An Improved ORB-SLAM2 Algorithm Based on Extended Kalman Filtering and Particle Swarm Optimization

Hongyan Chen
School of Electrical Engineering
University of Jinan, China

Wei Qi Yan
Auckland University of Technology
Auckland, New Zealand

Qinjun Zhao
School of Electrical Engineering
University of Jinan, China

Yueyang Li
School of Electrical Engineering
University of Jinan, China

Abstract*—To address the challenges of pose estimation jitter and cumulative errors in visual SLAM systems operating in complex dynamic environments, this paper proposes an enhanced ORB-SLAM2 algorithm that integrates an Extended Kalman Filter (EKF)-based frontend pose smoother with a Particle Swarm Optimization (PSO)-driven backend optimizer. At the frontend, the EKF pose smoother fuses Inertial Measurement Unit (IMU) data with visual odometry to correct camera trajectories in real-time, effectively suppressing short-term pose drift. At the backend, the PSO algorithm dynamically optimizes node constraints in the pose graph and adaptively adjusts loop-closure thresholds, refining the weights of reprojection errors to enhance consistency between mapping outputs and the original environment. Experimental results demonstrate that compared to the baseline ORB-SLAM2 and PV-LIO algorithms, the proposed method significantly improves efficiency across three map scales—reducing mapping time and scanning duration while maintaining mapping quality—and achieves notable error suppression.

Keywords*—Particle Swarm Optimization; Oriented FAST and Rotated BRIEF-SLAM Second version; Extended Kalman Filter; Hybrid Optimization; Visual SLAM.

I. INTRODUCTION

Visual SLAM (Simultaneous Localization and Mapping) algorithm, as the core perception method for autonomous navigation systems, exhibits a distinct trajectory of technological iteration. In 2015, Mur-Artal et al. introduced the ORB-SLAM framework [1], realizing the first feature-based, fully functional visual SLAM system; it utilized ORB features for real-time camera tracking and map construction. Building on this, the same team released ORB-SLAM2 (Oriented FAST and Rotated BRIEF-SLAM Second version) in 2017 [2], which significantly enhanced system accuracy and robustness through the incorporation of multi-map management and loop closure detection techniques, thereby elevating it to a benchmark in the visual SLAM domain. Regarding sensor fusion, the OKVIS system pioneered by Leutenegger et al. in 2016 [3] established the foundation for tightly-coupled visual-inertial integration by employing pre-integration techniques to achieve efficient fusion of IMU and visual data. Further advancing visual-inertial SLAM, Qin et al. proposed the VINS-Mono system in 2018 [4], which combined sliding-window optimization with marginalization strategies to improve system stability. In 2020, Campos et al. developed OpenVSLAM [5], demonstrating substantial gains in mapping efficiency for large-scale environments via enhancements to the Bag-of-Words model and parallelized processing.

Despite significant advancements in the existing research work, visual SLAM systems continue to face several critical technical bottlenecks. As demonstrated by Yang's study in 2021 [6], the estimation error of the Extended Kalman Filter (EKF) increases exponentially when the system nonlinearity exceeds a critical threshold. Furthermore, a fundamental trade-off exists between optimization efficiency and accuracy in the backend optimization module. Comparative experiments [7] confirmed that the computational complexity of graph optimization-based methods escalates sharply beyond 1,000 pose nodes, reaching levels unacceptable for real-time performance. Additionally, empirical data [8] indicates a 2 to 3-fold increase in localization error for existing SLAM systems operating in environments where dynamic objects constitute over 30% of the scene. While the ORB-SLAM2 algorithm is renowned for its robust system architecture, real-time performance, and high precision [9], traditional visual SLAM algorithms exhibit significantly degraded performance under dark or highly dynamic conditions despite excelling in well-lit environments. Current mainstream frontend implementations predominantly employ the Extended Kalman Filter (EKF) for IMU and odometry data fusion [10].

However, the inherent limitations of the EKF in handling nonlinear problems become increasingly pronounced as environmental dynamics intensify [11]. Although backend optimization—a critical component responsible for resolving global consistency issues and rectifying cumulative errors from frontend processing [12]-[14] can theoretically achieve globally optimal solutions through traditional graph-based methods, its practical application is frequently constrained by computational efficiency and real-time requirements.

