AUG: Perception-Aware UAV Graph-Based Planning for GNSS-Denied Navigation in Feature-Sparse Environments

Enguang Feng, Haohuan Yuan, Hong Zhang, Hao Xu†, Yikun Dou, Jiarong Lin† and Xiwang Dong

***Abstract*— Long-range unmanned aerial vehicle (UAV) navigation in Global Navigation Satellite System (GNSS)-denied, feature-sparse outdoor environments remains challenging because LiDAR-inertial odometry (LIO) can become ill-conditioned, causing drift accumulation and eventual divergence even when collision-free geometric paths exist. In this work, we present AUG, a scalable perception-aware planning framework for LiDAR-inertial navigation that couples long-horizon connectivity reasoning with estimator-aware online replanning. AUG integrates three components: an incremental free-space topological graph for global routing without a global Euclidean signed distance field (ESDF), a robocentric *ROG-map* with a *MINCO* local optimizer for real-time trajectory generation, and a perception-constrained yaw planner that steers the sensor toward informative structures to improve LIO conditioning. In three 1.2 km × 1.2 km simulation benchmarks (10 runs per method per scenario), AUG achieves 100 % success under a unified criterion that treats collision, LIO divergence, or relative pose error (RPE) > 10 % as failure, while maintaining 1-2 % RPE and a nearly constant memory footprint of 0.75-0.81 GB. Compared with representative voxel-free and ESDF-based baselines, AUG consistently yields higher reliability and lower drift. Real-world experiments further validate a fully onboard implementation using only LiDAR-inertial sensing: in 10 indoor flights of approximately 30 m, results compared to ground truth verify stable onboard execution, and in 10 outdoor flights of approximately 200 m, AUG completes all trials with a mean GNSS-referenced drift ratio of 2.24 %, where GNSS is used only for offline evaluation.**

I. INTRODUCTION

Reliable long-range UAV navigation in GNSS-denied outdoor environments depends on stable onboard state estimation. However, even in the presence of collision-free geometric paths, localization drift can cause mission failure by invalidating the planned trajectory, thus highlighting the necessity for estimation-consistent planning in autonomous systems. While LIO is robust to changes in illumination [1], in open or sparsely structured environments (e.g., grasslands, wide open fields, or long tunnels), reduced geometric constraints weaken observability [2], leading to degenerate (ill-conditioned) LIO updates, drift accumulation, and eventual estimator divergence during kilometre-scale flights. Consequently, it is critical to adopt planning strategies that explicitly account for estimator conditioning, rather than relying solely on geometric considerations.

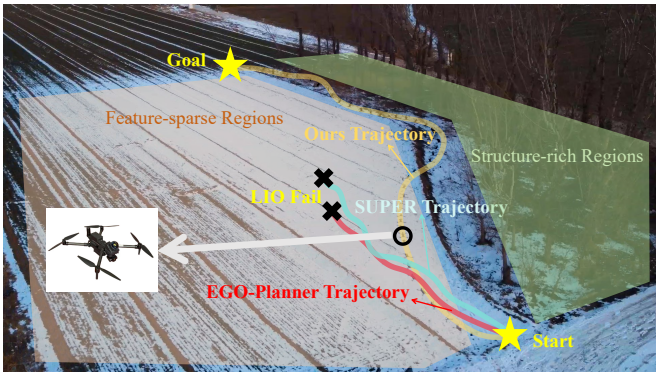

Fig. 1. Real-world illustration of the motivation and function of AUG in long-range GNSS-denied navigation. Although the direct route from start to goal is geometrically feasible, the open field is a feature-sparse region in which LiDAR-inertial odometry (LIO) is prone to degeneration, while the tree-lined boundary provides a structure-rich region with more informative constraints for LIO. As a result, baseline planners (*EGO-Planner* in red and *SUPER* in cyan) enter the feature-sparse area and fail due to localization degradation (black crosses), whereas AUG (yellow) selects a perception-aware route through the structure-rich corridor to maintain estimator conditioning and reach the goal reliably.

Most existing UAV planners focus on collision avoidance and kinodynamic feasibility [3]. While trajectory-optimization pipelines [4], [5] perform well in cluttered environments, they become memory-intensive for kilometre-scale missions. Voxel-free planners [6], [7] improve scalability by optimizing directly on sliding-window point clouds [7], but they typically assume reliable localization and apply geometric yaw heuristics (e.g., velocity alignment). In feature-sparse regions, however, these yaw heuristics can rapidly reduce the informative constraints used by LIO, leading to divergence [8], [9]. Moreover, even with a 360° LiDAR (e.g., the *Livox Mid-360*), the constraints selected and weighted by LIO remain anisotropic due to self-occlusion, limited vertical FoV, and range- and incidence-dependent feature selection, making yaw optimization crucial for maintaining estimator conditioning [10].

We propose a perception-aware UAV graph-based planning framework, **AUG**, that enables scalable kilometre-scale

This work was supported by the National Key RD Plan of China under Grant No. 2024YFB4708300, the National Natural Science Foundation of China under Grant U2241217, and the Beijing Natural Science Foundation under Grant JQ23019.

Corresponding authors: Hao Xu† and Jiarong Lin†.

Enguang Feng, Xiwang Dong, and Jiarong Lin are with the Institute of Unmanned System, Beihang University, Beijing, China.{fegkkd,xwdong,zivlin}@buaa.edu.cn

Haohuan Yuan is with the School of Artificial Intelligence, Beihang University, Beijing, China. zb2542218@buaa.edu.cn

Hong Zhang is with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. yiy1117@buaa.edu.cn

Yikun Dou is with the National Superior College for Engineers, Beihang University, Beijing, China. yiy1117@buaa.edu.cn

Hao Xu is with the School of Intelligence Science and Technology, Nanjing University, Suzhou, China. xxx@nju.edu.cn

missions while preventing perceptual degeneration. AUG incrementally constructs a lightweight free-space topological graph from local free space inferred by the *ROG-map* [6], and uses LiDAR feature statistics to bias routing, without the need for a global ESDF.

Guided by this topology, a robocentric *ROG-map* local planner generates real-time *MINCO* [11] position trajectories, and a perception-constrained yaw planner explicitly steers the sensor toward nearby well-conditioned structures and refines a smooth yaw profile within visibility corridors. The resulting motion jointly improves safety, dynamic feasibility, and LIO robustness.

