International Journal of Intelligent Robotics and Applications (2025) 9:1743–1754
https://doi.org/10.1007/s41315-025-00462-2

REGULAR PAPER

Vision-based autonomous navigation for quadrotor UAVs in unknown and constrained environments

Zhanhao Hu[1] · Muhammad Zohaib Butt[1] · Nazri Nasir[2] · Mohd Badrul Salleh[1]

Received: 18 January 2025 / Accepted: 1 May 2025 / Published online: 23 May 2025
© The Author(s) 2025

Abstract
The demand for autonomous UAV navigation in complex environments is growing rapidly. However, most existing studies focus on simulations, lacking real-world validation under challenging conditions. This study systematically evaluates a vision-based navigation framework combining VINS-Mono for localization and Ego-Planner for path planning in GPS-denied, low-light, high-obstacle-density environments. Unlike traditional approaches that struggle with UAV high-frequency motion and dynamic constraints, VINS-Mono provides accurate high-frequency positioning via visual-inertial fusion. At the same time, Ego-Planner enables smooth, dynamically feasible real-time trajectory generation. Real-world experiments on three randomly generated maps (minimum obstacle spacing: 0.8 m (twice UAV wheelbase); obstacle density: 0.72 obstacles/m^2) achieved a minimum 70% success rate, with 87% of trajectories smoother than A* in simulations. Results demonstrate the framework's robustness, though challenges such as field-of-view constraints and sensor-induced localization errors affecting replanning remain. This study presents a real-world evaluation of vision-based UAV navigation in constrained environments, offering insights into its feasibility and limitations. The findings contribute to improving trajectory optimization and UAV autonomy in constrained GPS-denied scenarios.

Keywords Vision-based UAV autonomous navigation · Unknown constrained environments · Real-world validation

1 Introduction

Quadrotor UAVs have seen significant advancements in aerial photography, reconnaissance, and infrastructure inspection (Erdelj et al. 2019). In particular, miniaturized UAVs have become a key research trend with their compact size and high manoeuvrability (Heng et al. 2011). However, navigation in constrained environments with low lighting and high obstacle density (e.g., underground mines, tunnels, and disaster sites) remains highly challenging see Fig. 1. These environments often lack GPS signals, have poor lighting conditions, and feature complex or dense obstacles, making it difficult for conventional navigation methods to ensure stability and safety. Additionally, according to Butt et al. (2024), human errors account for 80% of UAV accidents, with approximately 53% attributed to pilot mistakes. Therefore, research on autonomous UAV technologies with reliable information processing and planning capabilities can effectively reduce risks. However, existing autonomous navigation systems have rarely been tested in such constrained environments.

Autonomous UAV systems typically rely on various sensors for perception and path planning, including ultrasonic sensors, LiDAR, and cameras. However, limitations such as high sensor weight, low accuracy, or restricted applicability make certain methods unsuitable for ensuring agility and safety in miniaturized UAVs (Sanfourche et al. 2012). Additionally, miniaturized UAVs experience rapid angular variations during flight, making 2D sensing methods less effective in complex environments. While Petrlík et al. (2020) have explored tunnel navigation using LiDAR in low-light conditions, they incorporate additional lighting and operate in environments with minimal obstacles. In contrast, the method based on visual sensors (see Fig. 2) can provide rich environmental information, is highly scalable, and is the

✉ Nazri Nasir
mnazrimnasir@utm.my

[1] Department of Aeronautics, Automotive and Ocean Engineering, Faculty of Mechanical Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia

[2] UTM Aerolab, Institute for Vehicle Systems and Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia

mainstream perception technology for autonomous UAVs. However, efficiently processing large volumes of visual data remains a significant challenge (Xie et al. 2021).

Visual Odometry (VO) and Visual-Inertial Odometry (VIO) are widely studied for UAV positioning and tracking. ORB-SLAM2 (Mur-Artal and Tardós 2017), a well-known lightweight VO algorithm, struggles in low-light conditions, leading to frequent tracking failures. In contrast, VINS-Mono (Qin et al. 2018), the VIO algorithm used in this study, integrates visual and inertial data to provide high-frequency positioning. When visual tracking fails, it compensates using Inertial Measurement Unit (IMU) data, ensuring higher stability and adaptability. Beyond localization, modern UAV path planning emphasizes trajectory smoothness and adherence to dynamic constraints rather than simple reachability (Zafar and Mohanta 2018). Ego-Planner (Zhou et al. 2020a, b), adopted in this study, optimizes path search efficiency by eliminating Euclidean Signed Distance Field (ESDF) computation, which accounts for 70% of processing time in conventional methods as explained by Usenko et al. (2017). The ESDF-free enables real-time performance in complex environments, making Ego-Planner a state-of-the-art solution. However, its practical adaptability in highly constrained, low-light environments with dense obstacles remains to be further examined.

This study systematically evaluates the real-world performance of VINS-Mono and Ego-Planner in GPS-denied, low-light, and high-obstacle-density environments, which present significant challenges and are often underexplored in existing research. Unlike prior studies, which primarily rely on simulations, we conduct real-world experiments to assess the robustness and effectiveness of these state-of-the-art methods under extreme conditions. We quantify the reliability and efficiency of VIO-based localization and trajectory optimization in highly constrained environments through extensive empirical analysis. The findings of this study provide critical insights into the limitations of current approaches and serve as a foundation for improving UAV autonomy and navigation reliability in real-world deployments. This paper is organized as follows: Section 2 reviews localization, path planning, and optimization methods. Section 3 details the experimental setup and evaluation. Section 4 presents results and discussion, while Sect. 5 concludes with future work.

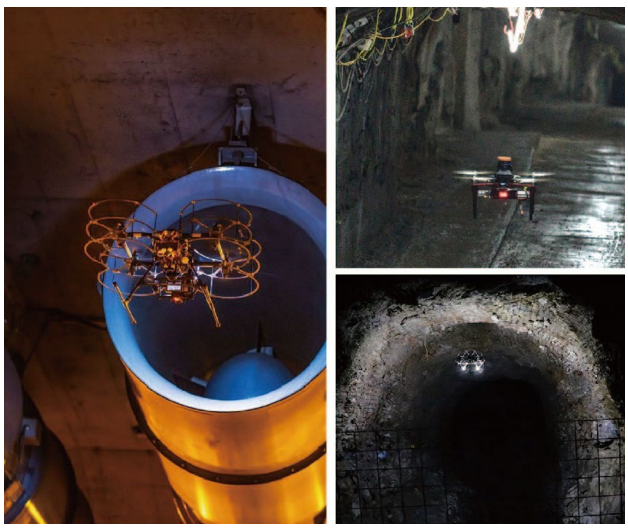


Fig. 1 Schematic diagram of UAV exploring a constrained environment (Petrlík et al. 2020)



Fig. 2 Five mainstream types of vision sensors

2 Related works

Exploring autonomous UAV navigation remains a crucial research challenge, requiring seamless localization, trajectory generation, and optimization integration. Among these, localization forms the foundation for UAV stability and attitude awareness in constrained and unknown environments. Trajectory generation and optimization are indispensable for ensuring safe, efficient, and feasible autonomous navigation. This chapter reviews key prior works in the three core modules underpinning UAV autonomous navigation: UAV localization and tracking (Sect. 2.1), trajectory generation (Sect. 2.2), and optimization (Sect. 2.3).

