or moving targets. The traditional A* algorithm, as a classical global path search method, can guarantee optimality in static environments. However, in dynamic environments, it suffers from several limitations: lack of predictive capability regarding moving obstacles, high replanning overhead when the environment changes, paths with numerous sharp turns and insufficient smoothness, and difficulty in fusing real-time sensor information. Therefore, investigating prediction-aware planning methods that predict obstacle motion and integrate it with local real-time obstacle avoidance is valuable for improving the flight safety and mission efficiency of UAVs in dynamic environments.

Related studies on UAV path planning, A* improvements, and hybrid planning in dynamic environments are reviewed in Section 2. In summary, existing methods still exhibit two gaps. First, most A*–DWA hybrids forward-simulate the UAV against the obstacles' *current* positions, implicitly assuming obstacles are static over the prediction window; a recent 3D extension applies a short-horizon constant-velocity prediction inside the DWA forward simulation [1], but it does not propagate a Kalman prediction with growing uncertainty, adapt to turning obstacles, or couple the prediction to a global risk field. Prediction that is both uncertainty-aware and consistent across the two layers therefore remains uncommon. Second, the coupling between global and local planning is loose, and the adaptive capability of heuristic functions is limited. To address these gaps, this paper proposes a prediction–planning–avoidance framework that injects uncertainty-aware Kalman-filter motion prediction into *both* the global A* search and the local DWA. Over 100 randomized trials with 12 static and 10 dynamic obstacles under UAV acceleration constraints, the complete framework attains an 87% success rate, versus 0% for traditional A*, 35% for IA*-only, and 70% for a plain A*–DWA baseline (McNemar test, $p = 0.0005$). Beyond the nominal scene, the framework's most counterintuitive result appears in constrained geometry (Section 5.7): A non-predictive reactive DWA oscillates in narrow passages and even underperforms plain A*, and the prediction-aware local layer is the minimal intervention that removes this failure mode. The architecture is realized through the following components:

- **Prediction layer: dynamic environment modeling based on Kalman filtering.** A constant-velocity motion model is adopted for state estimation, augmented with an online turn-rate estimate so that the trajectory prediction follows an adaptive coordinated-turn model, and a spatiotemporal risk field is constructed that incorporates the growing prediction uncertainty into the environmental cost assessment, providing a forward-looking cost map for the global planner.
- **Planning layer: risk-adaptive heuristic A* and path smoothing.** The evaluation function of traditional A* is extended with a sigmoid-type adaptive weight coupled to the local risk level and a risk-based heuristic term, enabling the search to pursue efficiency in safe regions while proactively avoiding high-risk regions; the resulting path is further smoothed through redundant-waypoint removal and cubic spline interpolation.

https://doi.org/10.3390/aerospace13121102

- **Avoidance layer: prediction-aware Dynamic Window Approach.** The DWA local planner is made prediction-aware: Its forward simulation propagates the predicted obstacle motion instead of treating obstacles as static, and risk-triggered mode switching passes risk information between the global and local layers, closing the temporal gap in which sudden, close-range motion changes occur.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 formulates the system model, including the A* preliminaries, environment gridding, Kalman-filter-based obstacle prediction, and risk field construction. Section 4 details the risk-adaptive heuristic, path smoothing, and the prediction-aware hybrid planning architecture. Section 5 presents the numerical simulation setup, comparative and ablation studies, robustness under maneuvering and crossing obstacles, scalability with obstacle density, structured-environment evaluation, a parameter-sensitivity study, the failure-mode and learned-predictor analyses, and a three-dimensional 6-DOF validation. Section 6 concludes the paper.

2. Related Work

2.1. UAV Path Planning Algorithms

UAV path planning algorithms can be broadly classified into three categories: classical algorithms, intelligent optimization algorithms, and hybrid algorithms. Classical algorithms include Dijkstra's algorithm, the A* algorithm, Rapidly exploring Random Trees (RRT), and the Artificial Potential Field (APF) method. These algorithms possess a solid theoretical foundation but suffer from low search efficiency or susceptibility to local optima in complex environments. Intelligent optimization algorithms include Particle Swarm Optimization (PSO) [2], Genetic Algorithm (GA) [3], Ant Colony Optimization (ACO) [4], and Grey Wolf Optimizer (GWO) [5]. These algorithms exhibit strong global search capability but are characterized by slow convergence and complex parameter tuning. Hybrid algorithms attempt to combine the advantages of both categories and have become a research focus in recent years [6].

In recent years, learning-based planning methods have also made notable progress. Rivière et al. proposed the SETS algorithm, which integrates spectral expansion with Monte Carlo tree search, enabling real-time optimal replanning for UAVs in dynamic wind fields with moving obstacles [7]. Izzo et al. proposed the G&CNet end-to-end neural network architecture, capable of generating optimal control commands in real time under onboard computational resource constraints, providing a new way to tightly couple path planning and control [8]. Furthermore, He et al. proposed a bilevel optimization method for UAV path planning that fuses A* with the Dynamic Window Approach, optimizing the node expansion strategy of A* by defining neighbor node pruning rules, thereby significantly improving search efficiency in complex environments [9].

Beyond single-agent planning, cooperative and swarm path planning for multiple UAVs has attracted increasing attention. Cooperative planning must jointly optimize individual trajectories while respecting inter-vehicle collision avoidance and communication constraints. L. Xu et al. proposed a comprehensive-learning dynamic multi-swarm particle swarm optimization (CL-DMSPSO) for cooperative path planning of multiple UAVs under communication constraints [10]; X. Xu et al. developed a multi-objective evolutionary algorithm with two balancing mechanisms for heterogeneous UAV swarm path planning [11]; and X. Xu et al. proposed a two-stage optimization method that jointly optimizes trajectory and communication constraints for real-time path planning of heterogeneous UAV swarms [12]. Yildiz et al. combined a consensus-based virtual leader tracking swarm algorithm with GDRRT*-PSO for formation-preserving path planning of multiple UAVs [13].

Zhou et al. proposed a task-decomposed multi-agent twin delayed deep deterministic policy gradient (TD-MATD3) algorithm for multi-UAV autonomous path planning [14].

Cooperative path planning for marine vehicles faces similar challenges in dynamic and uncertain environments. Xiong et al. applied Voronoi-based ant colony optimization to the path planning of multiple autonomous marine vehicles for adaptive sampling [15]; Zeng et al. addressed rendezvous path planning for multiple autonomous marine vehicles [16]; Yin et al. proposed a robust algorithm for path planning of multiple autonomous underwater vehicles under ocean currents [17]; Zhuang et al. studied cooperative path planning of multiple autonomous underwater vehicles operating in a dynamic ocean environment [18]; and Yao and Qi considered obstacle-avoiding path planning for multiple autonomous underwater vehicles with simultaneous arrival [19].

2.2. A Algorithm and Its Improvements*

Research on improving the A* algorithm has primarily focused on the following directions:

Heuristic function optimization. Traditional A* uses Euclidean distance or Manhattan distance as the heuristic estimate. Researchers have improved the heuristic function by introducing weight factors, terrain costs, and threat costs, making the search direction better aligned with practical requirements. Sun et al. proposed an autonomous vehicle path planning algorithm based on an improved heuristic function, balancing search efficiency and path quality through dynamic adjustment of heuristic weights [20].

Search space reduction. Sparse A* Search (SAS) incorporates the UAV kinematic model to constrain node expansion directions, substantially reducing the search space. Ding et al. proposed a memetic algorithm for curvature-constrained path planning of a messenger UAV in air–ground coordination, modeling the UAV as a Dubins vehicle so that the search is restricted to flyable, curvature-bounded trajectories [21].

Path smoothing. To address the problem of excessive turning points in A* paths, researchers have proposed smoothing methods such as turning point optimization, B-spline curve fitting, and circular arc transitions. Wang et al. constructed a digital twin for three-dimensional path planning of a large-span curved-arm gantry robot, achieving smooth obstacle-free motion in a three-dimensional production environment through a three-dimensional navigation mesh [22].

Dynamic adaptability enhancement. Some studies have attempted to combine A* with dynamic environment modeling methods. In terms of path smoothing, Sun et al. proposed a gradient-based B-spline trajectory planning method for autonomous UAV obstacle avoidance, optimizing trajectory safety using gradient information from a Euclidean signed distance field and achieving a favorable balance between smoothness and safety [23]. In terms of fusion algorithms, He et al. incorporated an extended Kalman filter into an improved A* and DWA hybrid framework, enhancing localization accuracy and path planning robustness through the fusion of odometry and IMU data [24]. More recently, Liao et al. integrated a Kalman-filter-based dynamic-obstacle prediction into a voxel-based improved A* for indoor UAV planning, triggering replanning when the predicted trajectories violate safety thresholds [25]. Hai et al. proposed the PENC framework, which combines nonlinear model predictive control with moving horizon estimation, maintaining high prediction accuracy even under loss of target observation information, providing a novel approach for reliable state estimation in dynamic environments [26]. For incremental replanning, the D* algorithm proposed by Stentz [27] and the D* Lite algorithm subsequently developed by Koenig and Likhachev [28] can efficiently update paths in partially known environments, yet both still have limited trajectory prediction capability for dynamic obstacles.

2.3. Hybrid Planning in Dynamic Environments

Path planning in dynamic environments typically employs a hybrid architecture: A global planner is responsible for generating a coarse reference path, while a local planner performs dynamic adjustments based on real-time perceptual information. Common hybrid combinations include A* fused with the Dynamic Window Approach (DWA), A* fused with the Artificial Potential Field (APF) method, and A* fused with the Velocity Obstacle (VO) method. In the A*–DWA fusion framework, A* generates the global path, while DWA performs local velocity-space search under robot kinematic constraints to achieve real-time obstacle avoidance. Wang et al. extended the A*–DWA fusion to three dimensions, predicting dynamic obstacles under a constant-velocity assumption over a short horizon within the DWA forward simulation [1]; this is the closest prior work to our local prediction-aware layer, but it does not model prediction uncertainty, estimate a turn rate, or feed the prediction into a global risk field. Chen and Lu proposed a real-time framework that identifies and avoids simultaneous static and dynamic obstacles directly from point-cloud input, using Kalman filters to track obstacle position and velocity [29]. Fnadi et al. addressed local obstacle-skirting for a fast bi-steerable rover using cubic Bézier curves tracked by a constrained model predictive controller, prioritizing lateral stability at high speed [30]. Beyond the A*–DWA paradigm, other hybrid strategies have been explored. Ajeil et al. proposed a hybrid particle swarm optimization–modified frequency bat (PSO-MFB) algorithm for multi-objective path planning of an autonomous mobile robot in static and dynamic environments, coupling global optimization with a local search module and an obstacle detection-and-avoidance module [31]. Rao et al. proposed an artificial potential field–A* (APF-A*) algorithm for dual-UAV cooperative suspension transport, introducing a repulsive-force term into the heuristic function to keep the path at a safe distance from obstacles while improving search efficiency [32]. Ren et al. proposed PG-ITD3, a potential-field-guided deep reinforcement learning approach that dynamically adjusts the repulsive gain to adapt UAV path planning to post-disaster environments [33].

To summarize the positioning of the proposed framework relative to representative prior work, Table 1 compares the capabilities addressed by several established and recent methods. A check mark (✓) indicates that the capability is explicitly addressed by the cited work; a dash (–) indicates it is not.

Table 1. Capability comparison of representative path-planning methods and the proposed framework (based on the scope described in each cited work). Abbreviations: Dyn. obs. = dynamic-obstacle handling; Pred. = obstacle-motion prediction; Uncert. = prediction-uncertainty modeling; Risk = risk-aware planning; Adapt. heur. = adaptive heuristic; Local avoid. = local collision avoidance; and Smooth. = path smoothing. Validation: S = numerical simulation, and R = robot/hardware experiments.

Method	Dyn. Obs.	Pred.	Uncert.	Risk	Adapt. Heur.	Local Avoid.	Smooth.	Valid.
A* [34]	–	–	–	–	–	–	–	S
D* Lite [28]	✓	–	–	–	–	–	–	S
DWA [35]	✓	–	–	–	–	✓	–	R
A*–DWA [9]	✓	–	–	–	–	✓	–	S
IA–DWA [24]	✓	–	–	–	–	✓	–	S
B-spline avoidance [23]	✓	–	–	✓	–	✓	✓	S
Prediction-based [36]	✓	✓	✓	–	–	✓	✓	R
APF–A* [32]	✓	–	–	✓	–	–	–	S
Point-cloud avoidance [29]	✓	✓	–	–	–	✓	–	R
Bézier obstacle-skirting [30]	✓	–	–	–	–	✓	✓	R
Voxel + KF A* [25]	✓	✓	–	✓	–	–	✓	S
3D A*–DWA [1]	✓	✓	–	–	–	✓	✓	S
Proposed framework	✓	✓	✓	✓	✓	✓	✓	S

3. System Model and Problem Formulation

The A* algorithm is a heuristic graph search algorithm proposed by Hart et al. [34], which extends Dijkstra's algorithm by introducing a heuristic estimate to guide the search toward the goal. Its evaluation function is defined as

$$f(n) = g(n) + h(n) \tag{1}$$

where $g(n)$ is the actual cost from the start point to node n and $h(n)$ is the heuristic estimate of the remaining cost from node n to the goal. With an admissible heuristic that never overestimates the remaining cost, A* guarantees the shortest path in static environments; this paper adopts the Euclidean distance as the baseline heuristic. However, when the environment contains moving obstacles, this classical formulation exhibits the following fundamental limitations.

The notation used throughout Sections 3 and 4 is summarized in Table 2.

Table 2. Notation used in Sections 3 and 4.

Symbol	Description	Symbol	Description
Ω	two-dimensional task space	δ	grid resolution
r_d	dynamic-obstacle radius	$\mathbf{p}$	obstacle position $[x, y]^\top$
$\mathbf{v}$	obstacle velocity $[v_x, v_y]^\top$	$\mathbf{x}_k$	Kalman state vector
$\mathbf{F}$	state transition matrix	$\mathbf{w}_k$	process noise
$\mathbf{Q}$	process-noise covariance	$\mathbf{R}$	measurement-noise covariance
$\mathbf{H}$	observation matrix	$\mathbf{z}_k$	position observation
$\mathbf{K}_k$	Kalman gain	$\mathbf{P}$	state covariance
Δt	discrete time step (0.2 s)	κ	prediction-step index
$R_\kappa(\mathbf{p})$	risk contribution at step κ	$\mathbf{\Sigma}_\kappa$	combined covariance
$\bar{r}$	local mean risk in the neighborhood $N(n)$	$N(n)$	5×5 window around node n
$\alpha(\bar{r})$	sigmoid adaptive weight	β	sigmoid steepness
r_0	risk-sensitivity inflection point (0.6)	h_{dist}	distance heuristic
h_{risk}	risk heuristic	$d_{\max}$	map diagonal (normalization)
$g(n)$, $h(n)$, $f(n)$	A* accumulated cost, heuristic, and total	$J(v, \omega)$	DWA objective
v, ω	linear/angular velocity	σ_1, σ_2, σ_3	DWA weighting coefficients
τ_{safe}	waypoint-removal safety threshold (0.7)	τ_{trig}	DWA activation risk threshold (0.3)
ω_k, $\hat{\omega}_k$	obstacle instantaneous/smoothed turn rate	τ_ω	turn-rate smoothing time constant (1.0 s)
$\mathbf{F}_{\text{CT}}$	coordinated-turn transition matrix	$\omega_{\max}$	turn-rate clamp (1.2 rad/s)

3.1. Preliminaries and Problem Description

In two-dimensional dynamic environments, the core limitations of traditional A* are as follows:

1. **Inability to predict dynamic obstacles.** Traditional A* only considers the obstacle distribution at the current moment and lacks the ability to predict motion trends. It is highly prone to selecting a path that appears safe at present but will encounter moving obstacles within a short time.
2. **Low efficiency of full replanning.** Each time the environment changes, traditional A* must restart the search from the start point, incurring high computational overhead. Moreover, historical planning information is discarded, easily leading to path discontinuities that violate the motion continuity requirements of UAVs.
3. **Insufficient path smoothness.** Paths obtained from grid-based search consist of discrete nodes connected together, containing numerous sharp corners that not only increase flight energy consumption but also hinder tracking by the low-level controller.