The main contributions of this work are listed as follows:

- We propose topology-guided long-range planning without a global ESDF, which builds a lightweight free-space topological graph via bubble-based [12] [13] free-space decomposition and feature-aware node selection, enabling kilometre-scale navigation with bounded memory.
- We propose perception-constrained yaw optimization for LIO robustness. To this end, we design a yaw planner that increases the *effective* informative constraints available to LIO, and it steers yaw toward nearby well-conditioned structures under dynamic limits.
- We validate AUG in sparse long-range settings via large-scale simulations and real-world flights. Compared to *EGO-Planner* [5] and *SUPER* [7], results show that our method achieves a higher success rate, lower drift, and real-time performance with nearly constant memory.

II. Related Work

A. Perception-Aware Planning

Perception-aware planning [14], [15] has received increasing attention for UAV navigation in feature-sparse environments, where state estimation can become ill-conditioned and may eventually diverge. A common paradigm is to quantify the *informativeness* of candidate motions and incorporate it as a planning objective or constraint, thereby discouraging trajectories that traverse perceptually degenerate regions.

In vision-based localization, Zhang and Scaramuzza [16] propose Fisher-information-based maps to enable differentiable perception-aware planning, whereas Bartolomei *et al.* [17] leverage semantic segmentation to identify texture-poor areas and penalize motions that rely on weak visual cues. For LiDAR-based systems, Chai *et al.* [9] introduce a perturbation-induced metric together with a lightweight perceptual map to accelerate perception queries during planning. Despite their effectiveness, these approaches typically rely on a perception field (or its surrogate) that must be constructed and maintained, which can limit adaptability in previously unseen, large-scale scenes with highly non-uniform feature distributions.

To improve online adaptability, several works evaluate perceptual quality on-the-fly within a receding-horizon loop. Costante *et al.* [18] incorporate information gain into planning to avoid feature-sparse regions, and Takemura and Ishigami [19] study perception-aware receding-horizon navigation with LiDAR-based simultaneous localization and mapping (SLAM). However, most online methods reason primarily in the 3D position space and handle yaw using simple geometric heuristics (i.e., velocity alignment) [7]. In practice, yaw influences which structures fall into the *effective* informative set and thus dominate the LIO update (Sec. IV-C); consequently, purely geometric yaw heuristics can be brittle in feature-sparse scenes.

Overall, existing perception-aware frameworks either depend on static or preconstructed priors (reducing adaptability in unknown environments) or do not explicitly co-optimize yaw to maintain an informative constraint set. In addition, prior work is largely vision-centric; for LiDAR, perception-aware planning is comparatively under-explored even though LIO exhibits LiDAR-specific degradation mechanisms, where the *effective* constraints are highly anisotropic due to self-occlusion, limited vertical FoV, and range-/incidence-dependent feature selection. These gaps motivate a planner that evaluates perceptual structure online, jointly reasons about position and yaw, and remains scalable for kilometre-scale outdoor navigation.

B. UAV Trajectory Planning

Classical UAV navigation frameworks [3], [20] commonly adopt receding-horizon strategies that repeatedly compute locally optimal trajectories toward distant goals in unknown environments. Zhou *et al.* [4] combine kinodynamic search with trajectory optimization to enable fast and safe flight in cluttered scenes. *EGO-Planner* [5] further improves reactivity using an ESDF-free, gradient-based local optimization pipeline; however, for kilometre-scale missions its mapping-related data structures typically require *pre-declared* map bounds that must expand with the operational range, increasing memory demand on resource-constrained platforms.

When the operational range extends to hundreds of meters or beyond, scalable mapping becomes a key requirement. *SUPER* [7] addresses this by optimizing trajectories directly on point clouds using a robocentric sliding *ROG-map* representation [6], enabling long-range navigation without constructing a global voxel/ESDF map. While such voxel-free planners substantially improve scalability, they typically assume reliable state estimation and treat yaw in a purely geometric manner. As a result, they can still be vulnerable in perception-degraded environments, where feature visibility varies significantly and LiDAR-inertial localization may drift or fail.

In summary, few existing planners simultaneously provide (i) long-horizon global connectivity reasoning, (ii) online preference for perceptually informative corridors in unknown large-scale scenes, and (iii) explicit yaw optimization to preserve an informative constraint set for LiDAR-inertial estimation. Building on the *ROG-map*-based optimization framework, our work introduces a topology-guided global routing layer together with a perception-constrained yaw optimization backend to enable robust, long-range navigation

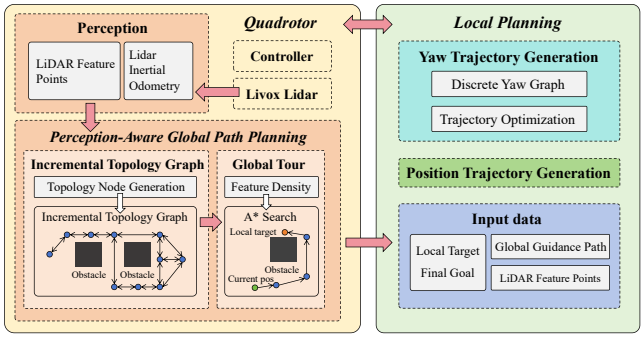

Fig. 2. Overview of our proposed perception-aware LiDAR planning framework. The Perception and UAV subsystems provide LiDAR feature points and LIO odometry. The Perception-Aware Global Path Planning layer incrementally builds a feature-aware topological graph and computes a global guidance path. The Local Planning layer takes the global guidance path, local targets, and LiDAR features as input, generating a *MINCO*-based position trajectory together with a visibility-aware yaw trajectory.

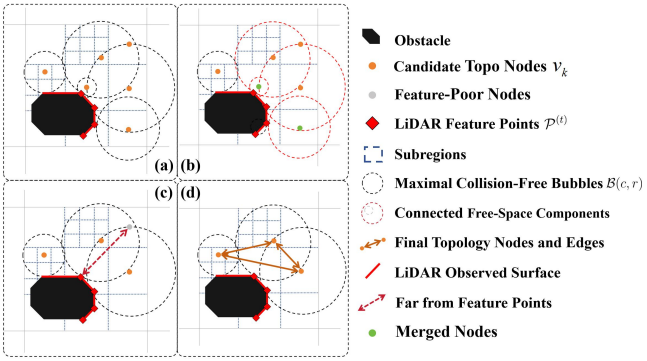

Fig. 3. Topology node generation from LiDAR observations. Maximal collision-free bubbles are grown in a local update region, merged into connected free-space components, filtered by feature density, and finally connected into an incremental topological graph that captures large-scale connectivity.

with consistent LiDAR localization in feature-sparse environments.