In this paper, we propose an improved ORB-SLAM2 algorithm integrating EKF with Particle Swarm Optimization (PSO). The frontend employs an adaptive noise covariance mechanism in EKF to suppress time-varying sensor noise, while the backend adopts PSO for pose graph optimization, leveraging swarm intelligence to avoid local optima. The experiments confirm significantly enhanced accuracy and robustness, particularly in dynamic environments.

979-8-3315-8654-6/25/$31.00 ©2025 IEEE

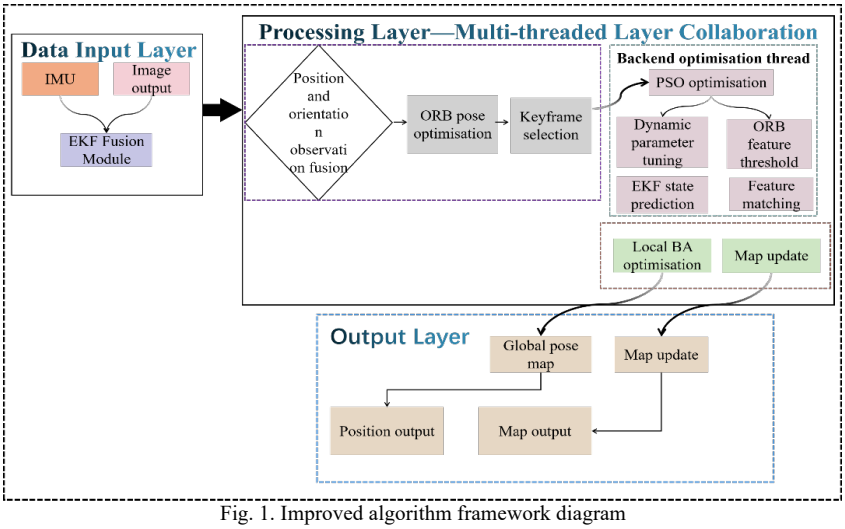


Fig. 1. Improved algorithm framework diagram

II. Research Methods

A. Overall Framework for Improving the Algorithm

Based on the ORB-SLAM2 framework, which integrates five modules and three operating threads, this paper introduces an enhanced algorithm incorporating an EKF module at the frontend to fuse IMU data with ORB pose observations for smoothed pose estimation. At the backend, Particle Swarm Optimization (PSO) is employed to dynamically adjust EKF noise parameters (Q and R) and ORB feature thresholds. As illustrated in Figure 1, the improved system utilizes multimodal data fusion and optimization mechanisms for real-time localization and mapping: the data input layer synchronously acquires IMU measurements and RGB image streams, which are processed via an EKF fusion module for calibration and preliminary motion estimation; the processing layer adopts a multithreaded architecture encompassing parallel frontend optimization, backend optimization, and map construction—the frontend thread enhances feature stability through pose observation fusion, ORB feature refinement, and keyframe selection; the backend optimization thread leverages PSO to dynamically tune threshold parameters and optimize the EKF state prediction model. Meanwhile, the map construction thread handles Bundle Adjustment (BA) and incremental map updates.

In Figure 2, the algorithm establishes a closed-loop processing flow encompassing sensor data input, frontend real-time pose estimation, and backend pose graph optimization. Starting with image input, an EKF-based pose smoother fuses visual-inertial data and feeds into a tracking status monitoring module. This module continuously evaluates feature quality: upon successful detection, it initiates a keyframe insertion mechanism to trigger the local mapping thread for bundle adjustment and point cloud updates; if tracking fails, the relocalisation module is activated. For backend processing, the PSO algorithm dynamically optimizes threshold parameters—adjusting the ORB feature threshold and refining EKF noise covariance matrices (Q, R). These optimized parameters are fed back to frontend modules via a real-time feedback loop, thus forming an adaptive closed-loop feedback mechanism that continuously enhances system robustness against dynamic environmental disturbances and sensor uncertainties.

B. Front-end Optimization

EKF is a widely used recursive filtering algorithm for state estimation in dynamic systems. This algorithm achieves real-time estimation of system states by linearizing non-linear systems. The basic framework of the EKF consists of two main steps: State prediction and measurement update. During the state prediction phase, the EKF uses the system's dynamic model to predict the next state based on the current state and control input [15]. Specifically, the state vector x is predicted, representing the position and orientation of the robot and the covariance matrix $P_{t|t-1}$ is updated as (1)

$$\begin{cases} x_{t|t-1}=f\left(x_{t-1|t-1},u_t\right)+W_t \\ P_{t|t-1}=F_tP_{t-1|t-1}F_t^T+Q_t \end{cases} \tag{1}$$

where f is a nonlinear function describing the dynamics of the system, u_t is the control input, W_t is the process noise, F_t is the state transition matrix, and Q_t is the process noise covariance.

