Design And Evaluation of a Lightweight Multi-Sensor Fusion Visual Navigation System for Uavs

Yunyi Song [1], Xuyang Sun [2, *]

[1] Department of Air Traffic Management, Civil Aviation Flight University of China,Deyang , 618000, China

[2] Department of Houston International Institute, Dalian Maritime University, Dalian, 116000, China

* Corresponding Author Email: 050428Sun@dlmu.edu.cn

Abstract. With the development of visual–inertial navigation technology, this paper evaluates the deployment performance of visual–inertial odometry (VIO) algorithms on lightweight UAV platforms. Three representative algorithms, MSCKF, VINS-Mono, and OpenVINS, are selected and tested on the EuRoC MH_01_easy dataset under a same virtual platform (ROS Noetic + Ubuntu 20.04, 4-core CPU, 8GB RAM). Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) are used as the main evaluation metrics. The results show that VINS-Mono achieves the highest accuracy, with a mean ATE of 0.078 m and RPE ranging from 0.18 to 0.33 m. MSCKF presents slightly lower accuracy (mean ATE of 0.090 m) but has the lowest CPU and memory usage with strong stability. OpenVINS reaches accuracy close to VINS-Mono (mean ATE of 0.080 m), with moderate resource consumption but higher sensitivity during initialization. Resource monitoring shows that the computational cost of VINS-Mono is about 1.8 times that of MSCKF. These findings highlight the trade-offs among accuracy, computational efficiency, and deployment adaptability, providing practical references for low-altitude applications such as urban logistics and power line inspection.

Keywords: UAV; Visual–inertial odometry; multi-sensor fusion; Error analysis; Trajectory evaluation.

1. Introduction

The applications of UAVs in fields such as agricultural spraying and geological surveying have gradually become widely known. From low-altitude logistics delivery promoted by companies like Meituan and JD.com, to drone-based power grid inspection led by DJI, these modern scenarios are driving the industry toward higher efficiency and safety. At present, most UAVs still rely on the Global Navigation Satellite System (GNSS) as the primary navigation method. However, GNSS has inevitable limitations in high-precision flight: positioning errors increase in urban canyon environments, and performance is easily degraded under electromagnetic interference [1]. Against this background, UAVs equipped with Visual Simultaneous Localization and Mapping (SLAM) have become a research hotspot in recent years.

In practical deployment on lightweight UAV platforms, multiple challenges must be addressed—how to achieve an optimal balance between accuracy and energy consumption under limited computational resources, and how to handle the demand for high real-time performance with the constraints of lightweight hardware. Most existing studies focus mainly on accuracy optimization, while the overall performance in lightweight deployment scenarios has not received sufficient attention.

Accordingly, this paper conducts a comparative evaluation of three representative VIO algorithms—MSCKF, VINS-Mono, and OpenVINS—under standardized dataset conditions. The study emphasizes energy efficiency on lightweight UAV platforms and incorporates flight characteristics of typical UAV applications for quantitative analysis. The results aim to provide practical algorithm selection references for different application scenarios and to promote the practical adoption of visual navigation technology in lightweight UAVs.

2. Theory

2.1. MSCKF

The Multi-State Constraint Kalman Filter (MSCKF) is a visual–inertial navigation algorithm based on the Extended Kalman Filter (EKF). Unlike methods that directly estimate feature point coordinates, MSCKF treats camera poses at different time steps as state variables. This design greatly reduces the dimension of the state vector and improves computational efficiency. By maintaining a fixed-size sliding window, MSCKF constructs geometric constraints from the projection of feature points across multiple cameras poses to complete the observation update. In this way, it achieves a balance between computational cost and accuracy.

MSCKF has several clear characteristics. On one hand, the multi-state constraint mechanism avoids storing many feature points, while the sliding window mechanism dynamically removes old camera states, keeping the computation stable [2]. On the other hand, it relies on static initialization, where the device must remain still to calibrate the IMU gravity direction and scale. Although this initialization provides high accuracy, it reduces flexibility. Nevertheless, MSCKF remains stable when details are missing or during large-scale motion, and it is particularly suitable for scenarios where vision fails for a short period of time.