III. System Overview

We target long-range UAV navigation in GNSS-denied, feature-sparse environments using only onboard LiDAR-inertial sensing. The system follows a voxel-free, point-cloud-centric paradigm: a robocentric sliding map is maintained via *ROG-map* [6], enabling real-time trajectory optimization directly from incoming LiDAR scans with near-constant memory. To add long-horizon guidance and estimation robustness, we augment this local optimizer with (i) a topology-guided global routing layer that incrementally builds a sparse free-space topological graph with perception-weighted costs, and (ii) a perception-aware yaw module that searches and refines a visibility-constrained yaw profile along the position trajectory. Fig. 2 summarizes the pipeline.

IV. Methodology

A. Perception-Aware Global Path Planning

As shown in Fig. 2, the global layer incrementally builds a sparse free-space topological graph and assigns perception-weighted costs using LiDAR features, without maintaining a global voxel/ESDF map.

1) Topology Node Generation: At time t, we maintain an inflated occupancy set $\mathcal{O}^{(t)}$ from the local *ROG-map* and LiDAR feature points $\mathcal{F}^{(t)}$ in the sliding window, as illustrated in Fig. 3. We update topology only in a local region $\mathcal{R}^{(t)}$ by growing maximal collision-free spheres ("bubbles") from free cells, while treating unknown space as occupied.

For each candidate center $c \in \mathcal{R}^{(t)}$, we define a maximal collision-free sphere

$$\mathcal{B}(c,r) = \{x \in \mathbb{R}^3 \mid \|x-c\|_2 \le r\}, \quad r = \min_{o \in \mathcal{O}^{(t)}} \|c-o\|_2, \tag{1}$$

where r denotes the clearance to the nearest inflated occupied element. We then recursively subdivide uncovered subregions in $\mathcal{R}^{(t)}$ until a bubble satisfies $r > r_{\min}$ (kept) or the region size falls below δ_{reg} (discarded).

To recover connectivity, two bubbles (c_a, r_a) and (c_b, r_b) overlap if

$$\|c_a - c_b\|_2 < r_a + r_b - \delta_{\text{conn}}, \tag{2}$$

where δ_{conn} is a margin to avoid spurious links through narrow gaps, and we merge overlapping bubbles into free-space components (union-find) and take the largest-radius center in each component as a candidate node c_k.

To enforce perceptual quality, we discard nodes in feature-sparse areas. For a candidate c_k, let $\mathcal{F}(c_k, r_{\text{feat}})$ denote feature points within radius r_{feat} that are effectively visible under the current pose estimate. We keep c_k only if

$$\left|\mathcal{F}(c_k, r_{\text{feat}})\right| \ge N_{\min}, \tag{3}$$

where $N_{\min} > 0$ is a minimum feature-count threshold.

Surviving candidates are promoted to topological nodes with positions $v_k = c_k$, and the resulting node set at time t is

$$\mathcal{V}^{(t)} = \{v_1, \dots, v_{N_v}\}, \tag{4}$$

where N_v is the number of nodes.

2) Incremental Topological Graph and Global Guidance Path: The global free space is represented as an undirected graph

$$\mathcal{G}^{(t)} = \left(\mathcal{V}^{(t)}, \mathcal{E}^{(t)}\right), \tag{5}$$

where $\mathcal{E}^{(t)}$ is the edge set. Each node tracks re-observations: nodes exceeding N_{obs} are marked confirmed, while stale nodes are pruned together with incident edges.

Edges connect nearby confirmed nodes within r_{nbr} after a ray-based collision check on the occupancy map (unknown treated as occupied). Since updates are limited to newly observed regions, the per-step update cost is largely independent of total flight distance.

To bias routing toward regions that better support LiDAR-inertial estimation, each node v_i is assigned a local feature density

$$\rho_i = \frac{1}{|\mathcal{N}_i|} \sum_{p \in \mathcal{N}_i} \exp\!\left(-\frac{\|p - v_i\|_2^2}{2\sigma_\rho^2}\right), \tag{6}$$

where $\mathcal{N}_i$ are nearby feature points within r_ρ, and σ_ρ controls smoothing. Each edge $(i,j) \in \mathcal{E}^{(t)}$ is then assigned a perception-weighted cost

$$\begin{aligned} w_{ij} &= \|v_i - v_j\|_2 \Big(1 + \lambda_\rho (1 - \bar{\rho}_{ij})\Big), \\ \bar{\rho}_{ij} &= \tfrac{1}{2}(\rho_i + \rho_j), \end{aligned} \tag{7}$$

where $\lambda_\rho > 0$ balances geometric efficiency and perceptual quality.

Let $x_s, x_g \in \mathbb{R}^3$ denote the current robot position and the goal. They are attached to $\mathcal{G}^{(t)}$ via virtual nodes connected to their nearest neighbors in $\mathcal{V}^{(t)}$. A standard A* search on $\mathcal{G}^{(t)}$ using w_{ij} as edge cost yields a perception-aware global guidance path

$$\Gamma = \big[v_s,\, v_{k_1},\, \ldots,\, v_{k_M},\, v_g\big], \tag{8}$$

where v_s and v_g are associated with x_s and x_g, and $\{v_{k_m}\}_{m=1}^{M}$ are intermediate nodes.

B. Position Trajectory Generation

Given Γ, we generate a collision-free, dynamically feasible trajectory $p(t) \in \mathbb{R}^3$ using the *ROG-map+MINCO* pipeline in *SUPER* [7], with local targets selected from our perception-aware topology.