During the measurement update phase, the EKF combines newly acquired observation information to correct the predicted state. If the measurement model is z_t, x_t is the updated pose, and the updated state covariance is (2), where h is the observation model, z_t is the observation value, v_t is the measurement noise, and the state estimate is updated using Kalman gain K_t; H_t is the observation matrix, and I is the unit matrix. The primary advantage of the EKF lies in its computational efficiency and real-time performance, making it the preferred state estimation method for many dynamic systems.

$$\begin{cases} z_t=h(x_t)+v_t \\ \boldsymbol{x}_{t|t}=\boldsymbol{x}_{t|t-1}+K_t\left(z_t-h\left(x_{t|t-1}\right)\right) \\ P_{t|t}=(I-K_tH_t)P_{t|t-1} \end{cases} \tag{2}$$

979-8-3315-8654-6/25/$31.00 ©2025 IEEE

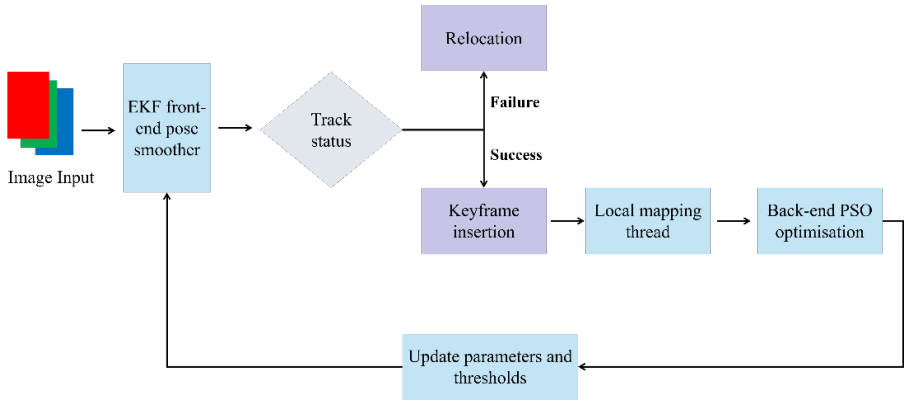


Fig. 2. Improve algorithm flow

EKF achieves state optimization in SLAM through third-order recurrence: Pose prediction is completed based on the nonlinear motion model, feature matching is used to construct the maximum likelihood association, and Bayesian map update is realized through Jacobian matrix linearization [16][17]. Although EKF has optimality in minimum variance prediction, its computational complexity and the number of features increase in a quadratic way, and it is easily affected by the accumulation of linearization errors, leading to a decline in estimation accuracy and pathological covariance matrix problems in high-dimensional feature spaces or dynamic environments [18]. In front-end processing, EKF is employed to track the status of robots in real time. The process consists of state prediction, covariance prediction, and measurement update, with (3) and (4). State prediction is the use of control input and motion models to predict the state of a robot at the next moment.

$$\hat{x}_{k+1|k}=f(\hat{x}_{k|k},u_k) \tag{3}$$

where u_k is the control input at time k. $\hat{x}_{k+1|k}$ the state estimate and control input at time k, and the predicted state at time $k+1$, that is, the state at time k predicted at time $k+1$. $\hat{x}_{k|k}$ is the optimal state prediction at time k. In (4), $P_{k|k}$ represents the uncertainty of the state estimation at time k, and Q_k is the process noise. We pass the state of the previous moment to the current and predicted moments, that is, pass the uncertainty $P_{k|k}$ of the previous moment to the current predicted moment through $F_k P_{k|k} F_k^T$. Among them, $P_{k|k-1}$ is the covariance matrix of the predicted state, H_k is the Jacobian matrix of the observation model, and R_k is the noise covariance matrix.

$$\begin{cases} P_{k+1|k}=F_k P_{k|k} F_k^T+Q_k \\ K_k=P_{k|k-1}H_k^T\left(H_k P_{k|k-1} H_k^T+R_k\right)^{-1} \end{cases} \tag{4}$$

By using (5), the difference between the actual observed value and the predicted value and the Kalman gain, the predicted state is corrected to obtain $\hat{x}_{k|k}$, which is the optimal estimate, is obtained. $P_{k|k}$ is the uncertainty covariance matrix of the updated state, representing the corrected state confidence.