4. **Lack of integration with real-time perception.** As a global planner, traditional A* cannot directly utilize high-frequency local obstacle information acquired by onboard sensors and is powerless against suddenly appearing obstacles.

Therefore, effective path planning in dynamic environments must simultaneously possess predictive capability, adaptive search capability, path smoothing capability, and global–local coordination capability. These are precisely the motivations for the improvements proposed in this paper.

3.2. Environment Gridding and Dynamic Obstacle Representation

This paper investigates path planning in a two-dimensional plane, with the task space denoted as $\Omega \subset \mathbb{R}^2$. The static environment is represented using a grid map with resolution δ. Each grid cell stores whether it is occupied by a static obstacle.

A dynamic obstacle is modeled as a circular region with radius r_d. Its motion state is described by position $\mathbf{p} = [x, y]^\top$ and velocity $\mathbf{v} = [v_x, v_y]^\top$. All dynamic obstacles are regarded as potential threats and move within the scene according to a boundary-bouncing pattern, as illustrated in Figure 1.

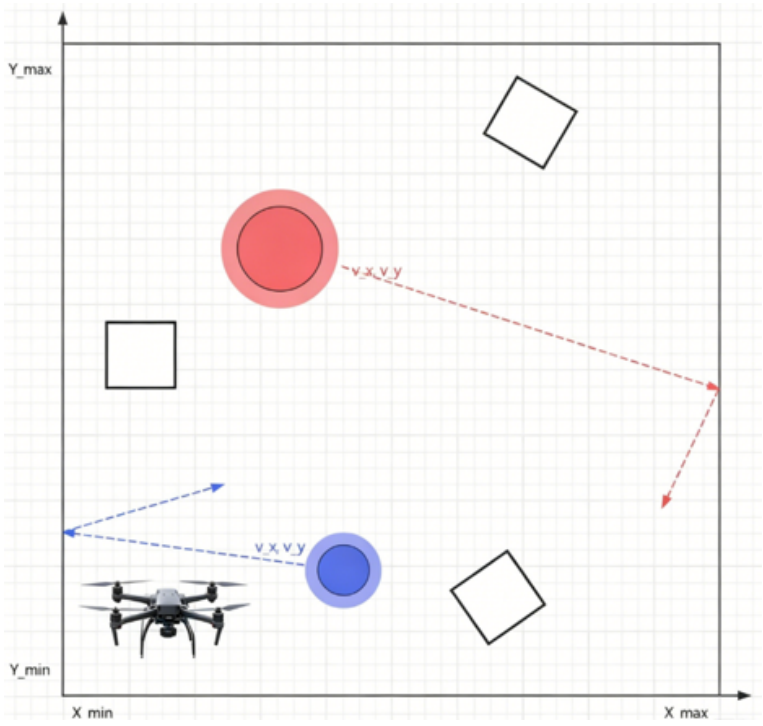


Figure 1. Schematic of environment modeling. White squares denote static obstacles; the red and blue circles with shaded halos denote dynamic obstacles of radius r_d, and the dashed arrows show their boundary-bouncing motion; the UAV is shown at the lower left.

3.3. Obstacle State Prediction Based on Kalman Filtering

To predict the future trajectories of dynamic obstacles, this paper adopts a constant-velocity motion model and deploys a Kalman filter [37]. The state vector is defined as

$$\mathbf{x}_k = [x_k, y_k, v_{x,k}, v_{y,k}]^\top \tag{2}$$

The discrete-time state transition equation is:

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \mathbf{w}_k \tag{3}$$

where

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

$\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ is the process noise, and Δt is the discrete time step.

Assuming that the sensor can directly observe the position of the obstacle, the observation equation is:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{5}$$

where $\mathbf{H} = [\mathbf{I}_{2\times 2}\ \mathbf{0}_{2\times 2}]$ extracts the position components, and $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ is the observation noise, with covariance $\mathbf{R} = 2.0\,\mathbf{I}$ treated as the filter's nominal design parameter (Section 5.1). The onboard sensor is modeled generically as an idealized detector that reports each dynamic obstacle's exact position within a detection radius of 35 m (Section 3.4), rather than as a specific sensor model; position measurements are therefore noiseless in the simulations and are acquired at every control step, i.e., at a rate of $1/\Delta t = 5$ Hz with $\Delta t = 0.2$ s, so the measurement update below runs at the same 5 Hz rate as the control loop. In the numerical study, the global planner is additionally given the simulated state of every dynamic obstacle over the whole map (an oracle assumption), while the 35 m radius limits only the obstacle list assembled by the local DWA layer; this idealization is stated as a limitation in Section 5.12.

The filter executes a prediction step and a measurement-update step in each cycle. In the prediction step, the prior state estimate and its covariance are propagated forward as

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}_{k-1|k-1}, \qquad \mathbf{P}_{k|k-1} = \mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^\top + \mathbf{Q} \tag{6}$$

In the update step, the Kalman gain $\mathbf{K}_k$ and the posterior estimates are computed as

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}^\top \left(\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^\top + \mathbf{R}\right)^{-1} \tag{7}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1}\right), \qquad \mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_{k|k-1} \tag{8}$$

From the posterior estimate, the state and covariance κ steps into the future are obtained by repeatedly applying the transition model:

$$\hat{\mathbf{x}}_{k+\kappa|k} = \mathbf{F}^\kappa\hat{\mathbf{x}}_{k|k}, \qquad \mathbf{P}_{k+\kappa|k} = \mathbf{F}^\kappa\mathbf{P}_{k|k}(\mathbf{F}^\kappa)^\top + \sum_{i=0}^{\kappa-1}\mathbf{F}^i\mathbf{Q}(\mathbf{F}^i)^\top \tag{9}$$

The predicted covariance $\mathbf{P}_{k+\kappa|k}$ reflects the growing prediction uncertainty that is subsequently injected into the risk field. In the simulations, the prediction horizon is $\kappa = 25$ steps, i.e., a physical look-ahead of $25 \times 0.2 = 5.0$ s, chosen to match the 5 s global replanning period so that the risk field remains forward-looking over the entire interval between successive replans (Section 5.1).

A constant-velocity model, however, extrapolates straight-line motion and therefore degrades when an obstacle turns or accelerates, since the propagated future positions deviate increasingly from the true curved trajectory. To relax this assumption while retaining the constant-velocity filter as the underlying state estimator, we augment it with an online turn-rate estimate. After each measurement update, the instantaneous turn rate is recovered from the change in the estimated velocity direction between consecutive steps:

$$\omega_k = \frac{1}{\Delta t}\arctan\left(\frac{\hat{v}_{x,k-1}\hat{v}_{y,k} - \hat{v}_{y,k-1}\hat{v}_{x,k}}{\hat{v}_{x,k-1}\hat{v}_{x,k} + \hat{v}_{y,k-1}\hat{v}_{y,k}}\right) \tag{10}$$

and smoothed by a first-order low-pass filter with time constant $\tau_\omega = 1.0$ s,

$$\hat{\omega}_k = \hat{\omega}_{k-1} + \frac{\Delta t}{\Delta t + \tau_\omega}(\omega_k - \hat{\omega}_{k-1}), \tag{11}$$

clamped to $|\hat{\omega}_k| \le \omega_{\max} = 1.2$ rad/s. The κ-step prediction of Equation (9) then propagates, instead of $\mathbf{F}$, a coordinated-turn transition parameterized by the estimated turn rate,

$$\mathbf{F}_{\mathrm{CT}}(\hat{\omega}_k) = \begin{bmatrix} 1 & 0 & \dfrac{\sin(\hat{\omega}_k \Delta t)}{\hat{\omega}_k} & \dfrac{\cos(\hat{\omega}_k \Delta t) - 1}{\hat{\omega}_k} \\ 0 & 1 & \dfrac{1 - \cos(\hat{\omega}_k \Delta t)}{\hat{\omega}_k} & \dfrac{\sin(\hat{\omega}_k \Delta t)}{\hat{\omega}_k} \\ 0 & 0 & \cos(\hat{\omega}_k \Delta t) & -\sin(\hat{\omega}_k \Delta t) \\ 0 & 0 & \sin(\hat{\omega}_k \Delta t) & \cos(\hat{\omega}_k \Delta t) \end{bmatrix} \tag{12}$$

i.e., $\hat{\mathbf{x}}_{k+\kappa|k} = \mathbf{F}_{\mathrm{CT}}(\hat{\omega}_k)^\kappa \hat{\mathbf{x}}_{k|k}$. When the obstacle does not maneuver ($\hat{\omega}_k \to 0$), $\mathbf{F}_{\mathrm{CT}}(\hat{\omega}_k)$ reduces to the constant-velocity transition $\mathbf{F}$ defined above, so the prediction of non-maneuvering obstacles is unchanged and the nominal results are unaffected; residual speed changes are absorbed by the process noise $\mathbf{Q}$, exactly as in the constant-velocity filter. This adaptive coordinated-turn predictor directly addresses the model mismatch experienced by a pure constant-velocity filter under turning and accelerating obstacles.

Kalman filtering has been extensively studied for dynamic obstacle trajectory prediction. Richter et al. employed a polynomial trajectory prediction method for aggressive quadrotor flight in dense indoor environments, validating the critical role of probabilistic prediction frameworks in enhancing flight safety [36]. The specific procedures are illustrated in Figures 2 and 3.

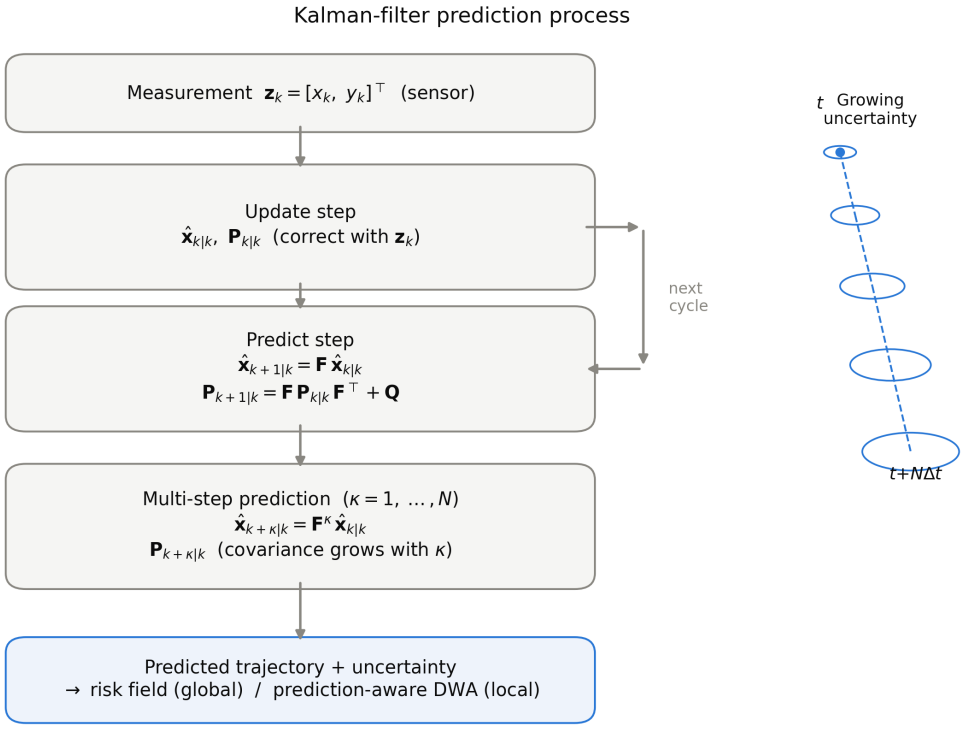


Figure 2. Schematic of the Kalman filter prediction process.

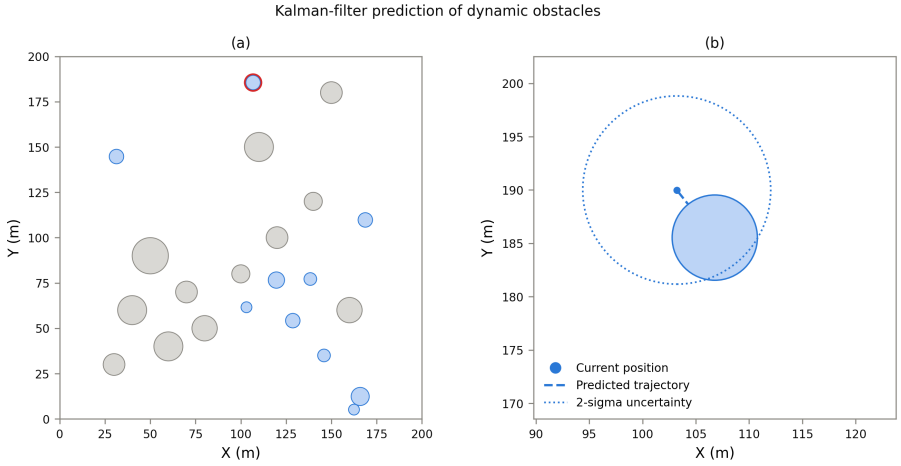


Figure 3. Prediction of dynamic obstacles. (a) Current positions of all dynamic obstacles on the map; the red outline marks the obstacle magnified in (b). (b) Magnified view: the filled circle, dashed line, and dotted ellipse denote the current position, the predicted trajectory over the 5 s prediction horizon, and the 2σ prediction uncertainty, respectively.

3.4. Dynamic Risk Field Construction

Based on the predicted states of obstacles, the potential collision risk in the future can be quantified as a risk field over space. For a single dynamic obstacle, its risk contribution at future step κ, corresponding to time $t + \kappa\Delta t$, to a spatial point $\mathbf{p}$ is defined in the form of a Gaussian function:

$$R_\kappa(\mathbf{p}) = \exp\left(-\frac{1}{2}(\mathbf{p} - \hat{\mathbf{p}}_{k+\kappa|k})^\top \boldsymbol{\Sigma}_\kappa^{-1}(\mathbf{p} - \hat{\mathbf{p}}_{k+\kappa|k})\right) \tag{13}$$

where $\hat{\mathbf{p}}_{k+\kappa|k}$ is the predicted position, and the covariance matrix is jointly composed of the predicted position covariance and the obstacle radius:

$$\boldsymbol{\Sigma}_\kappa = \mathbf{P}^{pp}_{k+\kappa|k} + r_d^2\mathbf{I} \tag{14}$$

When the covariance matrix is not invertible, the formulation degenerates to a Gaussian function based on Euclidean distance. This is illustrated in Figure 4.

