

CO-PRE: A Multi-Sensor Fusion Framework with Visual-Inertial Navigation and Curvature-Optimized Path Planning for UAVs

Yuecai Yang

Department of Intelligent Engineering, Hebei Chemical & Pharmaceutical College, Shijiazhuang 050026, China

Keywords: multi-sensor fusion, visual inertial navigation, UAV, autonomous navigation, path planning

Received: April 22, 2025

With the widespread application of unmanned systems in various fields, achieving high-precision autonomous navigation of unmanned aerial vehicles has become a research hotspot. Therefore, a navigation system that integrates vision, lidar, inertial sensor and barometer is designed in this study. Based on the factor graph optimization structure, the effective fusion of multi-source information is realized, thereby improving the consistency of positioning and mapping. Meanwhile, a probabilistic path planning method based on curvature constraints is proposed to reduce trajectory discontinuity and support dynamic feasibility requirements. The results show that the positioning error of the complete system is 0.43 m and the mapping completeness is 98.7%. Compared with the traditional A algorithm, particle swarm optimization algorithm, and probabilistic rapid expansion random tree method, the improved algorithm reduces the path length by about 17.04% and shortens the planning time by about 16.71%. In the simulation test, its average energy consumption is 3.42 J/m, the average path deviation is 0.65 m, and the number of obstacle avoidance re-planning is 2.4 times/task. The research results show that the system has good operating performance in complex environments and has certain advantages over the baseline method in terms of efficiency, stability and obstacle avoidance ability.*

Povzetek: Predstavljena je avtonomna navigacija UAV. Uvaja faktor-grafno fuzijo podatkov (kamera, LiDAR, IMU, barometer) z načrtovanjem poti CO-PRE s krivinskimi omejitvami; 0,43 m RMSE, 98,7% zemljevid, ~17% krajše/hitrejše poti.

1 Introduction

As intelligent unmanned systems continue to evolve, quadcopter unmanned aerial vehicles (UAVs) have gained widespread application in diverse scenarios including urban logistics, disaster search and rescue, and environmental monitoring, thanks to their straightforward design, agile maneuverability, and robust adaptability [1]. Particularly in enclosed settings like indoor structures, underground corridors, and post-disaster debris where satellite positioning is unreliable, the achievement of autonomous UAV navigation has emerged as a critical challenge demanding urgent attention. UAVs need to have precise environmental perception and state estimation capabilities in such scenarios to complete core tasks such as positioning, mapping, and path planning, which puts higher demands on the system's perception reliability and computational efficiency [2-3]. In recent years, with the development of sensor technology and computing platforms, various sensors such as vision, LiDAR, inertial measurement units (IMUs), etc. have been widely used in autonomous navigation systems. Multi-sensor fusion has become a key direction for improving localization robustness and map accuracy [4-5].

Zhang et al. introduced an innovative modeling and algorithmic approach that integrates the Internet of Things with edge computing to address the limitations in intelligence within the collaborative optimization of path planning and control for UAVs. They developed an

integrated optimization model for path and control, along with a metaheuristic solution framework, thereby achieving a comprehensive enhancement in flight efficiency and collaborative scheduling [6]. Tong et al. introduced a novel optimization algorithm that integrates a logarithmic spiral with an adaptive step size strategy. This approach enhances population diversity through hierarchical modeling and improvements in spatial search. Simulation results demonstrated that the algorithm was capable of generating high-quality, feasible paths while satisfying time coordination constraints, thereby verifying its effectiveness and advantages [7].

Shiri was inspired by attention mechanisms in natural language processing and proposed an iterative single head attention mechanism suitable for multi-UAV path control. This method assigned individual scores through communication assistants to strengthen the modeling of multi-machine collaborative relationships in control. The results indicated that this mechanism could improve flight speed and reduce collision risk [8]. Bashir et al. proposed a path planning method based on connectivity awareness to address the risk of collision and communication interruption between UAVs and obstacles in urban environments. This method constructed a graph structured path and combined it with fleet formation strategy to ensure communication continuity and path safety [9].

In intricate environmental settings and task-oriented scenarios, the autonomous path planning of UAVs

necessitates a delicate equilibrium among flight efficiency, the quality of task coverage, and the robustness of obstacle avoidance. Researchers have proposed various path optimization methods that integrate intelligent algorithms and task models. Ko et al. proposed a UAV visual coverage method that combines velocity function and dynamic trajectory planning to meet the resolution requirements of images in different regions. This method optimized regional access by adjusting flight altitude and path sequence, combined with auxiliary travel merchant strategies [10]. Jayarajan N proposed a path planning method that combines artificial bee colony algorithm and fast exploration of random tree stars to address the low efficiency of UAV path planning in complex obstacle environments. This algorithm combined global search capability with path optimization characteristics to achieve efficient obstacle avoidance and path optimization in obstacle environments [11].

Wu et al. proposed the Q-Learning algorithm with adaptive transition speed to address the issues of low path efficiency and unstable learning in UAV search and rescue missions in unknown environments. This method involved phased design and state space optimization, combined with sensor information to initialize the learning process. The simulation results showed that the algorithm had fast learning speed, shorter path, and more stable convergence [12]. Liu et al. proposed a collaborative path planning model for UAVs and transport vehicles. Through multi-objective optimization and boundary intersection algorithm, UAV task scheduling and path optimization were achieved under dynamic objectives and variable path conditions [13]. Table 1 summarizes the main contents of the related work section.

In summary, the current UAVs still face three key problems in performing autonomous navigation tasks in complex environments: First, there is heterogeneity in multi-sensor observations, and the data fusion accuracy is not high. Second, path planning lacks continuity and real-time in dynamic environments. Third, the control system does not respond sufficiently to disturbances, affecting flight stability. Although existing studies have made some progress in visual-inertial navigation mapping or path planning, most of them have failed to achieve deep integration of perception, planning and control under a unified framework, and lack collaborative modeling of path feasibility and consistency of multi-source

observations. To this end, this study proposes a visual-inertial navigation autonomous navigation system based on multi-sensor fusion, constructs an integrated closed-loop architecture of perception-planning-control, and aims to provide efficient, accurate and adaptable solutions for autonomous navigation and real-time decision-making of UAVs in unknown environments. The study aims to evaluate whether the proposed multi-sensor fusion navigation framework can achieve reliable performance in complex environments. The specific goals are to keep the average path deviation below 0.7 meters and the energy consumption below 3.5 J/m while ensuring a high success rate and minimal re-planning during the navigation mission.

The novelty of this research resides in the application of the factor graph optimization method to achieve tight coupling and fusion of multiple sensor data. Additionally, to enhance the efficiency and precision of path planning, a Curvature Optimized Probabilistic Path Rapid Expansion Algorithm (CO-PRE) based on curvature optimization is introduced. By incorporating curvature smoothing techniques, this approach improves path smoothness and computational efficiency, while optimizing path feasibility and stability through the integration of dynamic constraints. Furthermore, anti-interference adaptive control methods are employed to bolster the system's steady-state response and resistance to disturbances.

2 Methods and materials

2.1 Autonomous positioning and mapping methods for multi-source information fusion

The autonomous localization and map construction of UAVs in unknown and communication limited environments rely on deep fusion of information from multiple heterogeneous sensors [14-15]. A synchronous positioning and mapping system based on tight coupling of multi-source information was studied and designed, which integrates observation information from laser radar, IMU, optical flow sensor, and barometer. State estimation and map construction were achieved through factor graph optimization. The architecture is shown in Figure 1.

Table 1: Summary of related UAV navigation methods.

