D-Nav: End-to-End Dynamic UAV Navigation with Dual-Resolution Motion Awareness

Liang Qin[1,2], Min Wang[3*], Xingyu Lu[1,2], Wengang Zhou[1,2], and Houqiang Li[1,2]

[1]MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition, China
[2]University of Science and Technology of China, China
[3]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, China
{qinliang6218,luxingyu010}@mail.ustc.edu.cn; wangmin@iai.ustc.edu.cn; {zhwg,lihq}@ustc.edu.cn;

***Abstract*—Autonomous navigation in dense, dynamic clutter remains a fundamental challenge for Unmanned Aerial Vehicles (UAVs) due to the heterogeneous obstacle scales and complex motion patterns. Existing methods often rely on fragile explicit tracking or noise-sensitive implicit flow estimation, both of which struggle with irregular geometries and real-time constraints. We propose D-Nav, a novel end-to-end reinforcement learning framework that directly maps raw LiDAR observations to control actions through a saliency-driven dual-resolution spatio-temporal representation. At the global level, D-Nav constructs a spatio-temporal spherical depth representation that encodes scene structure and motion trends directly from sequential LiDAR measurements. Building on this global context, a saliency-based refinement mechanism is designed to identify dynamically critical regions and extract fine-grained geometric and motion cues at the local level. This formulation enables the policy to reason about both large-scale dynamic context and small, irregular, fast-moving obstacles in complex dynamic environments. In addition, D-Nav introduces a new mission-aware waypoint-goal condition mechanism that explicitly guides the policy to balance collision avoidance with the need to traverse task-critical regions. Extensive simulations demonstrate a significant improvement in navigation success, increasing from 0.37 to 0.56 in complex dynamic environments. Real-world experiments further validate the robustness and real-time capability of the proposed system across diverse dynamic scenarios. The code is available at https://github.com/qinliangql/D-NAV.git.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) are increasingly essential across a wide range of industries, from search and rescue and industrial inspection to urban logistics [6, 17, 34, 11, 8, 9, 22, 35, 12, 1, 21, 20]. However, achieving autonomous navigation in dense, dynamic environments remains a fundamental challenge in robotics. Safe flight in such scenarios requires a system that simultaneously analyzes the various geometries and captures the motion trends of obstacles. The primary difficulty stems from the vast diversity of potential obstacles, ranging from large, moving barriers to thin, irregular structures such as wires or branches. Robust navigation demands a system that generalizes across these varied geometries while maintaining the temporal sensitivity necessary for high-speed maneuvers.

[1]* Corresponding author.

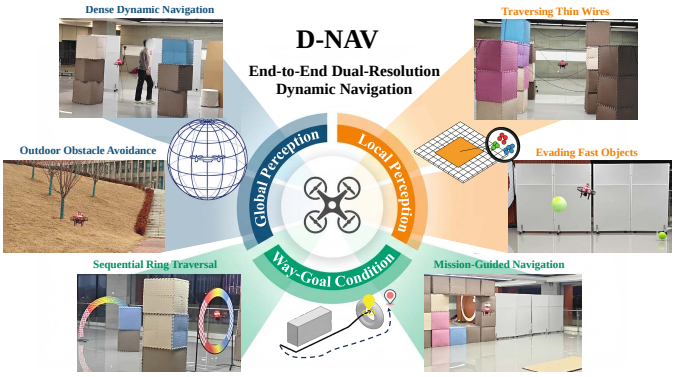


Fig. 1. The D-Nav framework for robust autonomous navigation in dense, dynamic environments. We decompose the navigation challenge into three synergistic dimensions: (1) Global Perception establishes a wide-field spatio-temporal view to capture overall scene structure and motion tendencies, facilitating *Dense Dynamic Navigation* and *Outdoor Obstacle Avoidance*; (2) Local Perception employs saliency-driven refinement to extract fine-grained geometric and motion cues, enabling the system to resolve micro-scale and agile threats like *Traversing Thin Wires* and *Evading Fast Objects*; and (3) Way-Goal Condition modulates decision-making to balance reactive safety with mission-critical progress, supporting purposeful maneuvers such as *Sequential Ring Traversal* and *Mission-Guided Navigation*.

Existing navigation methods struggle to meet these requirements. Some approaches [13, 14, 24, 26, 32, 25] rely on explicit object detection and tracking to model dynamic entities, while such strong geometric or semantic priors limit their versatility. Consequently, these methods become fragile when facing irregular obstacles—like rotating structures or continuously moving surfaces—that are difficult to cluster. Conversely, implicit motion estimation [4, 5, 2, 10, 28] techniques, such as those employing optical flow, accommodate more motion patterns but necessitate low-resolution processing to meet real-time constraints. This low resolution introduces significant noise, making it difficult to detect small-scale obstacles or resolve precise motion in tight, high-speed situations.

To address these challenges, we introduce D-Nav, a novel end-to-end reinforcement learning–based navigation framework for autonomous flight in dense dynamic environments using a single LiDAR sensor. The framework develops a saliency-driven dual-resolution spatio-temporal perception module and a mission-aware condition mechanism, as illus-

trated in Fig. 1, to address multi-scale environmental dynamics and mission-driven navigation demands. The dual-resolution module provides a coherent global view of scene structure and motion trends, and adaptively preserves local geometric details where salient dynamic interactions occur. The way-goal condition mechanism steers policy execution toward task-relevant regions, thereby reconciling immediate collision avoidance with mission objectives.

D-Nav first constructs a global spatio-temporal representation from sequential LiDAR frames to capture large-scale structure and dominant motion tendencies. Concretely, raw lidar point clouds are projected into a spherical depth view and stacked over time to preserve wide angular coverage and temporal continuity. A lightweight encoder processes this stacked volume to produce spatio-temporal features that highlight scene-level dynamics such as bulk motion, approach vectors, and broadly occluding structures. These global features provide the policy with coherent contextual information that supports anticipatory navigation decisions.

While global context captures coarse dynamics, it is insufficient for resolving fine geometric details that directly affect collision safety. D-Nav therefore designs a saliency-driven refinement module to identify regions that exhibit strong temporal saliency. Such regions are characterized by rapid changes in depth or motion patterns and indicate potential local interactions. Within such regions, the framework emphasizes fine-grained geometric and motion cues derived from raw LiDAR point clouds. Local features are extracted from the identified regions using a shared multi-layer perceptron (Shared MLP), and temporal variation features are obtained through a differential MLP (Diff MLP), enabling precise observation of local geometry and motion. This local branch enables the policy to account for small, irregular, and fast-changing obstacles that are overlooked at the global scale.

Beyond perception, effective navigation in dynamic environments requires integrating both reactive safety behaviors and mission objectives within a unified decision-making process. D-Nav addresses this requirement by incorporating a mission-aware Way-Goal Condition into the reinforcement policy, which computes progress toward intermediate waypoints to adaptively modulate the state encoding and explicitly guide task execution. All perception features and way-goal state features are fused and provided to a deep reinforcement learning policy, which maps the combined observations directly to control actions. The policy is trained end-to-end through interaction with dynamic environments, optimizing for both safe navigation and task completion.

Extensive experiments conducted in large-scale simulated environments populated with hundreds of heterogeneous static and dynamic obstacles demonstrate a substantial performance improvement. In the most challenging dynamic scenarios, the proposed method improves the state-of-the-art success rate from 0.37 to 0.56. Real-world deployments further validate the robustness and real-time efficiency of the system, showing that the UAV reliably handles sudden dynamic disturbances across diverse environments and task configurations.