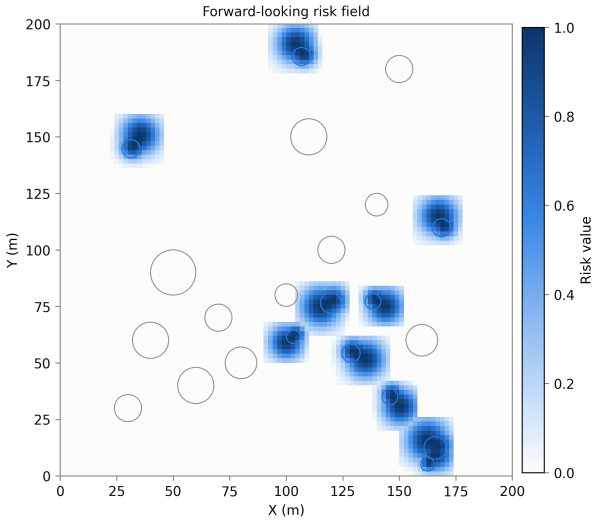


Figure 4. Schematic of the risk field. Thin grey outlines denote static obstacles, dark blue circles denote dynamic obstacles, and the shaded regions give the risk value on the scale shown at the right.

For each grid cell, the maximum risk value across all prediction steps and all dynamic obstacles is taken as the risk value of that cell. Since each Gaussian contribution is already bounded by one, the resulting field lies in the interval $[0, 1]$ without further renormalization. This risk field incorporates both the current positional information of obstacles and their future motion trends and prediction uncertainty, providing a forward-looking threat assessment for path search.

In each planning cycle, the Kalman filters of all dynamic obstacles are advanced and the risk field is recomputed over the whole map, so that the global planner's forward-looking cost map reflects the predicted future positions of obstacles across the entire search space. The UAV's onboard sensor, covering a detection circle of radius 35 m, is used by the local DWA layer to assemble the obstacle list for real-time collision checking. Figure 5 shows the evolution of this field during a representative run.

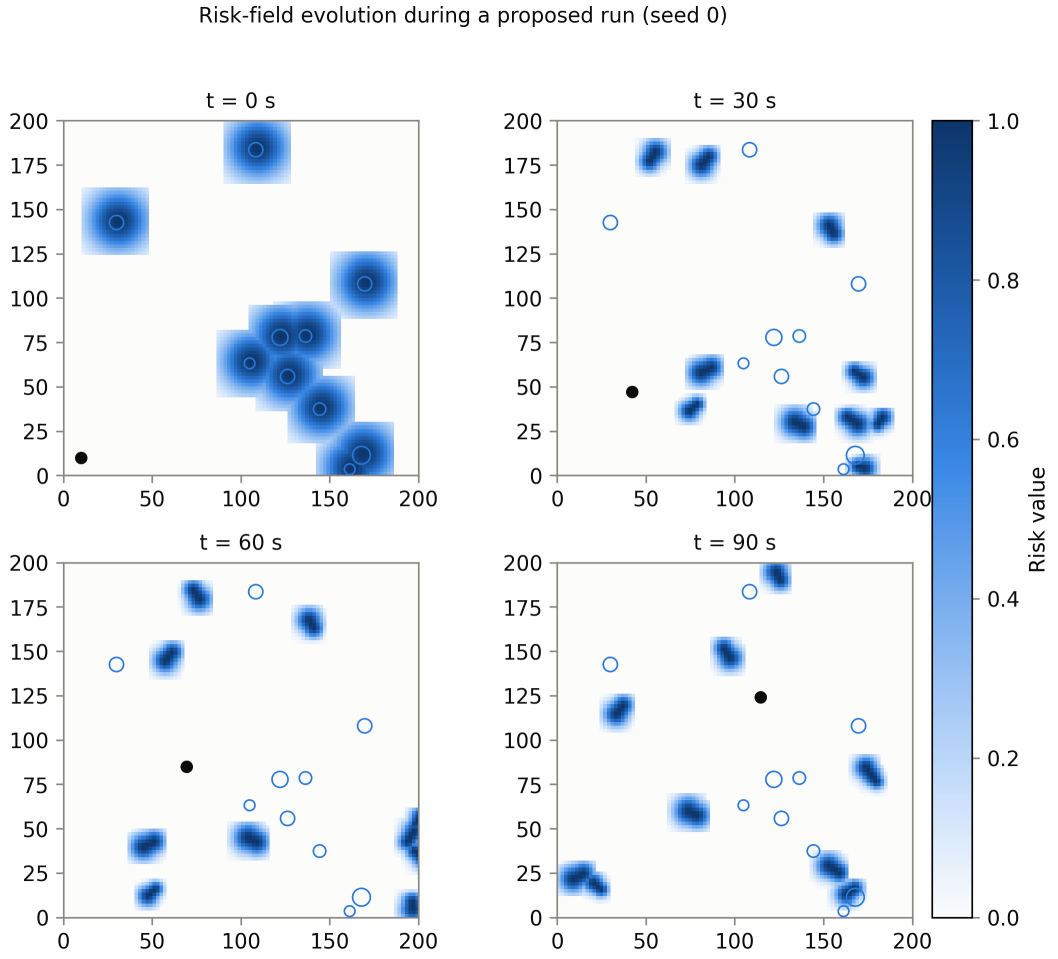


Figure 5. Evolution of the risk field during a representative run of the proposed algorithm, shown at $t = 0$, 30, 60, and 90 s. Hollow circles denote predicted dynamic-obstacle positions, the filled black dot marks the UAV, and the colour scale gives the risk value. The field follows the predicted positions of the dynamic obstacles.

4. Proposed Method

4.1. Risk-Adaptive Heuristic Function

The evaluation function of traditional A* is $f(n) = g(n) + h(n)$, where $h(n)$ is typically the Euclidean distance to the goal. To enable the search to automatically adjust according to environmental risk, this paper designs the following adaptive heuristic:

$$f(n) = g(n) + (1 - \alpha(\bar{r})) \cdot h_{\text{dist}}(n) + \alpha(\bar{r}) \cdot h_{\text{risk}}(n) \tag{15}$$

where

$$\alpha(\bar{r}) = \frac{1}{1 + e^{-\beta(\bar{r} - r_0)}} \tag{16}$$

and $\bar{r} = \frac{1}{|N(n)|} \sum_{m \in N(n)} R(m)$. Here, $\bar{r}$ is the average risk field value within the neighborhood $N(n)$ of node n, a window centered on n with a side length of 5 grid cells, representing the local risk level of that region. The parameter β controls the steepness of the sigmoid transition, and r_0 is the inflection point of the risk sensitivity.

The distance heuristic term is defined as

$$h_{\text{dist}}(n) = \|\mathbf{p}_n - \mathbf{p}_{\text{goal}}\|_2 \tag{17}$$

The risk heuristic term is defined as

$$h_{\text{risk}}(n) = \bar{r} \cdot d_{\max} \tag{18}$$

where $d_{\max}$ is the maximum possible distance within the map, taken here as the diagonal length of the environment, serving as a normalization factor to ensure dimensional consistency between the two heuristic terms.

The design rationale is as follows: When the local risk $\bar{r}$ is low, $\alpha(\bar{r}) \to 0$, and the evaluation function degenerates to $f(n) \approx g(n) + h_{\text{dist}}(n)$, i.e., the traditional distance-priority A*, enabling the algorithm to pursue efficiency in safe regions. When the local risk $\bar{r}$ is high, $\alpha(\bar{r}) \to 1$, and the evaluation function becomes $f(n) \approx g(n) + h_{\text{risk}}(n)$, making the risk term dominant and causing the search to proactively avoid high-risk regions. The sigmoid function ensures a smooth transition between the two extremes, avoiding path oscillations caused by abrupt switching.

We adopt a sigmoid for three reasons, although the comparison in Section 5.3 shows that the exact shape is not performance-critical. First, a hard threshold switch ($\alpha \in \{0, 1\}$ according to whether $\bar{r}$ exceeds r_0) makes the heuristic discontinuous: An infinitesimal change in $\bar{r}$ flips the search between two distinct cost landscapes, which in turn causes the path to oscillate between successive replans as the local risk hovers near r_0. Second, a linear blend ($\alpha = \bar{r}$ clipped to $[0, 1]$) is smoother but does not saturate: The risk term remains active and growing across the whole risk range, so the distance term is never fully trusted even in clearly safe regions, and its transition slope is fixed rather than tunable. Third, the sigmoid $\alpha(\bar{r}) = 1/(1 + e^{-\beta(\bar{r} - r_0)})$ is smooth, bounded in $[0, 1]$, and saturates at both extremes, so the search reverts to pure distance-priority A* in safe regions and to pure risk avoidance in hazardous ones, while exposing β and r_0 as two interpretable controls over the transition sharpness and location. A learned mapping (e.g., a small network that outputs α from $\bar{r}$) could in principle fit a better shape, but for a scalar-to-scalar mapping, it offers little benefit over two interpretable parameters and would require training data that is unavailable in this setting. The paired comparison in Section 5.3 confirms that a linear blend and a hard threshold are statistically indistinguishable from the sigmoid at 100 trials (largest difference 5 percentage points, $p = 0.383$): The sigmoid is adopted for its continuity, boundedness, and interpretability, not because a different smooth shape

would fail. The neighborhood window is a 5×5 grid region (i.e., 10 m $\times$ 10 m at $\delta = 2$ m), which averages the risk over a region comparable to the UAV's local sensing footprint and thereby suppresses cell-level noise; a single-cell sample would be dominated by grid discretization, whereas a much larger window would average away the local hazard signal that the heuristic is meant to capture. In the implementation, a grid cell whose risk exceeds r_0 is also treated as non-traversable during A* expansion, using the same $r_0 = 0.6$ as the sigmoid inflection point. The risk field therefore acts both through the heuristic and as a hard local exclusion; the "w/o adaptive heuristic" ablation in Section 5.3 removes both, so factor A there denotes the risk-adaptive global layer as a whole rather than the sigmoid shape alone.

4.2. Path Smoothing and Redundant Waypoint Optimization

The initial path obtained from the A* search typically contains a large number of redundant turning points. This paper adopts a two-stage post-processing strategy to optimize path smoothness.

4.2.1. Redundant Waypoint Removal

Given the original sequence of path points $\mathcal{P} = \{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_M\}$, the algorithm starts from the start point and attempts to directly connect to the farthest possible subsequent node: If the line segment $\overline{\mathbf{p}_i \mathbf{p}_j}$ does not intersect any static obstacle and the risk values of all sampled points along the segment are below a safety threshold $\tau_{\text{safe}} = 0.7$, then the intermediate nodes $\mathbf{p}_{i+1}, \ldots, \mathbf{p}_{j-1}$ can be removed. This process is repeated until the endpoint is reached, yielding an optimized path that retains only the key turning points.

4.2.2. Cubic Spline Interpolation

For the key path points remaining after redundant waypoint removal, cubic spline interpolation [38] is applied to generate a continuous and curvature-smooth final path. The interpolation is implemented using SciPy's `splprep` and `splev` functions [39], generating a specified number of high-density path points to ensure that the path can be smoothly tracked by the UAV.

4.3. Hybrid Planning Architecture Integrating the DWA

The global adaptive A* provides the UAV with a long-distance reference path but is unable to respond in real time to suddenly appearing obstacles. To address this, the Dynamic Window Approach (DWA) is introduced as a local planner, forming a hybrid planning framework.

4.3.1. Principles of the Dynamic Window Approach

The Dynamic Window Approach (DWA) was proposed by Fox et al. [35]. It achieves effective control of local obstacle avoidance behavior for robots by discretely sampling in velocity space and forward-simulating trajectories based on a kinematic model. DWA discretely samples multiple candidate commands in velocity space (linear velocity v and angular velocity ω) and forward-simulates the motion trajectory over a prediction time window for each candidate command. Collision detection is performed for each point along the trajectory against both static and dynamic obstacles within the perception range; the command is discarded if a collision occurs. For collision-free trajectories, an evaluation function is computed:

$$J(v, \omega) = \sigma_1 \cdot \text{heading}(v, \omega) + \sigma_2 \cdot \text{obs_dist}(v, \omega) + \sigma_3 \cdot \text{goal}(v, \omega) \tag{19}$$

where the heading term evaluates the consistency between the terminal orientation of the trajectory and the local goal direction; the obs_dist term penalizes excessively close proximity to obstacles (in the implementation, a linear penalty that decreases from one at zero clearance to zero at the 3.0 m safety distance); and the goal term rewards the proximity of the trajectory endpoint to the local goal. The command with the highest evaluation score is selected for execution.

4.3.2. Prediction-Aware Forward Simulation

The standard DWA forward-simulates the UAV against the obstacles' *current* positions, implicitly assuming that obstacles are static over the prediction window. This approximation is poor when obstacles move at a speed comparable to the UAV's: A fast-moving obstacle can enter the planned trajectory within the prediction horizon and be missed by the collision check. The proposed framework therefore makes the DWA layer *prediction-aware*. At each planning cycle, the predicted trajectory of every detected dynamic obstacle is obtained from its Kalman filter: The k-step-ahead predicted position $\hat{\mathbf{p}}_{t+k|t}$ is used as the obstacle's location at the k-th forward-simulation step, so the collision check evaluates the UAV against where each obstacle *will* be rather than where it currently is. In our implementation, these predicted positions are taken from the same Kalman filters that construct the global risk field. Sharing one predictor is an engineering choice: It keeps the two layers' beliefs about obstacle motion consistent and avoids maintaining two independent predictors. We do not claim this sharing as a separate algorithmic contribution; the contribution evaluated here is the injection of prediction into the local forward simulation itself. The velocity sampling, evaluation function, and look-ahead target selection remain unchanged; only the obstacle positions used in the forward simulation are propagated in time. The propagation mechanism itself is the same as in the short-horizon constant-velocity variant of [1]; what differs here is the uncertainty-aware Kalman/ACT predictor that supplies the positions and its consistency with the global risk field, not the mechanism of advancing the obstacle positions. This single change lets the local planner anticipate close-range encounters and resolve them before they become imminent, which is a key source of the framework's robustness gain over a static-obstacle DWA.

4.3.3. Local Goal Selection Guided by the Global Path

To achieve global guidance, the local planner does not directly target the final goal point. Instead, a look-ahead target point is dynamically selected from the globally smoothed path. The specific method is as follows: along the global path, the path point closest to the UAV's current position is identified; then, searching forward from that reference point, the first point whose distance exceeds a minimum look-ahead distance of 3–8 m is selected as the local goal for DWA. In this way, the local planner inherits the overall direction of the global path while retaining the capability for autonomous avoidance of nearby obstacles.

4.3.4. Mode Switching and Replanning Mechanism

The hybrid framework includes two operating modes: **Path Following Mode** and **DWA Active Avoidance Mode**. Under normal conditions, the UAV operates in Path Following Mode, with DWA only fine-tuning velocity and angular velocity under the guidance of the global path. When the local risk value around the UAV exceeds a trigger threshold $\tau_{\text{trig}} = 0.3$ for two consecutive frames, the system automatically switches to DWA Active Avoidance Mode. In this mode, the look-ahead target on the global path is shortened from 8 m to 3 m, so the local planner tracks the path more tightly and can deviate around a close obstacle. After the mode has been active for at least 3 s, a drop of the local risk below 0.15 triggers a global replanning, which generates a new path with the current position as the start point, after which Path Following Mode is resumed.

In addition, to prevent the accumulation of errors during long-term operation, a periodic global replanning is configured every 5 s. Each replanning cycle updates the Kalman filters, recomputes the risk field, runs the adaptive A*, and refreshes the global path along with the visualization of the Kalman prediction trajectories. The complete control loop is summarized in Algorithm 1.