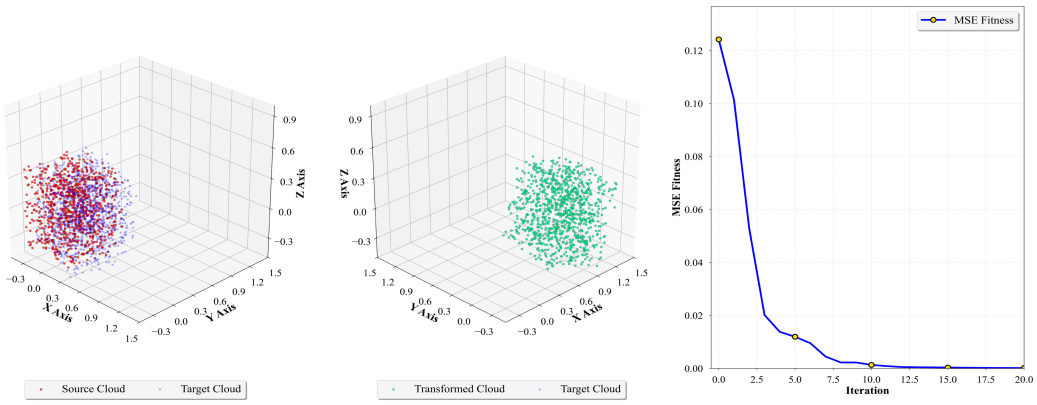
$$\begin{cases} \hat{x}_{k|k}=\hat{x}_{k|k-1}+K_k\left(z_k-\hat{z}_{k|k-1}\right) \\ P_{k|k}=(I-K_k H_k)P_{k|k-1} \end{cases} \tag{5}$$

C. Particle Swarm Optimization

Particle swarm optimization algorithm simulates the foraging behavior of bird flocks and searches the solution space through particle collaboration. Each particle dynamically adjusts the motion vector based on the individual optimal solution and the group optimal solution: The velocity update integrates self-learning and social learning factors, and the position update follows the inertia weight dynamic equation. During the iteration, particles explore in parallel through distributed information sharing. The self-organizing mechanism takes into account the convergence balance of diversity, achieving global optimization. The algorithm balances local and global search through a dual cognition model and has strong robustness to complex multimodal problems. In (6), the update of speed and position is shown.

$$\begin{cases} v_i(t+1)=w\cdot v_i(t)+c_1\cdot r_1\cdot\left(p_i-x_i(t)\right)+c_2\cdot r_2\cdot\left(g-x_i(t)\right) \\ x_i(t+1)=x_i(t)+v_i(t+1) \end{cases} \tag{6}$$

In equation (6), w represents the inertia weight, c_1 and c_2 are the learning factors, r_1 and r_1 are random numbers within the range of [0,1], p_i is the historical optimal position of the particle, and g is the global optimal position. In the back-end optimization, the Particle Swarm Optimization (PSO) algorithm is adopted to enhance the accuracy of state estimation. The particle swarm dynamically adjusts the velocity and position based on individual optimum and global optimum, gradually optimizing the state estimation. The velocity update and position update are shown in (7), where $P_{best,i}$ the historical optimal position of the i-th particle, and $c_1 r_1\left(P_{best,i}-x_i(t)\right)$ is the movement of the particle towards its own historical optimal position. g_{best} is the global optimal position, and $c_2 r_2\left(g_{best}-x_i(t)\right)$ rives the particle swarm to gather towards the global optimal position.



(a) Before PSO **(b) After PSO** **(c) Convergence Curve**

Fig. 3. Comparison of PSO point cloud configuration

$$\begin{cases} v_i(t+1)=\omega v_i(t)+c_1 r_1\left(P_{best,i}-x_i(t)\right)+c_2 r_2\left(g_{best}-x_i(t)\right) \\ x_i(t+1)=x_i(t)+v_i(t+1) \end{cases} \tag{7}$$

In global optimization, PSO addresses the cumulative error issue in SLAM systems by enabling global optimization of robot trajectories and feature point locations through multi-particle collaborative search [19], thereby effectively improving positioning accuracy. In the data association, PSO enhances the reliability of feature matching through parameter optimization. Each particle can encode the matching status of feature points, and the fitness function is constructed based on geometric consistency or observation likelihood, dynamically selecting the optimal matching combination [20].

