drones

MDPI

Article

A Lightweight UAV System: Utilizing IMU Data for Coarse Judgment of Loop Closure

Hongwei Zhu [1,2], Guobao Zhang [1,2,*], Zhiqi Ye [1] and Hongyi Zhou [1]

[1] School of Automation, Southeast University, No. 2, Sipailou, Nanjing 210018, China; 230219174@seu.edu.cn (H.Z.)
[2] Nanjing Shendi Intelligent Construction Technology Research Institute, 7th Floor, Building A1, No. 8 Bailongjiang East Street, Jianye District, Nanjing 210019, China
* Correspondence: guobaozh@seu.edu.cn

Abstract: Unmanned aerial vehicles (UAVs) can experience significant performance issues during flight due to heavy CPU load, affecting their flight capabilities, communication, and endurance. To address this issue, this paper presents a lightweight stereo-inertial state estimator for addressing the heavy CPU load issue of ORB-SLAM. It utilizes nonlinear optimization and features to incorporate inertial information throughout the Simultaneous Localization and Mapping (SLAM) pipeline. The first key innovation is a coarse-to-fine optimization method that targets the enhancement of tracking speed by efficiently addressing bias and noise in the IMU parameters. A novel visual–inertial pose graph is proposed as an observer to assess error thresholds and guide the system towards visual-only or visual–inertial maximum a posteriori (MAP) estimation accordingly. Furthermore, this paper introduces the incorporation of inertial data in the loop closure thread. The IMU data provide displacement direction relative to world coordinates, which is serving as a necessary condition for loop detection. The experimental results demonstrate that our method maintains excellent localization accuracy compared to other state-of-the-art approaches on benchmark datasets, while also significantly reducing CPU load.

Keywords: sensor; SLAM; lightweight; UAV system; loop closure; EuRoC UAV dataset

Citation: Zhu, H.; Zhang, G.; Ye, Z.; Zhou, H. A Lightweight UAV System: Utilizing IMU Data for Coarse Judgment of Loop Closure. *Drones* **2023**, *7*, 338. https://doi.org/10.3390/drones7060338

Academic Editors: Yu-Shen Liu, Xiaoping Zhou, Jia-Rui Lin, Ge Gao, Yi Fang and Anthony Tzes

Received: 25 April 2023
Revised: 20 May 2023
Accepted: 22 May 2023
Published: 23 May 2023

1. Introduction

With the continuous advancement of modern technology, the application range of UAVs is becoming increasingly widespread. In recent years, UAVs have been appearing more frequently in people's sight. During UAV flights, surveying tasks need to be completed in preparation for future navigation inspections. The SLAM process in the UAV system requires high real-time performance, especially in terms of CPU information processing speed, which has become the main research direction at present. Based on the type of sensor used, SLAM systems can be categorized as visual SLAM, lidar SLAM, and multi-sensor fusion SLAM.

The camera offers abundant visual information at a low cost and in a compact form factor, enabling robot localization and navigation. However, purely visual SLAM performance often deteriorates in low-textured environments. To address this issue, researchers have combined points with other geometric entities such as lines as lines [1] or planes [2]. In human-made environments, a pose-graph optimization strategy can be used to take advantage of structural constraints such as parallelism or orthogonality of walls. Another well-known approach for reducing rotation drift is to adopt the Manhattan World (MW) assumption [3]. However, most of the methods discussed above use RGB-D cameras in human-made environments, which may not be universally applicable. Moreover, the accuracy of the system depends heavily on the estimation of the ground plane and Manhattan Axes (MA). Recently, with the development of deep learning, a combined system of SLAM and a Convolutional Neural Network (CNN) has emerged. In [4], a table retrieval method

is proposed for data association and loop closure using semantic information in a dynamic environment. Each landmark is associated with its own semantic and location information to improve the accuracy of the system.