Algorithm 1 Prediction-aware hybrid A*–DWA planning: one control cycle

Input: UAV state, global goal $\mathbf{p}_{\text{goal}}$, static obstacle map, dynamic-obstacle detections, and current global path.
Output: velocity command (v, ω) for the current control step.

1. Advance each obstacle's Kalman filter and predict its positions $\hat{\mathbf{p}}_{k+\kappa|k}$ over the horizon (Section 3.3).
2. Build the forward-looking risk field from the predicted states (Section 3.4).
3. **If** the path is empty **or** a 5 s replan is due **then** run the risk-adaptive A* and smooth the path (Sections 4.1 and 4.2).
4. Compute the local risk around the UAV and update the mode state (Section 4.3.4).
5. Select the look-ahead target on the global path (Section 4.3.3).
6. Sample candidate commands (v, ω) and forward-simulate each against the *predicted* obstacle positions (Section 4.3.2).
7. Discard colliding trajectories; score the remaining ones with $J(v, \omega)$ and select the best.
8. Execute the selected command and update the UAV state.

4.4. Algorithm Complexity Analysis

The complexity of the adaptive A* is primarily influenced by the number of expanded nodes and the risk computation. In a 100×100 grid space with 8-neighbor expansion, the worst-case number of expanded nodes is far smaller than the total number of grid cells. A single heuristic computation requires querying the average risk within the neighborhood, which is of $O(|N(n)|)$ complexity. A full global replan takes approximately 0.02 s on an ordinary consumer-grade computer, as measured in Section 5, far below the 5 s replanning period and meeting the real-time requirements of periodic replanning. The DWA is evaluated once per 0.2 s control step; its trajectory sampling and evaluation account for roughly 0.015 s per step (Section 5.2), leaving ample margin for the 5 Hz control loop and ensuring a fast response to dynamic events.

5. Numerical Simulations and Discussion

5.1. Simulation Setup

Simulations were conducted in a 200 m $\times$ 200 m two-dimensional plane with a grid resolution of 2 m. The scene contained 12 circular static obstacles with radii of 5–10 m and 10 dynamic obstacles with radii of 3–5 m and fixed, obstacle-dependent initial velocity vectors (the initial positions are randomized across trials, as detailed in Section 5.2). The UAV start point was set at $(10, 10)$ m and the goal point at $(180, 180)$ m, with a radius of 1.0 m, a maximum linear velocity of 2.5 m/s, maximum angular velocity of 0.9 rad/s, maximum linear acceleration of 2.0 m/s^2, maximum angular acceleration of 1.5 rad/s^2, and a detection radius of 35 m; these acceleration constraints ensure the UAV's commanded motion is dynamically feasible and that the reference path is actually trackable. The Kalman filter executes a predict–update cycle each step, with a process noise variance of 0.5 and an observation noise variance of 2.0, and a prediction horizon of 25 steps spanning 5.0 s, matching the 5 s replanning period. The algorithm configuration was as follows: adaptive A* step size of 4 m, risk threshold of 0.6, a waypoint-removal safety threshold of 0.7, and a sigmoid steepness of $\beta = 10$; DWA prediction time of 1.5 s, with velocity sampling of

12 linear velocity values and 25 angular velocity values. The weights of the DWA evaluation function (Section 4.3.1) were fixed at $\sigma_1 = 1.0$, $\sigma_2 = 2.0$, and $\sigma_3 = 0.5$, with the goal term defined as $1/(1+d)$ for endpoint-to-local-goal distance d and an obstacle safety distance of 3.0 m. These are conventional DWA weights, fixed a priori and not tuned in this work; the weighting structure follows the original formulation [35]. All programs were implemented in Python, encompassing complete environment updating, planning, smoothing, and animation demonstrations. Multiple simulation runs were conducted with the above settings, and a representative example from the final results is shown in Figure 6.

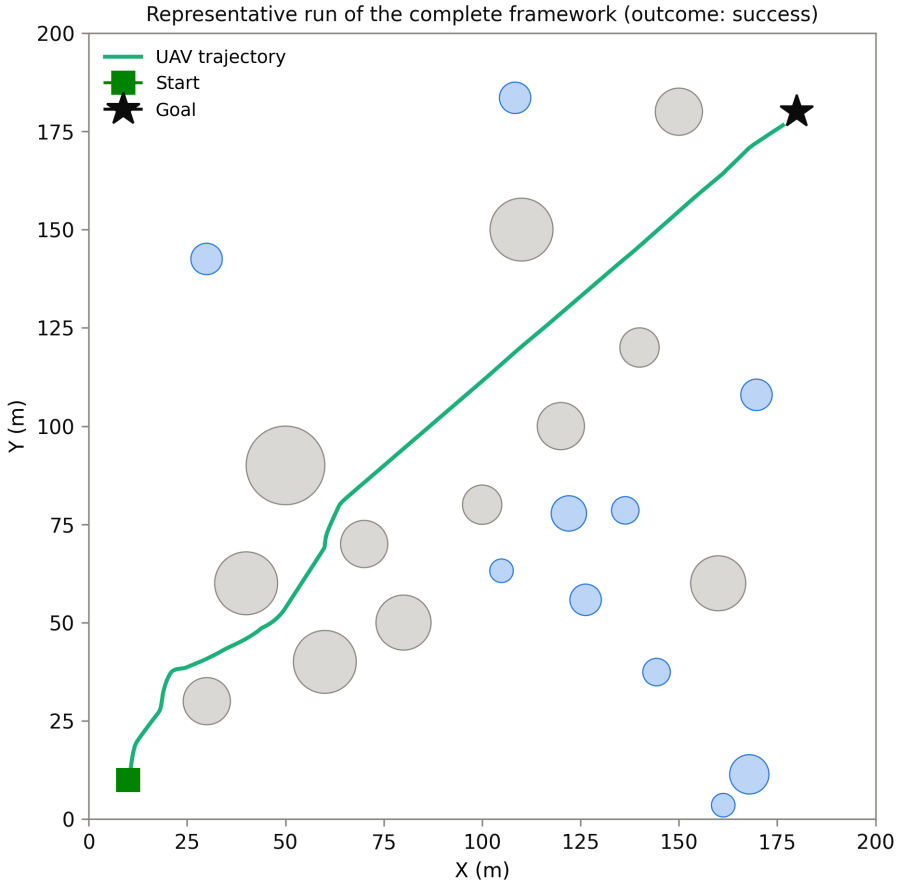


Figure 6. Simulation results. Grey circles denote static obstacles, light blue circles the initial positions of the dynamic obstacles, the green line the UAV trajectory, the green square the start point, and the black star the goal.

5.2. Comparative Simulation Studies and Analysis

To validate the performance of the proposed framework, the following methods were compared under identical conditions:

1. **Traditional A*:** Euclidean heuristic over the static map, with no prediction, smoothing, or local planning; it replans every 5 s but does not incorporate dynamic obstacles.
2. **D* Lite [28]:** an incremental replanner that updates only the cells affected by obstacle changes and replans every 0.2 s simulation step; it still reacts to instantaneous positions and provides no local avoidance.
3. **DWA-only:** the local DWA layer without any global path, steering directly toward the goal.
4. **A* + DWA:** a plain hybrid of Euclidean A* (static obstacles only) and the standard DWA, whose forward simulation treats obstacles as static; this is the existing A*–DWA baseline.

5. **IA*-only (improved A*):** the risk-adaptive A* and path smoothing of Sections 4.1 and 4.2 without the DWA layer (pure-pursuit tracking).
6. **Proposed framework:** the risk field, risk-adaptive heuristic, and DWA of Sections 4.1–4.3, but with a static-obstacle DWA (i.e., without prediction-aware forward simulation).
7. **Proposed framework + prediction-aware DWA:** the complete method of Section 4.3.

Each method was run 100 times against the same 100 randomly generated environments (paired seeds), so every method faces an identical set of scenarios; the initial positions of the dynamic obstacles were randomized while their radii and velocity vectors were held fixed. The evaluation metrics are:

- **Success rate (%):** A collision is defined as the UAV's minimum distance to any obstacle being less than the UAV radius + obstacle radius + 0.5 m. The success rate is the proportion of runs reaching the goal without collision, reported with a 95% Wilson confidence interval.
- **Path length (m):** the total length of the flown trajectory, averaged over successful runs.
- **Tracking error (m):** the mean minimum distance from the UAV to the planned path, averaged over successful runs, quantifying how closely the reference path is followed under the UAV's acceleration constraints. The distance is geometric: It is measured to the nearest point on the current reference path, not to a time-parameterized reference, so a UAV that is on the path but lagging along it records zero tracking error. This is the appropriate interpretation for a DWA-tracked path; temporal lag is reflected in the success rate and path length rather than in this metric.
- **Replanning time (s) and full control-cycle time (s):** The former is the cost of one global replan, executed every 5 s; the latter is the average cost per 0.2 s control step, comprising Kalman prediction, risk-field construction, and DWA evaluation with the periodic global replan amortized over the 5 s interval, and is the cost that matters for real-time operation.

Because the simulations are paired by seed, differences in success rate are assessed with an exact McNemar test, and differences in continuous metrics with paired t-tests and Wilcoxon signed-rank tests. The results are presented in Table 3.

Table 3. Simulation results for each method over 100 randomized trials (paired seeds). Success-rate 95% confidence intervals (Wilson) are reported in brackets. Path length and tracking error are averaged over successful runs only. Traditional A* and DWA-only never reached the goal, so their path-quality metrics are undefined (N/A); the D* Lite path metrics are averaged over only its two successful runs.

Method	Success Rate (%)	Path Length (m)	Tracking Error (m)	Replanning Time (s)
Traditional A*	0.0 [0.0, 3.7]	N/A	N/A	0.021
D* Lite	2.0 [0.6, 7.0]	238.8	0.79	0.009
DWA-only	0.0 [0.0, 3.7]	N/A	N/A	N/A
A* + DWA	70.0 [60.4, 78.1]	243.7	3.18	0.014
IA*-only	35.0 [26.4, 44.7]	241.6	3.77	0.021
Proposed algorithm	77.0 [67.8, 84.2]	247.0	3.35	0.017
Proposed + Pred-DWA	**87.0 [79.0, 92.2]**	**247.8**	**3.33**	**0.016**

Every failure is a collision—predominantly with a dynamic obstacle, with a small number of static-obstacle collisions (Section 5.9)—and no timeouts or stuck episodes occurred on the nominal scene. The complete framework fails in 13 of 100 runs, versus 30 for plain A* + DWA and 100 for traditional A*. The full control cycle (Kalman prediction, risk-field construction, DWA evaluation, and the global replan amortized over its 5 s period) requires

about 0.015 s on average per 0.2 s step, confirming real-time feasibility on the evaluation hardware (Windows 11, 8-core CPU, Python 3.14).

Table 3 shows a clear hierarchy in the success rate, the primary safety indicator. Traditional A* and DWA-only never reach the goal: The former has neither prediction nor local reactive avoidance, and the latter, lacking global guidance, cannot find a feasible route across the map. D* Lite, which replans incrementally every 0.2 s, reaches only 2%—fast incremental replanning is not by itself sufficient, because it still reacts to instantaneous obstacle positions and offers no local avoidance. The plain A*–DWA baseline attains 70%, confirming that a global planner coupled with reactive DWA is a strong but incomplete solution. The proposed framework, by adding the Kalman risk field and the risk-adaptive heuristic, raises this to 77%, and adding prediction-aware DWA raises it further to 87%. The 17-percentage-point advantage of the complete framework over plain A*–DWA is statistically significant under a paired McNemar test ($p = 0.0005$), and prediction-aware DWA itself contributes a significant 10-point gain over the non-predictive framework ($p = 0.002$). Representative trajectories of the three methods in the same environment are shown in Figure 7.

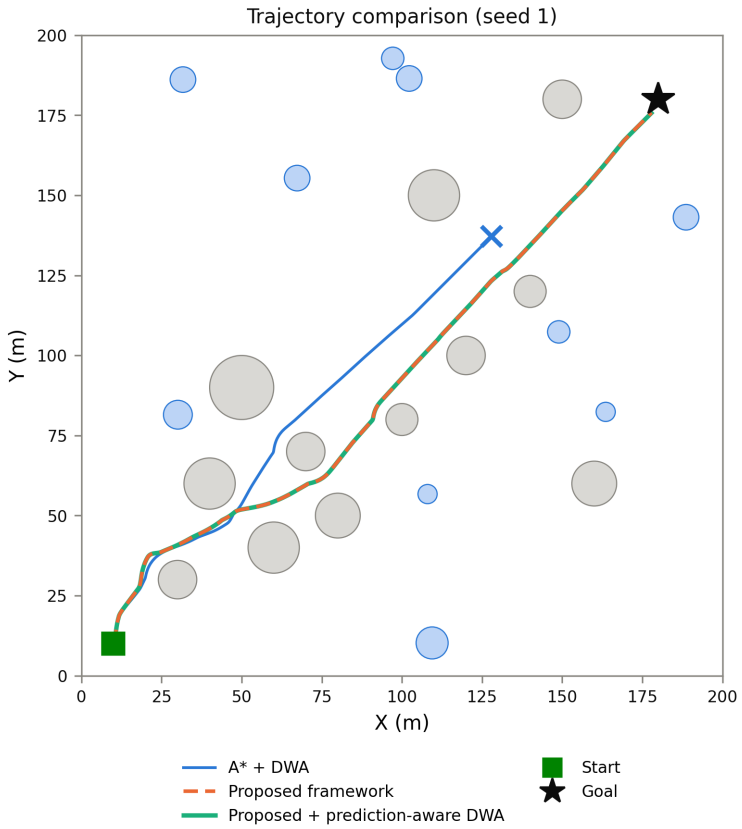


Figure 7. Representative trajectories of three methods in the same environment. Blue curve: A* + DWA, which ends in a collision (marked with a cross). Orange dashed curve: proposed framework. Green curve: proposed framework + prediction-aware DWA. The latter two nearly coincide and both reach the goal.

The comparison also isolates the source of each gain. IA*-only, which retains prediction and the adaptive heuristic but drops DWA, reaches 35%, so the DWA layer accounts for the largest single improvement (the success rate falls from 87% to 35% when it is removed). Conversely, prediction-aware DWA is what closes the remaining gap against the state-of-the-art hybrid, adding 10 percentage points on top of the risk-aware framework. We do not attribute this gain to using one motion model in both layers: The factorial study of

Section 5.3 shows that prediction-aware DWA adds only 1 percentage point without the adaptive heuristic on the nominal scene, and 22–41 percentage points in the structured scenes regardless of the heuristic, while the w/o-Kalman ablation shows that replacing the global Kalman covariance with CV extrapolation changes the success rate by a borderline 10 points (77% versus 87%, $p = 0.053$). These analyses do not isolate sharing a predictor from maintaining two separate predictors; they indicate that the local layer benefits from receiving *a* prediction and that the specific global Kalman covariance adds only a modest margin. What drives the gain is the local prediction-aware forward simulation itself, which resolves the close-range encounters that a static-obstacle DWA misses.