Reference	Sensor Types	Optimization Method	Key Metrics
[6] Zhang X	IoT + Edge	Metaheuristic	Flight efficiency, scheduling improvement
[7] Tong H et al.	IMU + Visual	LASSA (log-spiral based)	Path feasibility, diversity, time coordination
[8] Shiri H et al.	Communication modules	ISHA	Flight speed, collision risk reduction
[9] Bashir N et al.	Connectivity sensors	Graph-based with fleet formation	Connectivity, path safety
[10] Ko Y C et al.	Camera	Velocity + Dynamic Planning	Coverage quality, task time
[11] Jayarajan N et al.	General UAV sensors	ABC-RRT*	Path efficiency in obstacle environments
[12] Wu J et al.	Sensor-driven learning	ACSQ	Path length, convergence stability
[13] Liu X et al.	UAV + Vehicle sensors	Penalty-based boundary intersection+Pareto technique	Efficiency, resource utilization

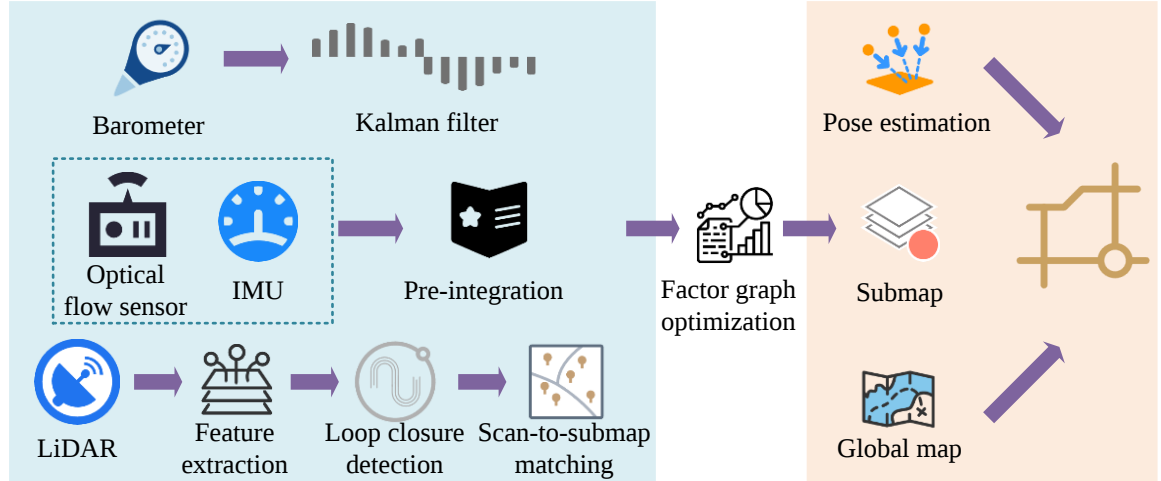


Figure 1: Synchronous localization and mapping system architecture based on tight coupling of multi-source information.

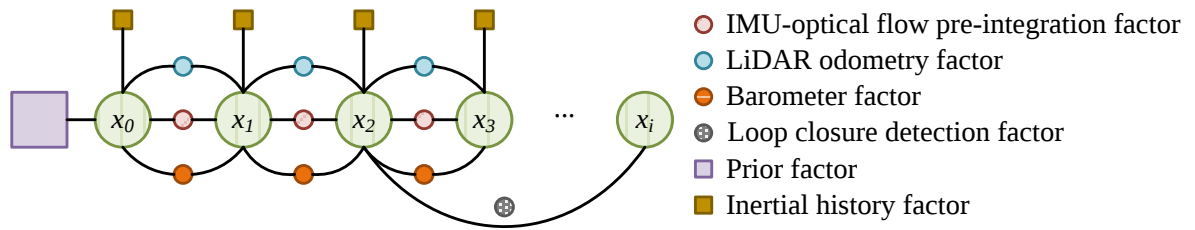


Figure 2: Schematic diagram of factor graph optimization structure under multi-sensor constraint fusion.

As illustrated in Figure 1, the system incorporates laser radar, IMU, optical flow sensors, and barometers to gather multi-source observation data. After pre-integration, filtering, and feature extraction, the data is fused via a factor graph framework to enable accurate pose estimation and dense map construction. The system's front-end handles data association and loop closure detection, while the back-end produces continuous subgraphs and global maps, thereby establishing a closed-loop process for autonomous positioning and mapping.

Initially, a systematic model of the quadcopter UAV's state is developed. The complete state vector of UAV at time t_k is represented by equation (1).

$$x_k = [p_k^T, v_k^T, q_k^T, b_{a_k}^T, b_{w_k}^T, p_{O_B}^T]^T \quad (1)$$

In equation (1), p_k represents the position vector of the UAV in the world coordinate system W , and v_k represents the velocity. q_k is a posture represented by quaternions. b_{a_k} and b_{w_k} are the zero bias terms of the accelerometer and gyroscope, respectively, and p_{O_B} represents the position of the origin of the optical flow coordinate system relative to the body coordinate system B .

To achieve unified fusion of multi-source sensor information, the system constructs a state estimation framework based on factor graphs, in which each state node represents the position of the UAV at a certain moment, and the edges correspond to the observation constraint factors of different sensors. The spatial relationship between the body coordinate system B and

the optical flow coordinate system O is shown in Figure 2.

Figure 2 shows the composition of the multi-sensor factor graph optimization structure, where each state node represents the pose estimation of the UAV at a specific moment, and the measurements from different sensors are connected to the state nodes in the form of factors. The data is modeled as four types of observation factors, corresponding to inertial-visual pre-integration, lidar mileage constraint, air pressure measurement, and loop detection. Different information sources are embedded in the state estimation graph in a unified form to provide a high-precision initial state for subsequent pose optimization and path planning.

To enhance the mathematical clarity and technical operability of factor graph fusion modeling, a joint residual function is introduced to characterize the constraints of multi-source observation information on state estimation. Let the UAV state vector be x , including position, velocity, attitude quaternion and inertial sensor bias, then the optimization objective function can be expressed as equation (2).

$$J(\hat{x}) = \sum_i \|r_{imu}^i\|_{\Sigma_{imu}^{-1}}^2 + \sum_j \|r_{lidar}^j\|_{\Sigma_{lidar}^{-1}}^2 + \sum_k \|r_{flow}^k\|_{\Sigma_{flow}^{-1}}^2 + \sum_l \|r_{baro}^l\|_{\Sigma_{baro}^{-1}}^2 \quad (2)$$

In equation (2), r_{imu}^i , r_{lidar}^j , r_{flow}^k , r_{baro}^l represent the residual terms of the i , j , k , l th sensor observations respectively. Σ_{imu} , Σ_{lidar} , Σ_{flow} , and Σ_{baro} are the covariance matrices of the corresponding observation

residuals, which are used to weight the errors of different sensors. $\|r\|_{\Sigma^{-1}}^2$ represents the Mahalanobis distance norm, which is used to consider the noise uncertainty of each residual. To clearly present the relationship between the system state composition and sensor observations, Table 2 lists the state variables and their source sensors.

In addition, to improve the accuracy of LiDAR point cloud registration and the global consistency of the system, the study adopts an initial registration strategy based on voxel filtering and KD tree matching, and introduces an iterative closest point method for residual refinement. In terms of system closed-loop detection, a revisit detection module based on the scan context descriptor is designed. Once the loop node is identified, the pose graph optimization module is triggered to adjust the global constraints, and the graph scale is controlled through the node pruning mechanism to ensure real-time operation efficiency.

Firstly, based on the pre integration factor of optical flow and IMU, high-frequency constraints are established using the continuous acceleration and angular velocity information obtained by IMU, and combined with the relative displacement observation of optical flow sensor on the image plane. The IMU measurement model is represented by equation (3).

$$\begin{aligned}\hat{w}_t &= w_t + b_w + n_w, \\ \hat{a}_t &= R_w^B(a_t - g) + b_a + n_a\end{aligned}\quad (3)$$

In equation (3), $\hat{w}_t$ and $\hat{a}_t$ respectively represent the angular velocity and acceleration measured by IMU. b_w and b_a are zero bias terms. n_w and n_a are for noises. R_w^B is for the rotation matrix and g is for the gravitational acceleration. The optical flow method is based on the assumption of constant image grayscale, and its constraints are shown in equation (4).

$$I(\tilde{x} + \Delta\tilde{x}, \tilde{y} + \Delta\tilde{y}, t + \Delta t) = I(\tilde{x}, \tilde{y}, t) \quad (4)$$

In equation (4), $I(\tilde{x}, \tilde{y}, t)$ represents the grayscale of the image at time t and $(\tilde{x}, \tilde{y})$ is the pixel coordinate. $\Delta\tilde{x}$ and $\Delta\tilde{y}$ are the displacement amounts, and Δt is the time interval between frames.

A first-order Taylor expansion is performed on the equation, the definitions of pixel velocities $u_{pix} = \frac{d\tilde{x}}{dt}$ and

$w_{pix} = \frac{d\tilde{y}}{dt}$ are introduced, and linear system constraints are further constructed and then solved using the least squares method, as shown in equation (5).

$$\begin{aligned}A \begin{bmatrix} u_{pix} \\ w_{pix} \end{bmatrix} &= -b, \\ \begin{bmatrix} u_{pix} \\ w_{pix} \end{bmatrix} &= -(A^T A)^{-1} A^T b\end{aligned}\quad (5)$$

In equation (5), A is a matrix composed of image gradients of multiple pixels, with each row representing the gradient $[I_{\tilde{x}}, I_{\tilde{y}}]$ of a point in the $\tilde{x}$ and $\tilde{y}$ directions.

b corresponds to the time gradient I_t of each pixel.