2.1 UAV visual positioning and tracking

The process of self-localization and mapping using only visual sensors is known as simultaneous localization and mapping (SLAM). Within the visual SLAM (VSLAM) framework, VO and VIO are two primary localization techniques. VO estimates camera motion solely based on visual information, whereas VIO enhances the estimation by integrating IMU measurements to improve robustness and accuracy. VSLAM further incorporates loop closure detection and global optimization to achieve long-term, scalable

localization (Taketomi et al. 2017) and has attracted significant research attention. Existing studies on visual odometry can be broadly categorized into three approaches: indirect methods, direct methods, and multi-sensor fusion methods, where both indirect and direct methods fall under VO, while multi-sensor fusion methods typically employ VIO techniques.

Indirect methods estimate the motion between consecutive frames of UAV flight by identifying and tracking specific regions in visual images (e.g., points, lines, and planes). Early feature extraction methods used corner detection techniques, with PTAM (Klein and Murray 2007) being a representative application of corner detection. Meanwhile, feature detection methods have emerged, offering more stable local image features, with ORB-SLAM2 making significant progress using feature point detection methods.

However, indirect methods perform poorly in scenes with sparse textures and limited environmental features. In contrast, direct methods use the image's overall pixel intensity (including edge information and intensity variations) to extract information directly for optimization (Forster et al. 2016). Studies based on photometric consistency have notably improved localization in low-texture environments (Kerl et al. 2013).

Despite these advancements, both indirect and direct methods have limitations. Indirect methods require substantial computational resources for feature matching and processing, while direct methods demand good initial guesses and are less robust. To solve those issues, Lynen et al. (2013) proposed a loosely coupled sensor fusion approach, treating the IMU as an independent module to assist vision-based pose estimation. Given that IMU data acquisition is significantly faster than cameras, effectively leveraging IMU pre-integration to reduce errors is crucial. VINS-Mono utilized a tightly coupled approach, combining IMU pre-integration with visual observations in a joint optimization framework, significantly improving localization accuracy and robustness.

2.2 Trajectory generation

After solving the localization problem, UAVs must find a safe path from the starting point to the destination. Extensive studies have been conducted on path generation and optimization for quadrotor UAVs, but this paper only provides a limited overview of the representative approaches. Sampling-based localization algorithms, such as Probabilistic Road Map (PRM) and Rapidly-exploring Random Tree (RRT), explained by Madridano et al. (2021) and Rocha et al. (2023), both used random point sampling to identify feasible paths in the environment quickly. These algorithms are particularly suitable for real-time path adjustment in complex and dense environments, a core task of local trajectory optimization. However, their randomness may lead to unstable or low-precision paths, creating challenges for subsequent optimization.

Moreover, optimization-based methods, such as Ant Colony Optimization (ACO) used for multiple UAVs by Ali et al. (2021) and Particle Swarm Optimization (PSO) applied by Zhang and Zhang (2022), simulate natural group behaviours to exhibit strong global search capabilities in high-dimensional and complex environments, particularly when computational resources are not a constraint. However, these methods often rely on expert parameter tuning, limiting their suitability for real-time tasks. In contrast, classic global search algorithms, such as Dijkstra and A*, generate flight strategies through global modelling. These methods produce reliable paths under limited computational resources but rely heavily on static environmental assumptions (Xu et al. 2019). Consequently, the generated paths often aim for the shortest distance and tend to stay close to obstacles, which may lack flexibility.

2.3 Optimization

The primary goal of trajectory optimization is to refine the initial path using various constraints (e.g., dynamic constraints) to enhance flight smoothness and meet specific requirements (e.g., low energy consumption). Optimization methods can be categorized into two levels: geometric constraints and parametric constraints.

Geometric constraints focus on smoothing the trajectory derived from the initial path, typically represented using piecewise polynomials, spline functions, or discrete point sets. For example, the Mini-Snap method (Mellinger and Kumar 2011) represents trajectories as piecewise polynomials and achieves smoothing by solving a quadratic programming (QP) problem. Building on this, Richter et al. (2016) introduced dynamic programming to add waypoints for further trajectory optimization iteratively. Additionally, Gao et al. (2019) proposed constructing convex corridors (e.g., spheres or cubes) in complex environments to generate safe and dynamically feasible trajectories using Bézier curves with segment constraints. However, geometric methods often rely on simple heuristic time allocation strategies, which may significantly reduce trajectory quality.

Parametric constraints formalize the trajectory as a nonlinear optimization problem by imposing additional constraints on UAV dynamics (e.g., velocity and acceleration) while incorporating gradient or time-optimization terms. These approaches minimize the cost function to avoid abrupt velocity or acceleration changes, ensuring flight continuity. Oleynikova et al. (2016) used gradient-based methods to directly parameterize time as continuous piecewise polynomial trajectories, reducing numerical differentiation errors. However, this method requires continuous iteration and has

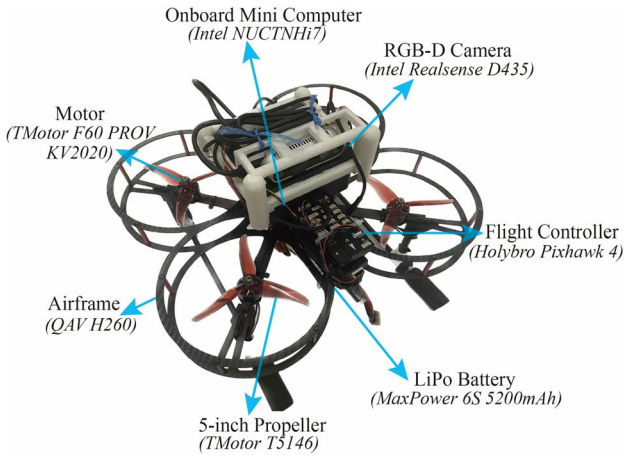


Fig. 3 The assembled UAV used in the experiment

difficulty meeting real-time demands, especially when trajectory generation costs are high. To overcome this, Usenko et al. (2017) leveraged the convex hull properties of B-splines to parameterize trajectories as uniform B-spline curves demonstrated that the locality and continuity of B-splines significantly improve optimization efficiency and robustness in replanning tasks. Guo et al. (2020) introduced a varying predictive duration method, optimizing numerical iterations to accelerate initial solution acquisition and enhance control performance. Furthermore, Zhang et al. (2021) used extended uniform B-splines to avoid collisions of cost functions on multiple UAVs.

Furthermore, Ego-Planner further integrated the convex corridor geometry method with a B-spline parameter optimization framework. It employs gradient-based optimization to enforce dynamic feasibility constraints on the trajectory, followed by time reallocation and local trajectory replanning. Its modular design significantly reduced planning costs in complex environments while ensuring real-time performance requirements.

3 Material and methodology

This chapter outlines the materials and methodologies employed in this study to evaluate UAV navigation and trajectory planning in real-world constrained unknown environments. The research comprehensively explains the UAV's hardware configuration, experimental environment, parameter details, algorithm settings, and evaluation methods.