To address the limitations of purely visual SLAM, the fusion of vision and IMU data have become mainstream. The IMU is primarily used to measure acceleration and rotational motion, providing high-frequency and outlier-free inertial measurements. However, the IMU's long-term operation may result in significant accumulated drift, which necessitates the initialization of all IMU parameters and real-time optimization in the later stages. This is critical for visual–inertial odometry (VIO) and visual–inertial SLAM systems, and researchers are actively seeking ways to quickly complete the initialization of the IMU and suppress its noise and bias. Currently, the initialization methods of VIO systems can be broadly categorized into two main approaches: loosely coupled [5,6] and tightly coupled [7]. The loosely coupled approach involves separate initialization processes for the IMU and the camera, followed by minimizing the distance between their poses. In VINS-Mono [5], keyframes and map points are initially obtained using visual odometry, and IMU parameters are optimized through aligning the IMU pre-integrated rotation with visual measurements by covariance propagation of the error term. However, this approach estimates the velocity as an unknown variable and overlooks the accelerometer bias, leading to incomplete initialization information. On the other hand, ORB-SLAM3 [6] is a visual–inertial tightly coupled system that employs MAP estimation to estimate scale, gravity direction, biases, and velocity during IMU initialization, while the tightly coupled approach directly establishes constraints between the camera and IMU during the initialization process to optimize various parameters. OpenVINS [7] is a tightly coupled initialization approach that leverages camera poses to establish visual constraints, enabling the estimation of initial velocity, gravity, and three-dimensional coordinates of feature points. Subsequently, multiple-frame velocity and position relationships are obtained through first-order and second-order integration, respectively. BASALT [8] employs a two-level SLAM system that optimizes the noise and bias of the IMU in both stages. In contrast to other systems, it does not directly utilize the pre-integrated IMU measurements in the mapping stage. Instead, it extracts short-term visual–inertial tracking information from the marginalized information of the VI-odometry stage. This approach not only reduces the dimensionality of the global optimization problem but also enhances the accuracy of the optimization results. GVINS [9] employs a coarse-to-fine approach to initialize GNSS visual–inertial states using MAP estimation and integrates their raw measurements within a probabilistic framework. It is capable of providing drift-free 6-DoF global pose estimation in complex environments where GNSS signals may be obstructed or entirely unavailable.

One of the challenges in developing SLAM systems is ensuring algorithm robustness and real-time performance while working with limited computing resources, such as cheap and low-performance processors. This is especially important for battery-powered robots, where computational efficiency is crucial for extending the robot's endurance. To address these challenges, researchers have proposed various approaches. For example, Ref. [10] uses a direct method to initialize the system and tracks non-keyframes for state estimation at the front-end. At the back-end, sliding window and marginalization are adopted to limit the number of keyframes and perform nonlinear optimization. Similarly, FastORB-SLAM [11] tracks keypoints between incoming frames without computing descriptors, exploiting motion smoothness and constraints on epipolar geometry to refine the correspondence. ORB-SLAM2S [12] includes a lightweight front-end that uses a sparse optical flow method for non-keyframes and descriptors, achieving faster speed performance compared to ORB-SLAM2. However, these methods often replace ORB features with direct methods and optical flow methods to track feature points, which can lead to reduced system accuracy if feature points are not extracted properly.

Our laboratory is developing an inspection robot that relies on the Robot Operating System (ROS) and object detection algorithms to achieve mapping and monitoring. Endurance and real-time tracking are key factors for inspection robots. Therefore, we

propose a lightweight stereo-inertial SLAM based on nonlinear optimization and feature tracking, which achieves fast tracking, better robustness, and a lower CPU load. The overall system architecture is shown in Figure 1. Our three main contributions focus on speeding up tracking, reducing CPU consumption, and maintaining system accuracy. The main contributions are summarized as follows:

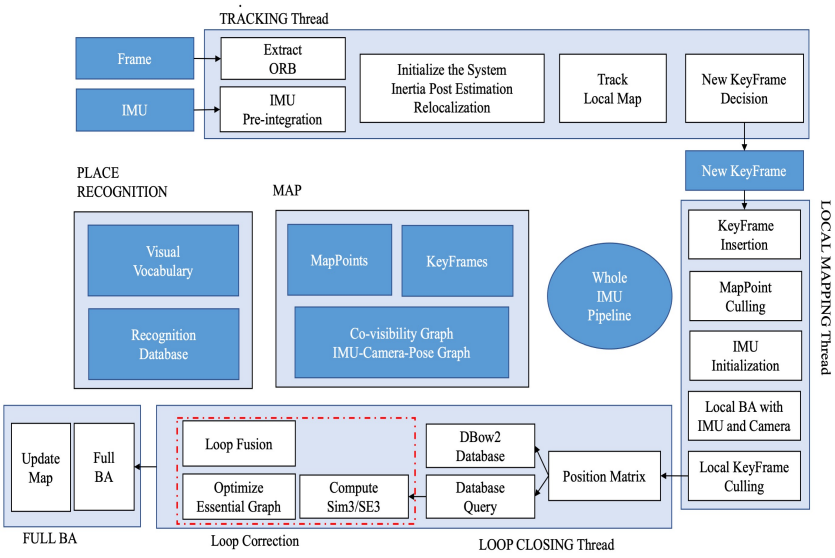


Figure 1. The whole frame of system.

- A coarse-to-fine optimization approach. Coarse optimization is for faster IMU initialization to replace the constant velocity model and speed up the tracking process, while fine optimization ensures localization accuracy.
- A novel visual–inertial pose graph as an observer decides which variables need to be optimized to prevent over-optimization.
- Fusion of IMU data with loop closure to further reduce CPU load.

The rest of the paper is structured as follows: in Section 2, we introduce the state-of-the-art relevant systems. Section 3 is the main contribution and framework of the system. The experiment setup and comparison with other systems are given in Section 4. The last section covers the conclusions of this work.

2. Related Work

In recent years, visual SLAM has gained increasing attention from researchers due to advancements in sparse nonlinear optimization theory and computer performance. Most visual SLAMs rely on point features and MAP estimation because of their general applicability. In the feature-based method, the system's robustness and localization accuracy are improved by minimizing the feature reprojection error, while photometric Bundle Adjustment (BA) is used to optimize the pose by minimizing the photometric error of a set of pixels in the direct method. Cameras provide rich visual information at low cost. However, point features have several drawbacks. First, point features extracted by vision sensors are highly sensitive to environmental conditions and fail to track when the texture is poor or the image is blurred. Moreover, they are vulnerable to illumination changes. Finally, point features are sparse, making them challenging to use in robot path planning.

SLAM involves three types of data association [13]: short-term data association for feature point tracking, medium-term data association for bundle adjustment in local maps, and long-term data association for loop closure. This approach is followed by most current visual SLAM systems. Nonlinear optimization methods have been shown to have better accuracy than filtering, so the current mainstream approach is to select representative frames (keyframes) for input into backend optimization. The keyframe-based approach

provides better accuracy with less computation, and has become an important standard for current SLAM systems. PTAM [14] is a representative system for keyframes, with two parallel threads for camera pose tracking and mapping to achieve short-term and medium-term data association. ORB-SLAM [15,16] has three parallel threads for tracking, local mapping, and loop closing, representing short-term, medium-term, and long-term data association, respectively. ORB features are used for short-term data association to compute the pose between frames. Medium-term data association uses keyframes and map points to minimize reprojection errors with bundle adjustment, while the loop closure thread uses the bag-of-words library DBoW2 [17] for long-term data association. These methods have greatly improved the accuracy of ORB-SLAM.