$[u_{pix}, w_{pix}]^T$ represents the average optical flow velocity vector of pixels on the image plane. Pixel velocity is converted to ground velocity and UVA camera installation height H and scaling factor are introduced. When there is a change in attitude of UVA, it is corrected based on the gyroscope output, and the corrected velocity observation is obtained as shown in equation (6).

$$\hat{v}_o = v_o - H w_z + n_o \quad (6)$$

In equation (6), w_z is the rotational speed around the vertical axis. n_o is the observation noise term. Subsequently, in the factor graph optimization, a joint pre integration model is constructed based on the observation information of IMU and optical flow sensor to estimate the state transition of UAV between two laser radar keyframes. The prediction of the state between time t_k and t_{k+1} is described by the integral equation of equation (7).

$$\begin{cases} p_{k+1} = p_k + v_k \Delta t_k + \\ \int_{t_k}^{t_{k+1}} \int_{t_k}^{\tau} (R_w^B(\hat{a}_t - b_a - n_a) - g) d\tau d\tau \\ v_{k+1} = \\ v_k + \int_{t_k}^{t_{k+1}} (R_w^B(\hat{a}_t - b_a - n_a) - g) d\tau \\ q_{k+1} = \\ q_k \otimes \text{Exp}\left(\frac{1}{2} \int_{t_k}^{t_{k+1}} (R_w^B(\hat{w}_t - b_w - n_w) d\tau\right)\end{cases} \quad (7)$$

Table 2: UAV state elements, sensor observations, and factor addition time.

State Variable / Observation	Sensor Source	Factor Addition Time (ms)
UAV position in world frame	LiDAR, IMU, Optical Flow, Barometer	0.4 (IMU), 1.8 (LiDAR), 0.6 (Flow), 0.2 (Baro)
UAV velocity	IMU, Optical Flow	0.4 (IMU), 0.6 (Flow)
UAV orientation $\sum_{lidar}$	IMU	0.4
Accelerometer bias	IMU	0.4
Gyroscope bias	IMU	0.4
Image-plane displacement observation	Optical Flow	0.6
Altitude estimate	Barometer	0.2
Point cloud geometric residuals	LiDAR	1.8

Note: The listed factor addition times were measured on a system with Intel Core i7-12700H and 32GB RAM using ROS middleware. Values represent average computation time per frame for inserting the corresponding factor into the graph optimization.

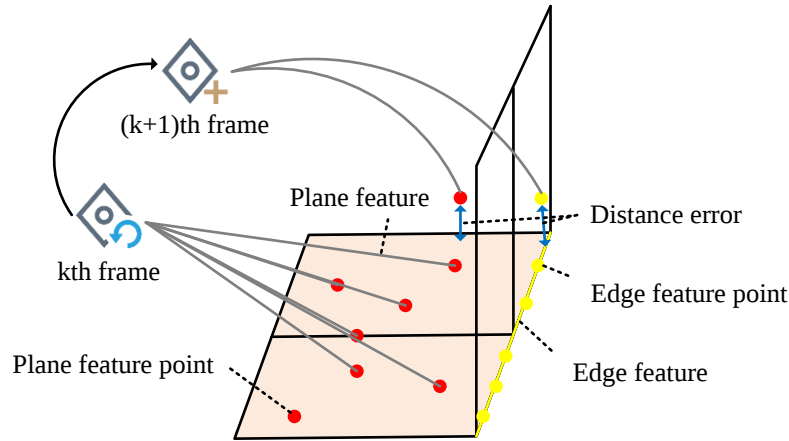


Figure 3: Schematic diagram of laser radar feature extraction and inter-frame matching.

In equation (7), Δt_k represents the time difference. $\otimes$ is quaternion multiplication. $Exp(\cdot)$ represents exponential mapping. Under the factor graph optimization framework, all observation information is ultimately fused through residual terms to construct the optimization objective function.

Furthermore, in the joint optimization of LiDAR and IMU, the feature points extracted by LiDAR are combined with IMU data to estimate the relative motion of UAV. The distance calculation in feature matching is shown in equation (8).

$$d_e = \left\| \frac{e_o \times e_2}{\|e_o\|} \times e_2 \right\|, \quad (8)$$

$$d_p = \left\| \frac{(p_3 p_2) \times (p_1 p_2)}{p_2 o} \right\|$$

In equation (8), e_o and e_2 are the edge features of the feature points. p_1 , p_2 , and p_3 are the coordinates of the point cloud, and o is the origin of the beam. The schematic diagram of LiDAR feature extraction and inter frame matching is shown in Figure 3.

As shown in Figure 3, the feature points of the current frame are compared with those of the previous frame by rotating the matrix T_k , to estimate the relative displacement between the two frames. The calculated distance is used to optimize the objective function, which is shown in equation (9).

$$e = \sum d_e + \sum d_p \quad (9)$$

In equation (9), d_e and d_p are the distances from the feature point matching point to the edge and from the point to the surface, and e is the residual. Finally, the barometer is used to estimate the altitude of the UAV, but its measurement is greatly affected by noise. Therefore, a low-pass filter is used to smooth the barometer data. The calculation of the measured value of the barometer is shown in equation (10).

$$z_k = b_0 h_k + b_1 h_{k-1} + b_2 h_{k-2} + a_1 z_{k-1} + a_2 z_{k-2} + n_h \quad (10)$$

In equation (10), h_k is the raw measurement value of the barometer, b_0 , b_1 , b_2 , a_1 , and a_2 are filter coefficients, and n_h is the noise term. As a result, the measurement value of the barometer is smoothed and corrected.

2.2 3D path planning algorithm integrating smooth and constrained optimization

The integration of multi-source information and state estimation has established a fundamental basis for the autonomous positioning and mapping of UAVs. However, to attain full closed-loop control within autonomous navigation systems, it remains imperative to tackle the pivotal challenges of path planning and dynamic obstacle evasion. Therefore, in response to the practical application requirements of quadcopter UAVs, a path planning method based on Probabilistic Rapidly-exploring Random Tree (P-RRT) algorithm is proposed. P-RRT quickly finds feasible paths from the starting point to the target point by randomly expanding the tree structure, which is suitable for avoiding dynamic environments and complex obstacles [16-17]. To further improve the smoothness and computational efficiency of the path, curvature smoothing technology will be introduced to reduce the randomness of the planning.

After generating the local path, a sliding window is first used to calculate the curvature smoothness of each point in the path. In the path planning process, a sliding window containing $|S|$ path nodes is used to smooth the current node x'_i , and the curvature smoothness of the path nodes is calculated using equation (11).

$$c_i = \frac{1}{|S|} \left\| \sum_{j \in S, j \neq i} (x'_j - x'_i) \right\| \quad (11)$$

In equation (11), c_i represents the average position deviation between the i th node and the nodes in its sliding window neighborhood, which can be used to approximately measure the local non-smoothness or "sharpness" of the point, and thus serve as an heuristic

indicator of path smoothness. S represents a sliding window set centered on the current path node x_i' and containing $|S|$ neighboring nodes. Based on the calculation of curvature smoothness, the sliding window method is used to optimize the path. According to the calculated curvature smoothness, if the curvature of a node is less than a predetermined threshold c_r , the node is deleted and the path is adjusted through interpolation algorithm to achieve smoother path planning.

To further smooth the generated path, a B-spline-based formulation is used as shown in equation (12), which defines the trajectory parametrically through control point interpolation

$$P(s(t)) = s_{t+1} M_{t+1} q_a \quad (12)$$

In equation (12), P represents the path planning result. q_a is the speed of the path node. M_{t+1} represents the constraint matrix at the current time. This expression serves as a structural representation of the path, providing the basis for subsequent optimization. To enforce velocity and acceleration limits during path execution, dynamic constraints are incorporated into the optimization process via the cost function defined in equation (13).

$$\begin{cases} V = Q_{m+1} - Q_m, & A = \frac{V}{\Delta t} \\ f_c = \sum (\lambda_1 (V)^2 + \lambda_2 (A)^2) \end{cases} \quad (13)$$

In equation (13), V and A respectively represent the velocity and acceleration of the path nodes. λ_1 and λ_2 are weight coefficients for adjusting speed and acceleration limits, ensuring that the path satisfies dynamic constraints simultaneously during the optimization process. This designated objective function serves to mitigate abrupt

fluctuations in both speed and acceleration throughout the optimization procedure. Consequently, it facilitates the attainment of dynamic feasibility and enables precise control over the smoothness of the path.

To efficiently explore unknown environments, the Frontier exploration strategy is introduced. By dividing the environment into known and unknown areas, quadcopter UAVs continuously update the map to find the boundaries of unknown areas and use them as new exploration targets [18]. After completing the exploration of a local area, the system calculates the unknown area adjacent to the known area and uses it as the next target for path planning. In three-dimensional space, considering the limitation of computing resources, the study optimizes the exploration area range through spatial segmentation to reduce computational burden. The improved Frontier exploration framework is shown in Figure 4.