At each replanning step, we project the current position p_0 onto Γ and choose a fixed-arc-length look-ahead target p_{ref}. A fixed-size *ROG-map* window is updated from recent scans, and an A* path from p_0 to p_{ref} is inflated into convex safe flight corridors [21]

$$\mathcal{C}_k = \big\{x \in \mathbb{R}^3 \,\big|\, A_k x \le b_k\big\}, \tag{9}$$

where $A_k \in \mathbb{R}^{m_k \times 3}$ and $b_k \in \mathbb{R}^{m_k}$ encode half-space constraints that keep the corridor away from occupied cells.

We represent $p(t)$ as a *MINCO* spline with M_p segments over horizon T and optimize its parameters by minimizing

$$J_{\text{pos}} = \int_0^T \big\|p^{(4)}(t)\big\|_2^2 \, dt + \lambda_{\text{dyn}} J_{\text{dyn}} + \lambda_{\text{sfc}} J_{\text{sfc}}, \tag{10}$$

where the snap term promotes smoothness, J_{dyn} penalizes velocity/acceleration limit violations, and J_{sfc} enforces corridor adherence at collocation points. We use weights $\lambda_{\text{dyn}}, \lambda_{\text{sfc}} > 0$ and enforce boundary conditions on position/velocity at $t = 0, T$.

C. Yaw Trajectory Generation

We plan a dynamically feasible yaw trajectory $\psi(t)$ that keeps informative structures in view; unlike velocity-aligned yaw, we model visibility via a discrete yaw graph followed by spline refinement (Fig. 5).

1) LiDAR Visibility and Discrete Yaw Graph: Let $c(t) \in \mathbb{R}^3$ denote the LiDAR center at time t, and let $x \in \mathbb{R}^3$ be a world-frame point, where the line-of-sight vector is $d = x - c(t)$, and we model yaw-dependent *effective* visibility using an informative sector rather than the physical horizontal FoV. Although some 360° LiDAR provide 360° horizontal coverage, the dominant constraints used by LIO are anisotropic due to self-occlusion, limited vertical FoV, and range-/incidence-dependent feature selection and weighting.

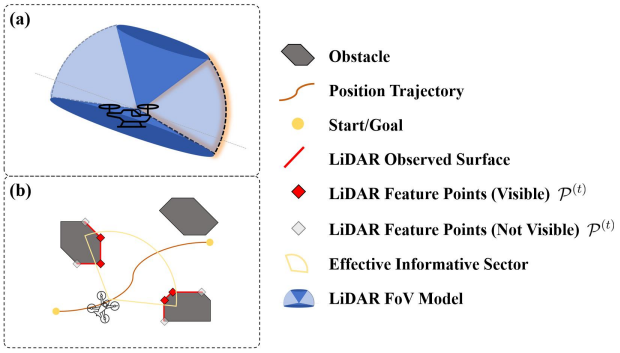

Fig. 4. Illustration of yaw-dependent effective visibility for yaw planning. (a) LiDAR FoV model: due to the sensor mounting angle and body self-occlusion, only the forward conical sector is considered as the effective visible region. (b) Visibility example along a position trajectory $p(t)$ (orange curve), where the yellow sector denotes the *effective informative sector*. Red diamonds indicate feature points that are visible and not occluded by obstacles.

We therefore approximate an *effective informative sector* by a polyhedral cone bounded by four planes with outward normals $\{n_i\}_{i=1}^{4}$ defined in the LiDAR frame (Fig. 4). Let $R_{WL} \in \mathrm{SO}(3)$ be the rotation from LiDAR to world frame, and $R_{\max} > 0$ the LiDAR range. A point x is considered effectively visible if

$$\|d\|_2 \le R_{\max}, \quad (R_{WL} n_i)^\top d < 0, \ \forall i, \tag{11}$$

and the ray segment from $c(t)$ to x does not intersect any occupied cell in the LiDAR map. For a given position $p(t)$ and yaw angle ψ, the number of visible feature points is denoted $N(t,\psi) \in \mathbb{N}$.

We discretize the planning horizon $[0, T]$ into time samples $\{t_i\}_{i=0}^{N_\tau}$ with step sizes $\Delta t_i = t_{i+1} - t_i$. At each time t_i, a reference yaw ψ_i^{ref} is defined from the forward tangent of $p(t)$,

$$\begin{aligned} \psi_i^{\text{ref}} &= \operatorname{atan2}(u_{i,y}, u_{i,x}), \\ u_i &= \frac{p(t_i + \delta) - p(t_i)}{\|p(t_i + \delta) - p(t_i)\|_2}, \end{aligned} \tag{12}$$

where $\delta > 0$ is a small look-ahead time and u_i is the unit tangent vector.

At each t_i, we sample candidates $\{\psi_{i,j}\}$ within the reachable interval $[\psi_i^{\text{ref}} - \dot{\psi}_{\max}\Delta t_i,\ \psi_i^{\text{ref}} + \dot{\psi}_{\max}\Delta t_i]$ and retain those satisfying $|\psi_{i,j} - \psi_i^{\text{ref}}| \le \theta_{\max}$ and $N(t_i, \psi_{i,j}) \ge N_{\min}^{\psi}$. Retained candidates form a layered directed graph over time.

Two candidates (i,j) and $(i+1,k)$ are connected if

$$\big|\psi_{i+1,k} - \psi_{i,j}\big| \le \dot{\psi}_{\max}\Delta t_i, \tag{13}$$

ensuring yaw-rate feasibility. Each edge is assigned a cost

$$\begin{aligned} c_{i,j\to i+1,k} = w_s \big(\psi_{i+1,k} - \psi_{i,j}\big)^2 + w_d \big(\phi_i + \phi_{i+1}\big) \\ - w_f \frac{N(t_i, \psi_{i,j}) + N(t_{i+1}, \psi_{i+1,k})}{2}, \end{aligned} \tag{14}$$

where $w_s, w_d, w_f > 0$ are weighting coefficients, $\phi_i = \big(\psi_{i,j} - \psi_i^{\text{ref}}\big)^2$ and $\phi_{i+1} = \big(\psi_{i+1,k} - \psi_{i+1}^{\text{ref}}\big)^2$. A shortest-path

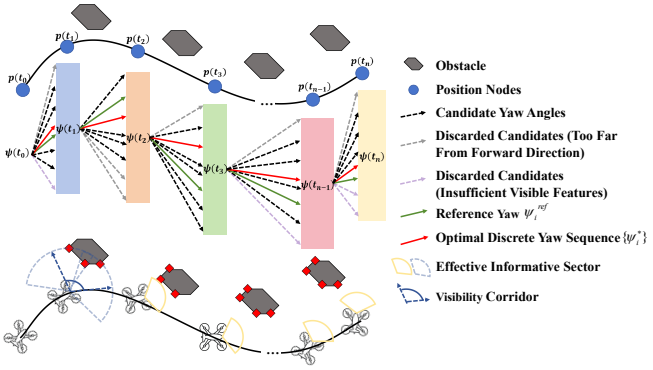

Fig. 5. Perception-aware yaw planning pipeline. Along a fixed position trajectory, candidate yaw angles are sampled at each time step, filtered by dynamic and visibility constraints, and connected into a layered graph. A shortest-path search returns an optimal discrete yaw sequence, which is then refined to a smooth spline within visibility corridors.