Multi-sensor fusion systems can significantly improve state estimation accuracy and robustness due to the complementarity between sensors. Adding an IMU can solve the problem of scale in monocular SLAM, where the image frame lacks depth information of the environment. Most visual–inertial fusion SLAM systems are tightly coupled and classified as either filter-based or optimization-based systems. The earliest multi-sensor fusion SLAM, MSCKF [18], relies on the feature method and adopts the EKF filtering method for optimization, adding camera poses at different times to the state vector. On the other hand, OKVIS [19] is the most representative system based on the nonlinear optimization method and uses keyframes, relying on the error propagation model to optimize the inertial. Some previous methods, such as [20,21], have limitations in their solution process or initialization scale accuracy. Recently, Ref. [22] proposed a robust stereo inertial odometry based on self-supervised feature points, using an improved multi-task CNN to extract feature points and incorporating an IMU to deal with rapid camera movements.

3. System Overview

The ORB-SLAM3's three threads can be taxing for certain processors and lack real-time monitoring of optimized inertial parameters. Therefore, our proposed system builds upon it with further refinement to improve real-time performance and decrease computation load, which is especially beneficial for low-end Intel Next Unit of Computing (NUC). We chose stereo because it directly measures scale without the need for additional computation and optimization via inertial information. According to our experiments (presented in Section 4.1), the stereo-inertial system is more accurate than the monocular-inertial system. The system is divided into three threads: tracking, local mapping, and loop closure, with each thread serving a specific function, which will be discussed later. It is worth noting that some of the formulas presented are not a contribution of this work, but are mainly from [23].

3.1. Coarse-to-Fine IMU Optimization

3.1.1. IMU Pre-Integration

From the parameters of inertial measurement between frame b_k and b_{k+1}, the position, velocity, and rotation of the object are expressed by

$$p^{\omega}_{b_{k+1}} = p^{\omega}_{b_k} + v^{\omega}_{b_k} t + \iint_k^{k+1} \left(q_{wb_t} a^{b_t} - g^w d \right) \delta t^2 \tag{1}$$

$$v^{\omega}_{b_{k+1}} = v^{\omega}_{b_k} + \int_k^{k+1} \left(q_{wb_t} a^{b_t} - g^w \right) \delta\, t \tag{2}$$

$$q_{wb_{k+1}} = \int_k^{k+1} q_{wb_t} \bigotimes \begin{bmatrix} 0 \\ \frac{1}{2}\omega^{b_t} \end{bmatrix} \delta t \tag{3}$$

$$q_{wb_k} = q_{wb_t} \bigotimes q_{b_t b_k} \tag{4}$$

where $p^{\omega}_{b_k}$, $v^{\omega}_{b_k}$ are, respectively, the position and velocity of b_k frame in the world reference. $q_{wb_{k+1}}$ is a quaternion which represents the transformation from b_{k+1} frame to the world reference, $\bigotimes$ denotes the transformation operation, ω^{b_k}, a^{b_k} represent angular velocity and

acceleration in b_k frame reference, respectively, g is the gravity vector, t is the time variation between frame b_k and b_{k+1}.

Due to the high sampling frequency of the IMU, the IMU pre-integration method can convert the integration of multiple measurement values into a single one, which improves the calculation efficiency. Transform Equations (1)–(3) according to Equation (4):

$$p^{\omega}_{b_{k+1}} = p^{\omega}_{b_k} + v^{\omega}_{b_k} t - \frac{1}{2} g^{w} t^2 + q_{wb_k} \iint_k^{k+1} \left(q_{b_k b_t} a^{b_t} \right) \delta t^2 \tag{5}$$

$$v^{\omega}_{b_{k+1}} = v^{\omega}_{b_k} - g^{w} t + q_{wb_k} \int_k^{k+1} \left(q_{b_k b_t} a^{b_t} \right) \delta t \tag{6}$$

$$q_{wb_{k+1}} = q_{wb_k} \int_k^{k+1} q_{b_k b_t} \bigotimes \begin{bmatrix} 0 \\ \frac{1}{2} \omega^{b_t} \end{bmatrix} \delta t \tag{7}$$