3.1 UAV composition

We employed a self-assembled UAV (see Fig. 3) designed for experiments in constrained unknown environments. The primary components of the UAV include an integrated carbon fiber protective frame (with a horizontal wheelbase of 0.4 m), which is capable of absorbing impact forces during potential collisions to safeguard other hardware components. The propulsion system features 5-inch three-blade propellers (model T5146, manufactured by TMotor) and high-KV small brushless motors (model F60 PROV, KV rating of 2020, manufactured by TMotor). Power is supplied by a 22.2 V battery (model Maxpower 5200 mAh 6S1P 60 C LiPo battery). The flight control system utilizes a Holybro Pixhawk 4 flight controller with a built-in multi-sensor fusion module, enabling precise and real-time control for flight missions.

To meet the requirements of real-time perception in unknown environments, the UAV is equipped with an advanced RGB-D camera (model Intel RealSense D435). The RGB module offers a resolution of 1920 × 1080 at a frame rate of 30 fps with a FoV of 69° × 42° (horizontal × vertical). The depth module provides a broader FoV of 87° × 58° and supports a higher frame rate of 90 fps, achieving a depth accuracy of over 98% within a 2-m range. The camera is mounted on the top of the onboard computer using a buffer pad and a 3D-printed protective frame, ensuring its FoV remains unobstructed by the UAV structure. This setup also maintains stable perception performance during attitude adjustments.

The UAV's mission processing module runs on a compact onboard computer (model Intel Tiger Canyon NUC11 TNHi7, dimensions 117 mm × 112 mm × 54 mm). It has an Intel i7-1165G7 processor, Iris XE graphics card, 8 GB RAM, and 256 GB SSD storage. This onboard computer handles multiple critical modules, including GPS-denied self-localization, state estimation, mapping, path planning, optimization, and real-time control. Finally, the total weight of the UAV is 2.24 kg.

3.2 Environment setup

In the path planning algorithm used in this study, the UAV is modelled as a point mass with no physical volume. To account for the UAV's actual physical dimensions during path planning, the obstacles are expanded with virtual boundaries (see Fig. 4), similar to the safety region strategy described by Al-Kaff et al. (2017). Where h_w represents the UAV's horizontal wheelbase (0.4 m) and the minimum safe distance is set to 2 h_w (0.8 m). Durning real-world experiments, each side of the obstacle is inflated by 0.5 h_w (0.2 m), forming medium gray block regions. Since the experimental trajectories naturally align within the 2 h_w distance from the obstacle, and the A* algorithm tends to generate paths close to obstacles (Zhou et al. 2020a, b). The obstacles in simulations are further inflated by an additional 0.15 m, creating light gray regions with a gap distance of 0.1 m.

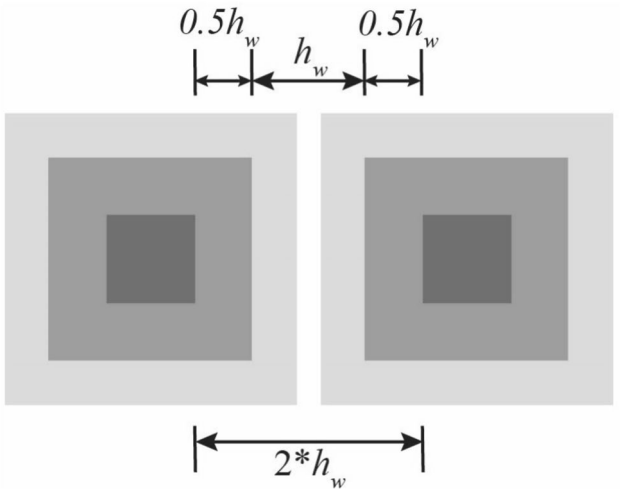


Fig. 4 Illustration of obstacle visual inflation. The dark gray square represents the actual size of the obstacle; the medium gray square indicates the inflated obstacle size (0.5 h_w) during UAV flight; the light gray square shows the inflated size (further 3/8 h_w each) of the obstacle used in A* simulation, to force A* trajectory is in the middle of the gap

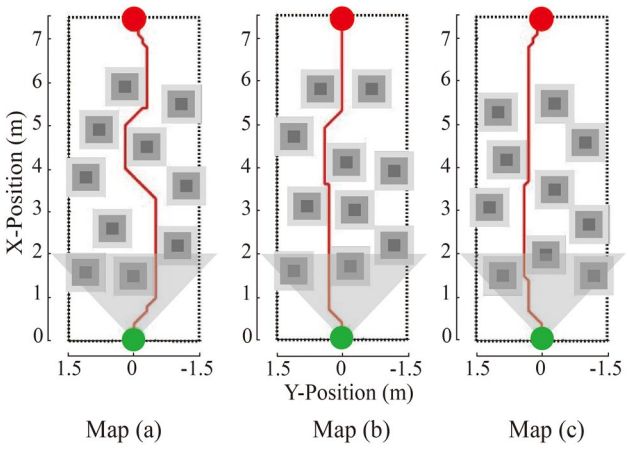


Fig. 5 Randomly generated Map (**a**), (**b**) and (**c**) with the corresponding actual obstacle map illustrations. The map boundaries are marked with black dashed lines, the starting and ending points are represented by green and red dots, respectively, gray triangles indicate the UAV's FoV, and the red line represents the trajectory simulated using the A* shortest path length algorithm consistent with the source code

The experiment utilized 10 column obstacles with low-texture features (made of foam material, dimensions: 0.3 m × 0.3 m × 1.5 m). The minimum distance between obstacles is set at 0.8 m (twice h_w). The experimental environment spans an area of 3 m × 7.5 m, and obstacles are randomly distributed within a width of 3 m and a length of 1.5 m to 6 m (see Fig. 5). The starting and destination points are set at (0.0) and 7.5 m further in the x-axis from the starting, respectively. The lighting conditions of the experimental scene are maintained at a constant low illumination level of 6 lx.

Under identical environmental conditions, we used the same pathfinding algorithm adopted in Ego-Planner to perform a global shortest-path simulation based on the A* algorithm. The obstacle inflation settings in the simulation were consistent with the virtual boundary settings to ensure the generated trajectories were feasible for real-world flights. However, it should be noted that the simulation was solely conducted under ideal conditions (e.g., without hardware noise or state estimation errors) to validate the theoretical shortest path and analyze the distribution of the generated waypoints. All other data were obtained from actual flights, with the simulation results serving only as a baseline for comparison with the actual flight trajectories.

3.3 Implementation details

All experimental software was conducted on an Ubuntu 20.04 Long-Term Support (LTS) operating system, with the Robot Operating System (ROS) as the primary software framework. ROS, a modular and node-based robotics software architecture, facilitates efficient message publishing and subscription (Quigley et al. 2009). Multiple ROS nodes were established to handle visual perception, path planning, and state feedback tasks. ROS also provides built-in tools, such as RVIZ for sensor data visualization and ROS Bag for data recording and analysis, ensuring seamless inter-module communication and robust and stable software support throughout the experiments.

$$Obstacle\ Density = \frac{Number_{obstacle}}{Area_{region}} \tag{1}$$

The obstacle density in our experiments is 0.72 $obstacle/m^2$ (see Eq. (1)). The configuration in our experiments exceeds the density reported by Ding et al. (2019) by more than threefold and is approximately 0.5 times higher than the original Ego-Planner configuration. It also approaches the maximum density used in simulated environments, as reported by Zhang et al. (2024).