II. Related Work

A. Explicit Object-aware Motion Modeling

Traditional strategies for navigating dynamic environments typically partition the problem into discrete detection, tracking, and planning phases. Early reactive frameworks employ artificial potential fields generated by moving events [3] or utilize Model Predictive Control (MPC) for real-time trajectory refinement [31]. While computationally efficient, these local optimization techniques often prioritize immediate safety at the expense of global task efficiency, leading to suboptimal behaviors in high-speed scenarios. More sophisticated systems attempt to bridge this gap by integrating geometric clustering with static occupancy mapping to facilitate long-term prediction [13, 14, 24]. However, these sequential pipelines are prone to error propagation; any failure in the tracking module directly leads to unreliable planning.

Recent research attempts to replace these rigid pipelines with learning-based object-aware strategies, where detection results are encoded into compact state vectors or spatial images [26, 32]. Although these methods enhance adaptability, they inherit the fundamental vulnerabilities of their detection and tracking front-ends. Specifically, they remain fragile when encountering unconventional geometries or non-semantic obstacles that do not conform to predefined categories. Furthermore, the reliance on explicit object instances causes computational complexity to scale poorly as the number of agents increases. In contrast, our approach uses a detection-free paradigm. By treating the environment as a unified spatio-temporal structure, our system remains robust across diverse geometries without relying on fragile tracking modules.

B. Implicit Motion Perception

Directly mapping sensory observations to control actions allows for high-frequency inference and improved robustness to unmodeled disturbances. End-to-end controllers utilize single-frame LiDAR scans [18, 29] or low-resolution depth maps [33], yet these static representations fail to generalize to dynamic settings where obstacle velocity constitutes a primary threat. To capture temporal dependencies, subsequent works employ the integration of past scan vectors into temporal images [5]. However, without explicit supervision, these models often struggle to isolate meaningful motion patterns from raw data, leading to poor generalization in new dynamic scenes.

More explicit motion awareness is explored through specialized feature extraction, such as point cloud alignment [2] or recurrent architectures including Long Short-Term Memory (LSTM) networks [7]. While these methods improve safety, the specialized modules often introduce training instabilities when co-optimized within a reinforcement learning framework. State-of-the-art architectures attempt to mitigate this by incorporating flow-aided representations, aligning geometric history with pixel-level motion semantics [28]. Nevertheless, these flow-based paradigms remain constrained by a critical resolution-efficiency trade-off. To maintain real-time performance, flow estimation is typically performed at low spatial

resolutions, where quantization noise and sampling sparsity obscure micro-scale obstacles and motion cues. Our work addresses this bottleneck by introducing a dual-resolution architecture. Instead of relying on noisy explicit flow, we combine a macro-scale spatio-temporal view with a saliency-driven refinement. This approach preserves fine details where they matter most, ensuring safety across different obstacle scales and speeds.

III. Method

A. Problem Formulation

We model the autonomous navigation task as a Partially Observable Markov Decision Process (POMDP) [19]. At each time step t, the agent receives an observation:

$$\mathbf{O}_t = \{\mathbf{PC}_t, \mathbf{X}_t, c_t\}. \tag{1}$$

Here, $\mathbf{PC}_t$ represents the raw LiDAR point cloud, and $\mathbf{X}_t = [\mathbf{p}_t, \mathbf{v}_t, \mathbf{a}_t]^\top \in \mathbb{R}^9$ encapsulates the UAV's position, velocity, and acceleration in the local inertial frame. The goal is denoted by $\mathbf{g}_t \in \mathbb{R}^3$, while $c_t \in \{0, 1\}$ indicates the traversal status of intermediate waypoints. To ensure representational consistency and invariance to global rotations, the UAV's yaw is strategically aligned with the direction of $\mathbf{g}_t - \mathbf{p}_0$ at each step. The policy $\pi(\mathbf{A}_{t+1}|\mathbf{O}_t)$ maps these observations to a normalized action vector $\mathbf{A}_t = [\hat{\mathbf{v}}_{t+1}]^\top$. The primary objective is to optimize π to maximize the expected cumulative return for safe and goal-directed flight through dynamic clutter.

B. System Framework

The overall architecture of D-Nav is illustrated in Fig. 2. The system is designed as an end-to-end reinforcement learning framework that maps raw LiDAR point clouds and the robot's kinematic state directly to agile control commands. To address the challenges posed by heterogeneous obstacle scales and rapidly moving dynamics, D-Nav introduces a new dual-resolution spatio-temporal perception architecture that decomposes environmental understanding into complementary global and local reasoning processes.

Specifically, we design a global spatio-temporal perception branch that projects sequential LiDAR observations into a spherical depth representation. Consecutive depth grids $\{D_t, D_{t-1}, D_{t-2}\}$ are stacked along the temporal dimension and encoded by a CNN backbone to extract global spatio-temporal features F_{global}. This branch provides the policy with a comprehensive understanding of the environmental topology and the general motion trends of obstacles. Simultaneously, we introduce a novel saliency-driven local perception branch to explicitly address fine-grained, safety-critical interactions that struggle at the global resolution. This module identifies the *Top-K* temporally salient regions exhibiting significant depth difference, which are indicative of rapid local motion or geometric discontinuities. Within these regions, point subsets from consecutive frames are encoded by a shared MLP to produce time-aligned features $\{F_t, F_{t-1}, F_{t-2}\}$. A differential encoding module (Diff-MLP) then leverages these features to model local motion residuals across frames, yielding F_{diff}. Finally, the motion-aware local feature F_{local} is formed by concatenating F_{diff} with the current-frame feature F_t.

The final stage of the architecture integrates these dual-resolution features with the UAV state. The robot's kinematic data and mission-critical information are processed through a dedicated state encoder, which incorporates our proposed Way-Goal Condition mechanism to manage transitions between intermediate waypoints and the final destination. The resulting feature vector F_{state} is concatenated with F_{global} and F_{local}, and further fed into the Action Net. This network outputs the optimal velocity commands $\hat{\mathbf{v}}_{t+1}$ required for safe, goal-directed flight in complex dynamic environments.

C. Spatio-Temporal Depth Projection for Global Branch

Temporal point cloud alignment. To make the learned policy invariant to the drone's heading and robust to high-frequency motion, we first transform the raw LiDAR stream into a goal-aligned coordinate system.

Let $\mathbf{PC}_t^{\text{ori}} \in \mathbb{R}^{M\times 3}$ be the raw point cloud captured in the UAV's body frame at time t, with the drone's orientation defined by (ϕ, θ, ψ) representing pitch, roll, and yaw. We define a reference yaw angle, ψ_0, aligned with the vector pointing from the UAV to the current goal in the XY-plane. To remove the influence of the UAV's attitude, the point cloud is rotated into a local frame aligned with ψ_0:

$$\mathbf{PC}_t^{\text{proj}} = \mathbf{R}_z(\psi_0 - \psi)\mathbf{R}_y(-\theta)\mathbf{R}_x(-\phi)\mathbf{PC}_t^{\text{ori}}. \tag{2}$$

During this transformation, points beyond a predefined sensing radius r_s (e.g., 5m) are truncated to save computational resources on the immediate collision volume. To maintain a consistent spatio-temporal representation, historical point clouds from previous timestamps ($\mathbf{PC}_{t-1}, \mathbf{PC}_{t-2}$) are transformed into the current local coordinate system. Since all frames share the same goal-aligned orientation ψ_0, this alignment is primarily achieved through a translational shift, while further accounting for the UAV's instantaneous orientation residuals:

$$\mathbf{PC}_{t-k\to t}^{\text{proj}} = \mathbf{R}\Delta\psi^{-1}\left(\mathbf{PC}_{t-k}^{\text{proj}} + \mathbf{R}_0^{-1}(\mathbf{p}_{t-k} - \mathbf{p}_t)\right), \tag{3}$$

where $\mathbf{p}_t$ and $\mathbf{p}_{t-k}$ are the global positions, $\mathbf{R}_0$ is the rotation of the fixed goal-aligned frame, and $\mathbf{R}\Delta\psi$ compensates for small heading deviations from ψ_0 at each timestamp.