From Equations (5)–(7), the IMU's pre-integration between frame b_k and b_{k+1} can be expressed as follows:

$$\Delta p_{b_k b_{k+1}} = \iint_k^{k+1} \left(q_{b_k b_t} a^{b_t} \right) \delta t^2 \tag{8}$$

$$\Delta v_{b_k b_{k+1}} = \int_k^{k+1} \left(q_{b_k b_t} a^{b_t} \right) \delta \, t \tag{9}$$

$$\Delta q_{b_k b_{k+1}} = \int_k^{k+1} q_{b_k b_t} \bigotimes \begin{bmatrix} 0 \\ \frac{1}{2} \omega^{b_t} \end{bmatrix} \delta t \tag{10}$$

where $\Delta p_{b_k b_{k+1}}$, $\Delta v_{b_k b_{k+1}}$, and $\Delta q_{b_k b_{k+1}}$ are the position, velocity, and rotation variation, respectively.

3.1.2. Coarse IMU Optimization

Since both the accelerometer and gyroscope of IMU have noise and bias, it will have a bad influence on the measurement results. We set a covariance matrix $\tau_{k,k+1}$ to contain $\Delta p_{b_k b_{k+1}}$, $\Delta v_{b_k b_{k+1}}$, and $\Delta q_{b_k b_{k+1}}$. Residual models can be used for IMU initialization and visual–inertial BA optimization.

$$r_{\Delta q_{b_k b_{k+1}}} = \Delta q_{b_{k+1} b_k} \bigotimes \left(q_{b_k w} \bigotimes q_{w b_{k+1}} \right) \tag{11}$$

$$r_{\Delta p_{b_k b_{k+1}}} = q_{b_k w} \left(p^{\omega}_{b_{k+1}} - p^{\omega}_{b_k} - v^{\omega}_{b_k} t + \frac{1}{2} g^{w} t^2 \right) - \Delta p_{b_k b_{k+1}} \tag{12}$$

$$r_{\Delta v_{b_k b_{k+1}}} = q_{b_k w} \left(v^{\omega}_{b_{k+1}} - v^{\omega}_{b_k} + g^{w} t \right) - \Delta v_{b_k b_{k+1}} \tag{13}$$

$$r_{\tau_{b_k b_{k+1}}} = \left[r_{\Delta q_{b_k b_{k+1}}}, r_{\Delta v_{b_k b_{k+1}}}, r_{\Delta p_{b_k b_{k+1}}} \right] \tag{14}$$

In local mapping thread, we adopt a maximum a posteriori estimation to estimate the IMU variables as a coarse but fast optimization. Assuming that all variables are independent, the noise of the variables follows a Gaussian distribution with a mean of zero. The scale of the stereo camera is known and does not require optimization. The estimated parameters are as follows:

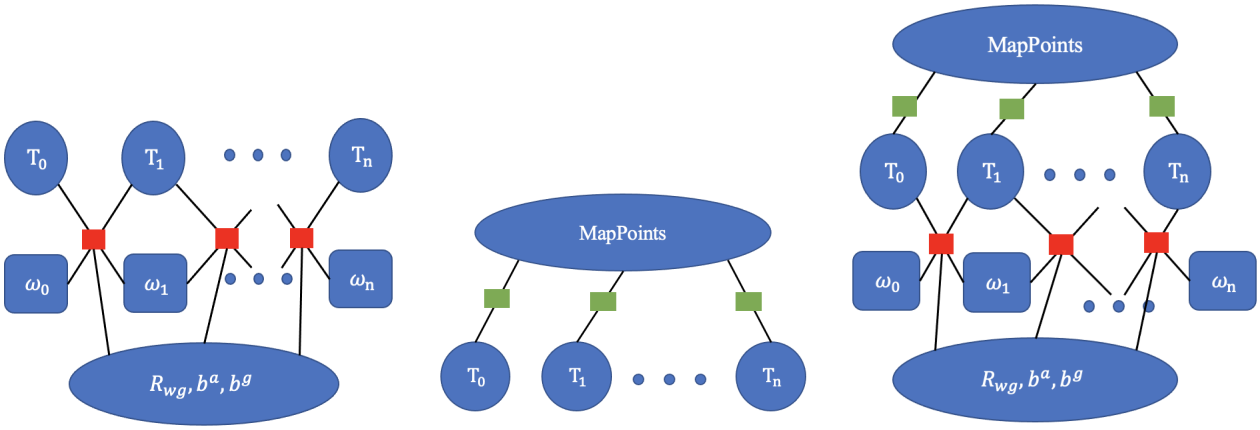
$$y_k = \{ R_{wg}, \, b^a, \, b^g, \, v_{0:k} \} \tag{15}$$