As shown in Figure 4, the framework indicates the division of known and unknown regions during the exploration process. Quadcopter UAVs continuously update the boundaries of known environments and use the boundaries of unknown areas as new exploration targets for path planning. Therefore, the process of combining the improved exploration strategy with the path planning algorithm is shown in Figure 5.

Figure 5 illustrates that during the path planning process, the algorithm initially identifies the global path objective. Subsequently, it constructs a three-dimensional map utilizing the current position of the UAV and environmental data. Following this, the algorithm optimizes the three-dimensional path planning to generate the most efficient route, ensuring a seamless navigation of the UAV towards the target zone. Ultimately, the algorithm refines the entire exploration and path planning procedure through a hybrid path planning strategy.

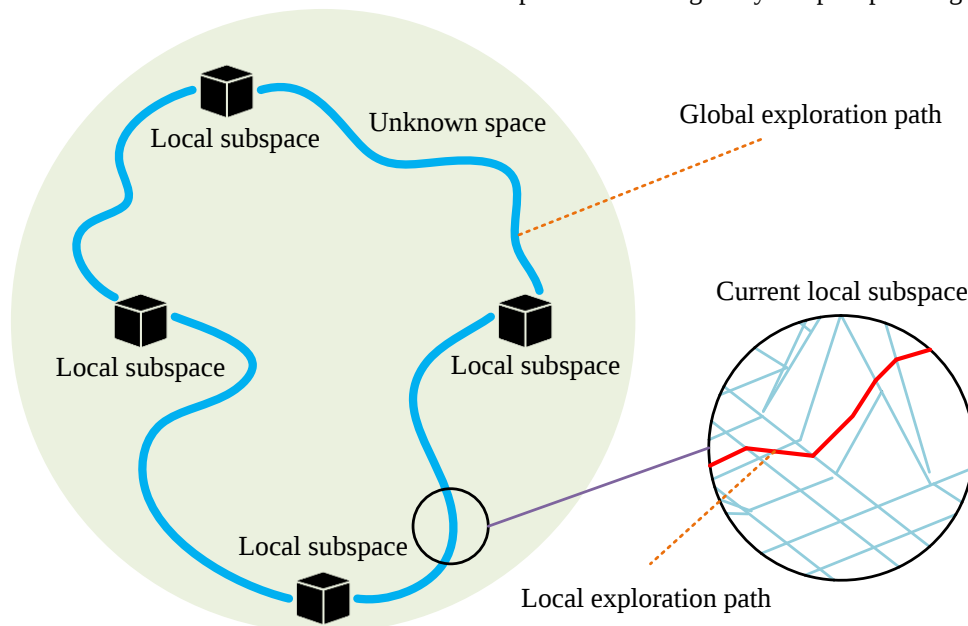


Figure 4: Improved Frontier exploration framework.

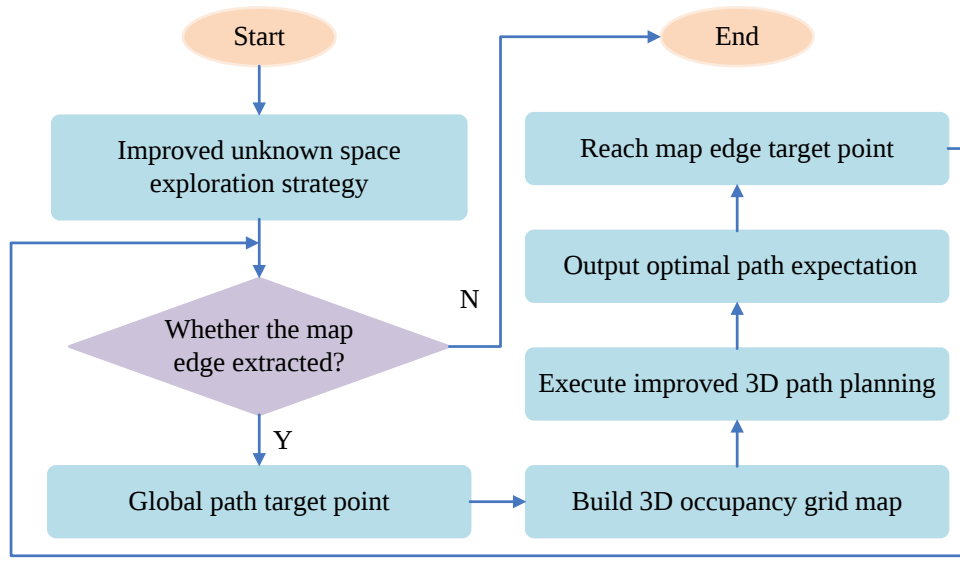


Figure 5: Process of combining improved exploration strategy with path planning algorithm.

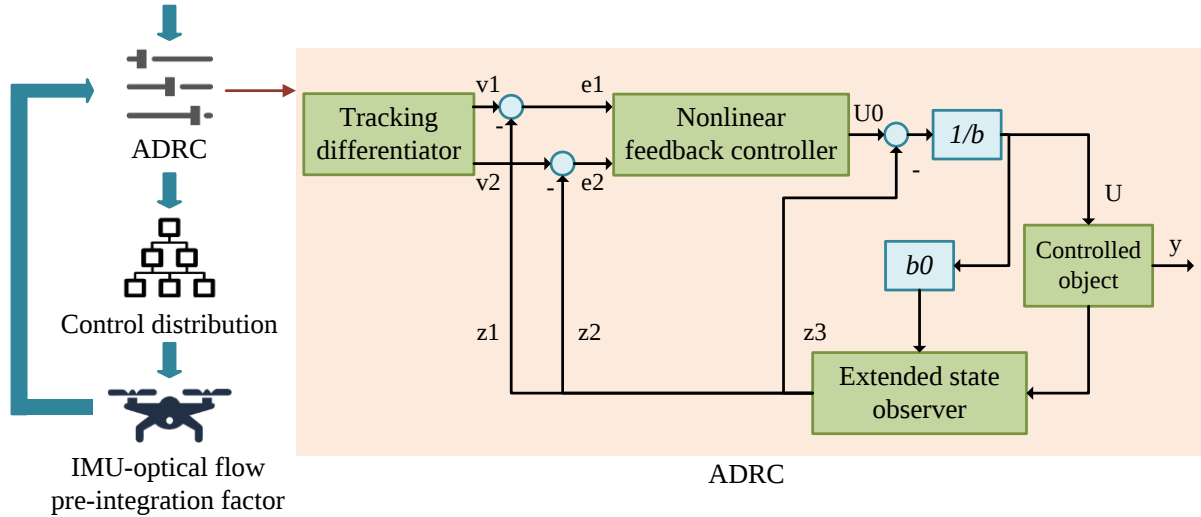


Figure 6: UAV control framework based on ADSC.

Subsequently, to achieve precise navigation and stable control, the research will adopt Active Disturbance Rejection Control (ADRC) method. ADSC, as a powerful control strategy, can effectively handle external disturbances and internal model uncertainties in the system [19–20]. Its core includes nonlinear state error feedback control law and extended state observer (ESO). Assuming that the current UAV system is a second-order uncertain system, the internal dynamics and external disturbances are uniformly modeled as generalized disturbance terms, and the extended state observer is constructed as shown in equation (14).

$$\begin{aligned} \dot{z}_1 &= z_2 - \beta_1(z_1 - y) \\ \dot{z}_2 &= z_3 - \beta_2(z_1 - y) \\ \dot{z}_3 &= -\beta_1(z_1 - y) \end{aligned} \quad (14)$$

In equation (14), z_1 , z_2 , and z_3 are the position, velocity, and generalized disturbance estimates,

respectively. β is the ESO and controller gain parameter. The final control law is shown in equation (15).

$$u = \frac{1}{b_0}(-k_1(z_1 - y_{ref}) - k_2 z_2 - z_3) \quad (15)$$

In equation (15), b_0 is the nominal system gain, and k_1 is the ESO and controller gain parameter. This control law is capable of effectively mitigating external disturbances and compensating for modeling inaccuracies, all without the need for a precise system model. The overall control architecture is depicted in Figure 6.

As shown in Figure 6, the framework is based on the ADRC strategy and includes key modules such as input commands, control quantity allocation, nonlinear feedback, and extended state observer. By monitoring the status of UAVs in real-time and compensating for interference, ADRC can effectively suppress external disturbances and model uncertainties, thereby achieving stable flight control. This control framework can

effectively improve the control performance of quadcopter UAVs, ensuring task completion in dynamic and unknown environments.