Spherical depth projection. The aligned point clouds are then discretized into a structured spherical depth grid. For a point (x, y, z), its spherical coordinates (d, ϕ_h, θ_v) are calculated as:

$$d = \sqrt{x^2 + y^2 + z^2}, \phi_h = \text{atan2}(y, x), \theta_v = \text{asin}(z/d). \tag{4}$$

Points outside the reliable vertical field-of-view $[\theta_{\min}, \theta_{\max}]$ are filtered, where d denotes the Euclidean distance from the point to the UAV body center. Each point is then mapped to a specific grid cell (u, v) based on angular resolutions $\Delta\phi$ and $\Delta\theta$:

$$u = \lfloor \phi_h // \Delta\phi \rfloor, \quad v = \lfloor \theta_v // \Delta\theta \rfloor. \tag{5}$$

The depth value for each grid cell $D_{u,v}$ is defined as the minimum radial distance d_{min} among all points projected into that cell. If no point falls into a given cell, the depth value is set

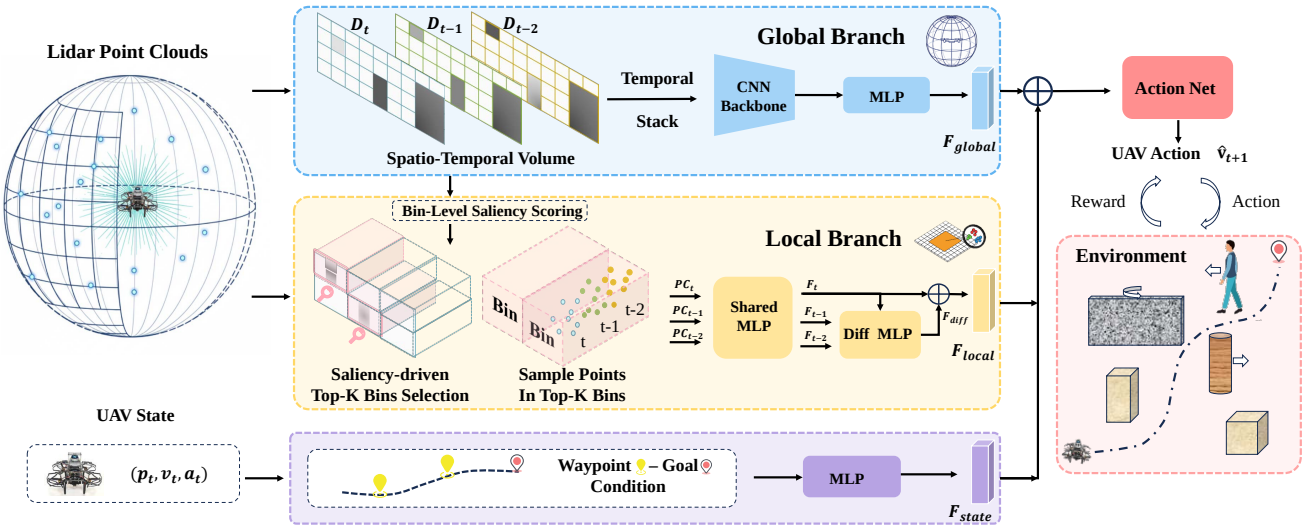


Fig. 2. Overview of the D-Nav framework. The architecture processes sequential LiDAR point clouds through two parallel streams: (1) The **Global Branch** aggregates temporal stacks of spherical depth projections using a CNN backbone to capture scene-level structure and motion tendencies; (2) The **Local Branch** performs saliency-driven refinement by selecting *Top-K* temporal bins with high depth variance and extracting fine-grained geometric features from sampled points. These perceptual features (F_{global}, F_{local}) are fused with the **UAV State** and **Way-Goal Condition** to feed the Action Net. The framework is trained via reinforcement learning, where the policy generates optimal control commands $\hat{\mathbf{v}}_{t+1}$ by resolving complex motion gradients from heterogeneous obstacles to ensure safety-critical and goal-oriented flight.

to a max sensing range r_s. This yields a temporal sequence of depth maps $\{D_t, D_{t-1}, D_{t-2}\}$ that captures the macro-scale environmental topology.

Spatio-temporal global encoding. To synthesize these discrete representations into a motion-aware context, the temporal sequence of depth maps $\{D_t, D_{t-1}, D_{t-2}\}$ is concatenated along the channel dimension to form a spatio-temporal volume $\mathbf{V} \in \mathbb{R}^{3\times H\times W}$. This volume is subsequently processed by a CNN-based backbone. The network employs a series of convolutional layers with progressively increasing receptive fields, enabling the extraction of both static environmental topology and macro-scale motion gradients across consecutive frames. The resulting high-dimensional feature maps are flattened and passed through an MLP-based integration layer, which compresses the spatial information into a compact global feature vector F_{global}. This global representation leverages the temporal consistency of the depth sequence. It provides a clear understanding of the scene geometry and the motion of large obstacles.

D. Saliency-Driven Refinement For Local Branch

Temporal saliency estimation. While depth grids effectively represent global geometry, the discretization process inevitably obscures micro-scale obstacles. We address this by introducing a saliency-driven mechanism to selectively refine regions containing critical motion or geometric details. As visualized in Fig. 3, the refinement process begins by quantifying the temporal saliency $S_{u,v}$ of each grid cell (u, v) based on the cumulative depth difference across the temporal window:

$$S_{u,v} = |D_{u,v,t} - D_{u,v,t-1}| + |D_{u,v,t-1} - D_{u,v,t-2}|. \quad (6)$$

Bin-level saliency aggregation. To bridge the gap between fine-grained grid cells and coarser processing units, we define

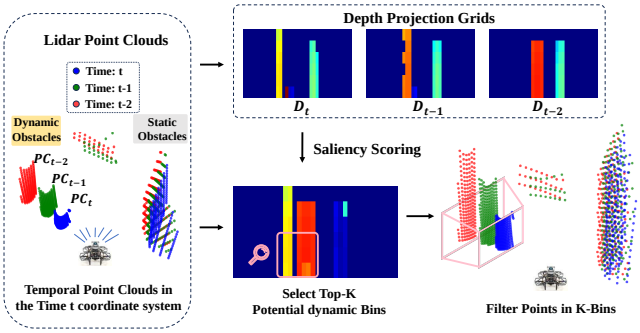


Fig. 3. Visualization of the saliency-driven refinement pipeline. Temporal LiDAR point clouds are projected into multi-frame depth grids, where deep blue regions denote the maximum sensing range. By computing the cumulative temporal depth variation ("Diff Depth"), the system identifies high-saliency bins (pink rectangle) likely containing dynamic or critical geometric features. Finally, raw point subsets are retrieved specifically from these top-K bins (*i.e.*, the points in the pink volume), facilitating high-fidelity local feature extraction while filtering out static background noise.

the saliency of each bin $B_{m,n}$ as the maximum variation among its constituent cells:

$$S_{m,n} = \max\{S_{u,v} \mid (u, v) \in B_{m,n}\}, \quad (7)$$

where the indices (u, v) are mapped to the bin $B_{m,n}$ via the downsampling relations $m = \lfloor u/\Delta u_{\text{bin}} \rfloor$ and $n = \lfloor v/\Delta v_{\text{bin}} \rfloor$(e.g. $m = 6, n = 2$). Based on this metric, we select the top-K bins with the highest saliency scores for refinement.