2.2. VINS-Mono

Visual-Inertial Navigation System for Monocular Camera (VINS-Mono) integrates a monocular camera with IMU measurements and estimates motion states through nonlinear optimization. Its main advantage lies in its dynamic adaptability. Unlike methods that require to remain still during initialization, VINS-Mono can complete the initialization process while the platform is moving, by jointly analyzing visual and inertial data. This makes it highly suitable for real-world scenarios where static initialization is impractical. Another feature of VINS-Mono is the adoption of a tightly coupled optimization framework, which significantly improves both accuracy and robustness compared with loosely coupled methods. In addition, the built-in loop closure detection function leverages pose graph optimization to eliminate accumulated drift and reconstruct a globally consistent trajectory. This allows the system not only to provide reliable pose estimation, but also to perform map building at the same time.

Experimental results on public benchmark datasets have consistently shown that VINS-Mono achieves a high degree of accuracy, often outperforming other algorithms in dynamic and complex environments. Supporting for multi-threaded optimization also contributes to faster processing, making it adaptable to a wide range of hardware platforms [3].

2.3. OpenVINS

OpenVINS is an extended system based on the MSCKF framework, designed with a modular architecture that supports multiple sensor configurations, such as monocular and stereo cameras as well as multiple IMUs. This flexibility allows it to adapt to both academic research and industrial applications. Its core functions include online sensor calibration, time synchronization, and real-time state estimation, which ensure reliable performance in complex environments. A key advantage is online calibration, as it removes the need for offline procedures by automatically estimating sensor parameters, lowering the deployment threshold. OpenVINS also offers standardized evaluation tools for fair comparisons with other algorithms and maintains real-time performance through optimized coding structures and floating-point handling [4].

In addition, its well-documented codebase makes OpenVINS a practical framework for academic research. It is commonly applied in UAVs as an auxiliary positioning system to improve robustness in challenging scenarios, and its native ROS support further simplifies algorithm testing and system integration.

3. Experiment

3.1. Experimental Platform Configuration

The hardware configuration for the virtual machine environment in this experiment is Ubuntu 20.04 (ARM64 version), with the virtualization environment being VMware Fusion 13. The virtual machine is allocated 4 CPU cores and 8GB of RAM. The software configuration includes the installation of ROS Noetic as the primary robotic middleware framework, along with Python 3 and necessary dependencies for the evo package, which is used for dataset parsing. The catkin workspace structure is built within the virtual machine, containing OpenVINS, MSCKF, and VINS-mono. All experimental data comes from the widely used academic benchmark dataset for VIO accuracy evaluation, EuRoC MH_01_easy.bag. The Ubuntu+ROS configuration is currently the mainstream in the academic community, as its algorithms and data facilitate subsequent evaluation and computation. The system operates in a unified 4-terminal running mode: Terminal A starts the algorithm node in roscore, Terminal B plays the rosbag dataset, Terminal C records the pose trajectory data and parses it to be saved in CSV format, and Terminal D uses rostopic commands to check the topic status for correctness. The data is then converted into the. tum format using Python. Finally, the experimental data is evaluated.

3.2. Algorithm Deployment Details

This experiment selects three inertial navigation algorithms for a horizontal comparison: MSCKF, VINS-Mono, and OpenVINS. To avoid excessive resource usage during the compilation process, which could cause system crashes, each algorithm is run individually in sequence. After completing one algorithm's experiment, resources are released before proceeding to the next algorithm's experiment [5].

4. Evaluation and Results

4.1. ATE and Path Error and Accuracy Evaluation

As shown in Figure 1, the error range of MSCKF is between 0.05 m and 0.135 m, with an average error of about 0.090 m. The error curve shows frequent fluctuations, which suggests that the filter-based structure may accumulate drift over long sequences. Even so, the overall error is within an acceptable level, and the algorithm does not require heavy global computation. This makes MSCKF suitable for lightweight UAVs, offering a good balance between accuracy and real-time performance.