In the actual flight experiment, to meet the high-frequency planning requirements in such a constrained environment, we increased the IMU data publishing rate of the flight controller to 200 Hz, significantly improving UAV responsiveness. Additionally, the UAV's maximum velocity and acceleration were capped at ±0.4 m/s and ±0.4 m/s^2, respectively, matching its horizontal wheelbase to ensure the stability and safety of algorithm testing. The camera resolution was set to 0.1 m, and the visual inflation distance was fixed at 0.5 h_w (0.2 m). Additionally, the active infrared light function was disabled since the VINS-Mono algorithm

fundamentally relies on visual modules. Under low-light conditions, active infrared light can create fixed spots on the lens, ultimately affecting the algorithm's stability.

Preliminary experiments demonstrated that under these configurations, the UAV achieved a success rate of approximately 60% in navigating through two consecutive rows of obstacles with spacing 2 h_w. This setup balances rigorous algorithm testing with experimental safety while ensuring data remains generalizable by avoiding excessively low success rates.

3.4 Algorithm Framework

This study employs state-of-the-art algorithms to evaluate the performance of UAV autonomous navigation in constrained environments, focusing on self-localization, trajectory generation, tracking controller, and optimization. The following sections will discuss the technical details of these four key aspects.

3.4.1 Environment perception

The experiment utilized the VIS-Mono algorithm as a state localization and tracking feedback module. The algorithm processes data from an RGB-D camera and an IMU, achieving UAV self-localization and trajectory tracking through nonlinear optimization. Results affected by localization errors inherent to the algorithm were actively excluded from our experiment. Furthermore, as this localization algorithm has been widely studied, detailed discussions are omitted here; readers may refer to Qin et al. (2018) for further information.

However, it is worth noting that VINS-Mono uses a feature-based optical flow method for tracking and matching in this experiment. Due to the disparity in hardware acquisition frequencies (30 Hz for the camera vs. 200 Hz for the IMU), the pre-integration fusion and total fusion frequency of hardware data emerges as a critical aspect requiring attention (see Fig. 6).

3.4.2 Path generation

In implementing Ego-Planner, the core idea of path generation is to push the pre-generated trajectory away from obstacles to ensure safety. The planning task is divided into path pre-search and collision avoidance. During the experiment, an initial uniform B-spline curve is first generated to meet only the terminal constraints without considering obstacle avoidance (with control points placed approximately 0.3 m apart by default). Usenko et al. (2017) showed B-splines with convex hull properties play an essential role in the continuity and locality of trajectory replanning. Then,

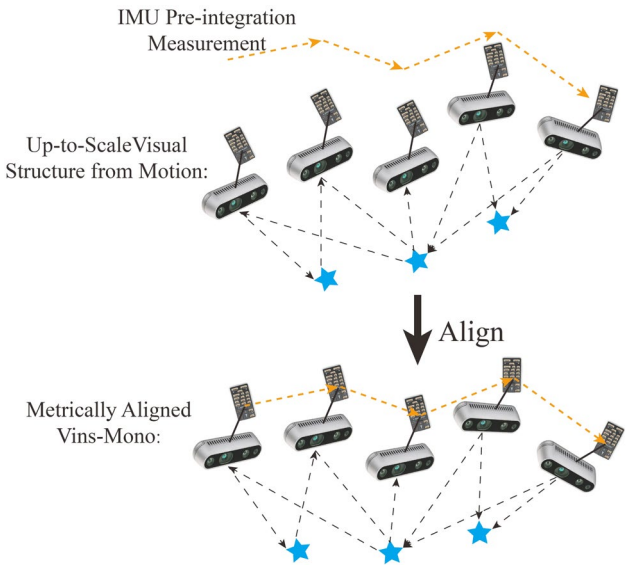


Fig. 6 Schematic diagram of the alignment of up-to-scale visual structure and IMU pre-integration matching. The yellow dotted line is the data line measured by IMU pre-integration, the black dotted line is the up-to-scale visual structure measurement, and the blue star is the visual feature, and there is a fixed spatial deviation between the camera and IMU

in iterative processing, the algorithm detects whether any control points lie within obstacles in the initial trajectory segments.

If a control point is found to be inside an obstacle, the velocity direction at that control point is computed and used as the normal vector, R_i. A line segment perpendicular to this normal vector is determined as l, with the nearest obstacle-free points before and after the control point serving as the start and end points, Q_{i-1} and Q_{i+1} respectively. A standard A* search algorithm generates a new collision-free trajectory for this segment. The line l then extended to intersect the trajectory generated by A*, effectively moving the control point Q_i inside the obstacle onto the A*-generated trajectory as a point p_{ij} (see Fig. 7). Due to the characteristic of A* to generate paths that naturally lie near the obstacle surface, the control points inside the obstacle can be directly adjusted to the trajectory generated by A* and considered as just outside the obstacle, without using computational resources to further search for obstacle planes.

$$d_{ij} = \left(Q_{ij} - p_{ij}\right) \cdot v_{ij} \tag{2}$$

Since the UAV is not a point without volume in reality, the generated trajectory must have a safe flight corridor to limit the UAV always being inside (Mellinger and Kumar 2011), which means directly placing control points onto the trajectory approximately equal to the obstacle surface

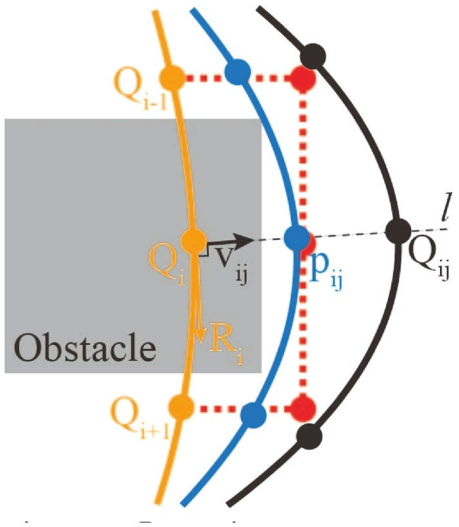


Fig. 7 Schematic diagram of initial trajectory generation with control points inside the obstacle. The yellow curve represents the initial trajectory without considering obstacles, the red segment is the collision-free trajectory generated by A*, the blue curve indicates the trajectory that the control point pushed onto the obstacle's outer surface, and the black curve represents the trajectory after applying the desired distance formula as Eq. (2). The solid circles of different colors are the control points of different curves

would be infeasible in actual flight. Therefore, a further adjustment is made based on the expected obstacle distance (see Eq. (2)). This formula pushes the trajectory further away from the obstacle, where p_{ij} represents the control point pushed onto A*-generated trajectory, Q_{ij} is the desired position to push the control point further, and v_{ij} is a fixed repulsive direction vector.