By leveraging the bin indices assigned during the initial projection phase, the system directly retrieves the raw point subsets $\{\mathbf{PC}_{B,t}, \mathbf{PC}_{B,t-1}, \mathbf{PC}_{B,t-2}\}$ associated with the selected bins. This indexing strategy avoids expensive clustering. It takes advantage of the spherical distribution of LiDAR data, where objects are naturally separated by their angles relative to the sensor. This "zero-cost" segmentation ensures that high-fidelity geometric features are preserved specifically where

environmental dynamics are most prominent.

Local sampling and encoding. Building on these efficiently retrieved point subsets, we further design a lightweight local sampling and encoding pipeline to minimize processing latency. Instead of adopting time-cost Farthest Point Sampling (FPS) [16], we employ a centroid-based sampling strategy. For each selected bin, we compute the spatial centroid $\mathbf{c}_B$ of the points. We then select the N_{sample} (e.g., 16) points nearest to $\mathbf{c}_B$ to represent the local geometry of each salient bin.

For each bin, the sampled point sets from consecutive frames are independently processed by a shared MLP, yielding per-frame bin-level geometric features $\{F_t, F_{t-1}, F_{t-2}\}$. To capture local temporal variation within the same bin, we further introduce a differential encoding module (Diff MLP), which takes the multi-frame features as input and learns motion residual representations across frames, producing F_{diff}. Finally, the motion-aware local feature of each bin is formed by concatenating F_{diff} with the current-frame feature F_t. Features from multiple selected bins are then concatenated to construct the overall local representation F_{local}.

E. Way-Goal Condition

Autonomous navigation requires balancing reactive safety and mission progress across different task stages. In our setting, the mission is specified as a sequence of navigation targets, including intermediate waypoints for traversal-critical regions and a final goal for task completion. These targets are provided by a higher-level planner or the experimental setup and are updated online as the UAV progresses.

To encode task-stage information in a unified manner, we introduce the *Way-Goal Condition*, which couples the current navigation stage with the relative kinematic relationship between the UAV and its active target to guide policy behavior. We use a binary indicator $c_t \in \{0, 1\}$ to denote the type of the active target, where $c_t = 0$ indicates an intermediate waypoint and $c_t = 1$ the final goal. The indicator reflects the current task stage without enforcing explicit traversal rules. The active target is updated online when a proximity threshold (e.g., 0.5m) is reached, and c_t switches only upon transitioning to the final goal.

Task progress is primarily conveyed through continuous relative state encoding. Given the UAV position $\mathbf{p}_t$ and the active target $\mathbf{g}_t$, we compute the relative displacement $\Delta\mathbf{p}_t = \mathbf{g}_t - \mathbf{p}_t$ and rotate it into a goal-aligned reference frame ψ_0:

$$\tilde{\mathbf{p}}_t = \mathbf{R}_z(-\psi_0)\Delta\mathbf{p}_t. \tag{8}$$

The rotated vector is decomposed into a normalized direction $\hat{\mathbf{e}}_t$ and planar and vertical distance components (d_{xy}, d_z), preserving vertical structure in sparse environments. The UAV velocity $\mathbf{v}_t$ and acceleration $\mathbf{a}_t$ are similarly transformed into the same frame, yielding $\mathbf{v}'_t$ and $\mathbf{a}'_t$. The maximum allowable speed $\mathbf{v}_{max}$ is appended to indicate action scaling. All mission-related signals are combined into the state vector $\mathbf{s}_t = [\hat{\mathbf{e}}_t, d_{xy}, d_z, \mathbf{v}'_t, \mathbf{a}'_t, \mathbf{v}_{max}, c_t]$, which is processed by an MLP-based state encoder to produce F_{state}. Through this formulation, the Way-Goal Condition serves as a conditioning mechanism on the state representation, enabling the policy to adapt its implicit risk preference and traversal behavior according to the current task stage and spatial context, rather than relying on explicit rule-based switching.

F. Reinforcement Learning

The encoded state F_{state} is concatenated with perception features F_{global} and F_{local} to form the complete latent observation F_{total}. The Action Net, a lightweight MLP, maps F_{total} to a normalized control vector $\mathbf{A}_{t+1} = [\hat{\mathbf{v}}_{t+1}]^\top \in [-1, 1]^3$, which is scaled by $\mathbf{v}_{max}$ to obtain physical velocity commands. The policy is trained end-to-end under an RL objective that balances collision avoidance and mission progress.

Two-stage training. The proposed navigation policy is trained using the Proximal Policy Optimization (PPO) [23] algorithm within the NVIDIA Isaac Sim [15, 27] environment, which provides high-fidelity physical simulations and efficient parallelization. To ensure the UAV develops robust obstacle-avoidance and mission-specific task capabilities, we adopt a two-stage training strategy. In the first, foundational safety-priority stage, the policy is trained using the base reward $\mathcal{R}_{base}$ to enable reliable collision-free navigation and direct goal-reaching, which already suffices for most generic navigation scenarios. In the second, mission-oriented fine-tuning stage, the same policy is further optimized with an additional functional reward $\mathcal{R}_{func}$, where the Way-Goal Condition mechanism is activated to handle mission constraints such as waypoint traversal.

1) Foundational safety-priority stage. The foundational reward $\mathcal{R}_{base}$ is designed to balance progress, safety, and control smoothness. The progress reward $\mathcal{R}_v$ incentivizes the UAV to maximize its velocity projection along the normalized relative vector to the target:

$$\mathcal{R}_v = \begin{cases} 0.01 \cdot (\mathbf{v}'_t \cdot \hat{\mathbf{e}}_t), & |\mathbf{p}_g - \mathbf{p}_t| \leq 3 \\ \mathbf{v}'_t \cdot \hat{\mathbf{e}}_t, & \text{otherwise,} \end{cases} \tag{9}$$

where $\mathbf{v}'_t$ is the current velocity and $\hat{\mathbf{e}}_t$ is the normalized direction toward the target. The reduced scaling factor applied within a $3m$ radius of the goal attenuates aggressive forward motion, promoting cautious approach behavior while preserving sufficient control authority for collision avoidance during the final phase of navigation.

To encourage the maintenance of a safe operational envelope, we define a safety reward $\mathcal{R}_s$ as the mean value of the projected spherical depth grid, promoting trajectories through more open volumes:

$$\mathcal{R}_s = r_s - \sum d_i / N_D, d_i \in D_t, \tag{10}$$

where N_D denotes the number of grids in D_t, and r_s is the maximum sensing range.