Among the A*-based methods, path lengths are broadly comparable (241.6–247.8 m), and the mean cross-track tracking error (3.1–3.8 m) confirms that the smoothed reference paths remain dynamically feasible under the UAV's acceleration constraints (D* Lite's two successful runs give 238.8 m and 0.79 m): The UAV follows the planned path with bounded deviation, and the slightly higher values of the proposed framework reflect deliberate avoidance maneuvers rather than a failure to track. Every observed failure was a collision (with a dynamic or, in a few cases, a static obstacle); no episode timed out or became stuck. The full control cycle—Kalman prediction, risk-field construction, DWA evaluation, and the global replan amortized over its 5 s period—requires about 0.015 s per step on the evaluation hardware, an order of magnitude below the 0.2 s planning cycle and therefore not a practical limitation. Figure 8 summarizes the run-to-run distribution of the main metrics.

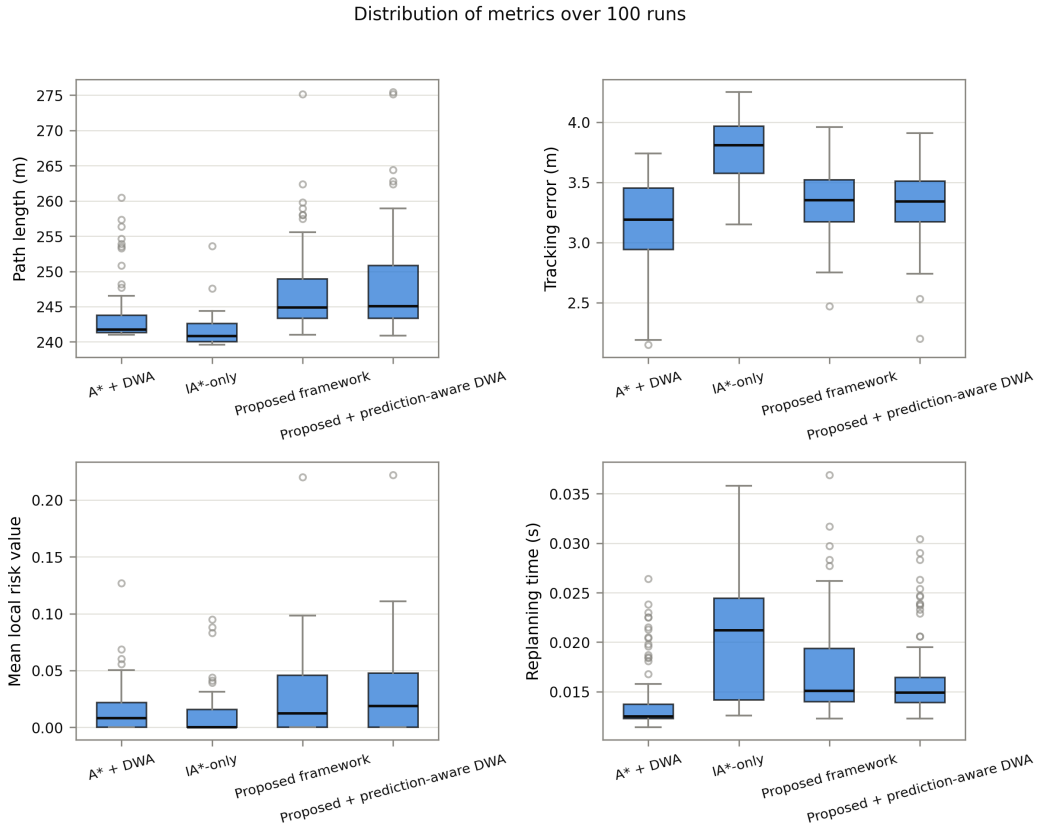


Figure 8. Distribution of the main metrics over the 100 runs. Path length and tracking error are computed over successful runs only, while replanning time is computed over all runs.

5.3. Ablation Study

To quantify the contribution of each module, the complete framework was degraded one component at a time, and each configuration was run 100 times against the same paired environments. The configurations are:

- **W/o prediction-aware DWA.** The DWA forward simulation reverts to the standard static-obstacle check, isolating the contribution of the prediction-aware local layer.
- **W/o Kalman prediction.** The risk field is built from constant-velocity extrapolation with a fixed diagonal covariance $\Sigma = \mathrm{diag}(1.0 + \mathrm{step} \cdot 0.5)$ instead of the Kalman prediction covariance.
- **W/o adaptive heuristic.** The risk-adaptive global layer is disabled: The A* heuristic reverts to the Euclidean distance and the risk-threshold node rejection of Section 4.1 is removed. The risk field is still computed for the DWA mode-switching trigger.
- **W/o DWA.** The local layer is removed entirely and tracking uses pure pursuit; this is identical to IA*-only.

The results are compared in Table 4.

Table 4. Ablation results over 100 randomized trials. Each row removes or replaces one component of the complete framework; success-rate 95% confidence intervals (Wilson) are shown in brackets. Path length and tracking error are averaged over successful runs only.

Configuration	Success Rate (%)	Path Length (m)	Tracking Error (m)
Complete framework (prediction-aware DWA)	87.0 [79.0, 92.2]	247.8	3.33
w/o prediction-aware DWA	77.0 [67.8, 84.2]	247.0	3.35
w/o Kalman prediction (CV extrapolation)	77.0 [67.8, 84.2]	247.0	3.40
w/o adaptive heuristic	71.0 [61.5, 79.0]	242.9	3.16
w/o DWA (pure pursuit)	35.0 [26.4, 44.7]	241.6	3.77

Removing prediction-aware DWA degrades the success rate from 87% to 77% (paired McNemar test, $p = 0.002$); removing the adaptive heuristic degrades it to 71% ($p = 0.002$); and removing the DWA layer entirely degrades it to 35% ($p < 0.001$). Replacing the Kalman prediction with constant-velocity extrapolation lowers the success rate to 77% (versus 87%, $p = 0.053$), a borderline difference that does not reach the 5% level.

Removing prediction-aware DWA—the paper's central local component—degrades the success rate from 87% to 77% ($p = 0.002$): A static-obstacle DWA misses the close-range encounters that a prediction-aware DWA anticipates and resolves ahead of time. Removing the adaptive heuristic has a comparable effect (87% to 71%, $p = 0.002$), confirming that the risk-adaptive term steers the global search away from hazardous corridors. Replacing the Kalman prediction with constant-velocity extrapolation lowers the success rate to 77% (versus 87%, $p = 0.053$), a borderline difference; the Kalman filter's specific advantage over simple extrapolation is therefore modest in this scenario, and its value lies primarily in the uncertainty-aware covariance that widens the risk field where prediction is least trustworthy, a property that matters most under maneuvering obstacles and would matter further under real sensor noise (Section 5.12). The same result cautions against treating the shared-model architecture as the source of the gain: The local layer benefits from receiving *a* prediction, not specifically from sharing the Kalman model with the global layer. Removing the DWA layer entirely—by far the largest single effect—collapses the success rate to 35% ($p < 0.001$): Without the high-frequency velocity-space search, even a risk-aware global path cannot react to sudden close-range motion changes between successive replans.

A caveat on the w/o-Kalman ablation is that it replaces the *global* risk field's Kalman prediction with constant-velocity extrapolation, while the local DWA still uses the Kalman

tracker. It therefore does not isolate a local CV-versus-Kalman comparison. Such a comparison would require driving the local DWA from CV extrapolation; we did not run it, because over the 1.5 s DWA horizon, the constant-velocity predictor is already accurate for non-maneuvering obstacles, and the Kalman filter's distinguishing output—the uncertainty covariance—is consumed by the global risk field rather than by the DWA score. The relevant distinction from the short-horizon CV local prediction of [1] is accordingly the uncertainty-aware global coupling and the ACT adaptation under maneuvering motion, not the local propagation step itself.

Taken together, the ablations reveal a clear division of labor: The DWA layer is the primary driver of success, prediction-aware DWA and the adaptive heuristic each add significant robustness on top of it, and the Kalman risk field provides a smaller, borderline complementary safety margin. This division—reactive avoidance for success, predictive and adaptive planning for robustness—supports the prediction–planning–avoidance architecture proposed in this paper.

The one-factor-at-a-time ablations show how each component behaves in isolation, but they cannot reveal whether the two predictive components interact. To test this, a full 2×2 factorial design crossed the risk-adaptive global layer (the adaptive heuristic plus the risk-threshold node rejection; factor A) with prediction-aware DWA (factor B), giving the four configurations of Table 5: plain A* + DWA (A off, B off), A* with prediction-aware DWA (A off, B on), the proposed framework without prediction-aware DWA (A on, B off), and the complete framework (A on, B on). Each cell was run on the nominal scene and on the three structured scenes (Section 5.7) over the same 100 paired seeds.

Table 5. Full 2×2 factorial over 100 paired trials: risk-adaptive global layer (adaptive heuristic plus risk-threshold node rejection; factor A) $\times$ prediction-aware DWA (factor B). Success rate (%) with Wilson 95% confidence intervals in brackets. "Proposed" denotes the framework without prediction-aware DWA.

Scenario	A* + DWA (A off, B off)	A* + Pred-DWA (A off, B on)	Proposed (A on, B off)	Proposed + Pred-DWA (A on, B on)
Nominal	70 [60.4, 78.1]	71 [61.5, 79.0]	77 [67.8, 84.2]	**87 [79.0, 92.2]**
Narrow passage	57 [47.2, 66.3]	79 [70.0, 85.8]	59 [49.2, 68.1]	**83 [74.5, 89.1]**
Bottleneck	51 [41.3, 60.6]	80 [71.1, 86.7]	45 [35.6, 54.8]	**86 [77.9, 91.5]**
Dead end	45 [35.6, 54.8]	71 [61.5, 79.0]	56 [46.2, 65.3]	**85 [76.7, 90.7]**

The factorial reveals a clean interaction structure. In the open nominal scene, prediction-aware DWA contributes little without the risk-adaptive heuristic (+1 percentage point, McNemar $p = 1.0$) but a significant +10 points when the heuristic is present ($p = 0.002$): The heuristic steers the global path into corridors where the local predictive layer then has something meaningful to resolve. In the structured scenes, the pattern inverts and sharpens: Prediction-aware DWA delivers a large and highly significant gain regardless of the heuristic (+22 to +41 points, all $p \leq 0.001$), whereas factor A alone (the risk-adaptive global layer) is weak or even mildly harmful there (+2 points in the narrow passage, $p = 0.82$; −6 points in the bottleneck, $p = 0.31$; and +11 points in the dead end, $p = 0.09$). The interaction is significant on the nominal scene (+9 percentage points, permutation $p = 0.012$): There the heuristic amplifies prediction-aware DWA's benefit from +1 to +10 points. In the structured scenes, the interaction is not significant (+2 to +12 percentage points, $p = 0.096$–0.87), so there the two factors are broadly additive, with a suggestive positive interaction in the bottleneck (+12 points, $p = 0.096$). This confirms the structured-environment finding as the framework's headline result—prediction-aware DWA is the minimal intervention that removes the reactive-layer failure mode—while

the adaptive heuristic is a complementary refinement whose largest contribution is in open terrain.

To test whether the specific shape of the risk-adaptive weight matters, the sigmoid of Equation (16) was replaced by the two natural alternatives discussed in Section 4.1: a linear blend $\alpha = \mathrm{clip}(\bar{r}, 0, 1)$ and a hard threshold $\alpha = \mathbb{1}[\bar{r} > r_0]$. Each shape was run on the same 100 paired seeds, in the proposed configuration and in the complete framework. Table 6 reports the paired differences relative to the sigmoid. The corresponding absolute success rates are given in Supplementary Table S2. No difference is statistically significant: The linear blend changes the success rate by at most 3 percentage points ($p \geq 0.629$) and the hard threshold by at most 5 points ($p \geq 0.383$). The exact shape of the weight is therefore not performance-critical. The sigmoid is retained for its continuity, boundedness, and interpretability, but the paper does not claim that a different smooth shape would fail.

Table 6. Paired comparison of the risk-adaptive weight shape over 100 seeds. Values are success-rate differences relative to the sigmoid (negative means lower than the sigmoid); McNemar p values are shown in parentheses. Because the three shapes were run in the same implementation build and on the same seed set, only the paired differences are interpreted.

Weight Shape	Proposed		Proposed + Pred-DWA	
	Δ (pp)	p	Δ (pp)	p
Linear blend	−1.0	1.000	−3.0	0.629
Hard threshold	−3.0	0.711	−5.0	0.383

5.4. Robustness Under Maneuvering Obstacles

The nominal scenario uses obstacles with constant velocity and boundary bouncing, a favorable case for the constant-velocity Kalman model. To test the framework under model mismatch, a second scenario was generated in which each dynamic obstacle turns at an angular rate uniformly sampled in $[-0.6, 0.6]$ rad/s and accelerates at a rate uniformly sampled in $[-0.4, 0.4]$ m/s^2 (speed clamped to $[0.3, 3.0]$ m/s), while keeping the same obstacle count, radii, and evaluation protocol over 100 paired trials. The results are reported in Table 7.

Table 7. Success rate over 100 paired trials in the maneuvering-obstacle scenario.

Method	Success Rate (%)	Tracking Error (m)
Traditional A*	0.0 [0.0, 3.7]	N/A
DWA-only	0.0 [0.0, 3.7]	N/A
IA*-only	29.0 [21.0, 38.5]	3.94
A* + DWA	71.0 [61.5, 79.0]	3.31
Proposed algorithm	78.0 [68.9, 85.0]	3.46
Proposed + Pred-DWA	80.0 [71.1, 86.7]	3.46

The complete framework still outperforms plain A* + DWA (80% versus 71%), although the difference is not statistically significant at 100 trials ($p = 0.093$), and the overall success rates are lower than in the nominal scenario, as expected for a more demanding environment. Crucially, the adaptive coordinated-turn predictor of Section 3.3 restores the benefit of the prediction-aware layer: Prediction-aware DWA now adds 2 percentage points under maneuvering obstacles (80% versus 78%, McNemar $p = 0.500$). The mechanism is confirmed at the prediction level: Averaged over 100 maneuvering obstacle trajectories, the adaptive predictor reduces the 5 s ahead position RMSE from 11.2 m (constant-velocity predictor) to 5.5 m, a 51% reduction. The end-to-end gain is positive but modest and does not yet reach significance at 100 trials, because the 1.5 s DWA forward-simulation horizon is

short relative to the turn; a larger gain is expected for longer horizons, and a full interacting multiple model (IMM) remains future work (Section 5.12).

The 2×2 factorial of Section 5.3 was extended to this scene (Supplementary Table S1). It provides no statistically significant evidence that the risk-adaptive global layer (factor A, including its risk-threshold node rejection) compensates for prediction-aware DWA's weakness under maneuvering motion. Prediction-aware DWA changes the success rate by -1 percentage point without factor A and $+2$ points with it (McNemar $p = 1.000$ and 0.500), whereas factor A adds $+7$ points without prediction-aware DWA and $+10$ points with it ($p = 0.210$ and 0.064); the interaction is $+3$ points ($p = 0.45$). The point estimates for the factor-A effect are positive but not significant, so a small compensation cannot be ruled out at 100 trials. Mechanistically, the local layer's weakness is a model-mismatch effect within its 1.5 s forward-simulation horizon, which global path shaping is not expected to remove; the effective mitigation is the adaptive coordinated-turn predictor itself. Under crossing traffic (Section 5.5), the pattern is the opposite: The global risk field and adaptive heuristic carry most of the gain, and prediction-aware DWA adds little.