ADRC estimates unknown disturbances online and performs feed-forward compensation through an extended state observer. It has the advantages of fast response, clear structure, and strong parameter independence. Related literature studies have shown that ADRC has stronger anti-disturbance ability and convergence stability under complex working conditions compared with traditional controllers such as PID and LQR [21–22]. In the study, the ADRC parameters were configured according to the bandwidth adjustment method. The observer bandwidth was set to 30 rad/s, the tracking differentiator bandwidth

was set to 20 rad/s, and the control gain parameters were $b_0=50$, $k_1=100$, $k_2=300$, and $k_3=500$ respectively. The parameter selection was optimized through typical wind disturbance simulation and noise test to ensure that the system still has a small steady-state error and fast recovery capability under unstable excitation.

To enhance reproducibility, the following pseudo code summarizes the overall workflow of the proposed CO-PRE algorithm, see Figure 7.

Figure 7 covers key steps such as sampling expansion, curvature smoothing, B-spline fitting and dynamic constraint optimization, and intuitively demonstrates the overall logic of path generation and optimization.

Pseudocode of CO-PRE Algorithm

Input:

- Occupancy map M
- Start point s
- Goal point g
- Curvature threshold κ_c
- Window size S
- Dynamic constraint weights λ_1, λ_2

Output:

- Smoothed path P

```

1: Initialize tree T ← {s}
2: while goal not reached do
3:   Sample random node q_rand in M
4:   Find nearest node q_near in T
5:   Extend from q_near to q_new
6:   if q_new is collision-free then
7:     Add q_new to T
8:   end if
9: end while

10: Extract raw path P_raw from T

11: for each node p in P_raw do
12:   Define sliding window W ← neighbors(p, S)
13:   Compute curvature  $\kappa_W$  using Eq.(11)
14:   if  $\kappa_W < \kappa_c$  then
15:     Remove node p and interpolate
16:   end if
17: end for

18: Apply B-spline smoothing to obtain P_spline
19: Optimize P_spline with dynamic constraints using Eq.(12)-(13)
20: Return final path P ← P_spline

```

Figure 7: Path planning comparison experiment results.

3 Results

3.1 Performance testing of multi-sensor fusion autonomous navigation system

The experiment was conducted under the Ubuntu 20.04 operating system, using ROS as the middleware platform, using Gazebo to build a three-dimensional flight scene, and using AirSim to simulate the dynamics of the aircraft. All modules were implemented in Python and C++ mixed programming, and the path planning module was encapsulated in the `co_pre_planner` function package for operation. Multi-source data such as lidar, IMU, optical flow sensor and barometer were generated in real time by the simulator and synchronously input into the system. The computing platform was configured with an Intel Core i7-12700H processor, 32GB of memory and an NVIDIA RTX 3060 graphics card. To ensure the uniformity of the evaluation, all experiments were conducted under the same system architecture and communication frequency settings. Firstly, sensitivity analysis was conducted on the key parameters in CO-PRE

to determine the optimal parameters and provide a basis for subsequent experiments. The results are shown in Table 3.

Table 3 systematically evaluated the effects of different combinations of curvature constraint thresholds c_r , sliding window sizes $|S|$, and dynamic constraint factors λ_1 , λ_2 on path length, average curvature, and planning time. In the experiment, c_r increased from 0.5 to 0.95, showing that too low a threshold would cause path oscillation at turns, while too high a threshold would limit trajectory following. The increase in sliding window size $|S|$ improved curvature smoothness while also bringing certain computational overhead. In particular, Groups 1 and 2 used extremely low dynamic constraint weights, which could be regarded as approximate ablation results of weakening dynamic constraints. The corresponding paths had the largest average curvature and reduced path feasibility. Considering the path quality and computational efficiency, the final parameter combination was selected as $c_r=0.9$, $|S|=7$, $(\lambda_1, \lambda_2)=(3,3)$.

Table 3: Key parameter sensitivity test results.

Serial number	c_r	$ S $	λ_1, λ_2	Path length (m)	Mean curvature (rad/m)	Planning time (ms)
1	0.5	5	(1,1)	11.8	0.164	145
2	0.7	5	(1,1)	11.2	0.118	158
3	0.7	7	(1,1)	11.0	0.109	169
4	0.7	7	(2,2)	10.7	0.102	176
5	0.7	7	(3,3)	10.4	0.096	190
6	0.9	7	(3,3)	10.2	0.090	203
7	1.1	7	(3,3)	10.1	0.087	215
8	0.5	7	(2,2)	11.5	0.130	160
9	0.9	5	(1,1)	10.5	0.095	178

Table 4: Area Coverage Rate over Time (%±SD, n = 5).

Time (s)	Classic Frontier	Greedy Frontier	Dynamic Frontier	Proposed method
10	22.5±1.4	25.1±1.7	27.9±1.2	31.4±1.1
20	47.3±2.1	50.2±1.9	55.6±2.2	61.8±1.6
30	63.9±2.3	66.4±2.0	71.5±2.4	76.5±1.8
40	76.2±2.5	78.7±2.1	83.3±2.0	88.9±1.5
50	85.4±2.1	86.3±1.9	89.1±1.8	94.6±1.2

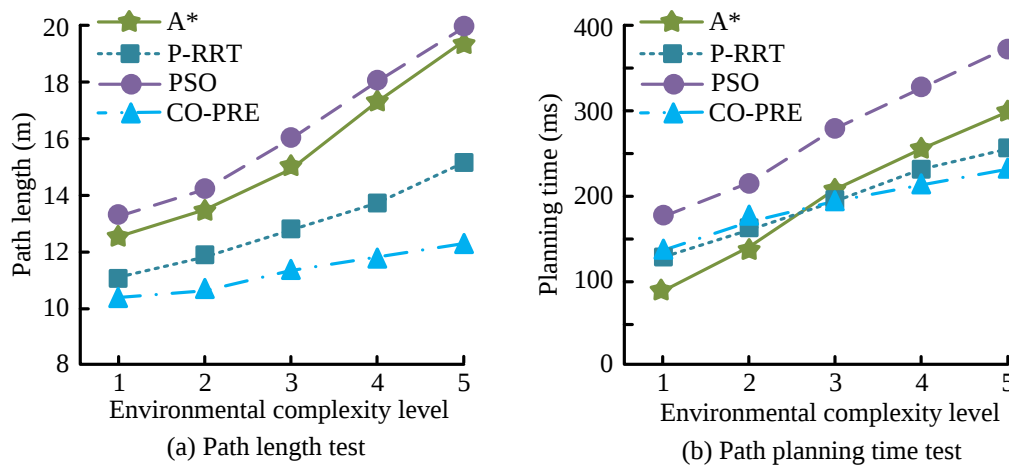


Figure 8: Path planning comparison experiment results.

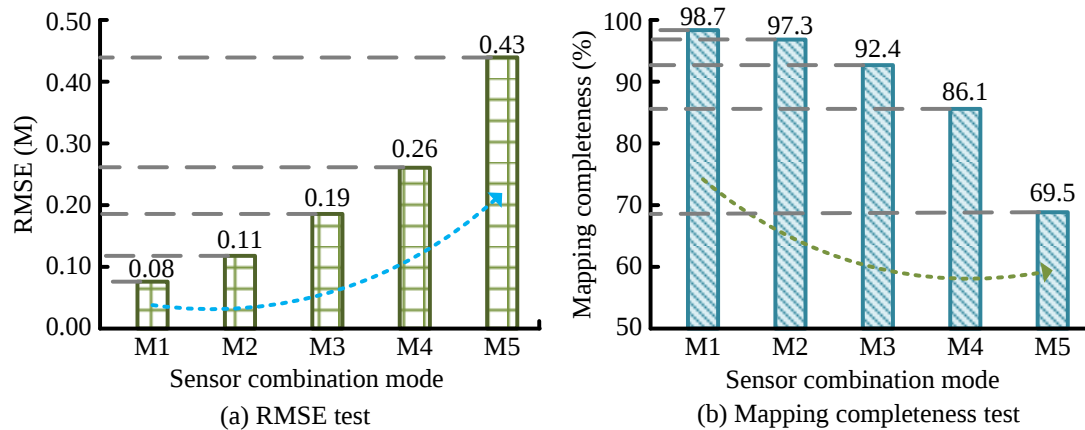


Figure 9: Multi-sensor combination comparison experimental results.

To verify the exploration efficiency of the improved Frontier strategy, a comparative experiment was designed with three typical methods: Classic Frontier, Greedy Frontier and Dynamic Frontier. Table 4 shows the average exploration coverage at different times. All results were the average of 5 independent experiments, with standard deviations attached.

The results in the table showed that the proposed improved Frontier strategy achieved higher area coverage in each stage, reaching 94.6% at 50 seconds. A paired t-test was further performed, and the results showed that this method had a statistically significant advantage over other strategies ($p < 0.01$).