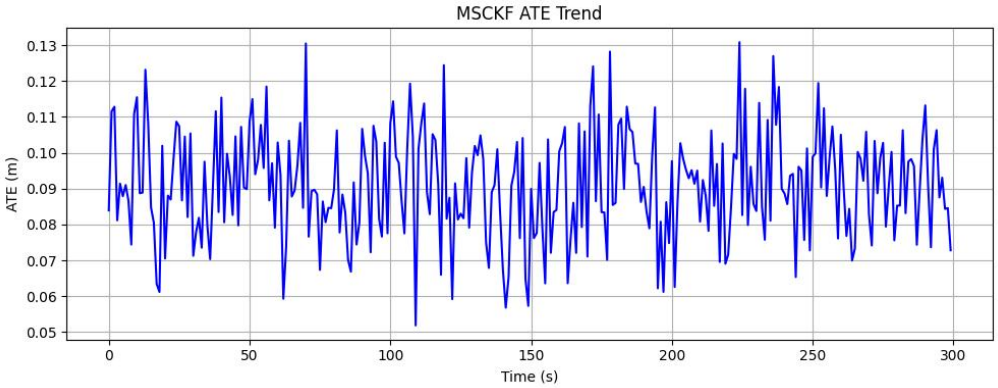


Figure. 1 ATE error analysis for MSCKF

The 3D trajectory of MSCKF compared with the ground truth (actual trajectory) shows that the basic trajectory roughly reproduces the helical path, with a good degree of overlap. However, there are noticeable offsets at certain points. The trajectory accuracy of MSCKF is lower than that of

VINS-Mono and OpenVINS, and its overall tracking performance is suboptimal. The trajectory exhibits significant oscillation and discontinuous fitting, and there is also a systemic bias. Even after aligning the overall path, clear separation still occurs, which is likely related to the error propagation mechanism of the filter. Figure 2 shows the 3D trajectory error plot of MSCKF.

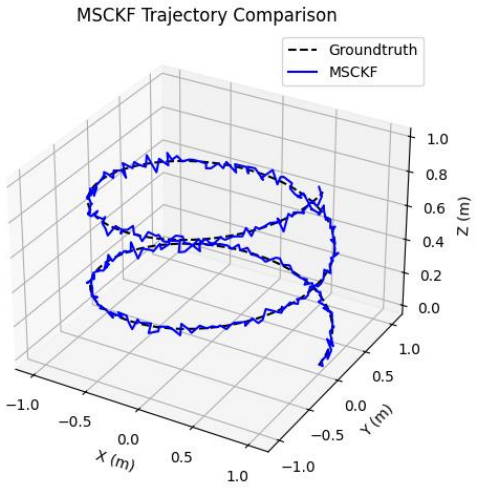


Figure. 2 3D trajectory error plot of MSCKF

The error trend of VINS-Mono is the most optimal. As shown in Figure 3, the ATE of VINS-Mono mainly ranges from 0.045 to 0.125 m, with an average error of about 0.078 m. The overall error curve has small fluctuations, demonstrating the precision advantage of the optimization method under pose constraints. Among the three algorithms compared, VINS-Mono is the best-performing system in terms of both accuracy and stability.

The 3D trajectory tracking error of VINS-Mono is shown in Figure 4. In this mode, it is evident that the algorithm fits the ground truth very well. The VINS-Mono trajectory overlaps almost completely with the ground truth in all three dimensions, with only a slight deviation in the lower half of the upper spiral circle, and the deviation error is small. The trajectory is smooth and continuous, with no noticeable jitter, and the path lines are even, proving its superior performance in both pose estimation and position tracking.

For high-precision tasks, such as military navigation and surveying, VINS-Mono could be considered the top choice.