5.5. Robustness Under Crossing and Head-On Obstacle Motion

The maneuvering scenario of Section 5.4 stresses the framework under obstacles that violate the constant-velocity assumption. A complementary and equally common threat is traffic that remains constant-velocity but crosses the UAV's route: other aircraft flying straight horizontal or vertical paths that repeatedly intersect the start-to-goal diagonal. To test this, a third scenario assigned the ten dynamic obstacles straight horizontal or vertical velocity vectors (magnitudes 1.0–1.3 m/s) while randomizing only their initial positions per trial, so that the UAV's natural diagonal route is crossed repeatedly by oncoming and crossing traffic. Because the obstacles move at constant velocity, the Kalman model remains a valid predictor here; this scenario therefore isolates the value of prediction under realistic, predictable-but-crossing motion, complementing the model-mismatch case of Section 5.4.

Table 8 reports the success rate over 100 paired trials. Traditional A* again fails outright (0%), and IA*-only attains 28%. The plain A*–DWA baseline reaches 68%, whereas the complete framework reaches 85%, a 17-percentage-point advantage. Notably, the risk field and adaptive heuristic already account for most of this gain (Proposed attains 82%): Because the crossing obstacles move in straight lines, the global risk field—which propagates the same constant-velocity prediction across the whole map—steers the search around the crossing corridors before they become imminent. Prediction-aware DWA adds only three further points here, in contrast to its larger role in the nominal and structured scenarios. Taken together with Section 5.4, this shows that prediction is valuable in two distinct regimes: crossing traffic, where the global layer suffices, and obstacles moving along the UAV's own path, where the local prediction-aware layer is decisive.

Table 8. Success rate over 100 paired trials in the crossing/head-on traffic scenario.

Method	Success Rate (%)
Traditional A*	0.0 [0.0, 3.7]
IA*-only	28.0 [20.1, 37.5]
A* + DWA	68.0 [58.3, 76.3]
Proposed algorithm	82.0 [73.3, 88.3]
Proposed + Pred-DWA	85.0 [76.7, 90.7]

5.6. Scalability with Obstacle Density

To assess how the framework degrades as the environment becomes more crowded, the number of dynamic obstacles was swept over $\{5, 10, 15, 20\}$ with the 12 static obstacles held fixed, and a crowded-static variant (24 static and 10 dynamic obstacles) was added. The extra obstacles were generated deterministically so that the layouts are reproducible, while the initial positions of the dynamic obstacles remained randomized per trial under the same paired-seed protocol. Table 9 reports the success rates; traditional A* attains 0% at every density and is omitted, and IA*-only falls from 43% at five obstacles to 20–23% at 15–20 obstacles. All reactive and predictive methods degrade as the scene becomes more crowded, but the complete framework degrades most gracefully: Its advantage over the plain A*–DWA baseline grows monotonically from 3 percentage points at five obstacles to 17, 18, and 31 points at 10, 15, and 20 obstacles, and reaches 24 points in the crowded-static scene. The full control-cycle time also grows with density (from 0.010 s at five obstacles to 0.025 s at 20), but remains an order of magnitude below the 0.2 s planning cycle, so crowding raises the required safety margin rather than the computational burden.

Table 9. Success rate (%) over 100 paired trials as the number of dynamic obstacles increases (12 static obstacles unless stated). Wilson 95% confidence intervals in brackets. Traditional A* attains 0% at every density and is omitted. "Proposed" denotes the framework without prediction-aware DWA.

Dynamic Obstacles	IA*-Only	A* + DWA	Proposed	Proposed + Pred-DWA
5	43.0 [33.7, 52.8]	92.0 [85.0, 95.9]	91.0 [83.8, 95.2]	95.0 [88.8, 97.8]
10	35.0 [26.4, 44.7]	70.0 [60.4, 78.1]	77.0 [67.8, 84.2]	87.0 [79.0, 92.2]
15	20.0 [13.3, 28.9]	59.0 [49.2, 68.1]	70.0 [60.4, 78.1]	77.0 [67.8, 84.2]
20	23.0 [15.8, 32.2]	40.0 [30.9, 49.8]	57.0 [47.2, 66.3]	71.0 [61.5, 79.0]
10 (24 static)	39.0 [30.0, 48.8]	66.0 [56.3, 74.5]	83.0 [74.5, 89.1]	90.0 [82.6, 94.5]

5.7. Structured Environments: Narrow Passages, Bottlenecks, and Dead Ends

The nominal, maneuvering, and crossing scenarios are open environments in which the local planner is never forced through a constrained opening. To expose failure modes that are invisible in open scenes, three structured environments were constructed from overlapping circular obstacles: a full-spanning vertical wall with a 16 m gap (narrow passage), a full-spanning horizontal barrier with a 20 m central gap (bottleneck), and a concave U-shaped pocket placed on the direct start-to-goal line (dead end). Each contains eight dynamic obstacles, with initial positions randomized per trial.

Table 10 reports the results over 100 paired trials. In the narrow passage and the bottleneck, the reactive layer is actively counterproductive when it is not prediction-aware: The static-obstacle DWA sees the walls and any nearby dynamic obstacle as stationary, oscillates in the gap, and becomes stuck, so A* + DWA and the non-predictive framework record 12–20 stuck episodes and path lengths near 450–490 m (roughly twice the direct route). Prediction-aware DWA eliminates every stuck episode and restores the direct $\sim$240 m path, reaching 83% and 86% versus 45–59% for the non-predictive variants. In the dead end, traditional A* fails in all 100 trials and only the complete framework reliably finds the detour around the pocket (85% versus 45–56%). Interestingly, in the narrow passage, traditional A* (67%) exceeds A* + DWA (57%): Because A* has no reactive layer, it simply follows its global path through the mostly empty gap by pure pursuit, whereas the static-obstacle DWA introduces an oscillation failure mode that prediction-aware DWA then removes. This is a concrete illustration of why prediction must be injected into the local layer rather than added only as a global cost map.

Table 10. Success rate (%) over 100 paired trials in three structured environments (walls formed from overlapping circular obstacles, eight dynamic obstacles). Wilson 95% confidence intervals in brackets. "Proposed" denotes the framework without prediction-aware DWA.

Scenario	Traditional A*	IA*-Only	A* + DWA	Proposed	Proposed + Pred-DWA
Narrow passage	67 [57.3, 75.4]	56 [46.2, 65.3]	57 [47.2, 66.3]	59 [49.2, 68.1]	**83 [74.5, 89.1]**
Bottleneck	59 [49.2, 68.1]	55 [45.2, 64.4]	51 [41.3, 60.6]	45 [35.6, 54.8]	**86 [77.9, 91.5]**
Dead end	0 [0.0, 3.7]	18 [11.7, 26.7]	45 [35.6, 54.8]	56 [46.2, 65.3]	**85 [76.7, 90.7]**

Table 11 decomposes the same runs into outcomes. The non-predictive variants lose 12–20 runs to stuck/oscillation and a further 24–41 to collisions; the prediction-aware variant records no stuck episodes at all, and every remaining failure is a collision. This is why the failure-mode taxonomy of Section 5.9 classifies collisions by geometry on the nominal scene, where no stuck episodes occur, while the structured scenes require the separate stuck/oscillation category reported here.

Table 11. Outcome breakdown over 100 paired trials in the structured environments (counts out of 100). "Stuck/oscillation" denotes a run that exhausted the simulation time limit while the reactive layer oscillated in the gap. "Proposed" denotes the framework without prediction-aware DWA.

Scenario	Configuration	Success	Collision	Stuck/Oscillation
Narrow passage	A* + DWA	57	26	17
Narrow passage	Proposed	59	24	17
Narrow passage	Proposed + Pred-DWA	**83**	17	0
Bottleneck	A* + DWA	51	37	12
Bottleneck	Proposed	45	40	15
Bottleneck	Proposed + Pred-DWA	**86**	14	0
Dead end	A* + DWA	45	41	14
Dead end	Proposed	56	24	20
Dead end	Proposed + Pred-DWA	**85**	15	0

5.8. Parameter Sensitivity

To assess the robustness of the framework to its manually tuned parameters, the prediction horizon, the adaptive-heuristic steepness β, and the Kalman noise variances Q and R were each swept over a range around the nominal setting, using the proposed configuration (without prediction-aware DWA) and 100 paired seeds per setting. The nominal values were fixed a priori (25 steps = 5.0 s, matching the 5 s replanning period; $\beta = 10$; $Q = 0.5$; and $R = 2.0$), and all main experiments use these values; the sweeps are exploratory and are not used to select the reported configuration. Because the sweeps use the non-predictive configuration, their rates correspond to the "Proposed" row of Table 3 rather than to the complete framework; the within-sweep variation is the quantity of interest. The results are reported in Table 12.

Table 12. Parameter sensitivity of the proposed framework (100 paired trials per setting; the success rate is reported in %; the nominal setting is marked in bold; Wilson 95% confidence intervals for every point are reported in Supplementary Table S3).

Parameter	Success Rate (%) at Each Value				
Prediction horizon (steps)	5: 69.0	10: 70.0	15: 68.0	**25: 77.0**	40: 85.0
β	1: 81.0	5: 76.0	10: 77.0	20: 78.0	50: 72.0
Observation noise R	0.5: 77.0	2: 77.0	5: 81.0	10: 80.0	–
Process noise Q	0.1: 78.0	**0.5: 77.0**	2: 85.0	5: 88.0	–

The success rate varies between 68% and 88% across the entire sweep, with overlapping Wilson intervals and no catastrophic degradation at any setting, indicating that the framework is not critically sensitive to any single parameter. The observation noise has only a weak effect (77–81%). The prediction horizon shows an upward trend at the long end: 68–70% for 5–15 steps, 77% at the nominal 25 steps (5 s), and 85% at 40 steps (8 s). The 40-step setting is not adopted because the nominal horizon was fixed a priori to match the 5 s replanning period; a horizon longer than the replanning interval relies on predictions that are not refreshed within the planning cycle, so the increase is reported as a sensitivity result rather than used to select the configuration post hoc. The process noise shows a clear upward trend (78% at $Q = 0.1$ to 88% at $Q = 5.0$), indicating that the nominal $Q = 0.5$ is conservative: A larger Q widens the prediction covariance and the risk field and improves robustness. The β sweep is flat within 72–81%, with no monotonic trend. Taken together, the sweep shows no collapse and identifies the prediction horizon and the process noise as the two parameters with the largest effect.

5.9. Failure-Mode Analysis

To understand *why* prediction-aware DWA helps and when it fails, the collisions of the three DWA-based configurations on the nominal scene were categorized by the relative geometry between the UAV and the colliding obstacle at the moment of impact, using the angle between their velocity vectors: head-on (approaching from opposite directions, angle $> 135°$), crossing (roughly perpendicular, 45°–135°), and overtaking (same direction, <45°). Table 13 reports the counts over 100 paired trials, and Figure 9 shows one representative failure trajectory per category. This analysis is deliberately restricted to the nominal scene, where failures are exclusively collisions. In the structured scenes, the dominant non-collision failure is stuck/oscillation, which is not a collision geometry and is therefore reported as a separate outcome category in Table 11 rather than forced into this taxonomy.

Table 13. Collision categories over 100 paired trials on the nominal scene, classified by the angle between the UAV and obstacle velocity vectors at impact. Only collisions are classified: The nominal scene has no non-collision failures. "Static" denotes a collision with a static obstacle. Non-collision failures in the structured scenes (stuck/oscillation) are reported separately in Table 11.

Configuration	Success	Head-On	Crossing	Overtaking	Static
A* + DWA	70	10	16	3	1
Proposed algorithm	77	3	16	2	2
Proposed + Pred-DWA	87	4	6	1	2

The dominant failure mode is crossing traffic, followed by head-on encounters; overtaking and static collisions are rare. The two predictive components attack *different* failure modes. The risk-adaptive heuristic primarily removes *head-on* collisions (10 for plain A* + DWA down to 3 for the proposed framework): By steering the global path away from predicted high-risk corridors, it routes the UAV around oncoming traffic before it becomes a close-range threat. Prediction-aware DWA, by contrast, removes *crossing* collisions (16 down to 6): A crossing obstacle that is still distant when the UAV commits to a maneuver is exactly the case a static-obstacle forward simulation misses, and propagating its predicted motion lets the local planner anticipate the intersection. Together, the two mechanisms reduce total collisions from 30 to 13, which explains their complementary roles in the factorial study of Section 5.3.

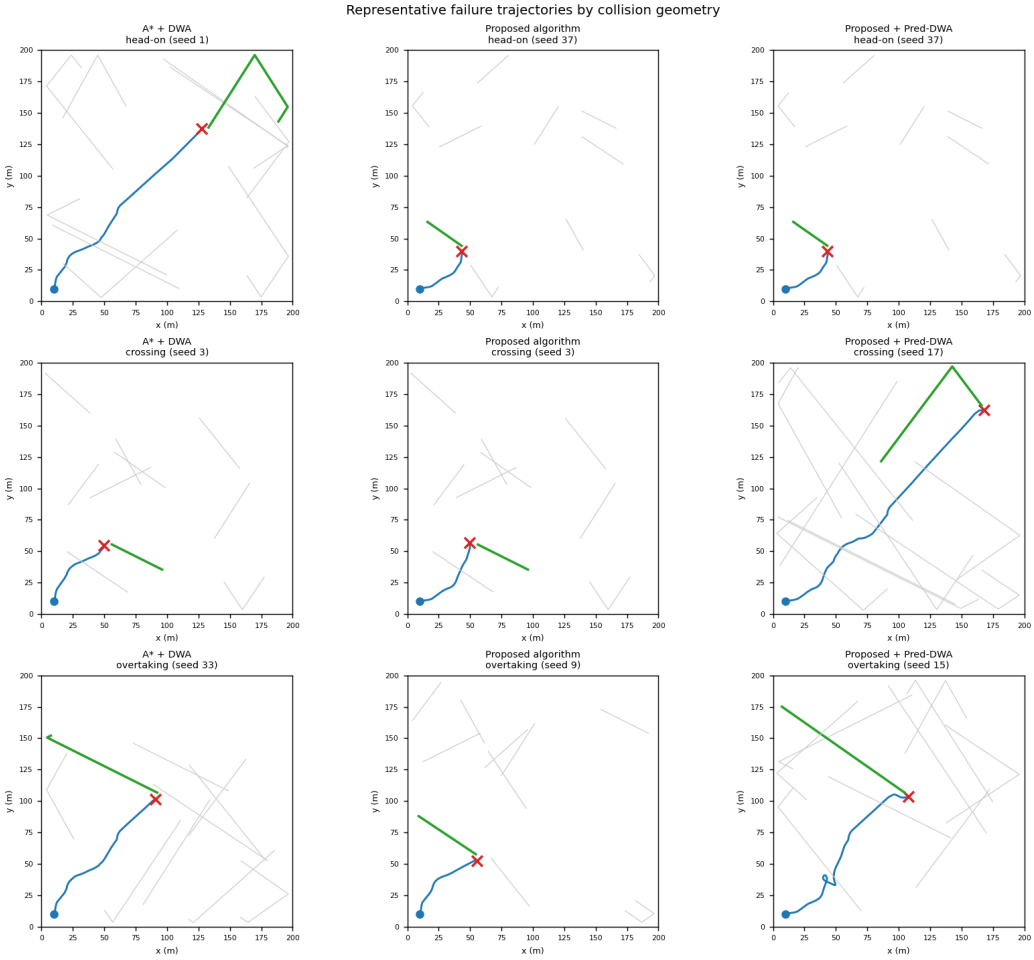


Figure 9. Representative failure trajectories categorized by collision geometry. Rows: head-on, crossing, and overtaking; columns: A* + DWA, proposed framework, and proposed framework + prediction-aware DWA. The blue line is the UAV trajectory (blue dot: start), the green line the trajectory of the colliding obstacle, the red cross the collision point, and the faint grey segments the static obstacles.