3.4.3 Controller model

In this experiment, we employ a cascade PID controller for trajectory tracking and a nonlinear thrust model to account for the UAV's propulsion characteristics. The control strategy consists of a position control loop, which minimizes the error between the current and desired positions, and a velocity control loop, which minimizes the error between the current and desired velocities. This cascaded structure enables precise UAV trajectory tracking, requiring only the desired position and velocity as inputs to guide the UAV stably toward its target (Idres et al. 2017). The position control loops can be expressed in Eq. (3), and the velocity control loop is shown in Eq. (4), where $\dot{p} = p_{des} - p_{act}$ represents the position error and $\dot{v} = \left(v_{des} - v_{act}\right)$ represents the velocity error. The cascade PID gains K for both control loops are set to 1.85 to achieve stable flight performance. Compared to SE(3) control and model predictive control (MPC), this approach offers more straightforward parameter tuning and lower computational complexity (Mo and Farid 2019). Given that this experiment is conducted in a static environment without high-acceleration manoeuvres, the response lag of this method is negligible.

$$v_{des} = K_{p,p}\dot{p} + K_{i,p}\int \dot{p}dt + K_{d,p}\frac{d}{dt}\dot{p} \tag{3}$$

$$a_{des} = K_{p,v}\dot{v} + K_{i,v}\int \dot{v}dt + K_{d,v}\frac{d}{dt}\dot{v} \tag{4}$$

$$F = K_1 \cdot V^{K_2} \cdot (K_3 \cdot u^2 + (1 - K_3) \cdot u) \tag{5}$$

The UAV thrust allocation is based on an empirically derived nonlinear thrust model, mapping the normalized thrust command u (range: 0–1) to actual thrust F, as shown in Eq. (5). Where V represents the battery voltage, and the thrust formulation follows an empirically fitted model referenced from the PX4[1] manual. The parameters K_1, K_2, K_3 are set to 0.8,1.45, and 0.48 in our experiment, respectively, representing the baseline thrust coefficient, voltage influence exponent, and nonlinear thrust ratio. The normalized thrust command u is 0.39, indicating that the UAV maintains a stable hover at 39% throttle. This dynamic model effectively compensates for battery voltage variations, ensuring consistent flight performance under different operating conditions.

3.4.4 Optimization

The initial path generation module only ensures reachability. In contrast, the optimization module refines the trajectory by generating control points that minimize the penalty values, resulting in a smoother and safer UAV motion trajectory. Based on the differential flatness property of quadrotor UAVs followed by Mellinger and Kumar (2011), the optimization problem is defined as shown in Eq. (6), where $\mathcal{J}_s$ represents the smoothness penalty, $\mathcal{J}_c$ represents the collision penalty, $\mathcal{J}_d$ represents the dynamic feasibility penalty, and λ denotes the weight of the penalty terms. The path generation module uses a standard B-spline curves formula with a uniform distribution of time intervals. By taking advantage of the convex hull property of B-splines, the smoothness cost $\mathcal{J}_s$ of the control points could be effectively reduced by minimizing the time integral of the squared norms of the acceleration and jerk.

$$\min_{Q} \mathcal{J} = \lambda_s \cdot \mathcal{J}_s + \lambda_c \cdot \mathcal{J}_c + \lambda_d \cdot \mathcal{J}_d \tag{6}$$

The collision term ensures the control points are kept away from obstacles. In the experiment, a twice continuously differentiable penalty function j_c is directly constructed,

[1] https://docs.px4.io/main/en/.

which restricts the safety margin while suppressing the slope as it decreases. The cost of all Q_i added together is the collision cost $\mathcal{J}_c$ (see Eq. (7)). Furthermore, the gradient information provides the direction in which the control points change the fastest, with the vector magnitude representing the rate of change in that direction (reflecting the optimization direction of the objective function in space). Instead of using the computationally intensive gradient difference-solving method of ESDF like Oleynikova et al. (2016) and Zhou et al. (2020a, b), the partial derivative of $\mathcal{J}_c$ is directly computed to obtain continuous gradient information.

$$\mathcal{J}_c = \sum_{i=1}^{N_c} j_c(Q_i) \tag{7}$$

Regarding dynamic feasibility, the UAV is constrained by physical limits on velocity, acceleration, and jerk (Gasparetto et al. 2015). Therefore, the higher-order derivatives in each dimension must not exceed the predefined thresholds, and these three physical limitations are enforced at the control points to control the entire trajectory equally (see Eq. (8)). Where w_v, w_a and w_j represent the weights of each term, and F is a smooth, second-order differentiable metric function based on the higher-order derivatives of the control points.

$$\mathcal{J}_d = \sum_{i=1}^{N_c-1} w_v F(V_i) + \sum_{i=1}^{N_c-2} w_a F(A_i) + \sum_{i=1}^{N_c-3} w_j F(J_i) \tag{8}$$

After obtaining the trajectory with the minimum penalty cost, time reallocation ensures feasibility while preserving smoothness. Since the complete trajectory information is not available before optimization, the initial assumption of uniform time distribution may not meet the physical constraints. In such cases, Gao et al. (2020) explained that some segments may exceed derivative limits, leading to iterative extensions of time intervals for those nodes, which can cause discontinuities in higher-order derivatives along the trajectory.

Following Ding et al. (2019), the algorithm also redistributes the time allocation for the collision-free trajectory before regenerating the uniform B-spline trajectory. The cost of the updated trajectory is then re-evaluated, replacing the collision cost with the curve fitting cost. To maintain feasibility, smoothness penalties are relaxed along the velocity direction, while collision constraints remain stringent in the perpendicular direction.

3.5 Evaluation standard

To ensure the comparability of experimental results, we conducted 10 actual flight experiments each on three randomly generated maps, adhering to identical map generation rules and environmental parameters while excluding planning failures caused by localization drift. After collecting the experimental data, we will analyze and evaluate algorithm performance based on the following criteria:

First, we calculated the success rate of the experiments. Since this algorithm has yet to undergo extensive research under constrained, unknown real-world conditions, and preliminary test results indicate some uncertainties, success rate analysis is deemed particularly essential. Second, hardware misread errors in real-world experiments are inevitable, leading to potential deviations from the ideal path and occasional stops for replanning during flight. These occurrences