To prevent collisions and discourage proximity to hazards, an exponential penalty $\mathcal{R}_c$ is applied when the UAV enters a safety buffer of $0.5\,\text{m}$:

$$\mathcal{R}_c = -\frac{e^{0.5-\min(D_t)} - 1}{e^{0.2} - 1} \times 3, \text{if } \min(D_t) \leq 0.5m. \tag{11}$$

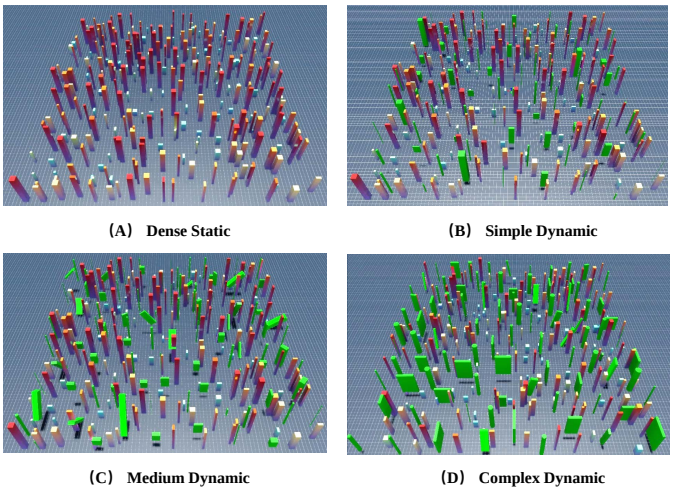


Fig. 4. Four distinct simulation scenarios categorized by dynamic complexity: (A) Dense Static; (B) Simple Dynamic; (C) Medium Dynamic; and (D) Complex Dynamic. Dynamic obstacles are highlighted in green to differentiate them from static entities.

This formulation ensures that the penalty grows sharply as the distance approaches the collision threshold. Additionally, a smoothness penalty $\mathcal{R}_m$ is applied to minimize abrupt changes in control inputs:

$$\mathcal{R}_m = -\lambda_v(|\mathbf{v}_t - \mathbf{v}_{t-1}| + |\mathbf{v}_{t-1} - \mathbf{v}_{t-2}|) - \lambda_a|\mathbf{a}_t - \mathbf{a}_{t-1}|, \quad (12)$$

where λ_v, λ_a are weighting factors. Including the out boundary penalty $\mathcal{R}_o = -5$ and the goal reward $\mathcal{R}_g = 3$, the total reward for the foundational stage is:

$$\mathcal{R}_{base} = 1 + \mathcal{R}_v + \mathcal{R}_s + \mathcal{R}_g + \mathcal{R}_m + \mathcal{R}_c + \mathcal{R}_o. \quad (13)$$

2) Mission-oriented stage. In the second stage, we extend the policy's capability to handle structured obstacles, such as rings, arches, and gates. A waypoint is assigned to the center of one such structure, and the agent must traverse this waypoint before the final goal reward becomes active. To explicitly reward successful traversal, we define two checkpoints located $0.3\,\mathrm{m}$ before and after the structure's center, perpendicular to its plane. A crossing reward $\mathcal{R}_{cross}$ is awarded incrementally: a value of 3 is granted when the UAV reaches the first checkpoint, and a value of 10 is awarded upon passing the second, ensuring that the agent learns to pass through the structure rather than merely approaching its vicinity. The combined reward for this functional expansion stage is

$$\mathcal{R}_{func} = \mathcal{R}_{base} + \mathcal{R}_{cross}, \quad (14)$$

where $\mathcal{R}_g$ in $\mathcal{R}_{base}$ is only attainable following the successful completion of the waypoint crossing.

IV. Experiments

A. Experimental Setup

The simulation experiments are conducted in NVIDIA Isaac Sim [15, 27], which provides realistic UAV dynamics and sensor modeling. As shown in Fig. 4, we construct a benchmark with four scenarios of increasing difficulty: (i) *Dense Static*, featuring 300 randomly distributed static obstacles; (ii) *Simple Dynamic*, with 200 static and 80 linear-moving agents; (iii) *Medium Dynamic*, containing 200 static and 100 obstacles exhibiting linear or non-linear motion (e.g., rotations); (iv) *Complex Dynamic*, which further intensifies the medium scenario with expanded obstacle size and increased motion velocities. The UAV is initialized at one boundary, with the target placed at a different boundary.

The D-NAV policy is trained only in the *Medium Dynamic* scenario using a two-stage procedure: 50k episodes of safety-priority training followed by 20k episodes of mission-oriented fine-tuning. Training uses a batch size of 256, a learning rate of 5×10^{-4}, and an episode horizon of 1500 steps at a control frequency of 20 Hz. Generalization is evaluated across all scenarios with multiple random seeds. Training and inference are performed on an NVIDIA A100 GPU and an Intel 8575C CPU. All experiments are conducted in a $40 \times 40 \times 4.5$ m^3 arena, where success requires reaching within 0.5m of the target while maintaining a minimum obstacle distance $d_{min} \geq 0.3$m. For fair comparison, the maximum flight speed is limited to 3m/s for all methods. Real-world experiments are carried out on a custom quadrotor equipped with a Jetson NX and a Mid360 LiDAR, with position estimation provided by Fast-LIO [30].

B. Comparative Benchmarking in Simulation

We compare D-Nav with state-of-the-art learning-based and traditional rule-based baselines. These include NavRL [32], which utilizes explicit tracking and prediction; FHD [5]: an end-to-end depth-field approach that processes temporal depth sequences; FAF [28]: a flow-aided method that computes optical flow from lidar depth maps; FOP [29]: a direct point-cloud RL policy. Traditional methods such as FAPP [14] (clustering-based motion awareness) and EGO-Planner [34] (static mapping) are also included. All learning-based baselines are fine-tuned in the same Isaac Sim environment for fair comparison, while D-Nav is evaluated using the base policy trained in the safety-priority stage.

As shown in Table I, D-Nav significantly outperforms both learning-based and rule-based methods in dynamic environments. In the *Complex Dynamic* scenario, D-Nav achieves a **56.2%** success rate, corresponding to a **53.1% relative improvement** over the strongest baseline (FAF, 36.7%). Traditional planners perform competitively in static scenes but degrade sharply in dynamic settings due to the lack of motion awareness. While FHD and FOP maintain higher average speeds, their collision rates exceed 80% in complex scenes, indicating poor safety. D-Nav consistently achieves higher success rates with low collision rates in both Medium and Simple Dynamic scenarios, demonstrating robust generalization across different dynamic scenes. In Dense Static, NavRL attains a success rate of 0.925 with a velocity of 0.85 m/s. In contrast, our D-Nav speedups the velocity by 180% while achieving a competitive success rate of 0.887, realizing a better balance between efficiency and safety.

Fig. 5 qualitatively compares trajectories in the *Medium Dynamic* scenario with identical start and goal positions.