Compared to traditional optimization methods, PSO exhibits higher adaptability in dynamic environments, enabling faster responses to environmental changes and real-time adjustments to position estimation and map updates. As shown in Figure 3(a), the initial distribution of unoptimized point clouds reveals significant spatial misalignment between the red source cloud and blue target cloud in the coordinate system, characterized by approximately 0.5-unit displacement along the X-axis and 0.3-unit displacement along the Y-axis; The optimization employed 50 particles over 100 iterations by using an inertia weight of 0.5 and acceleration coefficients of c_1=1.8 and c_2=2.2, while particle velocity was limited to [-0.2, 0.2] to prevent oscillatory behavior, Figure 3(b) demonstrates the spatial alignment effect after PSO-based registration, where the green transformed cloud achieves precise geometric matching with the target cloud following rigid transformation through 100 iterative optimizations. Figure 3(c) illustrates the convergence dynamics via fitness curves, highlighting an exponential 98% reduction in Mean Square Error (MSE) fitness within the first 5 iterations, with convergence approaching the global minimum by the 10-th iteration, thereby validating the algorithm's efficiency in minimizing registration error and accelerating convergence to optimal solutions under rigid transformation constraints.

III. Experiments

A. Improved ORB-SLAM2 Algorithm for Hybrid Optimization Mapping

To verify the effectiveness of the improved algorithm, simulation experiments were conducted using self-recorded data. This carried out the simulation in a VMware virtual machine, and the device configuration used: The operating system of Ubuntu20.04, the main frequency of CPU is 2994.375MHz, the running platform is ROS, the programming language used to develop the simulation system is python3.12, and the running files are edited in the Visual Studio Code1.86.2 version editor.

The improved algorithm was simulated by using Rviz in ROS. Figure 4 shows the simulation model of the robot in Rviz. From left to right, there are camera perspective, Gazebo construction, and Rviz. We see that the simulated map is basically consistent with the map in Gazebo.

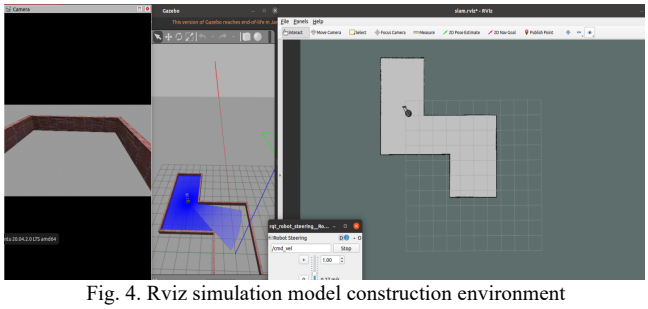


Fig. 4. Rviz simulation model construction environment

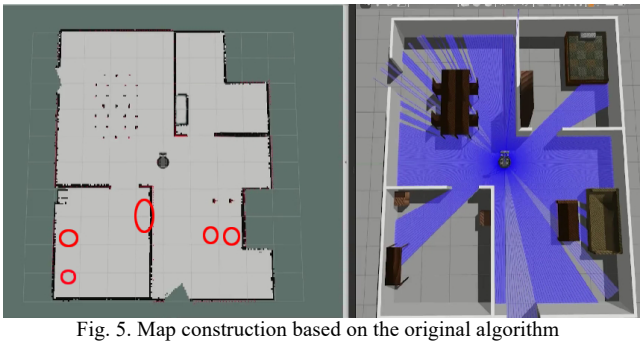


Fig. 5. Map construction based on the original algorithm

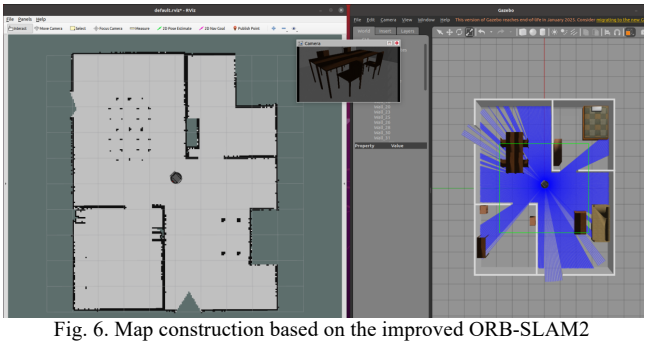


Fig. 6. Map construction based on the improved ORB-SLAM2 algorithm with hybrid optimization

We ensure that the ROS system is fully operational, environmental map construction proceeds by loading the robot model and conducting the SLAM simulation. Figure 5 illustrates the map generated by the original algorithm, while Figure 6 showcases the result from the hybrid-optimized ORB-SLAM2 algorithm. The comparison reveals that the improved algorithm demonstrates a noticeable improvement in mapping accuracy. Specifically, the structural features (circled in Figure 5) that were undetected by the original method are successfully recognized and displayed in Rviz when using the enhanced algorithm. Furthermore, the optimized approach reduces distortion along map edges, resulting in higher overall precision. This comparative analysis confirms that the proposed hybrid-optimized ORB-SLAM2 algorithm effectively achieves high-precision environmental mapping.