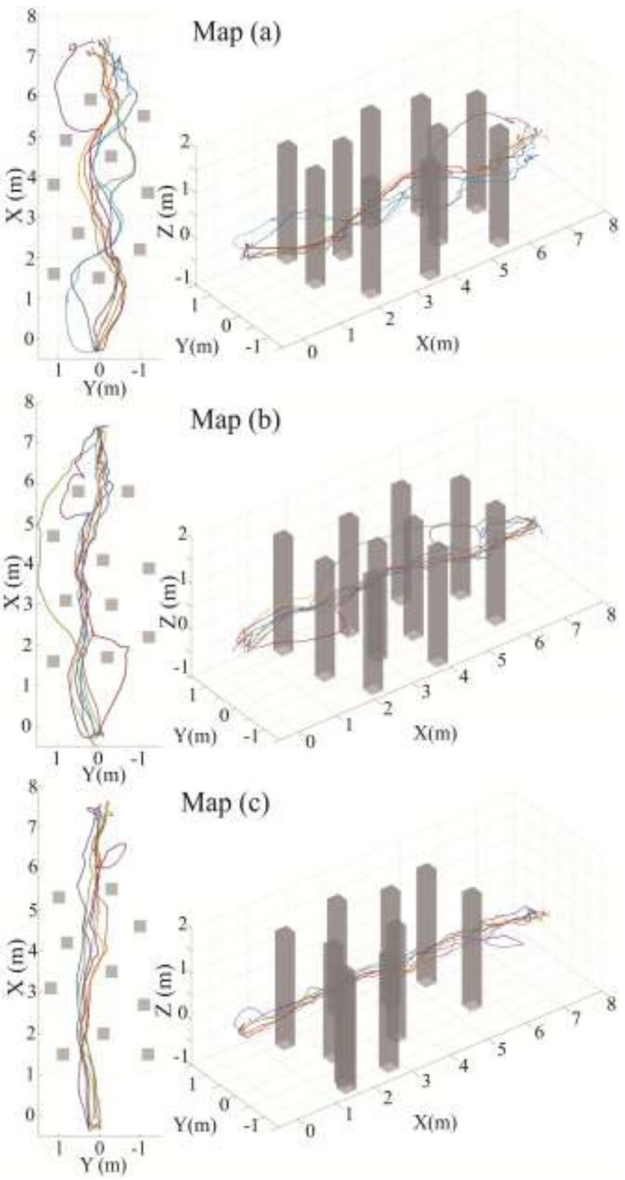


Fig. 8 The flight trajectories of all successful paths from start to end in the high-density obstacle environments of Map (**a**), Map (**b**), and Map (**c**)

were thoroughly recorded and analyzed to comprehensively evaluate the algorithm's applicability and robustness.

Second, the shortest theoretical trajectory length obtained through simulation algorithms serves as a baseline. Similar to Oleynikova et al. (2016), we compared the distance of the trajectory generated during actual flights with the extremum range of the shortest path.

$$\theta_i = cos^{-1}\left(\frac{\vec{v}_{i-1} \cdot \vec{v}_i}{\| \vec{v}_{i-1} \| \cdot \| \vec{v}_i \|}\right) \tag{9}$$

Furthermore, we performed a time-independent analysis of the local trajectory smoothness (see Eq. (9)) to evaluate its continuity and executability. Where $\vec{v}_i$ is the direction vector of the i^{th} path, and $\left\|\vec{v}_i\right\|$ is the Euclidean norm of the vector. By calculating the angular θ_i changes between consecutive segments at each turning point (acceleration direction change), we derive an evaluation metric of the maximum angular variation during flight. This quantitative measure of path smoothness provides valuable insights for optimizing flight control and path planning.

4 Results and discussion

After completing the experiments, the success rates of flight tests that achieved complete path planning in the environments of Map (**a**), Map (**b**), and Map (**c**) were 80, 80, and 70%, respectively. Overall, the UAV exhibited a relatively high success rate. The flight trajectories of all successful experiments are shown in Fig. 8, and the primary analyzed data are shown in Table 1. While the path lengths exceeded the shortest paths generated by the A* algorithm, the trajectory smoothness outperformed it. Specifically, 75, 100, and 86% of the trajectories in Maps (**a**–**c**), respectively, exhibited superior smoothness. The test environments'low-light conditions and high obstacle density posed considerable challenges for path planning, especially for vision-based UAVs with a limited FoV. As a result, the successful flight trajectories showed variability in decision-making and sometimes required instances of trajectory replanning.

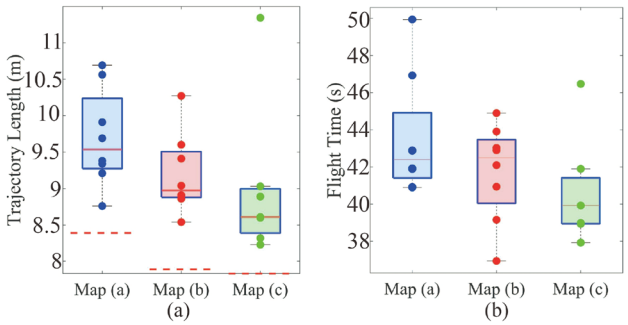


Fig. 9 **a** Distribution of trajectory lengths across the three map sets. **b** Distribution of total flight durations across the three map sets

The constrained environment conditions reduced Signal-to-Noise Ratio (SNR) in captured images, increasing image noise and impairing frame quality. Consequently, the stability of feature point tracking in the VINS-Mono algorithm decreased, and it is easier in scenes with sparse features (low-texture obstacle in the experiment), tracking failures occurred. It also hindered triangulation-based depth estimation to a certain extent, leading to the erroneous detection of virtual obstacles. However, such issues were mitigated by the UAV's replanning mechanism and did not directly cause mission failures.

Observing the shortest path distances corresponding to the three maps, it is apparent that Map (a) posed the most significant challenge to UAV trajectory planning, followed by Map (b), and Map (c) had the lowest difficulty. In all successful experiments, the overall trajectory length decreased with the complexity of the map (see Fig. 9a). The red dashed line represents the shortest path length without noise and errors under ideal conditions.

Furthermore, longer trajectory lengths naturally correspond to longer flight times (see Fig. 9b). Some data points deviate significantly from the median in the boxplot. These deviations are caused by noise in the visual sensor, affecting obstacle detection accuracy. Nevertheless, with the powerful

Table 1 Performance metrics across map (a), (b), and (c)

Experiment data		Map (a)	Map (b)	Map (c)
A*	Avg. distance (m)	8.39	7.89	7.83
	Max. Yaw difference (°)	90	90	45
In Experiment	Avg. distance (m)	9.69	9.19	9.00
	Max. Yaw difference (°)	106.18	71.97	119.97
	Avg. Yaw difference (°)	51.40	42.19	32.46
	Avg. flight time (s)	43.54	41.73	40.59
	Smoother rate (%)	75	100	86
	Success rate (%)	80	80	70
	Trajectory deviation (count)	4	3	0
	Max. replanning times (count)	2	2	1

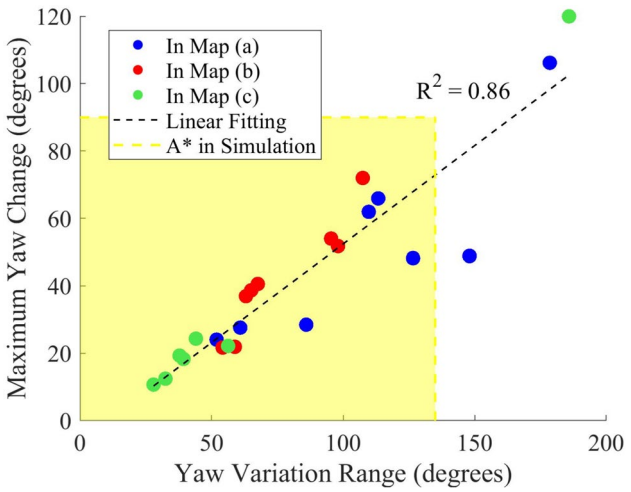


Fig. 10 The maximum yaw angles and yaw variation ranges of the UAV in all successful experiments. Red, blue, and green represent experimental data from maps (**a**), (**b**) and (**c**), respectively, while yellow indicates the maximum yaw variation and total range from A* algorithm simulations

replanning capabilities of the drone planner, we identified feasible alternative paths and completed the mission, but at the cost of unnecessary additional distance and time consumption.