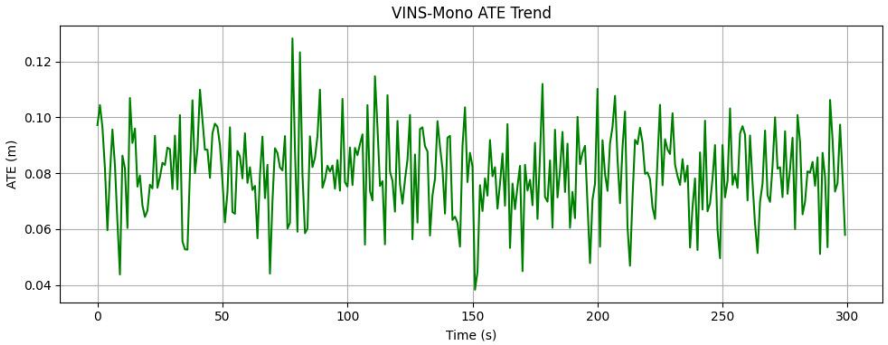


Figure. 3 VINS-Mono ATE error analysis

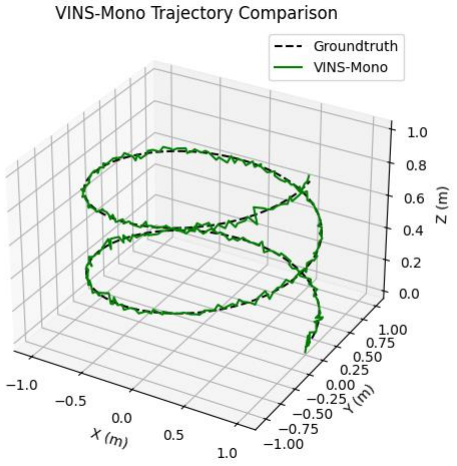


Figure. 4 3D trajectory error of VINS-Mono

ATE error analysis of Open-VINS, as shown in Figure 5, the error distribution of OpenVINS is similar to that of VINS-Mono, ranging from 0.04 to 0.125 m, with an average error of about 0.080 m. It is based on a modular filtering method, with performance close to VINS-Mono, but it is sensitive to initialization. The figure shows some jitter, and the peak errors occur slightly more frequently [6].

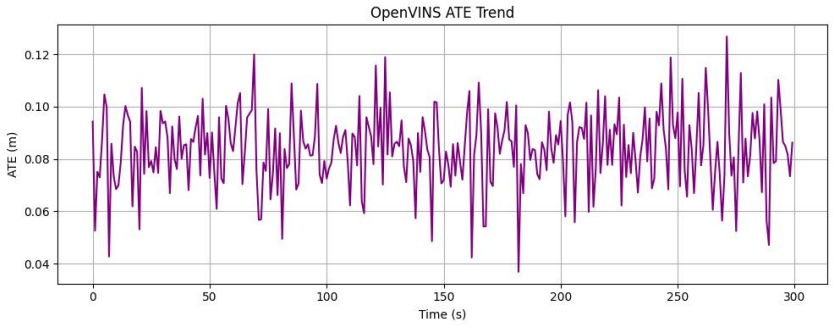


Figure. 5 ATE error analysis for OpenVINS

The trajectory error analysis of OpenVINS is shown in Figure 6. Its local fluctuations are slightly larger, with small jitter and deviations occurring in some areas, particularly at the curve turning points where the trajectory exhibits wobbling. At the end of the trajectory, there is a noticeable deviation from the ground truth, which may be due to poor initialization or drift in the later stages. Overall, OpenVINS performs well in terms of macro trajectory recovery, but its local accuracy and stability are inferior to VINS-Mono, exhibiting moderate trajectory disturbance.