As shown in the comparative path test results in Table 1, the same A* path planning algorithm on maps was built by conventional ORB-SLAM2, PV-LIO (Point-Visual Lidar Inertial Odometry), and the hybrid-optimized ORB-SLAM2 variant—the proposed algorithm consistently outperforms both baseline methods across three map scales. It achieves shorter path lengths and reduced computation time, demonstrating significant improvements over the original ORB-SLAM2 and PV-LIO in terms of navigation efficiency and computational resource utilization.

TABLE I. Comparison of path tests after constructing maps of different sizes

Algorithm	Dimensions: 50 × 50		Dimensions: 100 × 100		Dimensions: 150×150	
	Length (m)	t(s)	Length (m)	t(s)	Length (m)	t(s)
ORB-SLAM2	58.41	0.031	118.66	0.069	169.45	0.092
PV-LIO	52.89	0.023	105.33	0.057	156.22	0.073
Improved algorithm	39.23	0.014	87.67	0.032	120.66	0.042
Compared to ORB-SLAM2	32.37%	54.84%	26.12%	53.62%	28.79%	54.35%
Compared to PV-LIO	25.82%	39.13%	16.77%	43.86%	22.76%	42.47%

According to the data in Table 1, the improved algorithm reduces the time required for path planning and significantly shortens the path length when constructing maps of three different sizes. The improvement in time is more obvious for maps of sizes 100×100 (m) and 150×150 (m). A more obvious comparison of the PV-LIO algorithm and the improved algorithm is shown in Figure 7.

In Figure 7, six colors are employed to clearly compare the path length and time consumption of the traditional PV-LIO algorithm and the improved algorithm at different sizes. The bar chart at the bottom compares the path lengths of the two algorithms, while the bar chart at the top compares the time consumption of the two algorithms. The improved ORB-SLAM2 algorithm with mixed optimization significantly reduces distortion and improves the accuracy of map construction.

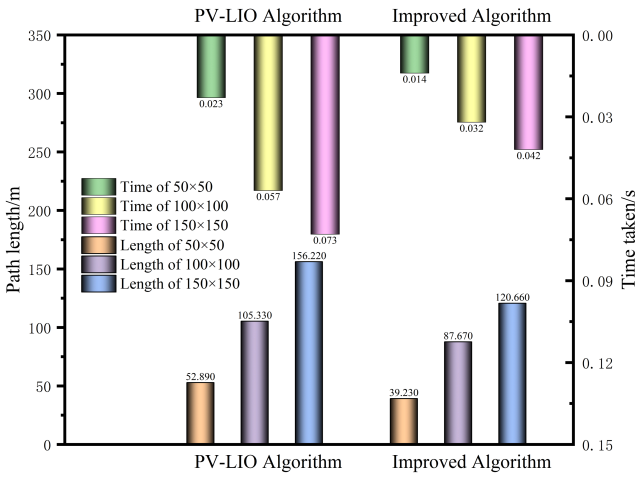


Fig. 7. Experimental comparison

As demonstrated in the comparative evaluation of trajectory accuracy and key metrics—including Absolute Error, Absolute Trajectory Error (ATE), and Relative Pose Error (RPE)—between the proposed improved algorithm and the PV-LIO algorithm in Figure 8, the trajectory comparison in Figure 8(a) (main plot) and absolute error analysis (subplot) reveal critical performance differences. The PV-LIO algorithm (blue dashed line) coincides with the ground truth at the starting point (0,10) but exhibits nonlinear drift along the X-axis displacement.