Scenario	Method	SR ↑	CR ↓	Vel ↑
Complex Dynamic	Ego [34]	$0.000_{\pm.00}$	$1.000_{\pm.00}$	1.73
	FAPP [14]	$0.110_{\pm.08}$	$0.863_{\pm.08}$	1.67
	NavRL [32]	$0.257_{\pm.07}$	$0.726_{\pm.07}$	0.81
	FOP [29]	$0.113_{\pm.05}$	$0.851_{\pm.05}$	2.86
	FHD [5]	$0.187_{\pm.08}$	$0.812_{\pm.08}$	**2.89**
	FAF [28]	$0.367_{\pm.06}$	$0.624_{\pm.07}$	2.05
	D-NAV(Ours)	**0.562**$_{\pm.06}$	**0.406**$_{\pm.06}$	2.11
Medium Dynamic	Ego [34]	$0.000_{\pm.00}$	$1.000_{\pm.00}$	1.79
	FAPP [14]	$0.196_{\pm.08}$	$0.782_{\pm.08}$	1.68
	NavRL [32]	$0.407_{\pm.07}$	$0.582_{\pm.07}$	0.84
	FOP [29]	$0.119_{\pm.06}$	$0.851_{\pm.06}$	2.88
	FHD [5]	$0.257_{\pm.07}$	$0.726_{\pm.07}$	**2.93**
	FAF [28]	$0.429_{\pm.05}$	$0.515_{\pm.05}$	2.14
	D-NAV(Ours)	**0.639**$_{\pm.05}$	**0.359**$_{\pm.05}$	2.33
Simple Dynamic	Ego [34]	$0.103_{\pm.08}$	$0.885_{\pm.08}$	2.15
	FAPP [14]	$0.463_{\pm.05}$	$0.501_{\pm.05}$	2.17
	NavRL [32]	$0.496_{\pm.06}$	$0.448_{\pm.06}$	0.94
	FOP [29]	$0.235_{\pm.04}$	$0.726_{\pm.05}$	2.97
	FHD [5]	$0.364_{\pm.05}$	$0.632_{\pm.05}$	**2.99**
	FAF [28]	$0.608_{\pm.05}$	$0.382_{\pm.05}$	2.69
	D-NAV(Ours)	**0.710**$_{\pm.04}$	**0.289**$_{\pm.04}$	2.68
Dense Static	Ego [34]	$0.809_{\pm.03}$	$0.136_{\pm.03}$	2.13
	FAPP [14]	$0.794_{\pm.03}$	$0.158_{\pm.03}$	2.24
	NavRL [32]	**0.925**$_{\pm.05}$	**0.074**$_{\pm.05}$	0.85
	FOP [29]	$0.871_{\pm.05}$	$0.092_{\pm.05}$	2.96
	FHD [5]	$0.598_{\pm.05}$	$0.373_{\pm.05}$	**2.98**
	FAF [28]	$0.742_{\pm.04}$	$0.246_{\pm.04}$	2.31
	D-NAV(Ours)	$0.887_{\pm.04}$	$0.110_{\pm.04}$	2.40

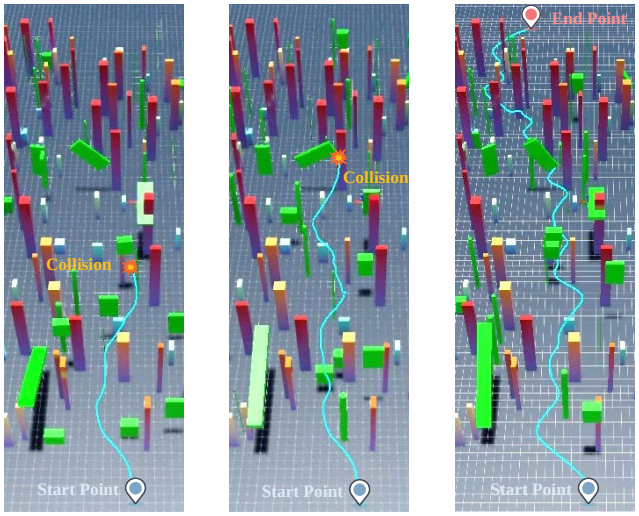
TABLE I

QUANTITATIVE PERFORMANCE COMPARISON AGAINST STATE-OF-THE-ART BASELINES ACROSS DIVERSE SIMULATED SCENARIOS. RESULTS ARE REPORTED AS MEAN$_{\pm\text{STD}}$. "SR" DENOTES SUCCESS RATE, "CR" DENOTES COLLISION RATE, AND "VEL" REPRESENTS THE AVERAGE VELOCITY (M/S). BOLD AND SHADED VALUES INDICATE THE **BEST** AND SECOND-BEST RESULTS, RESPECTIVELY.

NavRL collides with a thin moving obstacle, while FAF fails under rotating motion, reflecting the limits of low-resolution flow estimation. D-Nav successfully avoids both hazards and reaches the goal, demonstrating its ability to reason about global scene evolution and localized high-risk dynamics.

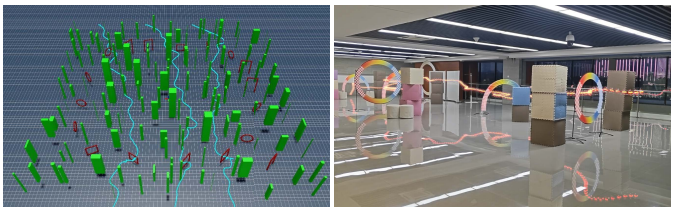
C. Ablation Study

1) Way-Goal Condition: To evaluate the trade-off between reactive safety and mission objectives, we conduct a "ring-crossing" task in simulation, requiring the UAV to navigate narrow, high-risk apertures (Fig. 6A). Table II reveals that foundational RL policies, including NavRL, FAF, and our base model, consistently fail the mission. This is primarily attributed to their safety-first optimization objectives, which prioritize collision avoidance and encourage the agent to detour around structural constraints rather than traversing them.



(A) NavRL (B) FAF (C) Ours

Fig. 5. Qualitative trajectory comparison in the Medium Dynamic scenario under identical environment initialization, start, and goal configurations. (A) NavRL [32] collides with a thin moving obstacle due to limited sensitivity to small-scale dynamics. (B) FAF [28] fails when encountering a rotating obstacle, illustrating the fragility of low-resolution flow estimation under non-linear motion. (C) D-Nav (Ours) successfully avoids both obstacles and reaches the goal, benefiting from saliency-driven local refinement and global



(A) Cross in Simulator (B) Cross in RealWorld

Fig. 6. Qualitative evaluation of mission-oriented navigation. (A) The ring-crossing simulation environment, featuring static structural obstacles (red) and potential dynamic disturbances (green). (B) Successful real-world deployment where the policy generalizes to physical constraints; the light trails illustrate the UAV's stable trajectory as it purposefully traverses the rings despite environmental noise and dynamic disturbances.

For the ablation variants, introducing the Way-Goal Condition during the crossing-oriented stage fine-tuning, with the reward unchanged, yields a moderate increase in cross rate with only a slight rise in collisions, indicating more structured task guidance. In contrast, applying the crossing reward $\mathcal{R}_{cross}$ without the Way-Goal Condition significantly increases the cross rate but also raises collision risk, reflecting overly aggressive traversal. When both are applied in the full model, the Way-Goal Condition effectively moderates the behavior induced by $\mathcal{R}_{cross}$, achieving the highest success rate with fewer collisions than the reward-only variant. This shows that structured goal conditioning and crossing-oriented fine-tuning work synergistically to enable stable task-level navigation.

2) Dual Resolution Perception: We analyze how the perception architecture affects navigation in dynamic conditions via ablation in the Medium Dynamic scenario (Table III). Relying solely on Global Perception provides a baseline success rate of 54.0%, confirming macro-topology as the navigation backbone. Integrating Local Refinement, where Top-3 salient

Method	Cross Rate ↑	Collision Rate ↓	Success Rate ↑
NavRL [32]	$0.032_{\pm.03}$	$0.314_{\pm.04}$	$0.010_{\pm.01}$
FAF [28]	$0.091_{\pm.06}$	$0.297_{\pm.06}$	$0.040_{\pm.04}$
Ours (Base)	$0.107_{\pm.05}$	$\mathbf{0.252}_{\pm.05}$	$0.064_{\pm.05}$
Ours (+way-goal)	$0.262_{\pm.05}$	$0.305_{\pm.06}$	$0.158_{\pm.05}$
Ours (+$\mathcal{R}_{cross}$)	$0.430_{\pm.06}$	$0.563_{\pm.06}$	$0.249_{\pm.06}$
Ours (Full)	$\mathbf{0.614}_{\pm.05}$	$0.476_{\pm.05}$	$\mathbf{0.511}_{\pm.05}$

TABLE II
QUANTITATIVE RESULTS OF THE RING-CROSSING TASK. THE TABLE COMPARES BASELINE METHODS WITH OUR METHOD AND ITS ABLATION VARIANTS, HIGHLIGHTING THE EFFECTS OF THE WAY-GOAL CONDITION MECHANISM AND THE CROSSING-ORIENTED TRAINING OBJECTIVE.