5.10. Learned Prediction Baseline

Section 5.4 showed that a pure constant-velocity model degrades under maneuvering obstacles, which the adaptive turn-rate estimate of Section 3.3 mitigates; a complementary question is whether a learned predictor would do even better. To answer it, a small gated recurrent unit (GRU) was trained to map a window of ten noisy position observations to the next true obstacle position, and compared against the constant-velocity Kalman filter and naive constant-velocity extrapolation on held-out trajectories generated by the paper's two motion models (constant velocity with boundary bouncing, and maneuvering with turning and acceleration). All three predictors observe the same noisy measurements ($R = 2.0$) and predict the true position κ steps ahead; the GRU is rolled out autoregressively. Table 14 reports the position RMSE, and Figure 10 plots it against the prediction horizon.

Table 14. Position RMSE (m) of obstacle-trajectory prediction as a function of the prediction horizon κ (steps of 0.2 s), for constant-velocity and maneuvering motion.

Scenario	Predictor	$\kappa = 1$	$\kappa = 5$	$\kappa = 15$	$\kappa = 25$
Constant velocity	CV extrapolation	4.50	15.74	44.21	72.73
Constant velocity	Kalman filter (CV)	2.15	3.61	7.54	11.75
Constant velocity	GRU (learned)	1.52	2.80	6.35	9.74
Maneuvering	CV extrapolation	4.50	15.74	44.25	72.89
Maneuvering	Kalman filter (CV)	2.35	4.04	8.96	14.54
Maneuvering	GRU (learned)	1.62	3.55	9.74	15.92

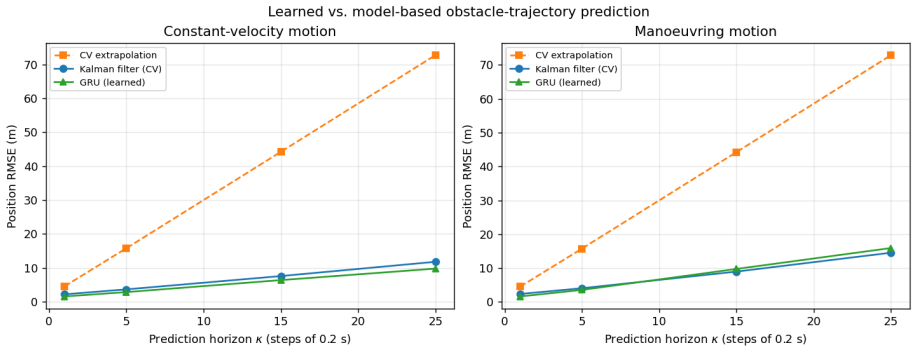


Figure 10. Prediction RMSE versus horizon for the three predictors on constant-velocity and maneuvering obstacle motion.

On constant-velocity motion, the learned predictor modestly exceeds the Kalman filter (RMSE 9.7 m versus 11.8 m at a 5.0 s horizon), while both are far better than naive extrapolation ($\sim$73 m), whose velocity estimate is dominated by measurement noise. On maneuvering motion, the learned predictor remains best at short horizons (1.6 m versus 2.4 m at one step) but its advantage shrinks and essentially vanishes at long horizons (15.9 m versus 14.5 m at 5.0 s): The short-horizon denoising benefit is offset by error compounding in the autoregressive rollout, and the constant-velocity model is equally wrong for both predictors over long horizons. These results are honest about the trade-off: The Kalman filter is competitive *without any training data* and, crucially, provides the uncertainty covariance that the risk field requires, so the Kalman predictor (augmented with the adaptive turn-rate estimate of Section 3.3) remains the appropriate online predictor for this framework; a learned predictor is an attractive future replacement specifically for short-horizon, strongly maneuvering obstacles, at the cost of training data and an uncertainty estimate.

A limitation of this comparison is that the GRU is trained and evaluated on the same synthetic motion models used to generate the scenarios. It therefore answers the question "can a learned predictor beat Kalman on the same motion model?" but not "would a predictor trained on real obstacle trajectories generalize better than Kalman on real obstacles?" The latter remains an open question: Testing it would require a dataset of real pedestrian or aircraft trajectories, which is outside the scope of this study.

5.11. Three-Dimensional Validation with a 6-DOF Quadrotor

The preceding sections use a planar kinematic model. To test whether the mechanisms survive a fully three-dimensional, dynamically constrained plant, this section evaluates the same planning stack on a rigid-body quadrotor in a $100 \times 100 \times 40$ m volume. The plant has mass 1.5 kg and inertia $\mathrm{diag}(0.02, 0.02, 0.04)$ kg m^2; its thrust is saturated at 1.8 mg, its body torques are saturated at $(0.35, 0.35, 0.15)$ N m, and its translational acceleration is limited to 4 m/s^2 horizontally and 3 m/s^2 vertically. A cascaded controller tracks the planner's velocity command through a velocity loop and a geometric attitude loop, so the

closed loop includes attitude dynamics, actuator saturation, and tracking lag rather than the instantaneous velocity changes of the kinematic model. The 3D study uses a direct 3D adaptation of the planner: the 3D risk-adaptive A*, a 3D constant-velocity Kalman predictor with the same predict–update structure, and a 3D DWA. Several simplifications relative to the 2D implementation are that the DWA layer is active throughout each run rather than risk-triggered; collision-free candidates are ranked by heading and goal proximity with collisions rejected as hard constraints (the soft obstacle-distance term of Equation (19) is omitted); the DWA forward simulation is kinematic and does not ramp velocity, yaw rate, and vertical velocity subject to the acceleration limits that the 6-DOF plant enforces in closed loop; both the 3D global planner and the DWA use the full simulated obstacle state rather than a 35 m detection cutoff; and the 3D A* applies the risk field only through the heuristic rather than also rejecting cells above r_0 as the 2D implementation does.

Two scenarios are evaluated. The *altitude-window* scenario retains the walls of the preliminary study—a full-spanning low ridge and a high slab that leave a ∼5 m altitude window—and adds three movers that cross the window vertically and four that cross it horizontally. The *3D crossing-traffic* scenario contains no static walls; eight movers at altitudes between 5 and 35 m cross the route with horizontal, vertical, or mixed velocities. Both scenarios use 100 paired seeds (well above the 20–30 requested) and four configurations: a true Euclidean 3D A* with pure-pursuit tracking (6 m look-ahead and 1.6 m/s speed cap; the baseline requested by the reviewer), the risk-adaptive 3D A* (denoted 3D IA*), and the latter with a static-obstacle or prediction-aware DWA. The 1.6 m/s cap is deliberately conservative: A faster pure-pursuit follower cuts the corner at the altitude window and crashes even on a collision-free path, whereas the DWA configurations command velocities up to 2.0 m/s (0.8 of the 2.5 m/s plant limit) and are therefore not favored by the speed cap.

Table 15 reports the results, and Figure 11 visualizes the success rates with their confidence intervals. Prediction-aware DWA is the best configuration in both scenarios (90% and 94%), 18 and 17 percentage points above the static-obstacle DWA; the paired McNemar test is significant in both scenarios ($p < 0.001$ in both; 18 and 17 discordant pairs in favor of prediction-aware DWA and none against). Against the global-only planners, the gain is +19 points in the window and +11 in crossing, and is also significant at 100 trials ($p = 0.0009$ and $p = 0.019$). The static-obstacle DWA is not significantly different from global-only planning (72% versus 71% in the window; 77% versus 83% in crossing; and $p = 1.0$ and $p = 0.36$), and is six points worse in crossing; unlike the 2D structured scenes, the 3D static DWA is therefore not clearly counterproductive, but it adds no consistent benefit. The Euclidean and risk-adaptive 3D A* planners are indistinguishable (71% versus 73% in the window, $p = 0.69$ and 83% versus 85% in crossing, $p = 0.50$), so in these scenarios, the global risk field does not materially change the success rate or route; the benefit comes from injecting prediction into the local layer. Path lengths of the DWA variants are within 0.5 m of the global-only paths, tracking errors stay below 1 m, and the full control cycle (0.002–0.012 s) remains far below the 0.2 s planning step; a global replan takes 0.004–0.027 s.

Table 15. 6-DOF quadrotor results over 100 paired trials in two three-dimensional scenarios. Success-rate 95% Wilson confidence intervals are shown in brackets; path length and tracking error are averaged over successful runs; full step time is the average cost per 0.2 s control step over all runs. All configurations use the same rigid-body plant, low-level attitude controller, and paired seeds. "3D A*" is the true Euclidean baseline; "3D IA*" is the risk-adaptive global planner.

Scenario	Configuration	Success (%)	Path (m)	Track Err. (m)	Full Step (s)
Altitude window	3D A* (Euclidean)	71 [61.5, 79.0]	94.8	0.81	0.0022
	3D IA* (risk-adaptive)	73 [63.6, 80.7]	94.8	0.80	0.0033
	3D IA* + DWA	72 [62.5, 79.9]	94.7	0.82	0.0115
	3D IA* + Pred-DWA	**90 [82.6, 94.5]**	94.9	0.83	0.0114
3D crossing	3D A* (Euclidean)	83 [74.5, 89.1]	86.3	0.54	0.0019
	3D IA* (risk-adaptive)	85 [76.7, 90.7]	86.7	0.55	0.0021
	3D IA* + DWA	77 [67.8, 84.2]	86.6	0.76	0.0115
	3D IA* + Pred-DWA	**94 [87.5, 97.2]**	86.8	0.76	0.0115

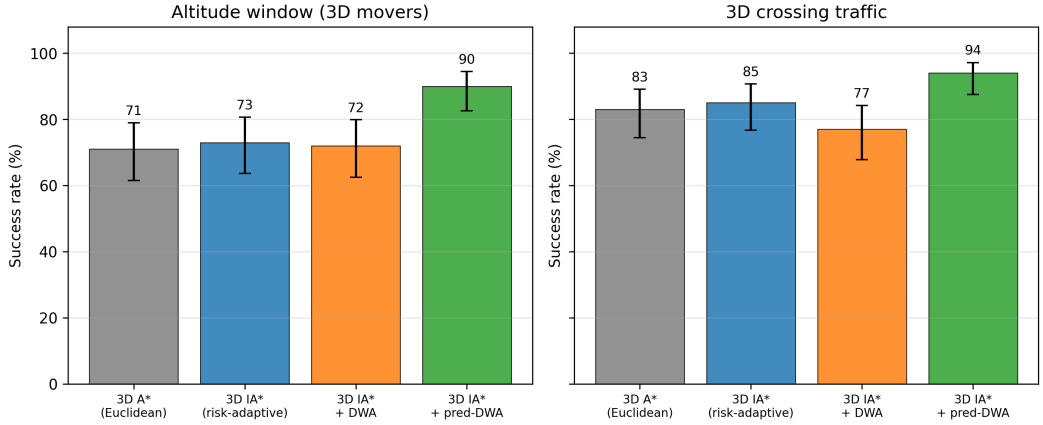


Figure 11. Success rate with Wilson 95% confidence intervals for the 6-DOF quadrotor in the two three-dimensional scenarios: altitude window (**left**) and 3D crossing traffic (**right**). Prediction-aware DWA is the best configuration in both scenarios.

These absolute success rates are lower than those of the earlier kinematic 3D preliminary study, as expected once attitude dynamics, thrust/torque saturation, and tracking lag are included; the benefit of injecting prediction into the local layer, however, is preserved, and the prediction-aware local layer remains the decisive component. This is a rigid-body simulation, not a Gazebo/ROS2 SITL campaign or an in situ flight experiment: It closes the kinematic/dynamic part of the sim-to-real gap (attitude dynamics, actuator limits, and three-dimensional obstacle motion) but not the perception, actuator, and outdoor-disturbance parts, which remain future work as stated in Section 5.12.

5.12. Discussion and Future Work

Although the proposed framework has demonstrated favorable overall performance in the numerical simulations, several limitations merit in-depth discussion.

At the level of modeling assumptions, the proposed method involves five interrelated simplifications. First, although Section 5.11 now evaluates a full 6-DOF quadrotor in a three-dimensional volume, the main comparative study is formulated in a two-dimensional plane, and the 3D extension remains simulation-only: It omits rotor-level aerodynamics and full sensor models and has not been validated in a physics engine or on hardware. The framework's applicability to complex three-dimensional scenes is therefore demonstrated, but not yet established at the fidelity of a flight campaign. Second, the Kalman filter prediction module adopts a constant-velocity motion model as its state estimator.

To mitigate the model mismatch that this assumption introduces under maneuvering obstacles, the prediction is augmented with an online turn-rate estimate and a coordinated-turn transition (Section 3.3), which restores the benefit of the prediction-aware layer under turning and accelerating obstacles (Section 5.4). The coordinated-turn model nevertheless assumes an approximately constant turn rate and speed over the prediction horizon, so obstacles whose turn rate or speed changes rapidly within the horizon can still degrade the multi-step prediction and bias the risk field, affecting the reliability of the global path safety assessment. Third, the main comparative study uses a planar kinematic UAV model without full quadrotor attitude dynamics or an inner-loop position/attitude controller; the 6-DOF study of Section 5.11 adds a rigid-body plant and a cascaded attitude controller for the 3D validation, but the 2D study does not, and tightly following the cubic-spline path under fast dynamics would still call for a higher-fidelity tracking controller such as nonlinear model predictive control (NMPC) to suppress the tracking lag typical of simple feedback loops. A hardware-in-the-loop or flight experiment would be needed to fully confirm real-world feasibility, but at present, we do not have the facilities, safety approvals, and flight-test infrastructure required for an in situ campaign; the 6-DOF simulation is therefore the strongest validation available in this revision cycle. Fourth, the onboard sensor is modeled generically as an idealized detector that reports obstacle positions within 35 m at 5 Hz without sensor noise, and without modeling a specific device, its field of view, or its false-positive/false-negative rates; this abstraction keeps the study hardware-agnostic but leaves the perception pipeline—including real sensor noise—to be validated on real sensors. In the same idealization, the global planner is given the simulated state of every dynamic obstacle over the whole map, so the 35 m radius constrains only the local DWA's obstacle list; a real deployment would need a global tracker that extrapolates beyond the sensing range, which this study does not model. Fifth, the framework implicitly assumes that threats maneuver on timescales longer than the 5 s replanning period; a fast intruder whose speed is comparable to or exceeds the UAV's and whose trajectory changes within the prediction horizon can outrun both the local prediction and the periodic global replan, an operating regime the current predictor only partially addresses. A related limitation concerns the learned-predictor comparison: The GRU of Section 5.10 is trained on synthetic trajectories generated by the paper's own motion models, so its generalization to real pedestrian or aircraft trajectories is untested.

At the level of algorithm design and scalability, the proposed method likewise faces two shortcomings. On the one hand, multiple key parameters involved in the algorithm—including the adaptive weight coefficient β and the risk trigger threshold—still rely on empirical manual tuning for specific scenarios and lack the capability for online adaptive optimization according to environmental characteristics. The quality of parameter configuration directly affects the generalization performance of the algorithm across different scenarios. Manual parameter tuning not only increases the complexity of engineering deployment but also makes it difficult to guarantee the sustained robustness of the algorithm in dynamically changing environments. On the other hand, the current algorithm is only designed for a single-UAV scenario and has not yet addressed mutual collision avoidance and distributed cooperative planning among multiple UAVs. In real-world scenarios involving multiple UAVs simultaneously executing missions, the real-time assessment of inter-UAV collision risks and cooperative avoidance will impose more complex requirements on the communication mechanisms and decision-making architecture of the planning algorithm.