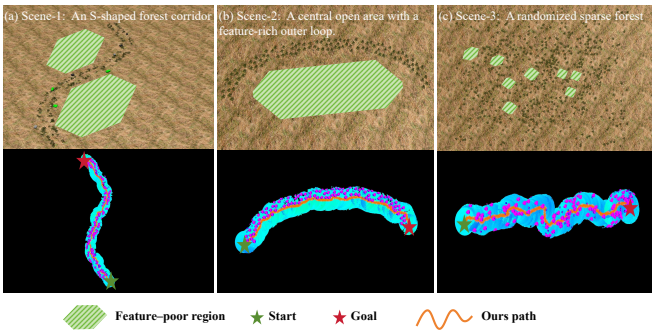

Fig. 6. Benchmark environments and representative trajectories. Top (green hatched regions indicate feature-sparse areas): (a) Scene-1: An S-shaped forest corridor, (b) Scene-2: A central open area with a structure-rich outer loop, and (c) Scene-3: A randomized sparse forest. Bottom: accumulated LiDAR map and a representative trajectory of AUG; cyan tubes show the flown corridor and magenta points indicate *FAST-LIO* features.

search over this graph yields a discrete perception-aware yaw sequence $\{\psi_i^\star\}_{i=0}^{N_\tau}$.

2) Trajectory Optimization: We refine $\{\psi_i^\star\}$ into a smooth *MINCO* spline $\psi(t)$ over the same time partition as $p(t)$. For each time step t_i, we approximate a feature-visible yaw interval

$$\mathcal{I}_i^{\text{feat}} = \left\{\psi \,\middle|\, N(t_i,\psi) \geq N_{\min}^{\psi}\right\} \tag{15}$$

and intersect it with the LiDAR horizontal FoV centered at ψ_i^{ref},

$$\mathcal{I}_i = \mathcal{I}_i^{\text{feat}} \cap \left[\psi_i^{\text{ref}} - \tfrac{\alpha_h}{2},\ \psi_i^{\text{ref}} + \tfrac{\alpha_h}{2}\right], \tag{16}$$

where $\alpha_h > 0$ denotes the horizontal aperture of the *effective informative sector* that captures the subset of returns providing dominant constraints to the LIO front-end. This sector is not the physical FoV, but a modeling abstraction reflecting self-occlusion, limited vertical FoV, and range-/incidence-dependent feature selection and weighting in LIO; steering yaw toward nearby structures increases both the number and conditioning of effective constraints.

The continuous yaw spline is obtained by minimizing

$$J_{\text{yaw}} = \int_0^T \dot{\psi}(t)^2\, dt + \lambda_{\text{dyn}} J_{\text{dyn}}^{\psi} + \lambda_{\text{corr}} J_{\text{corr}}, \tag{17}$$

where J_{dyn}^{ψ} penalizes yaw-rate/acceleration limit violations and J_{corr} enforces corridor membership at collocation points. With weights $\lambda_{\text{dyn}}, \lambda_{\text{corr}} > 0$, the resulting $\psi(t)$ is smooth, feasible, and maintains sufficient visible features for robust LIO.

V. Experiments

A. Experimental Setup

1) Simulator and Hardware: We conduct all simulations in Gazebo with physics-based LiDAR rendering. The onboard sensor suite is modeled as a *Livox Mid-360* (360° horizontal FoV, vertical FoV of −7-52°, and range up to 70 m), and we use *FAST-LIO* [1] as the odometry backend for *all* methods to ensure fair comparison. Unless otherwise specified, planning runs at 20 Hz on a desktop computer (Intel i7, 32 GB RAM). The UAV limits are set to $v_{\max} = 3.0\,\text{m/s}$ and $\dot{\psi}_{\max} = 90\,^\circ/\text{s}$, consistent with our real platform.

We quantify yaw-dependent LIO conditioning using $\lambda_{\min}^{\text{norm}}$ in Sec. V-A.3.

2) Benchmark Environments: We design three $1.2\,\text{km} \times 1.2\,\text{km}$ benchmark testing scenarios, where the start-goal distances range from 800 m to 1.2 km:

- **Scene-1: An S-shaped forest corridor.** Curved tree rows form an S-shaped corridor, which evaluates yaw behavior under moderate feature density.
- **Scene-2: A central open area with a structure-rich outer loop.** A feature-sparse open center with a nearby structure-rich loop, which stresses topology guidance and estimator robustness.
- **Scene-3: A randomized sparse forest.** Irregularly spaced trees that emulate realistic sparse structure, with no obvious corridor present.

Fig. 6 visualizes the environments and representative trajectories: the top row shows terrain layouts with feature-sparse regions highlighted (green hatching), and the bottom row shows the accumulated LiDAR map and a representative trajectory of our method (cyan tube), with *FAST-LIO* feature points in magenta. For each scenario and method, we report statistics over ten runs, each with a randomly sampled start-goal pair.

3) Evaluation Metrics: Following standard odometry evaluation practice [22], we report:

1) **Success Rate (%).** A trial is counted as successful only if the UAV reaches the goal *without collision* and localization remains reliable. We declare failure if *FAST-LIO* diverges *or* if drift becomes unacceptable (RPE $> 10\,\%$), even when the flight is collision-free. This unified criterion avoids overestimating planners that succeed geometrically but suffer severe drift in feature-sparse regions. We set the threshold to RPE $> 10\,\%$ because such drift makes waypoint-level navigation unreliable for kilometre-scale missions.
2) **ATE / RPE.** Absolute trajectory error (ATE) is

the RMSE after SE(3) alignment; relative pose error (RPE[%]) measures drift over fixed-length segments.