Due to the inherent limitations of the visual FoV, when the target point fell outside the UAV's current FoV, the flight logic prioritized yaw adjustments to align with the target direction before initiating pitch-based forward or backward movement. In the absence of lateral (roll) environmental information, lateral movement was avoided. Following this flight logic, we analyzed the maximum yaw angle changes and the range of yaw variations across successful flight paths to demonstrate overall smoothness (see Fig. 10).

The results reveal that trajectory smoothness varied significantly in the more complex Map (a). Most trajectories demonstrated better smoothness than the shortest path in A* simulation, but some involved significant replanning and large-angle turns. All trajectories show better trajectory smoothness performance for the moderately complex Map (b). In the least complex Map (c), all successful trajectories, except one impacted by obstacle misdetection, surpassed the A* shortest path in smoothness, exhibiting lower yaw angle changes and better trajectory smoothness. The overall trajectory smoothness is better than that of the A* algorithm in the simulation by 87%, and the linear fit value of the maximum yaw angle change and the overall yaw change range is 0.86, representing a high trend accuracy.

Although research on autonomous UAV planning emphasizes only successful experiments, and Zhou et al. (2020a, b) explicitly state that failed tests are meaningless, our study found that understanding the causes of failure remains significant. The primary failure causes can be categorized into hardware and optimization planner configuration limitations.

From the hardware perspective, the UAV's limitations became apparent in environments with high-density obstacles. Such environments amplify slight motion deviations, as tolerance for deviations is minimal. Although it is not within the scope of this paper, we still initially tuned the PID to achieve UAV flight stability. However, minor overshoots or oscillations during flight could cause the UAV's side to collide with obstacles, leading to mission failures.

On the algorithmic side, the failures were primarily attributed to the trajectory optimization process. In the final trajectory optimization step, the algorithm relaxes the cost-fitting constraints for smoothness (along the velocity direction) after temporal redistribution of the path. While this relaxation significantly improves trajectory smoothness, it increases the risk of collision in environments with high obstacle density. Thus, parameters related to this smoothness adjustment require stricter tuning for such complex environments to reduce the risk of failures.

5 Conclusion and future work

This study utilized the VINS-Mono localization algorithm and the Ego-Planner path planning and optimization algorithm to conduct flight tests in real-world low-light, high-obstacle-density environments. The results demonstrated the effectiveness of these methods in constrained scenarios. Experimental findings showed that the actual flight success rates for three randomly generated maps meeting the constraints were above 70%. The successful flight trajectories exhibited notable advantages in path smoothness. Specifically, compared to simulated paths generated by the A* algorithm, 87% of successful flight trajectories demonstrated superior smoothness. However, as the map's complexity increased, the paths'overall smoothness stability decreased, and noise and sparse feature points led to frequent replanning issues.

Despite these promising results, this study has several limitations. First, the restricted field of view (FoV) of the vision-based planning system led to a relatively conservative flight strategy. In dense obstacle environments, ensuring safety often came at the cost of trajectory smoothness, limiting adaptability to dynamic or more uncertain conditions. Second, the optimization process required careful fine-tuning of trajectory smoothness tolerance parameters, particularly in high-density obstacle environments, making the system more sensitive to environmental variations and increasing the likelihood of trajectory failures. Lastly, the reliance on feature point matching for localization and tracking made the system susceptible to noise and mismatches, reducing performance in feature-sparse or visually degraded environments.

Future work will focus on addressing these limitations by enhancing localization robustness and expanding sensor capabilities. Algorithmic improvements, such as noise reduction

modules or more robust feature extraction techniques, could mitigate the effects of visual disturbances and improve trajectory stability. Additionally, integrating additional sensors, such as wide-angle cameras or LiDAR, could expand the UAV's FoV, enabling real-time acquisition of lateral and rear distance information. This would enhance planning efficiency, reduce unnecessary trajectory adjustments, and improve flight performance in visually challenging conditions.

Acknowledgements The authors acknowledge Universiti Teknologi Malaysia (UTM Aerolab) for financially supporting this work through the grant vot number: R.J130000.7851.5 F573 (Fundamental Research Grant Scheme) and grant vot number: R.J130000.7324.4 J608 (Translational Research@Campus). We are grateful to Mr. Aiman Hanif for his invaluable advice and help in the UAV Lab during this work.

Author contributions Z.H was responsible for the integration of UAV hardware and software, the execution of experiments, and data analysis. M.B, N.N and M.S contributed to the design and refinement of experimental details and assisted with the preparation of experiment environment and certain figures. The initial draft of the manuscript was written by Z.H, and all authors reviewed and provided feedback on earlier versions of the manuscript. All authors have read and approved the final manuscript.

Funding Open access funding provided by The Ministry of Higher Education Malaysia and Universiti Teknologi Malaysia.

Data availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by-nc-nd/4.0/.

References

Ali, Z.A., Zhangang, H., Hang, W.B.: Cooperative path planning of multiple UAVs by using max–min ant colony optimization along with cauchy mutant operator. Fluct. Noise Lett. **20**(01), 2150002 (2021)

Al-Kaff, A., Garcia, F., Martin, D., De La Escalera, A., Armingol, J.M.: Obstacle detection and avoidance system based on monocular camera and size expansion algorithm for UAVs. Sensors (Basel) (2017). https://doi.org/10.3390/s17051061

Butt, M.Z., Nasir, N., Rashid, R.B.A.: A review of perception sensors, techniques, and hardware architectures for autonomous low-altitude UAVs in non-cooperative local obstacle avoidance. Robot. Auton. Syst. **173**, 104629 (2024)

Ding, W., Gao, W., Wang, K., Shen, S.: An efficient b-spline-based kinodynamic replanning framework for quadrotors. IEEE Trans. Rob. **35**(6), 1287–1306 (2019)

Erdelj, M., Saif, O., Natalizio, E., Fantoni, I.: UAVs that fly forever: uninterrupted structural inspection through automatic UAV replacement. Ad Hoc Netw. **94**, 101612 (2019)

Forster, C., Zhang, Z., Gassner, M., Werlberger, M., Scaramuzza, D.: SVO: semidirect visual odometry for monocular and multicamera systems. IEEE Trans. Rob. **33**(2), 249–265 (2016)

Gao, F., Wu, W., Gao, W., Shen, S.: Flying on point clouds: online trajectory generation and autonomous navigation for quadrotors in cluttered environments. J. Field Robot. **36**(4), 710–733 (2019)

Gao, F., Wang, L., Zhou, B., Zhou, X., Pan, J., Shen, S.: Teach-repeat-replan: a complete and robust system for aggressive flight in complex environments. IEEE Trans. Rob. **36**(5), 1526–1545 (2020)