Method	Success Rate ↑	Collision Rate ↓
Only Global Perception	$0.540_{\pm.05}$	$0.457_{\pm.05}$
Only Local Perception	$0.273_{\pm.07}$	$0.754_{\pm.07}$
Top-1 Bin Local + Global	$0.552_{\pm.05}$	$0.446_{\pm.05}$
Top-3 Bin Local + Global	$0.639_{\pm.05}$	$0.359_{\pm.05}$
Top-6 Bin Local + Global	$0.628_{\pm.06}$	$0.371_{\pm.06}$
Window Size $\Delta t = 6$	$\mathbf{0.642}_{\pm.07}$	$\mathbf{0.357}_{\pm.07}$
With Noise($\mu_{sense} = 0.3m$)	$0.610_{\pm.07}$	$0.388_{\pm.07}$

TABLE III
ABLATION STUDY OF THE DUAL-RESOLUTION PERCEPTION ARCHITECTURE CONDUCTED IN THE MEDIUM DYNAMIC SCENARIO.

regions are selected for refinement, increases the success rate to 63.9% by resolving high-fidelity motion cues and fine-grained geometries otherwise blurred in coarse representations. System performance is sensitive to refinement density: using only the highest-saliency region (Top-1 bin) yields marginal improvement, while excessive refinement (Top-6 bins) reduces success to 62.8% due to extra representational noise. Expanding the temporal window from 3 to 6 frames produces only minor gains with a higher computational cost. Moderate noise injection into the LiDAR range measurements and UAV position causes slight degradation, indicating that the method is efficient and robust to sensing perturbations.

D. Real-World Flight and Generalization

We evaluate D-Nav in a cluttered real-world scene with dense obstacles and dynamic disturbances over 20 trials (Fig. 7A). D-Nav achieves an 80% success rate (Table IV), significantly outperforming baseline methods. Runtime performance is evaluated on the Jetson NX platform (Table V), where *Data Process* refers to perception-side preprocessing and *Action Generate* corresponds to policy inference. By relying on lightweight depth-grid construction and point sampling,

Method	NavRL [32]	FAF [28]	Ours
Success Rate ↑	0.30	0.40	**0.80**

TABLE IV
SUCCESS RATE COMPARISON IN REAL-WORLD FLIGHT TESTS FEATURING DENSE INDOOR CLUTTER AND DYNAMIC DISTURBANCES.

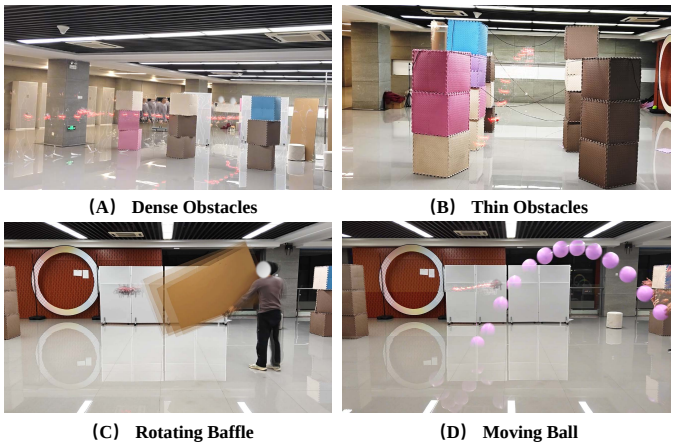


Fig. 7. Robustness of D-NAV across diverse real-world edge cases. The policy maintains navigational success when encountering: (A) dense, unstructured indoor clutter; (B) micro-scale structures such as thin wires and branches; (C) large-scale dynamic obstacles, exemplified by a rotating baffle; and (D) high-velocity projectiles (e.g., a tossed ball) requiring sub-second reactive maneuvers.

Method	Data Process ↓	Action Generate ↓	All Time ↓
NavRL [32]	41.81	23.15	65.90
FAF [28]	28.24	**15.19**	43.58
Ours	**17.63**	19.33	**37.06**

TABLE V
COMPUTATIONAL LATENCY (MS) BENCHMARKED ON THE NVIDIA JETSON NX ONBOARD PLATFORM.

D-Nav achieves the lowest overall latency (37.06 ms), avoiding expensive operations such as object detection or dense flow estimation. Beyond obstacle avoidance, D-Nav also demonstrates strong task-level generalization in real-world deployment. As shown in Fig. 6B, the policy consistently executes goal-directed traversal through narrow structures, confirming that this design remains effective under real-world conditions.

We further analyze robustness across diverse real-world edge cases in Fig. 7. The dual-resolution representation enables reliable perception across a wide range of spatial scales, from dense barriers (Fig. 7A) to micro-scale structures such as thin wires (Fig. 7B) that are challenging for low-resolution depth maps. In addition, the policy demonstrates strong temporal responsiveness when encountering large dynamic obstacles, including rotating baffles (Fig. 7C), and successfully performs sub-second evasive maneuvers against high-velocity projectiles (Fig. 7D). In addition, D-Nav generalizes zero-shot to outdoor 3D terrains, indicating that the proposed representation captures general geometric structure rather than overfitting to simulation. More experiments and videos are provided in the supplementary material.

V. CONCLUSION

This paper presents D-Nav, an end-to-end navigation framework for dense and dynamic environments using a saliency-driven dual-resolution spatio-temporal formulation. By structuring environmental understanding into global and local reasoning stages, the system captures both scene-level structure and fine-grained geometric details. This approach eliminates

reliance on fragile tracking and noise-sensitive flow estimation, enabling robust detection of heterogeneous obstacles, from large moving barriers to micro-scale agile threats. Moreover, the integration of a mission-aware way-goal condition balances reactive safety with goal-directed progress. Extensive evaluations in complex simulations and real-world deployments demonstrate that D-Nav improves navigation success while maintaining stable, real-time performance on resource-constrained platforms.

Despite these advances, the system still struggles with extremely high-speed, small-scale obstacles where sensing latency limits responsiveness. Future work will focus on enhancing temporal responsiveness and extending the framework to handle more aggressive, high-speed flight scenarios.

Acknowledgments

This work is supported by National Natural Science Foundation of China under Contract 62472141, Natural Science Foundation of Anhui Province under Contract 2508085Y040, the Youth Innovation Promotion Association CAS, the GPU cluster built by MCC Lab of USTC and the Supercomputing Center of USTC.

References

[1] Christoph Böhm, Martin Scheiber, and Stephan Weiss. Filter-based online system-parameter estimation for multicopter uavs. In *Robotics: Science and Systems*, 2021.

[2] Jorge de Heuvel, Xiangyu Zeng, Weixian Shi, Tharun Sethuraman, and Maren Bennewitz. Spatiotemporal attention enhances lidar-based robot navigation in dynamic environments. *IEEE Robotics and Automation Letters*, 9 (5):4202–4209, 2024.