3) **Memory [MB] / Runtime [ms].** We report the average RAM usage of mapping/planning modules and the mean per-step planning time.
4) N_{feat} **/** $\boldsymbol{\lambda}_{\min}^{\text{norm}}$**.** N_{feat} is the average number of key features *actually selected and used* by *FAST-LIO* in each update (from logs), which is distinct from the planner-side visibility count $N(t, \psi)$. $\lambda_{\min}^{\text{norm}} = \lambda_{\min}(\mathbf{H}_{\text{pose}})/(\text{tr}(\mathbf{H}_{\text{pose}}) + \epsilon)$ is the normalized minimum eigenvalue of the pose Hessian from each LIO update. It characterizes the weakest constrained direction in the LIO update: values closer to zero indicate stronger degeneracy, poorer estimator conditioning, and typically worse localization quality.

We visualize $\lambda_{\min}^{\text{norm}}$ as a time series in Fig. 8 (rather than aggregating it in Table I) to show when and how estimator degeneracy develops along the trajectory. In particular, decreases of $\lambda_{\min}^{\text{norm}}$ toward zero indicate increasingly ill-conditioned LIO updates and are typically accompanied by larger localization error. Quantitative results are summarized in Table I, and the temporal coupling among feature availability, estimator conditioning, and localization error is illustrated in Fig. 8.

4) Implementation Details: Unless otherwise specified, we use a single parameter set for all simulations and real-world flights. For the topology layer, $r_{\min}$, δ_{reg}, and δ_{conn} avoid unstable nodes and spurious links in narrow gaps. Nodes are retained with at least $N_{\min}$ nearby LiDAR features within r_{feat}, confirmed after N_{obs} re-observations, and connected within r_{nbr}. For yaw planning, candidates are sampled within $\theta_{\max}$, must satisfy $N_{\min}^{\psi}$ visible features, and use weights (w_s, w_d, w_f) and $(\lambda_{\text{dyn}}, \lambda_{\text{corr}})$. All methods use the same replanning frequency and vehicle limits.

Baseline fairness. For ***EGO-Planner***, we set the pre-declared map bounds to cover the entire $1.2\,\text{km} \times 1.2\,\text{km}$ workspace with a fixed safety margin; smaller bounds may cause the vehicle to exit the allocated map volume during kilometre-scale missions.

B. Simulation Benchmarks

1) Baselines and Ablations: We compare AUG with two representative planners:

- ***EGO-Planner* [5]:** a gradient-based local planner that optimizes dynamically feasible trajectories using collision costs from an incremental occupancy representation. For long-range missions, it typically requires *pre-declared* global map bounds/volumes, increasing memory usage as the mission scale grows; as a purely local method, it may also enter poorly constrained open regions.
- ***SUPER* [7]:** a scalable voxel-free baseline that performs *ROG-map*-based local optimization on point clouds [6], but without topology-guided routing or perception-aware yaw optimization.

We choose ***EGO-Planner*** and ***SUPER*** as representative ESDF-free local planners with widely used long-range variants, covering both gradient-based and *ROG-map*-based paradigms. The success criterion described above is applied identically to all methods and ablations to ensure a fair comparison under the same navigation reliability requirement.

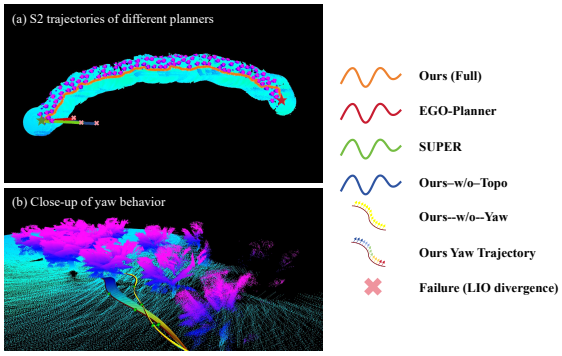

Fig. 7. Qualitative comparison in Scene-2. Top: accumulated LiDAR map and trajectories of different planners. *EGO-Planner*, *SUPER*, and Ours-w/o-Topo enter the central feature-sparse region and fail due to LiDAR drift, whereas AUG and Ours-w/o-Yaw follow the structure-rich loop. Bottom: close-up of yaw behavior; colored arrows indicate yaw directions. Ours-w/o-Yaw keeps the yaw approximately aligned with the velocity direction, whereas AUG steers the LiDAR toward nearby structure-rich regions through yaw optimization.

We additionally evaluate two ablations: **Ours-w/o-Yaw** (yaw aligned with velocity) and **Ours-w/o-Topo** (no global topology; local targets selected purely from the local planner). All methods share the same *FAST-LIO* backend [1] and identical vehicle limits. We declare failure if the UAV collides, *FAST-LIO* diverges, or RPE $> 10\,\%$ (Sec. V-A.3). In Scene-1, ***EGO-Planner*** often completes the geometric corridor but accumulates large drift in weakly constrained segments, frequently triggering the RPE $> 10\,\%$ criterion.

Fig. 7 provides a qualitative comparison in Scene-2. ***EGO-Planner***, ***SUPER***, and **Ours-w/o-Topo** tend to cut through the central feature-sparse region and fail due to feature starvation, whereas AUG and **Ours-w/o-Yaw** follow the peripheral structure-rich loop. The bottom subfigure further highlights yaw behavior: AUG actively steers the LiDAR toward nearby structures, while **Ours-w/o-Yaw** keeps yaw close to the velocity direction, reducing effective feature visibility.

2) Long-Range Localization Performance: Across all environments, AUG achieves the highest reliability and the lowest drift among the compared methods under the unified success criterion.

In Scene-1, AUG and **Ours-w/o-Yaw** complete all trials with $100\,\%$ success, while ***EGO-Planner***, ***SUPER***, and **Ours-w/o-Topo** fail in all runs and therefore do not report ATE/RPE. Compared with **Ours-w/o-Yaw**, AUG reduces ATE from 4.18 m to 2.38 m and RPE from $2.7\,\%$ to $1.1\,\%$, indicating that perception-aware yaw planning improves long-range localization performance even when the global route remains feasible.