Subsequently, the CO-PRE algorithm was tested, introducing A* algorithm, Particle Swarm Optimization (PSO) algorithm, and traditional P-RRT as comparison algorithms. The experiment divided the complexity level of the environment from 1 to 5, with level 1 representing a simple environment with sparse obstacles and smooth paths. Level 5 simulates high difficulty navigation areas with dense obstacles and complex structures. The results are shown in Figure 8.

In Figure 8 (a), when the complexity level was 5, the path lengths of A*, PSO, P-RRT, and CO-PRE were 18.18 m, 19.78 m, 14.26 m, and 11.83 m, respectively. In Figure 8 (b), the planning times of the four algorithms at the highest complexity level were 299.7 ms, 360.2 ms, 247.1 ms, and 205.8 ms, respectively. The CO-PRE mechanism reduced redundant sampling and improved node selection. By applying dynamic constraints, it produced smoother and more controllable paths, balancing quality and real-time efficiency. In contrast, A* increased its time consumption in complex environments due to graph search inflation, PSO was the slowest due to computationally intensive iterative optimization, and P-RRT had more path redundancy despite its fast expansion, which affected overall efficiency.

To verify the influence of various sensors in the multi-source fusion system on the UAV positioning and mapping performance, five sensor combination modes were designed for comparative experiments, and different sensors were removed to observe the system performance changes. The specific configuration was as follows: M1 was in full fusion mode (LiDAR, IMU, optical flow, and

barometer were all enabled), M2 removed the barometer, M3 removed the optical flow and barometer, M4 removed the LiDAR, and M5 only retained the IMU single sensor. The results of root mean square error (RMSE) and mapping completeness for each combination are shown in Figure 9.

In Figure 9 (a), the RMSE of M1, M2, M3, M4, and M5 were 0.08 m, 0.11 m, 0.19 m, 0.26 m, and 0.43 m, respectively. The positioning accuracy decreased significantly as the number of sensors decreased. M1 integrated all sensors and had the strongest state observation capability. M2 only removed the barometer, with little change in positioning accuracy. After M3 lost optical flow, its ability to observe velocity decreased and the error increased significantly. Removing the LiDAR from M4 resulted in a decrease in map construction accuracy, indirectly affecting pose estimation. However, M5 relied solely on IMU, with the largest error, and simple inertial navigation was difficult to support long-term positioning. In Figure 9 (b), the completeness of mapping for the five groups was 98.7%, 97.3%, 92.4%, 86.1%, and 69.5%, respectively. M4 and M5 showed the most significant decrease, indicating that LiDAR contributed the most to map accuracy, while IMU could not provide spatial structure information. This verified the key role of multi-sensor collaboration in stable mapping and accurate navigation.

3.2 Testing of UAV autonomous navigation system based on simulation environment

Furthermore, the study designed an autonomous navigation experimental scenario. Firstly, the experiment set up the UAV to take off from its original position to an altitude of 1.6 meters, maintain a brief hover, and then translate about 1.5 meters along the x-axis direction before returning to the origin and landing. The flight process was closed-loop controlled by the controller based on the output of the navigation system. The system recorded the real-time position information of the UAV in the x, y, and z axes, and compared the error with the preset trajectory to test the tracking ability of the system in three-

dimensional space. The experimental results are shown in Figure 10.

In Figures 10, during the takeoff phase, the system experienced short-term oscillations in response to sudden changes, then quickly stabilized and entered a steady tracking phase. The x-axis and y-axis converged within 2 seconds, while the z-axis remained within a fluctuation range of ± 0.1 meters. The system exhibited good response speed and control accuracy during autonomous flight, verifying the adaptability and robustness of the state estimation and control system for trajectory tracking under the multi-sensor fusion architecture.

To further verify the path planning and execution capability of the proposed autonomous navigation system in complex three-dimensional space, several three-dimensional obstacles were randomly arranged in the experimental scene. The UAV needed to start from the starting point and use various path planning algorithms to generate trajectories to bypass the obstacles and reach the target point. All obstacle maps in the experiment were generated using the seed randomization method, and the environmental parameters were fixed. All methods used the same random seed to ensure the consistency of obstacle layout and scene configuration. The comparison algorithms were A*, Rapid exploring Random Tree Star Algorithm (RRT*), and Path Optimization Algorithm

based on Bezier Curve. The results are shown in Figure 11.

In Figure 11 (a), Algorithm A* was based on regular grid search, and the path presented multiple polylines. Although it could ensure accessibility, it did not have a trajectory smoothing mechanism, resulting in multiple turns and discontinuous paths, which led to an overall increase in the length of the route. In Figure 11 (b), the RRT algorithm used a sampling extension tree to construct the path, which had strong global search ability. However, its random sampling and reconnection process was prone to local oscillations, resulting in severe path jitter and affecting navigation stability. In Figure 11 (c), the Bezier curve relied on control points to generate an overall smooth curve with excellent curvature continuity. However, due to the lack of explicit consideration of obstacles, the path approached or even partially crossed obstacles, lacking environmental adaptability and resulting in infeasible areas. The CO-PRE algorithm in Figure 11 (d) introduced a sliding window mechanism based on curvature optimization, dynamically adjusted path nodes, and combined probability sampling and spatial feasibility discrimination to effectively balance path smoothness and obstacle avoidance ability. The final generated path was smooth, continuous, and risk averse, demonstrating good adaptability and practical application potential for 3D navigation.

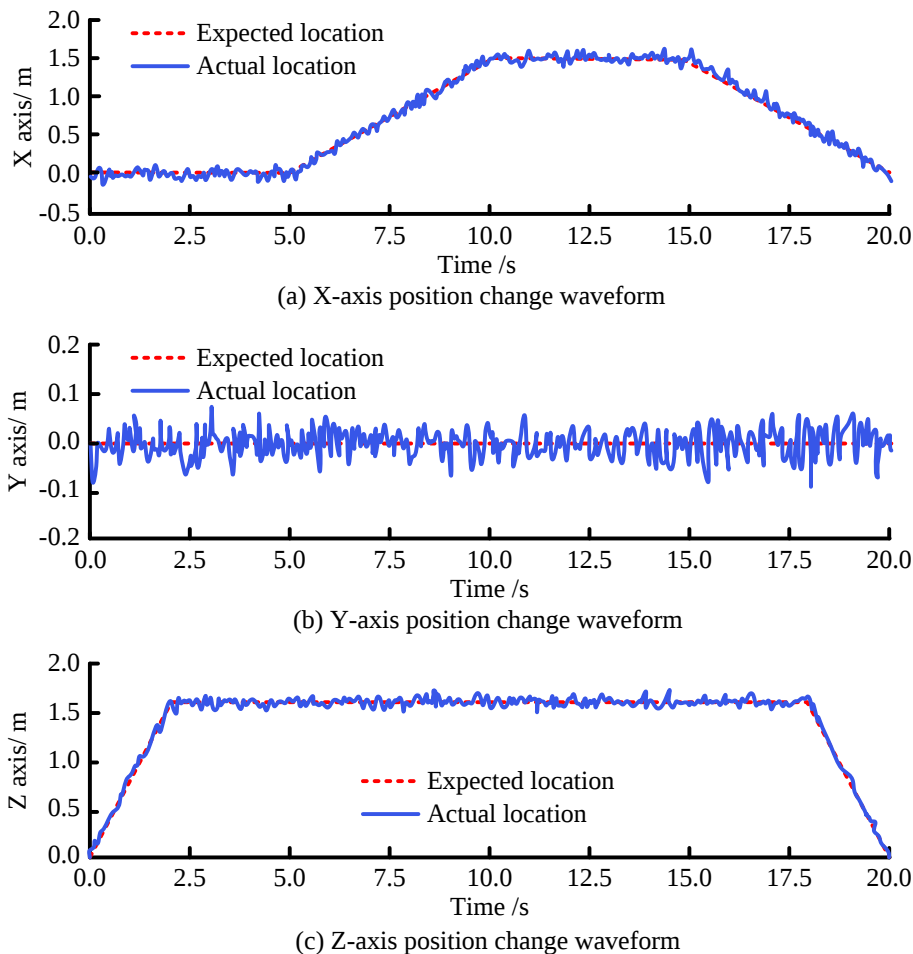


Figure 10: Three-axis trajectory tracking comparison chart.