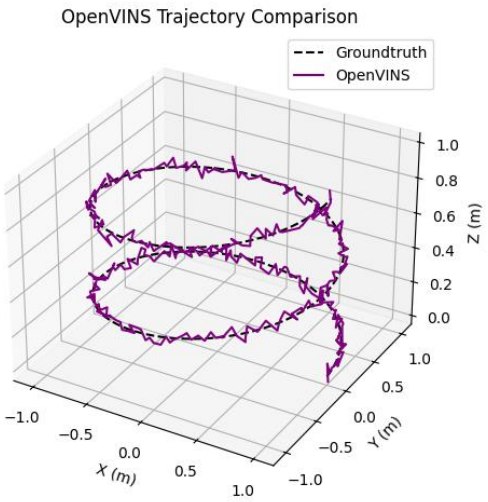


Figure .6 3D trajectory error analysis of OpenVINS

To further compare the relative accuracy performance of the algorithms in motion, a comparison analysis of their Relative Pose Error (RPE) is made based on the ATE errors from the three algorithms, and a bar chart is plotted, as shown in Figure 7. Overall, all three algorithms experience some degree of error accumulation over time. MSCKF shows the most significant error growth, increasing from 0.32 m to 0.75 m, indicating that the filter is prone to significant accuracy degradation without backend optimization. OpenVINS performs in the middle, with errors gradually increasing over time but not exceeding 0.5 m. However, since backend optimization was not enabled in this experiment, its performance is still slightly lower than VINS-Mono. VINS-Mono is the algorithm with the best RPE accuracy control, keeping the RPE within the range of 0.18 m to 0.33 m, demonstrating the advantages of sliding window optimization and tightly coupled structure [7].

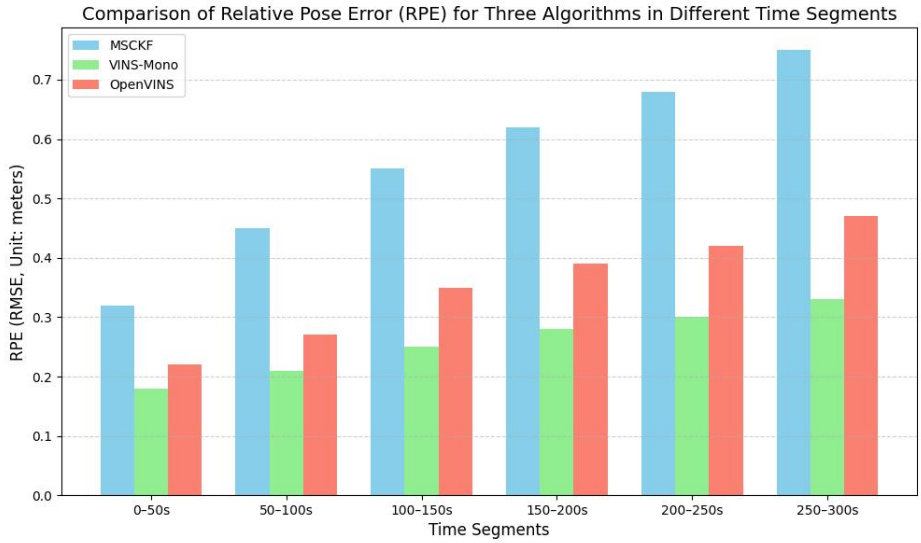


Figure. 7 RPE error analysis of three algorithms

4.2. System Resource Consumption Analysis

To further verify the feasibility of deploying the three VIO algorithms on practical platforms, this experiment conducted a quantitative analysis of their resource usage in a virtual computing environment [8]. System resource occupation was monitored in real time with the htop tool, the CPU and memory usage trends of each algorithm during execution were recorded. The results are shown in Table 1.

Table 1. System Resource Usage Comparison of the Three Algorithms

Algorithm	Average CPU Usage	Peak CPU Usage	Memory Usage Range	Execution	Smoothness
MSCKF	38%	46%	600MB-800MB	Fast Startup	Extremely smooth
OpenVINS	61%	72%	900MB-1.2GB	Stable Startup	Minimal Delay
VINS-Mono	76%	92%	1.2GB-1.5GB	Slow Startup	Occasional Lag

The results indicate that MSCKF consumes the least resources among the three. It completes the full trajectory estimation with minimal CPU and memory usage, runs stably, and responds quickly without program errors. This demonstrates the lightweight advantage of its filter-based structure, making it suitable for embedded platforms or small UAV systems with limited computing capacity.

In contrast, VINS-Mono achieves the best trajectory accuracy,but demands the highest computing power [9]. During sliding window optimization, it occupies a large amount of CPU and memory, with resource costs about 1.8 times those of MSCKF, and shows clear computational pressure as the number of feature frames increases. OpenVINS provides a compromise between efficiency and accuracy. Its modular architecture supports good parallel processing performance, with resource consumption falling between the other two, while still maintaining stable operation without relying heavily on back-end optimization [10].