In the highly degenerate Scene-2, ***EGO-Planner***, ***SUPER***, and **Ours-w/o-Topo** fail in all trials, whereas AUG and **Ours-w/o-Yaw** achieve $100\,\%$ success by following

TABLE I

COMPREHENSIVE SIMULATION RESULTS (10 RUNS PER SCENARIO)

Scenario	Method	Success (%)	ATE (m)	RPE (%)	N_{feat}	Mem. (MB)	Time (ms)
Scene-1	*EGO-Planner*	0	-	-	420	1558	41
	SUPER	0	-	-	510	**741**	**18**
	Ours-w/o-Yaw	100	4.18	2.7	910	796	22
	Ours-w/o-Topo	0	-	-	503	753	19
	Ours (Full)	**100**	**2.38**	**1.1**	**1280**	799	24
Scene-2	*EGO-Planner*	0	-	-	310	2363	54
	SUPER	0	-	-	290	**741**	**18**
	Ours-w/o-Yaw	100	6.57	3.3	880	805	23
	Ours-w/o-Topo	0	-	-	350	753	19
	Ours (Full)	**100**	**3.68**	**1.5**	**1370**	811	25
Scene-3	*EGO-Planner*	30	16.93	7.1	860	1760	47
	SUPER	40	11.72	5.5	920	**748**	**19**
	Ours-w/o-Yaw	100	5.24	2.9	1100	804	23
	Ours-w/o-Topo	50	7.92	4.0	950	754	20
	Ours (Full)	**100**	**3.13**	**1.7**	**1430**	814	26

structure-rich corridors. Moreover, AUG further reduces ATE from 6.57 m to 3.68 m and RPE from 3.3 % to 1.5 % relative to **Ours-w/o-Yaw**, highlighting the complementary benefits of topology guidance and yaw optimization.

In Scene-3, AUG and **Ours-w/o-Yaw** achieve 100 % success with RPE 1.7 % and 2.9 %, respectively. In contrast, ***EGO-Planner*** and ***SUPER*** complete only 30 % and 40 % of the runs, with substantially larger drift (RPE 7.1 % and 5.5 %). These results demonstrate robust performance of AUG in sparse and irregular environments.

3) Feature Visibility and Observability: We report both the feature count N_{feat} and the conditioning metric $\lambda_{\min}^{\text{norm}}$ from *FAST-LIO* logs. N_{feat} denotes the number of key features selected and used in each update (averaged over a short temporal window), while $\lambda_{\min}^{\text{norm}}$ is the normalized minimum eigenvalue of the pose Hessian $\mathbf{H}_{\text{pose}}$ (Sec. V-A.3). Since it reflects the weakest constrained direction in the LIO update, smaller values indicate stronger degeneracy; in particular, values approaching zero correspond to severely ill-conditioned updates and are typically associated with poorer localization.

Fig. 8 shows a representative run in Scene-2. For ***EGO-Planner***, ***SUPER***, and **Ours-w/o-Topo**, $\lambda_{\min}^{\text{norm}}$ drops rapidly toward zero after entering the central open region, accompanied by a sharp drop in N_{feat}; localization error then grows rapidly and eventually diverges. **Ours-w/o-Yaw** follows the loop and maintains a moderate N_{feat}, but still exhibits intervals with reduced $\lambda_{\min}^{\text{norm}}$ due to suboptimal orientation, which leads to weaker conditioning and larger error than the full method. In contrast, AUG sustains higher $\lambda_{\min}^{\text{norm}}$, indicating better-conditioned LIO updates, and maintains bounded localization error throughout the trajectory.

4) Scalability and Runtime: Scalability is reflected by the *Mem.[MB]* and *Time[ms]* columns in Table I. ***EGO-Planner*** requires a *pre-declared* map volume; thus its memory increases with mission scale (up to 2.36 GB in Scene-2 and over 1.7 GB in Scene-3). In contrast, all *ROG-map*-based methods (***SUPER***, our ablations, and AUG) maintain a nearly constant footprint of 750-810 MB (0.75-0.81 GB), largely independent of explored distance.

All variants of our method complete replanning within 24-26 ms per step, below the 50 ms budget implied by the 20 Hz replanning frequency. ***EGO-Planner*** is slower (up to 54 ms in Scene-2), while ***SUPER*** is slightly faster but less robust, indicating that our topology and yaw modules preserve real-time performance.

C. Real-World Experiments

For real-world validation, we deploy AUG on a custom UAV equipped with a *Livox Mid-360* and an NVIDIA Jetson NX (16 GB RAM). LIO, mapping, planning, and control run fully onboard using only LiDAR-inertial sensing. We conduct 10 indoor flights (approximately 30 m) for algorithm verification and onboard flight-stability validation, and 10 outdoor flights (approximately 200 m) for field evaluation and baseline comparison.

Ground-truth for evaluation (not used for planning/estimation). For the indoor tests, motion capture is used only as ground truth for evaluation. For the outdoor field tests, we log a consumer-grade GNSS receiver as an *external reference* for drift evaluation. GNSS is *never* fused into *FAST-LIO* and is used *only* for offline evaluation. We time-synchronize GNSS with LIO odometry and align them using the initial pose. We report the normalized drift ratio

$$\text{RPE}_{\text{GNSS}}[\%] = \frac{\|p_{\text{LIO}}(T) - p_{\text{GNSS}}(T)\|_2}{L} \times 100\,\%, \quad (18)$$

where $p_{\text{LIO}}(T)$ and $p_{\text{GNSS}}(T)$ denote the final positions, and L is the traveled distance (approximately 200 m in our outdoor tests).

The outdoor test site is a feature-sparse open area with only a few trees, poles, and small buildings. In this environment, the UAV autonomously flies about 200 m between a manually specified start and goal. Table II compares AUG against ***SUPER*** and ***EGO-Planner*** under the same platform and outdoor flight setup, reporting the mean RPE_{GNSS} over the 10 outdoor trials only. AUG maintains bounded drift with $\text{RPE}_{\text{GNSS}} < 4\,\%$, whereas ***SUPER*** and ***EGO-Planner*** exhibit substantially larger drift and fail under the same setup.