[3] Davide Falanga, Kevin Kleber, and Davide Scaramuzza. Dynamic obstacle avoidance for quadrotors with event cameras. *Science Robotics*, 5(40):eaaz9712, 2020.

[4] Tingxiang Fan, Pinxin Long, Wenxi Liu, and Jia Pan. Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios. *The International Journal of Robotics Research*, 39(7): 856–892, 2020.

[5] Xiyu Fan, Minghao Lu, Bowen Xu, and Peng Lu. Flying in highly dynamic environments with end-to-end learning approach. *IEEE Robotics and Automation Letters*, 2025.

[6] Ismail Geles, Leonard Bauersfeld, Angel Romero, Jiaxu Xing, and Davide Scaramuzza. Demonstrating agile flight from pixels without state estimation. *Robotics Science and Systems online Proceedings*, 2024.

[7] David Hoeller, Lorenz Wellhausen, Farbod Farshidian, and Marco Hutter. Learning a state representation and navigation in cluttered and dynamic environments. *IEEE Robotics and Automation Letters*, 6(3):5081–5088, 2021.

[8] Minwoo Kim, Geunsik Bae, Jinwoo Lee, Woojae Shin, Changseung Kim, Myong-Yol Choi, Heejung Shin, and Hyondong Oh. Rapid: Robust and agile planner using inverse reinforcement learning for vision-based drone navigation. *arXiv preprint arXiv:2502.02054*, 2025.

[9] Dana Kulic, Gentiane Venture, Kostas Bekris, and Enrique Coronado. Mpcc++: Model predictive contouring control for time-optimal flight with safety constraints. In *Robotics: Science and Systems Conference*, 2024.

[10] Mihir Kulkarni and Kostas Alexis. Reinforcement learning for collision-free flight exploiting deep collision encoding. In *IEEE International Conference on Robotics and Automation*, pages 15781–15788. IEEE, 2024.

[11] Pratik Kunapuli, Jake Welde, Dinesh Jayaraman, and Vijay Kumar. Leveling the playing field: Carefully comparing classical and learned controllers for quadrotor trajectory tracking. *arXiv preprint arXiv:2506.17832*, 2025.

[12] Antonio Loquercio, Elia Kaufmann, René Ranftl, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Learning high-speed flight in the wild. *Science Robotics*, 6(59):eabg5810, 2021.

[13] Minghao Lu, Han Chen, and Peng Lu. Perception and avoidance of multiple small fast moving objects for quadrotors with only low-cost rgbd camera. *IEEE Robotics and Automation Letters*, 7(4):11657–11664, 2022.

[14] Minghao Lu, Xiyu Fan, Han Chen, and Peng Lu. Fapp: Fast and adaptive perception and planning for uavs in dynamic cluttered environments. *IEEE Transactions on Robotics*, 2024.

[15] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

[16] Carsten Moenning and Neil A Dodgson. Fast marching farthest point sampling. Technical report, University of Cambridge, Computer Laboratory, 2003.

[17] Md Safwan Mondal, Subramanian Ramasamy, Luca Russo, James D Humann, James M Dotterweich, and Pranav Bhounsule. How to coordinate uavs and ugvs for efficient mission planning? optimizing energy-constrained cooperative routing with a drl framework. *Robotics Science and Systems online Proceedings1*, 2025.

[18] Claudia Pérez-D'Arpino, Can Liu, Patrick Goebel, Roberto Martín-Martín, and Silvio Savarese. Robot navigation in constrained pedestrian environments using reinforcement learning. In *IEEE International Conference on Robotics and Automation*, pages 1140–1146. IEEE, 2021.

[19] Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

[20] Liang Qin, Min Wang, Peiwei Li, Wengang Zhou, and Houqiang Li. Active perception meets rule-guided rl: A two-phase approach for precise object navigation in complex environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages

7603–7612, 2025.

[21] Liang Qin, Min Wang, Haodong Wang, Wengang Zhou, and Houqiang Li. Dp-habitat: Bridging the gap between simulation and reality for visual navigation in dynamic pedestrian environments. In *IEEE International Conference on Robotics and Automation*, pages 7562–7568. IEEE, 2025.

[22] Friedrich M Rockenbauer, Simon Jeger, Liberto Beltran, Maximilian Berger, Marvin Harms, Noah Kaufmann, Marc Rauch, Moritz Reinders, Nicholas RJ Lawrance, Thomas Stastny, et al. Dipper: A dynamically transitioning aerial-aquatic unmanned vehicle. In *Robotics: Science and Systems*, volume 2021, pages 12–16, 2021.

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[24] Yingjian Wang, Jialin Ji, Qianhao Wang, Chao Xu, and Fei Gao. Autonomous flights in dynamic environments with onboard vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1966–1973. IEEE, 2021.

[25] Zhepei Wang, Xin Zhou, Chao Xu, and Fei Gao. Geometrically constrained trajectory optimization for multicopters. *IEEE Transactions on Robotics*, 38(5):3259–3278, 2022.

[26] Zhanteng Xie and Philip Dames. Drl-vo: Learning to navigate through crowded dynamic scenes using velocity obstacles. *IEEE Transactions on Robotics*, 39(4):2700–2719, 2023.

[27] Botian Xu, Feng Gao, Chao Yu, Ruize Zhang, Yi Wu, and Yu Wang. Omnidrones: An efficient and flexible platform for reinforcement learning in drone control. *IEEE Robotics and Automation Letters*, 9(3):2838–2844, 2024.

[28] Bowen Xu, Zexuan Yan, Minghao Lu Member, Xiyu Fan, Yi Luo, Youshen Lin, Zhiqiang Chen, Yeke Chen, Qiyuan Qiao, and Peng Lu. Flow-aided flight through dynamic clutters from point to motion. *IEEE Robotics and Automation Letters*, 11(1):218–225, 2025.

[29] Guangtong Xu, Tianyue Wu, Zihan Wang, Qianhao Wang, and Fei Gao. Flying on point clouds with reinforcement learning. *arXiv preprint arXiv:2503.00496*, 2025.

[30] Wei Xu and Fu Zhang. Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter. *IEEE Robotics and Automation Letters*, 6(2):3317–3324, 2021.

[31] Zhefan Xu, Di Deng, Yiping Dong, and Kenji Shimada. Dpmpc-planner: A real-time uav trajectory planning framework for complex static environments with dynamic obstacles. In *International Conference on Robotics and Automation*, pages 250–256. IEEE, 2022.

[32] Zhefan Xu, Xinming Han, Haoyu Shen, Hanyu Jin, and Kenji Shimada. Navrl: Learning safe flight in dynamic environments. *IEEE Robotics and Automation Letters*, 2025.

[33] Yuang Zhang, Yu Hu, Yunlong Song, Danping Zou, and Weiyao Lin. Learning vision-based agile flight via differentiable physics. *Nature Machine Intelligence*, pages 1–13, 2025.

[34] Xin Zhou, Zhepei Wang, Hongkai Ye, Chao Xu, and Fei Gao. Ego-planner: An esdf-free gradient-based local planner for quadrotors. *IEEE Robotics and Automation Letters*, 6(2):478–485, 2020.

[35] Xin Zhou, Xiangyong Wen, Zhepei Wang, Yuman Gao, Haojia Li, Qianhao Wang, Tiankai Yang, Haojian Lu, Yanjun Cao, Chao Xu, et al. Swarm of micro flying robots in the wild. *Science Robotics*, 7(66):eabm5954, 2022.