Gasparetto, A., Boscariol, P., Lanzutti, A., Vidoni, R.: Path planning and trajectory planning algorithms: a general overview. In: Motion and Operation Planning of Robotic Systems, pp. 3–27. (2015). https://doi.org/10.1007/978-3-319-14705-5_1

Guo, N., Zhang, X., Zou, Y., Lenzo, B., Zhang, T.: A computationally efficient path-following control strategy of autonomous electric vehicles with yaw motion stabilization. IEEE Trans. Transp. Electrific. **6**(2), 728–739 (2020)

Heng, L., Meier, L., Tanskanen, P., Fraundorfer, F., Pollefeys, M.: Autonomous obstacle avoidance and maneuvering on a vision-guided mav using onboard processing. In: 2011 IEEE International Conference on Robotics and Automation (2011)

Idres, M., Mustapha, O., Okasha, M.: Quadrotor trajectory tracking using PID cascade control. In: IOP Conference Series: Materials Science and Engineering (2017)

Kerl, C., Sturm, J., Cremers, D.: Dense visual SLAM for RGB-D cameras. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (2013)

Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality (2007)

Lynen, S., Achtelik, M. W., Weiss, S., Chli, M., Siegwart, R.: A robust and modular multi-sensor fusion approach applied to MAV navigation. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (2013)

Madridano, Á., Al-Kaff, A., Martín, D., De La Escalera, A.: Trajectory planning for multi-robot systems: methods and applications. Expert Syst. Appl. **173**, 114660 (2021)

Mellinger, D., Kumar, V.: Minimum snap trajectory generation and control for quadrotors. In: 2011 IEEE International Conference on Robotics and Automation (2011)

Mo, H., Farid, G.: Nonlinear and adaptive intelligent control techniques for quadrotor UAV—a survey. Asian J. Control **21**(2), 989–1008 (2019)

Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: an open-source slam system for monocular, stereo, and RGB-D cameras. IEEE Trans. Rob. **33**(5), 1255–1262 (2017)

Oleynikova, H., Burri, M., Taylor, Z., Nieto, J., Siegwart, R., Galceran, E.: Continuous-time trajectory optimization for online uav replanning. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2016)

Petrlík, M., Báča, T., Heřt, D., Vrba, M., Krajník, T., Saska, M.: A robust UAV system for operations in a constrained environment. IEEE Robot. Autom. Lett. **5**(2), 2169–2176 (2020)

Qin, T., Li, P., Shen, S.: VINS-Mono: a robust and versatile monocular visual-inertial state estimator. IEEE Trans. Rob. **34**(4), 1004–1020 (2018). https://doi.org/10.1109/tro.2018.2853729

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source Robot Operating System. In: ICRA workshop on open source software (2009)
Richter, C., Bry, A., Roy, N.: Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments. In: Robotics Research: The 16th International Symposium ISRR (2016)
Rocha, L.G., Kim, P.H., Teixeira Vivaldini, K.C.: Performance analysis of path planning techniques for autonomous robots: a deep path planning analysis in 2D environments. Int. J. Intell. Robot. Appl. **7**(4), 778–794 (2023)
Sanfourche, M., Delaune, J., Le Besnerais, G., De Plinval, H., Israel, J., Cornic, P., Treil, A., Watanabe, Y., Plyer, A.: Perception for UAV: vision-based navigation and environment modeling. Aerospace Lab (4):1–19 (2012)
Taketomi, T., Uchiyama, H., Ikeda, S.: Visual SLAM algorithms: a survey from 2010 to 2016. IPSJ Trans. Comput. vis. Appl. **9**, 1–11 (2017)
Usenko, V., Von Stumberg, L., Pangercic, A., Cremers, D.: Real-time trajectory replanning for MAVs using uniform B-splines and a 3D circular buffer. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2017)
Xie, G., Zong, Q., Zhang, X., Tian, B.: Loosely-coupled lidar-inertial odometry and mapping in real time. Int. J. Intell. Robot. Appl. **5**(2), 119–129 (2021)
Xu, Z., Liu, X., Chen, Q.: Application of improved Astar algorithm in global path planning of unmanned vehicles. 2019 Chinese Automation Congress (CAC) (2019)
Zafar, M.N., Mohanta, J.: Methodology for path planning and optimization of mobile robots: a review. Procedia Comput. Sci. **133**, 141–152 (2018)
Zhang, W., Zhang, W.: Efficient UAV localization based on modified particle swarm optimization. In: 2022 IEEE International Conference on Communications Workshops (ICC Workshops) (2022)
Zhang, X., Shen, H., Xie, G., Lu, H., Tian, B.: Decentralized motion planning for multi quadrotor with obstacle and collision avoidance. Int. J. Intell. Robot. Appl. **5**(2), 176–185 (2021)
Zhang, W., Li, J., Yu, W., Ding, P., Wang, J., Zhang, X.: Algorithm for UAV path planning in high obstacle density environments: RFA-star. Front. Plant Sci. **15**, 1391628 (2024)
Zhou, X., Wang, Z., Ye, H., Xu, C., Gao, F.: Ego-Planner: an ESDF-free gradient-based local planner for quadrotors. IEEE Robot. Autom. Lett. **6**(2), 478–485 (2020b)
Zhou, B., Gao, F., Pan, J., Shen, S.: Robust real-time uav replanning using guided gradient-based optimization and topological paths. In: 2020 IEEE International Conference on Robotics and Automation (ICRA) (2020)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Zhanhao Hu received his Bachelor of Engineering (Hons.) in Mechanical Engineering (Aeronautics) from Universiti Teknologi Malaysia in 2023. He is currently pursuing a Master of Philosophy (M.Phil.) degree in the same department at Universiti Teknologi Malaysia. His research interests include vision-based unmanned aerial vehicles (UAVs), autonomous navigation, and robotics.



Muhammad Zohaib Butt received his Bachelor of Engineering in Electrical Engineering at the National University of Sciences and Technology, Pakistan, in 2012, and M.S. in Systems Engineering from Pakistan Institute of Engineering and Applied Sciences, Islamabad, 2014. Currently he is a research assistant and Ph.D. candidate in the department of Mechanical Engineering, Universiti Teknologi Malaysia. His research interests include radar signal processing, embedded control systems, electronics, and robotics.



Nazri Nasir was born in Melaka, Malaysia in 1982. He received a B.S. degree from the University of Manchester, U.K. in 2005, an MSc degree from the Technical University of Delft, The Netherlands in 2008, and a PhD in applied aerodynamics from Technical University Darmstadt, Germany, in 2017. Since then, he has been a senior lecturer at the Faculty of Mechanical Engineering, Universiti Teknologi Malaysia, Malaysia. His research interests include Unmanned Aerial Vehicles (UAV), applied aerodynamics, wind tunnel testing and natural fliers.



Mohd Badrul Salleh received his Bachelor, MSc and PhD degrees in Aerospace Engineering from Universiti Sains Malaysia (USM). Currently, he is a senior lecturer in the Aeronautics Engineering program, Faculty of Mechanical Engineering, Universiti Teknologi Malaysia (UTM). His research interests include bio-inspiration aerodynamics, experimental aerodynamics for flight mechanics and renewable energy applications.