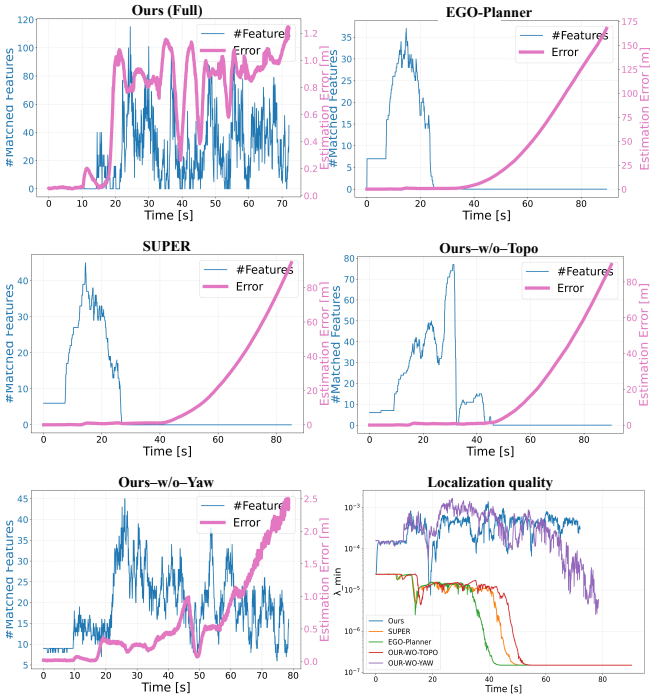

Fig. 8. Time evolution of the matched feature count N_{feat} (blue) and localization error (magenta) for representative runs in Scene-2 under different planners. The first five panels show the temporal evolution of feature availability and estimation error for AUG, *EGO-Planner*, *SUPER*, Ours-w/o-Topo, and Ours-w/o-Yaw, respectively. The bottom-right panel compares the corresponding $\lambda_{\min}$ over time, which reflects LIO update conditioning: values closer to zero indicate stronger degeneracy and typically poorer localization. Methods that enter the central open region rapidly lose matched features, their $\lambda_{\min}$ drops toward zero, and the localization error grows quickly, whereas AUG maintains better-conditioned updates and bounded error.

TABLE II

REAL-WORLD BASELINE COMPARISON UNDER GNSS-REFERENCED EVALUATION

Method	$\text{RPE}_{\text{GNSS}}(\%)$	Memory (GB)	Status
AUG (Ours)	**2.24**	**0.92**	**Success**
SUPER [7]	10.13	0.61	Failure
EGO-Planner [5]	16.80	2.46	Failure

VI. CONCLUSION

We presented AUG, a perception-aware and topology-guided planning framework for long-range UAV navigation in GNSS-denied, feature-sparse environments. By combining an incremental free-space topological graph for long-horizon guidance, a robocentric *ROG-map+MINCO* local planner for real-time replanning, and a visibility-constrained yaw planner for maintaining LIO conditioning, AUG enables scalable and reliable navigation without a global voxel/ESDF map.

Kilometre-scale simulations and real-world flights show that AUG achieves reliable navigation with low drift, near-constant memory, and stable fully onboard operation. Future work will incorporate online uncertainty into graph expansion and yaw planning for more dynamic and occluded environments.

REFERENCES

[1] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, 2021.

[2] G. Huang, A. Mourikis, and S. Roumeliotis, "Observability-based rules for designing consistent ekf slam estimators," *I. J. Robotic Res.*, vol. 29, pp. 502–528, 04 2010.

[3] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, pp. 2520–2525, 2011.

[4] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3529–3536, 2019.

[5] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao, "Ego-planner: An esdf-free gradient-based local planner for quadrotors," *IEEE*, 2021.

[6] Y. Ren, Y. Cai, F. Zhu, S. Liang, and F. Zhang, "Rog-map: An efficient robocentric occupancy grid map for large-scene and high-resolution lidar-based motion planning," *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8119–8125, 2024.

[7] Y. Ren, F. Zhu, G. Lu, Y. Cai, L. Yin, F. Kong, J. Lin, N. Chen, and F. Zhang, "Safety-assured high-speed navigation for mavs," *Science Robotics*, vol. 10, no. 98, 2025.

[8] Z. Zhang and D. Scaramuzza, "Perception-aware receding horizon navigation for mavs," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2534–2541, 2018.

[9] K. Chai, L. Xu, Q. Wang, C. Xu, P. Yin, and F. Gao, "Lf-3pm: a lidar-based framework for perception-aware planning with perturbation-induced metric," pp. 5372–5379, 10 2024.

[10] B. Zhou, J. Pan, F. Gao, and S. Shen, "Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1992–2009, 2021.

[11] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259–3278, 2022.

[12] Y. Zhang, X. Chen, C. Feng, B. Zhou, and S. Shen, "Falcon: Fast autonomous aerial exploration using coverage path guidance," *IEEE Transactions on Robotics*, vol. 41, pp. 1365–1385, 2025.

[13] S. Geng, Z. Ning, F. Zhang, and B. Zhou, "Epic: A lightweight lidar-based aav exploration framework for large-scale scenarios," *IEEE Robotics and Automation Letters*, vol. 10, no. 5, pp. 5090–5097, 2025.

[14] C. Yu, Z. Lu, J. Mei, and B. Zhou, "Perception-aware planning for quadrotor flight in unknown and feature-limited environments," 2025.

[15] X. Chen, Y. Zhang, B. Zhou, and S. Shen, "Apace: Agile and perception-aware trajectory generation for quadrotor flights," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 17858–17864, 2024.

[16] Z. Zhang and D. Scaramuzza, "Fisher information field: an efficient and differentiable map for perception-aware planning," *ArXiv*, vol. abs/2008.03324, 2020.

[17] L. Bartolomei, L. Teixeira, and M. Chli, "Perception-aware path planning for uavs using semantic segmentation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[18] G. Costante, C. Forster, J. Delmerico, P. Valigi, and D. Scaramuzza, "Perception-aware path planning," *IEEE Transactions on Robotics*, 2016.

[19] R. Takemura and G. Ishigami, "Perception-aware receding horizon path planning for uavs with lidar-based slam," in *2022 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pp. 1–8, 2022.

[20] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *International Symposium of Robotics Research*, 2016.

[21] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[22] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7244–7251, 2018.