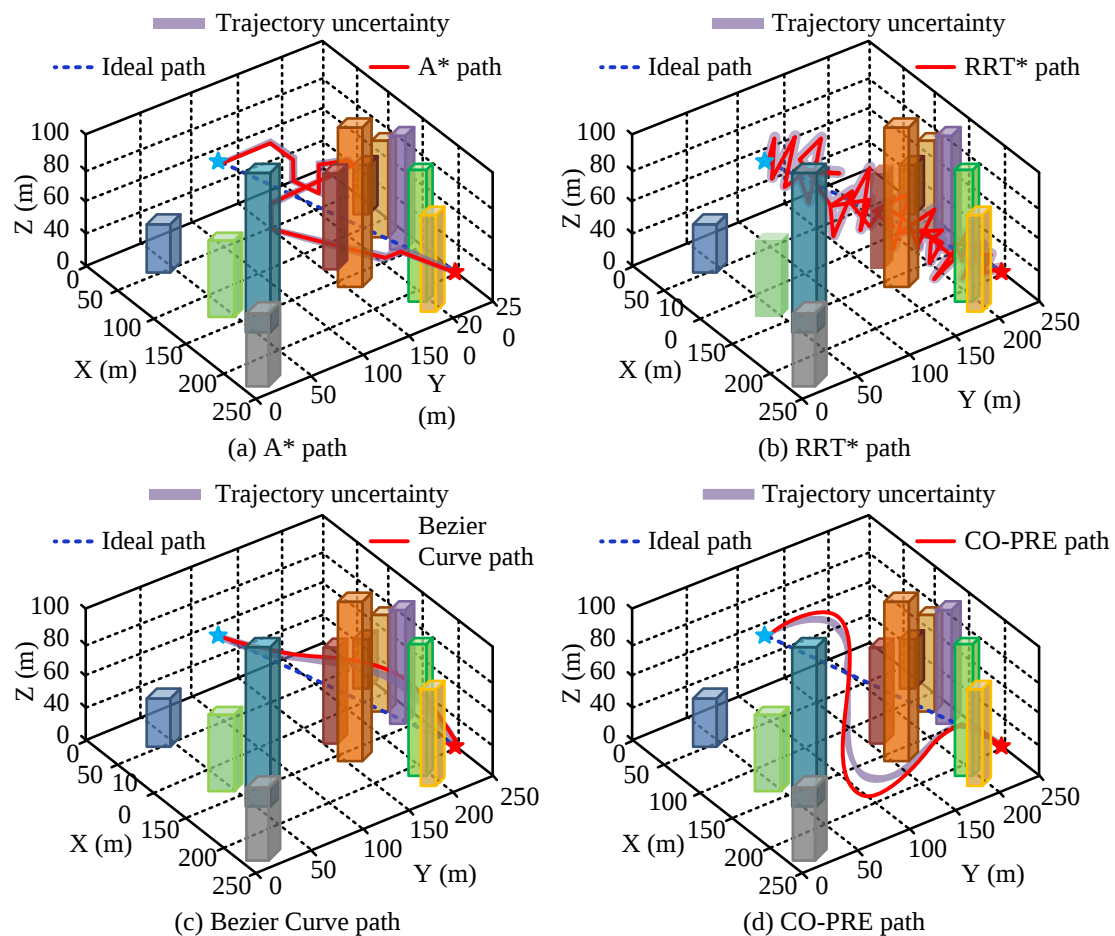


Figure 11: Three-dimensional navigation experiment based on urban building obstacle scene.

Table 5: Comprehensive test results (Mean $\pm$ SD, 95% CI, $n = 10$).

Performance Indicator	A*	RRT*	Bezier Curve	CO-PRE (Proposed)	p-value (vs. CO-PRE)
Success rate (%)	81.6 $\pm$ 1.5 (CI: $\pm$ 0.9)	87.4 $\pm$ 1.3 (CI: $\pm$ 0.8)	73.8 $\pm$ 2.1 (CI: $\pm$ 1.2)	96.2 $\pm$ 1.2 (CI: $\pm$ 0.7)	< 0.01
Average path deviation (m)	1.52 $\pm$ 0.07 (CI: $\pm$ 0.04)	1.13 $\pm$ 0.05 (CI: $\pm$ 0.03)	0.96 $\pm$ 0.04 (CI: $\pm$ 0.02)	0.65 $\pm$ 0.03 (CI: $\pm$ 0.02)	< 0.01
Mapping delay (ms)	219 $\pm$ 8 (CI: $\pm$ 4.8)	192 $\pm$ 6 (CI: $\pm$ 3.6)	240 $\pm$ 10 (CI: $\pm$ 6.0)	158 $\pm$ 5 (CI: $\pm$ 3.0)	< 0.01
Average energy consumption (J/m)	4.38 $\pm$ 0.12 (CI: $\pm$ 0.07)	4.01 $\pm$ 0.10 (CI: $\pm$ 0.06)	3.77 $\pm$ 0.09 (CI: $\pm$ 0.05)	3.42 $\pm$ 0.08 (CI: $\pm$ 0.05)	< 0.01
CPU load (%)	42.1 $\pm$ 2.5 (CI: $\pm$ 1.5)	48.5 $\pm$ 2.3 (CI: $\pm$ 1.4)	30.4 $\pm$ 1.8 (CI: $\pm$ 1.1)	51.9 $\pm$ 2.4 (CI: $\pm$ 1.4)	0.06
Replanning count (times/task)	6.1 $\pm$ 0.3 (CI: $\pm$ 0.2)	4.8 $\pm$ 0.3 (CI: $\pm$ 0.2)	7.3 $\pm$ 0.4 (CI: $\pm$ 0.2)	2.4 $\pm$ 0.2 (CI: $\pm$ 0.1)	< 0.01
Computational complexity (ms)	184 $\pm$ 6 (CI: $\pm$ 3.6)	142 $\pm$ 5 (CI: $\pm$ 3.0)	169 $\pm$ 7 (CI: $\pm$ 4.2)	127 $\pm$ 4 (CI: $\pm$ 2.4)	< 0.01

Finally, multiple synthetic environments were built using Gazebo and AirSim, and evaluated through repeated testing under varied start-end configurations and dynamic conditions to assess system robustness and generalization. The tested scenarios included: (1) a structured urban environment with dense buildings and grid-like corridors; (2) canyon-like terrain with narrow paths and elevation changes; and (3) indoor scenes simulating limited field of view and irregular obstacle layouts. Each environment incorporates factors such as dynamic obstacle updates, partial map observability, and sensor bandwidth

constraints to emulate real-world mission challenges. Evaluation metrics were averaged across multiple runs under consistent conditions, the results are shown in Table 5.

The average path deviation was used to measure the average spatial deviation between the flight trajectory and the reference path. It was calculated by sampling at equal time intervals during the mission and counting the mean Euclidean distance between the actual trajectory point and the reference trajectory point. Its expression was

$\frac{1}{N} \sum_{i=1}^N |p_i^{act} - p_i^{ref}|_2$, p_i^{act} , p_i^{ref} represented the actual trajectory point and the reference trajectory point at that moment, respectively, and N was the total number of samples. Mapping delay represented the time interval from the reception of sensor data to the completion of the corresponding occupancy grid map update, which was calculated using the topic timestamp in the ROS system. The CPU load indicator was sampled and recorded at a frequency of 1 Hz through the system performance monitoring tool. It was continuously monitored and its average value was calculated during the entire navigation mission, reflecting the system resource consumption level under each sensor configuration.

In Table 5, the success rate of the proposed CO-PRE algorithm reached 96.2% (± 1.2 , CI: ± 0.7), indicating reliable task completion. The average path deviation was maintained at 0.65 ± 0.03 meters, significantly lower than those of other methods ($p < 0.01$). Additionally, the system demonstrated reduced mapping delay and computational complexity, with energy consumption controlled at 3.42 ± 0.08 J/m. Although the CPU load was slightly higher than some baseline methods, the overall performance gain was statistically supported. All metrics were averaged over 10 trials, and 95% confidence intervals were reported. The p-values derived from paired t-tests confirmed that CO-PRE's improvements in path accuracy, efficiency, and obstacle re-planning were statistically significant.

4 Discussion

To comprehensively evaluate the performance of the proposed CO-PRE algorithm, the study compared it with the classic RRT and Bezier methods, focusing on indicators such as path smoothness, obstacle avoidance adaptability and system computational efficiency. The results showed that in complex environments, the average path deviation of CO-PRE was controlled at 0.65m, which was better than RRT's 1.13m and Bezier's 0.96m, with a higher success rate and fewer replanning times. In the test, the system showed good response speed and control accuracy during autonomous flight. The x-axis and y-axis basically converged within 2 seconds, and the z-axis maintained a fluctuation range of ± 0.1 meters.

Jayarajan et al. proposed the ABC-RRT* hybrid algorithm in the literature [11], but due to the randomness of the sampling tree structure, its path smoothness and local adaptability were still insufficient. In contrast, CO-PRE introduced a sliding window curvature optimization and feasibility judgment mechanism to generate more practical paths in high-density scenarios such as narrow urban passages. In the performance test, when the complexity level was 5, the path lengths of A*, PSO, P-RRT, and CO-PRE were 18.18 m, 19.78 m, 14.26 m, and 11.83 m, respectively, and the planning times were 299.7 ms, 360.2 ms, 247.1 ms, and 205.8 ms, respectively. The LASSA algorithm proposed by Tong et al. in the literature [7] had advantages for multi-machine systems, but did not directly optimize the local smoothness and dynamic

feasibility of a single path. CO-PRE focused more on the execution of a single trajectory in complex three-dimensional scenes, and could better cope with the actual navigation needs in GPS failure or densely built areas.