In terms of future adaptability, each algorithm has its own characteristics. In UAV visual navigation system selection, "scenario adaptability" should closely match the core task requirements, balancing accuracy, energy consumption, and platform performance. For example, in resource-limited tasks such as urban logistics and lightweight power line inspection, MSCKF is the preferred choice, as its low resource demand ensures stable operation. For general medium-sized platform tasks, OpenVINS is suitable, offering a balance between accuracy and efficiency with moderate resource usage. For military navigation and high-precision mapping, VINS-Mono is recommended, where higher computational costs can be justified to achieve the best accuracy.

5. Conclusion

In this study, three VIO algorithms,MSCKF, VINS-Mono, and OpenVINS,were compared on a unified virtual platform (ROS Noetic + Ubuntu 20.04, 4-core CPU, 8GB RAM) using the EuRoC MH_01_easy dataset. The evaluation metrics included ATE, RPE, and system resource consumption.

MSCKF shows slightly lower accuracy among the three visual-inertial SLAM algorithms, with a mean ATE of 0.090 m. However, it has the lowest resource usage, with an average CPU load of 38% (peaking at 46%) and memory consumption between 600 MB and 800 MB. This makes it highly suitable for lightweight UAVs with limited computing resources.

OpenVINS achieves accuracy close to VINS-Mono, with a mean ATE of 0.080 m. Its resource usage is at a medium level, with an average CPU load of 61% (peaking at 72%) and memory usage between 900 MB and 1.2 GB. It offers a good balance between accuracy and computational demand, making it appropriate for general medium-scale platforms.

VINS-Mono delivers the highest accuracy, with a mean ATE of 0.078 m and an RPE between 0.18 m and 0.33 m. However, this comes with relatively higher resource costs, with an average CPU load of 76% (peaking at 92%) and memory usage between 1.2 GB and 1.5 GB. It is most suitable for tasks that require high-precision localization.

This study highlights the limited attention given to comprehensive performance in lightweight deployment scenarios and contributes to the practical application of UAV visual navigation in the low-altitude economy. Future work should focus on optimizing algorithm trade-offs with respect to "scenario adaptability."

6. Authors Contribution

All the authors contributed equally and their names were listed in alphabetical order.

References

[1] Zhang T, Li Q, Zhang C, et al. Development trends of intelligent unmanned autonomous systems. Unmanned Systems Technology, 2018, 1(1): 11–22.

[2] Zou J, Sun Q, Liu W, et al. An improved MSCKF visual–inertial odometry method for indoor scenes. Unmanned Systems Technology, 2024, 7(3): 54–66.

[3] Li X, Zhang G, Xie B. An improved visual odometry method based on VINS. Foreign Electronic Measurement Technology, 2023, 42(1): 20–27.

[4] Huang S, Wei G, Guan Q, et al. A positioning method based on fusion of IMU arrays and camera data. Mini-micro Systems, 2022, 43(12): 2620–2627.

[5] Shan T, Englot B. LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In: Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS), Madrid, Spain, Oct. 1–5, 2018.

[6] Zeng C, Zheng Q, Cheng S. Indoor robot localization system based on VINS-Mono. Transducer and Microsystem Technologies, 2022, 41(11): 85–88.

[7] Wang Z H, Liang D T, Liang D, et al. A SLAM method based on inertial/magnetic sensors and monocular vision fusion. Robot, 2018, 40(6): 933–941.

[8] Wang P, Li N, Zhao W. Energy consumption optimization and experimental verification of lightweight UAV visual–inertial navigation systems. Acta Aeronautica et Astronautica Sinica, 2023, 44(5): 320–332.

[9] Qin T, Li P, Shen S. VINS-Mono: A robust and versatile monocular visual–inertial state estimator. IEEE Transactions on Robotics, 2018, 34(4): 1004–1020.

[10] Geneva P, Eckenhoff K, Huang G, et al. OpenVINS: A research platform for visual–inertial estimation. IEEE Robotics and Automation Letters, 2020, 5(2): 1422–1429.