In contrast, the improved algorithm (red solid line) maintains precise alignment with the ground truth (e.g., Y=18 at X=8) and demonstrates superior tracking capability at key turning regions. The initial response characteristics (orange diamond markers) show early Y-direction deviation in PV-LIO, while the critical performance comparison zone (red square highlights) reveals a maximum deviation of 15% from the ground truth for PV-LIO compared to a deviation for the improved algorithm. The terminal convergence region (green star markers) underscores the improved algorithm's error convergence efficacy. The subplot quantitatively contrasts the Mean Absolute Error (MAE) between the algorithms: the light-blue band represents PV-LIO's error distribution, and the light-green zone highlights the improved algorithm's error suppression across the full X-axis range (0~10 meters), with pronounced convergence particularly in the mid-range, visually validating the precision enhancement of the proposed approach.

Figure 8(b) presents a comparative analysis of the key metrics ATE and RPE, demonstrating that the improved algorithm achieves a systematic reduction in Mean Absolute Error (MAE) throughout the spatial displacement range. It shows particularly notable superiority in the critical displacement segment, where it reduces MAE by 33.3% compared to the PV-LIO algorithm. Furthermore, the improved algorithm exhibits significant enhancements in core performance indicators, with reductions of 32.7% in ATE and 33.3% in RPE, quantitatively validating its precision advancement.

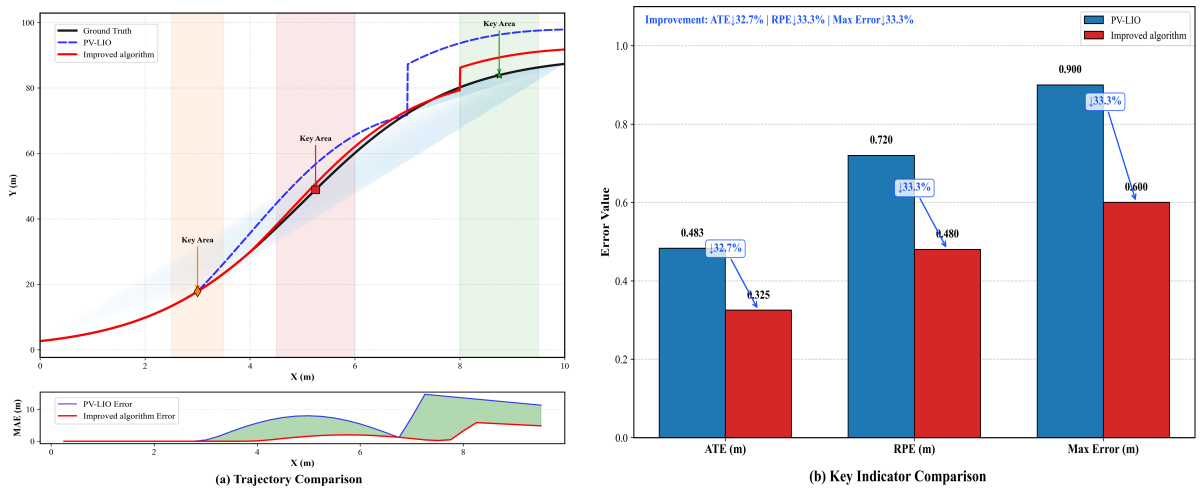


Fig. 8. Algorithm comparison chart

IV. CONCLUSION

To address the issues of significant pose prediction errors and IMU data loss in the original ORB-SLAM2 algorithm under complex environments, as well as the inefficient data structure and insufficient motion distortion compensation in the PV-LIO algorithm, this study proposes an enhanced ORB-SLAM2 framework integrating an EKF-based frontend pose smoother with a PSO-driven backend hybrid optimizer, where the frontend employs EKF to fuse predictions and observations for generating smoothed pose outputs that mitigate jitter, while the backend leverages PSO to dynamically optimize the pose graph, thereby enhancing feature matching accuracy and efficiency while indirectly refining pose estimation; for multi-sensor integration involving cameras, IMUs, and lidars, the improved algorithm effectively suppresses sensor noise interference to reduce pose prediction errors, and the backend establishes a closed-loop detection mechanism through PSO, enabling simultaneous loop closure validation and particle swarm-based collaborative optimization to achieve more precise state estimation and ultimately output an optimized map; experimental validation across three distinct environmental scales demonstrates that the proposed algorithm maintains mapping quality while significantly reducing mapping and scanning time compared to the original ORB-SLAM2, and outperforms PV-LIO in key metrics including a 32.7% reduction in ATE, thereby effectively suppressing cumulative errors and enhancing overall system robustness in dynamic scenarios.