The structured scenarios of Section 5.7 also warrant an explicit caveat. In a narrow passage, the plain A*–DWA baseline underperforms even traditional A*, because the purely reactive DWA oscillates in the gap and becomes stuck; this is a well-known shortcoming of reactive local planners in constrained geometry, and in this work, it is precisely the failure

that the prediction-aware layer removes. The structured environments were nevertheless built from rows of overlapping circles rather than arbitrary polygonal walls, and all of the newly added scenarios (maneuvering, crossing, density, and structured) remain numerical simulations, so their findings should be interpreted within that scope until confirmed on a physical platform.

Future work may be pursued along the following directions: extending the three-dimensional validation to a full physics engine and to more complex environments [40], drawing on BIM-supported inspection-oriented planning such as building exterior surface inspection [41]; extending the adaptive coordinated-turn predictor to a full Interacting Multiple Model (IMM) with a richer bank of motion models, or to learning-based trajectory prediction, to further improve prediction robustness against strongly maneuvering obstacles [42]; adopting Model Predictive Control (MPC) in place of DWA to handle dynamic constraints more precisely, drawing on safety-assured high-speed navigation [43] and aggressive flight trajectory generation [44]; leveraging reinforcement learning for online adjustment of algorithm parameters and end-to-end navigation [45]; and investigating distributed multi-UAV cooperative planning and collision avoidance under communication constraints, as studied for multi-vehicle real-time decision-making [46].

6. Conclusions

This paper addressed the core problems of obstacle-motion prediction, adaptive search, and global–local coordination for UAV path planning in dynamic environments. The central contribution is a prediction–planning–avoidance framework in which Kalman-filter motion prediction is injected into *both* planning layers: a prediction-uncertainty risk field and a risk-adaptive heuristic for the global A* search, and a prediction-aware Dynamic Window Approach (DWA) whose forward simulation propagates the predicted obstacle motion. Sharing a single predictor across the two layers is an implementation choice rather than a contribution in its own right; the empirical contribution is the demonstration that injecting uncertainty-aware prediction into the local layer is what removes a failure mode the reactive layer itself introduces in constrained geometry. In numerical simulations over 100 randomized trials with 12 static and 10 dynamic obstacles under UAV acceleration constraints, the complete framework attains an 87% success rate, versus 0% for traditional A*, 35% for IA*-only, and 70% for a plain A*–DWA baseline (McNemar test, $p = 0.0005$). The gain is driven primarily by the DWA avoidance layer, with prediction-aware DWA contributing a significant further 10 percentage points and the risk-adaptive heuristic an additional significant margin; the Kalman risk field supplies a smaller, borderline complementary benefit. Beyond the nominal scene, the framework's advantage over a plain A*–DWA baseline grows monotonically with obstacle density (from 3 to 31 percentage points), and prediction-aware DWA is decisive in narrow-passage, bottleneck, and dead-end environments (83–86% versus 45–59% for a static-obstacle DWA), where a non-predictive reactive DWA oscillates and even underperforms plain A*, confirming both that the benefit is not an artifact of a single open-field scenario and that prediction is most valuable when injected into the local reactive layer rather than added only as a global cost map.

From a broader perspective, the work shows that prediction is most valuable when it reaches the layer that makes the fast decisions. Global prediction shapes where to fly, but it is the prediction-aware local layer that resolves close-range encounters, and in constrained geometry, that layer is what removes a failure mode introduced by reactive avoidance itself. Future work will extend the framework to more complete three-dimensional perception and actuator models, and to multi-UAV cooperative scenarios, as discussed in Section 5.12.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/aerospace13121102/s1, Table S1: Full 2×2 factorial on the maneuvering scene; Table S2: Absolute success rates of the weight-shape comparison; and Table S3: Full parameter-sensitivity results.

Author Contributions: Conceptualization, T.J. and M.H.; methodology, Y.Z.; software, Y.Z.; validation, Y.Z.; formal analysis, Y.Z.; investigation, Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, T.J., M.H., T.W. and M.M.; visualization, Y.Z.; supervision, T.W.; and funding acquisition, M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (No. 72401291), and the Innovation Research Foundation of National University of Defense Technology (No. ZK23-34).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The simulation code and data supporting the findings of this study are openly available at the following GitHub repository: https://github.com/Mr-Merak/Prediction-Aware-Hybrid-A-DWA-Path-Planning-for-UAV-Obstacle-Avoidance-in-Dynamic-Environments (accessed on 14 November 2026). The code was implemented in Python 3.14.

Acknowledgments: During the preparation of this manuscript, the authors used DeepSeek-V4.1 Flash (https://www.deepseek.com/, accessed on 14 November 2026) for the purposes of partial code generation and Chinese-to-English translation. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle
DWA	Dynamic Window Approach
IA*	Improved A*
RRT	Rapidly exploring Random Trees
APF	Artificial Potential Field
PSO	Particle Swarm Optimization
GA	Genetic Algorithm
ACO	Ant Colony Optimization
GWO	Grey Wolf Optimizer
SAS	Sparse A* Search
CV	Constant Velocity
IMM	Interacting Multiple Model
MPC	Model Predictive Control

References

1. Wang, Y.; Hu, C.; Kang, Y. Global path planning and dynamic obstacle avoidance of UAV in 3D dynamic environment. In *Proceedings of the 5th International Conference on Computer, Artificial Intelligence and Control Engineering (CAICE '26)*; ACM: New York, NY, USA, 2026.
2. Hu, W.; Ma, X. Optimization algorithm of UAVs task assignment and path planning based on dynamic cluster particle swarm optimization. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 18157–18169.
3. Pehlivanoglu, Y.V.; Pehlivanoglu, P. An enhanced genetic algorithm for path planning of autonomous UAV in target coverage problems. *Appl. Soft Comput.* **2021**, *112*, 107796.
4. Hu, G.; Huang, F.; Shu, B.; Wei, G. MAHACO: Multi-algorithm hybrid ant colony optimizer for 3D path planning of a group of UAVs. *Inf. Sci.* **2025**, *694*, 121714.
5. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61.

6. Khaneghaei, M.; Asadi, D.; Ebrahimi, B.; Tutsoy, Ö.; Chashmi, Y.N. Intelligent hybrid optimization algorithms for multi-agent aerial robots path planning: Review of the recent emerging trends and open research directions. *Artif. Intell. Rev.* **2026**, *59*, 94.
7. Rivière, B.; Lathrop, J.; Chung, S.J. Monte Carlo tree search with spectral expansion for planning with dynamical systems. *Sci. Robot.* **2024**, *9*, eado1010.
8. Izzo, D.; Blazquez, E.; Ferede, R.; Origer, S.; De Wagter, C.; de Croon, G.C.H.E. Optimality principles in spacecraft neural guidance and control. *Sci. Robot.* **2024**, *9*, eadi6421.
9. He, Y.; Hou, T.; Wang, M. A new method for unmanned aerial vehicle path planning in complex environments. *Sci. Rep.* **2024**, *14*, 9257.
10. Xu, L.; Cao, X.; Du, W.; Li, Y. Cooperative path planning optimization for multiple UAVs with communication constraints. *Knowl.-Based Syst.* **2023**, *260*, 110164.
11. Xu, X.; Xie, C.; Ma, L.; Yang, L.; Zhang, T. Multi-objective evolutionary algorithm with two balancing mechanisms for heterogeneous UAV swarm path planning. *Appl. Soft Comput.* **2025**, *173*, 112927.
12. Xu, X.; Xie, C.; Liu, M.; Zheng, M.; Zhang, T. Real-time path planning for heterogeneous UAV swarm based on two-stage optimization. *Adv. Eng. Inform.* **2026**, *70*, 104208.
13. Yildiz, B.; Aslan, M.F.; Durdu, A.; Kayabasi, A. Consensus-based virtual leader tracking swarm algorithm with GDRRT*-PSO for path-planning of multiple-UAVs. *Swarm Evol. Comput.* **2024**, *88*, 101612.
14. Zhou, Y.; Kong, X.; Lin, K.-P.; Liu, L. Novel task decomposed multi-agent twin delayed deep deterministic policy gradient algorithm for multi-UAV autonomous path planning. *Knowl.-Based Syst.* **2024**, *287*, 111462.
15. Xiong, C.; Chen, D.; Lu, D.; Zeng, Z.; Lian, L. Path planning of multiple autonomous marine vehicles for adaptive sampling using Voronoi-based ant colony optimization. *Robot. Auton. Syst.* **2019**, *115*, 90–103.
16. Zeng, Z.; Sammut, K.; Lian, L.; Lammas, A.; He, F.; Tang, Y. Rendezvous path planning for multiple autonomous marine vehicles. *IEEE J. Ocean. Eng.* **2018**, *43*, 640–664.
17. Yin, L.; Yan, Z.; Tian, Q.; Li, H.; Xu, J. A novel robust algorithm for path planning of multiple autonomous underwater vehicles in the environment with ocean currents. *Ocean Eng.* **2024**, *312*, 119260.
18. Zhuang, Y.; Huang, H.; Sharma, S.; Xu, D.; Zhang, Q. Cooperative path planning of multiple autonomous underwater vehicles operating in dynamic ocean environment. *ISA Trans.* **2019**, *94*, 174–186.
19. Yao, P.; Qi, S. Obstacle-avoiding path planning for multiple autonomous underwater vehicles with simultaneous arrival. *Sci. China Technol. Sci.* **2019**, *62*, 121–132.
20. Sun, Y.; Yuan, Q.; Gao, Q.; Xu, L. A multiple environment available path planning based on an improved A* algorithm. *Int. J. Comput. Intell. Syst.* **2024**, *17*, 172.
21. Ding, Y.; Xin, B.; Dou, L.; Chen, J.; Chen, B.M. A memetic algorithm for curvature-constrained path planning of messenger UAV in air-ground coordination. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 3735–3749.
22. Wang, W.; Ding, W.; Hua, C.; Heng, Z.; Haibing, F.; Yao, Y. A digital twin for 3D path planning of large-span curved-arm gantry robot. *Robot. Comput.-Integr. Manuf.* **2022**, *76*, 102330.
23. Sun, W.; Sun, P.; Ding, W.; Zhao, J.; Li, Y. Gradient-based autonomous obstacle avoidance trajectory planning for B-spline UAVs. *Sci. Rep.* **2024**, *14*, 14458.
24. He, Q.; Wang, Z.; Li, K.; Zhang, Y.; Li, M. Research on autonomous navigation of mobile robots based on IA-DWA algorithm. *Sci. Rep.* **2025**, *15*, 2099.
25. Liao, Y.; Wang, Z.; Lin, Y.; Liu, C.; Hu, L.; Lu, J.; Wu, Z. Voxel-based path planning for UAVs in indoor dynamic environments. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2025**, *XLVIII-4/W14-2025*, 149–155.
26. Hai, S.; Na, X.; Feng, Z.; Shi, J.; Sun, Q. PENC: A predictive-estimative nonlinear control framework for robust target tracking of fixed-wing UAVs in complex urban environments. *Sci. Rep.* **2025**, *15*, 29753.
27. Stentz, A. Optimal and efficient path planning for partially-known environments. In *Proceedings of the IEEE International Conference on Robotics and Automation, San Diego, CA, USA, 8–13 May 1994*; IEEE: Piscataway, NJ, USA, 1994; pp. 3310–3317.
28. Koenig, S.; Likhachev, M. D* Lite. In *Proceedings of the AAAI Conference on Artificial Intelligence*; AAAI Press: Edmonton, AB, Canada, 2002; pp. 476–483.
29. Chen, H.; Lu, P. Real-time identification and avoidance of simultaneous static and dynamic obstacles on point cloud for UAVs navigation. *Robot. Auton. Syst.* **2022**, *154*, 104124.
30. Fnadi, M.; Du, W.; Gomes da Silva, R.; Plumet, F.; Benamar, F. Local obstacle-skirting path planning for a fast bi-steerable rover using Bézier curves. In *Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–1 August 2020*; IEEE: Piscataway, NJ, USA, 2020; pp. 10242–10248.
31. Ajeil, F.H.; Ibraheem, I.K.; Sahib, M.A.; Humaidi, A.J. Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm. *Appl. Soft Comput.* **2020**, *89*, 106076.
32. Rao, J.; Xiang, C.; Xi, J.; Chen, J.; Lei, J.; Giernacki, W.; Liu, M. Path planning for dual UAVs cooperative suspension transport based on artificial potential field-A* algorithm. *Knowl.-Based Syst.* **2023**, *277*, 110797.

33. Ren, X.; Geng, N.; Zhang, Y.; Xiao, L.; Gong, D. PG-ITD3: A potential field-guided deep reinforcement learning approach for UAV path planning after disaster. *IEEE Trans. Autom. Sci. Eng.* **2025**, *22*, 20221–20233.
34. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107.
35. Fox, D.; Burgard, W.; Thrun, S. The dynamic window approach to collision avoidance. *IEEE Robot. Autom. Mag.* **1997**, *4*, 23–33.
36. Richter, C.; Bry, A.; Roy, N. Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments. In *Proceedings of the International Symposium on Robotics Research (ISRR)*; Springer: Cham, Switzerland, 2016; pp. 1–16.
37. Kalman, R.E. A new approach to linear filtering and prediction problems. *J. Basic Eng.* **1960**, *82*, 35–45.
38. de Boor, C. *A Practical Guide to Splines*; Applied Mathematical Sciences 27; Springer: New York, NY, USA, 1978.
39. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J. et al. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nat. Methods* **2020**, *17*, 261–272.
40. Wang, J.; Zhao, Z.; Qu, J.; Chen, X. APPA-3D: An autonomous 3D path planning algorithm for UAVs in unknown complex environments. *Sci. Rep.* **2024**, *14*, 1231.
41. Huang, X.; Liu, Y.; Huang, L.; Stikbakke, S.; Onstein, E. BIM-supported drone path planning for building exterior surface inspection. *Comput. Ind.* **2023**, *153*, 104019.
42. AlMahamid, F.; Grolinger, K. Agile DQN: Adaptive deep recurrent attention reinforcement learning for autonomous UAV obstacle avoidance. *Sci. Rep.* **2025**, *15*, 18043.
43. Ren, Y.; Zhu, F.; Lu, G.; Cai, Y.; Yin, L.; Kong, F.; Lin, J.; Chen, N.; Zhang, F. Safety-assured high-speed navigation for MAVs. *Sci. Robot.* **2025**, *10*, eado6187.
44. Wang, M.; Wang, Q.; Wang, Z.; Gao, Y.; Wang, J.; Cui, C.; Li, Y.; Ding, Z.; Wang, K.; Xu, C.; et al. Unlocking aerobatic potential of quadcopters: Autonomous freestyle flight generation and execution. *Sci. Robot.* **2025**, *10*, eadp9905.
45. Mohiuddin, M.B.; Boiko, I.; Tran, V.P.; Garratt, M.; Abdallah, A.; Zweiri, Y. Reinforcement learning for end-to-end UAV slung-load navigation and obstacle avoidance. *Sci. Rep.* **2025**, *15*, 34621.
46. Yin, S.; Xiang, Z. Real-time distributed decision-making for simultaneous target assignment and path planning in multiple unmanned surface vehicles. *Expert Syst. Appl.* **2025**, *279*, 127457.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.