In summary, the system proposed in the study not only ensured the geometric continuity of the path, but also took into account the dynamic feasibility, providing an effective path for building a navigation system with high robustness and high executability. The research aimed to improve the interpretability and controllability of the UAV navigation system and provide a technical idea for realizing real-time planning under dynamic constraints.

5 Conclusion

To improve the smoothness and obstacle avoidance capability of UAV path planning in complex environments, a CO-PRE autonomous navigation algorithm integrating factor graph optimization and curvature-constrained path smoothing strategy was proposed. Experimental results showed that the algorithm had higher navigation stability and control accuracy. In the application experiment, the system successfully completed the closed-loop flight mission of take-off-hover-translation-return, verifying the feasibility of its structural design and control strategy. In summary, CO-PRE not only showed excellent performance in simulation scenarios, but also had the potential for actual deployment.

However, the research still has certain limitations. The system was sensitive to sensor calibration errors and lacked explicit fault tolerance, making it dependent on the integrity of multi-sensor inputs. In GPS-denied environments or under wind disturbances, control accuracy may degrade, affecting tracking stability. Feedback latency between perception and control may limit responsiveness, and the current framework relies on local mapping without integration of SLAM or semantic maps, reducing adaptability in long-term or multi-scenario tasks. While planning latency remained under 100 ms, CO-PRE incurred a higher CPU load than lightweight baselines such as Bezier, highlighting the need for further optimization. Future work will address redundancy-aware perception, real-time control robustness, and hardware-level deployment to improve system reliability in dynamic real-world environments.

References

- [1] Veselov Gennady, Sharma Ashutosh, Tselykh Alexey, and Huang Ruihang. Applications of artificial intelligence in evolution of smart cities and societies. *Informatica*, 45(5):603, 2021. <https://doi.org/10.31449/inf.v45i5.3600>
- [2] Soheila Ghambari, Mahmoud Golabi, Laetitia Jourdan, Julien Lepagnot, and Lhassane Idoumghar. UAV path planning techniques: A survey. *RAIRO-Operations Research*, 58(4):2951-2989, 2024. <https://doi.org/10.1051/ro/2024073>
- [3] Lisha Dong. Improved A* algorithm for intelligent navigation path planning. *Informatica*, 48(10):181-194, 2024. <https://doi.org/10.31449/inf.v48i10.5693>

- [4] Daqian Liu, Bowen Fei, Weidong Bao, Xiaomin Zhu, and Xiaoqing Li. DAWN: Dynamic task planning of multi-UAV with two-layer optimization mechanism in uncertain environments. *IEEE Internet of Things Journal*, 11(23):37813-37830, 2024. <https://doi.org/10.1109/JIOT.2024.3440017>
- [5] Chaohua Yan. Application of a graphical image pre-retrieval method based on compatible rough sets to the self-localization method of mobile robots. *Informatica*, 48(11):85-96, 2024. <https://doi.org/10.31449/inf.v48i11.5508>
- [6] Xiuzhu Zhang. Path planning and control of intelligent delivery UAV based on internet of things and edge computing. *International Journal of Advanced Computer Science & Applications*, 15(3):1068-1076, 2024. <https://doi.org/10.14569/ijacsa.2024.01503107>
- [7] Tong Han, Andi Tang, Huan Zhou, Dengwu Xu, and Lei Xie. Multiple UAV cooperative path planning based on LASSA method. *Journa of Systems Engineering and Electronics*, 44(1):233-241, 2022. <https://doi.org/10.12305/j.issn.1001-506X.2022.01.29>
- [8] Hamid Shiri, Hywoon Seo, Jihong Park, and Mehdi Bennis. Attention-based communication and control for multi-UAV path planning. *IEEE Wireless Communications Letters*, 11(7):1409-1413, 2022. <https://doi.org/10.1109/LWC.2022.3171602>
- [9] Nouman Bashir, Saadi Boudjit, and Gabriel Dauphin. A connectivity aware path planning for a fleet of UAVs in an urban environment. *IEEE Transactions on Intelligent Transportation Systems*, 24(10):10537-10552, 2023. <https://doi.org/10.1109/TITS.2023.3280995>
- [10] Yun-Chun Ko, and Rung-Hung. UAV velocity function design and trajectory planning for heterogeneous visual coverage of terrestrial regions. *IEEE Transactions on Mobile Computing*, 22(10):6205-6222, 2022. <https://doi.org/10.1109/TMC.2022.3182975>
- [11] Nireesh Jayarajan, Tamilselvan Ganesan, and Archana Naganathan. Optimal trajectories for UAV three-dimensional path planning using a hybrid ABC-RRT* algorithm. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 239(3):930-943, 2025. <https://doi.org/10.1177/09544062241288633>
- [12] Jiehong Wu, Ya'nan Sun, Danyang Li, Junling Shi, Xianwei Li, Lijun Gao, Lei Yu, Guangjie Han, and Jinsong Wu. An adaptive conversion speed Q-learning algorithm for search and rescue UAV path planning in unknown environments. *IEEE Transactions on Vehicular Technology*, 72(12):15391-15404, 2023. <https://doi.org/10.1109/TVT.2023.3297837>
- [13] Xiaofeng Liu, Zhong-Ren Peng, Li-Ye Zhang, and Qiang Chen. Real-time and coordinated UAV path planning for road traffic surveillance: A penalty-based boundary intersection approach. *International Journal of Control, Automation and Systems*, 20(8):2655-2668, 2022. <https://doi.org/10.1007/s12555-020-0565-8>
- [14] Linzhen Shi, Wensong Jiang, Zai Luo, Li Yang, and Yan Wang. Enhancing adaptability: Hierarchical frontier-based path planning for navigation in challenging environments. *IEEE Robotics and Automation Letters*, 9(10):8611-8616, 2024. <https://doi.org/10.1109/LRA.2024.3448137>
- [15] Rohit Dujari, Brijesh Patel, and Bhumeswar K. Patle. Fast and efficient drone path planning using riemannian manifold in indoor environment. *Automation*, 5(3):450-466, 2024. <https://doi.org/10.3390/automation5030026>
- [16] Gilang Nugraha Putu Pratama, Oktaf Agni Dhewa, Ardy Seto Priambodo, Faris Yusuf Baktiar, Rizky Hidayat Prasetyo, Mentari Putri Jati, and Indra Hidayatulloh. Benchmark analysis of sampling methods for RRT path planning. *Control Systems and Optimization Letters*, 2(2):217-233, 2024. <https://doi.org/10.59247/csol.v2i2.132>
- [17] S. Julius Fusic, and R. Sitharthan. Improved RRT* algorithm-based path planning for unmanned aerial vehicle in a 3D metropolitan environment. *Unmanned Systems*, 12(5):859-875, 2024. <https://doi.org/10.1142/S2301385024500225>
- [18] Ramirez-Neria M, Morales-Valdez J, Yu W. Active vibration control of building structure using active disturbance rejection control. *Journal of Vibration and Control*, 2022, 28(17-18): 2171-2186. doi: <https://doi.org/10.1177/10775463211009377>
- [19] Lianghong Zhu, Guoqiang Zhang, Runze Jing, Guangdong Bi, Runhua Xiang, Gaolin Wang, and Dianguo Xu. Nonlinear active disturbance rejection control strategy for permanent magnet synchronous motor drives. *IEEE Transactions on Energy Conversion*, 37(3):2119-2129, 2022. <https://doi.org/10.1109/TEC.2022.3150796>
- [20] Suiyuan Shen, Jinfa Xu, Pei Chen, and Qingyuan Xia. An intelligence attitude controller based on active disturbance rejection control technology for an unmanned helicopter. *IEEE Transactions on Vehicular Technology*, 72(3):2936-2946, 2022. <https://doi.org/10.1109/TVT.2022.3217300>
- [21] Jingqing Han. From PID to active disturbance rejection control. *IEEE transactions on Industrial Electronics*, 56(3):900-906, 2009. <https://doi.org/10.1109/TIE.2008.2011621>
- [22] Zhenlong Wu, Zhiqiang Gao, Donghai Li, Yangquan Chen, and Yanhong Liu. On transitioning from PID to ADRC in thermal power plants. *Control Theory and Technology*, 2021, 19:3-18, 2021. <https://doi.org/10.1007/s11768-021-00032-4>