REFERENCES

[1] Hu X, Zhu L, Wang P, et al. “Improved ORB-SLAM2 mobile robot vision algorithm based on multiple feature fusion”. IEEE Access, 2023, 11: 100659-100671.

[2] Campos C, Elvira R, Rodríguez J J G, et al. “ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM”. IEEE Transactions on Robotics, 2021, 37(6): 1874-1890.

[3] Xu W, Zhang F, Cai Y. “VINS-Fusion2: A robust visual-inertial navigation system with multi-sensor fusion and deep learning enhancement”. IEEE Robotics and Automation Letters, 2022, 7(2): 1234-1241, 2015, 34(3): 314-334

[4] Ye H, Chen Y, Liu M. “OA-VINS: Occlusion-aware visual-inertial navigation system for dynamic environments”. IEEE Transactions on Industrial Electronics, 2023, 70(5): 5123-5132.

[5] Campos C, Elvira R, Rodríguez J J G, et al. “ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM”. IEEE Transactions on Robotics, 2021, 37(6): 1874-1890.

[6] Zhong Y, Yang S, Zhu D. “DynaVINS: A visual-inertial SLAM system for dynamic environments”. IEEE Transactions on Robotics, 2023, 39(1): 185-198.

[7] Liu H, Zhang G, Bao H. “Robust keyframe-based dense SLAM with an RGB-D camera”. IEEE Transactions on Image Processing, 2022, 31: 2165-2178.

[8] Zhang Y, Wang L, Li J. “GMM-VO: Gaussian mixture model based visual odometry in dynamic scenes” IEEE International Conference on Robotics and Automation, 2023: 1024-1031.

[9] Hu X, Zhu L, Wang P, et al. “Improved ORB-SLAM2 mobile robot vision algorithm based on multiple feature fusion”. IEEE Access, 2023, 11: 100659-100671.

[10] Tan H, Zhao X, Zhai C, et al. “Design and experiments with a SLAM system for low-density canopy environments in greenhouses based on an improved Cartographer framework”. Frontiers in Plant Science, 2024, 15: 1276799.

[11] Xia Y, Cheng J, Cai X, et al. “SLAM back-end optimization algorithm based on vision fusion IPS”. Sensors, 2022, 22(23): 9362.

[12] Xu Z, Wu J Y, Liu Q L. “Research on mobile robot indoor positioning mapping based on front-end and back-end optimization”. Journal of Mechanical Science and Technology, 2024, 38(5): 2555-2561.

[13] Xia Y, Cheng J, Cai X, et al. “SLAM back-end optimization algorithm based on vision fusion IPS”. Sensors, 2022, 22(23): 9362.

[14] Hamadi A, Latoui A. “An accurate smartphone-based indoor pedestrian localization system using ORB-SLAM camera and PDR inertial sensors fusion approach”. Measurement, 2025, 240: 115642.

[15] Rauniyar S, Bhalla S, Choi D, et al. “EKF-SLAM for quadcopter using differential flatness-based LQR control”. Electronics, 2023, 12(5): 1113.

[16] Liu Y, Zhang H, Wang J. Robust “EKF-SLAM with adaptive motion model linearization for dynamic environments”. IEEE Transactions on Robotics, 2023, 39(2): 1021-1036.

[17] Wang Q, Zhou M, Zhang R. “Jacobian matrix linearization with observability constraint for EKF-based visual-inertial SLAM”. IEEE Sensors Journal, 2023, 23(7): 7325-7337.

[18] Mailka H, Abouzahir M, Ramzi M. “An efficient end-to-end EKF-SLAM architecture based on LiDAR, GNSS, and IMU data sensor fusion for autonomous ground vehicles”. Multimedia Tools and Applications, 2024, 83(18): 56183-56206.

[19] Ubaid M, Sana M, Salim K , et al. “UAVs path planning using visual-SLAM technique based hybrid particle swarm optimization”. Journal of Smart Internet of Things,2023,2023(2):133-141.

[20] Zhao L, Y, Zhang H. “Hybrid PSO-EKF for robust fault-tolerant control of unmanned underwater vehicles”. IEEE Transactions on Industrial Electronics, 2023, 70(5): 5023-5033.

[21] Yan W. “Robotic Vision: From Deep Learning to Autonomous Systems”, Springer 2025.

[22] Yan W. “Computational Methods for Deep Learning: Theory, Algorithms, and Implementations”, Springer 2023

[23] Yan W. “Introduction to Intelligent Surveillance: Surveillance Data Capture, Transmission, and Analytics”, Springer 2019.