where R_{wg} represents the rotation matrix from ENU reference to the world reference. The direction of gravity in ENU is (0, 0, g). b^a, b^g are the biases of the accelerometer and gyroscope, respectively. $v_{0:k}$ denotes the up-to-scale velocity vector accumulation from the first to current frame. The parameters in y_k are optimized by inertial-only MAP:

$$\begin{aligned} y_k^* &= argmax\ p(y_k \mid \tau_{k,k+1}) \\ &= argmax\ (p(y_k)p(\tau_{k,k+1} \mid y_k)) \\ &= argmin \left(\|b^a\|^2_{\sum_{b^a}^{-1}} + \|b^g\|^2_{\sum_{b^g}^{-1}} + \sum_{i=1}^{k} \|r_{\tau_{k,k+1}}\|^2_{\sum_{r_{\tau_{k,k+1}}}^{-1}} \right) \end{aligned} \tag{16}$$

where $p(\tau_{k,k+1} \mid y_k)$ is likelihood, and $p(y_k)$ stands for prior knowledge.

MAP estimation can further transform the graph optimization problem, using the parameters in y_k as vertices and gyroscope bias, acceleration bias, velocity, etc., as edges. The inertial residual model is then adopted to minimize the distance between IMU biases and zero, as shown in Figure 2a. This approach to IMU initialization is a coarse optimization method. Compared with [19–21], it is faster. Furthermore, feature point matching between consecutive frames is time-consuming, with a maximum time complexity of N^2 (where N is the number of feature points). However, this problem can be addressed by using optimized IMU data for the initial estimation of the camera pose, rather than relying on the constant velocity model. By leveraging IMU information, the feature points from the previous frame are projected onto the pixel plane of the current frame, and matching feature points can only be found within a few pixels, thereby speeding up feature point search and improving tracking performance.



(**a**) inertial-only (**b**) vision-only (**c**) vision-inertial

Figure 2. Three methods of graph optimizations along the system.

When the system starts, the pure vision module is required to initialize and generate map point clouds, and the world reference may not be aligned with the ENU, which will cause serious errors in the IMU integration and affect the positioning accuracy. After IMU initialization is completed, the optimized R_{wg} is used to align the Z axis of the world reference with the direction of the gravity G to ensure more accurate IMU preintegration. Finally, the system obtains all keyframes in the map, adjusts their poses according to Equation (17), and completes the transformation to the new world reference.

$$R_{wb_k} = R_{wg}^T R_{wb_k} \tag{17}$$

3.1.3. Fine IMU Optimization

When IMU initialization has not started, pure visual MAP estimation is adopted, which is converted into graph optimization as shown in Figure 2b. The system runs in pure vision mode for 2 s to initialize the entire system, generating keyframes at 4 Hz and 3D map points. We define a local window to contain the co-visible keyframes and other

keyframes that can observe the 3D map points in the current keyframe. Then, reprojection error is used in the local window to optimize the pose and 3D map points.

$$r_{b_i,j} = u_{b_i,j} - T_{b_i w} \bigotimes x_j \tag{18}$$

where $u_{b_i,j}$ is the observation of pixel j in the b_i frame, $T_{b_i w} \in \mathrm{SE}(3)$ represents the transformation between the world reference and b_i frame, $\bigotimes$ is the transformation operation of SE(3) group over R^3 elements, and x_j stands for the 3D map point j.

To judge whether IMU variables have good initialization, we introduce a vision–inertial pose graph:

$$r_{\Delta R_{b_k b_{k+1}}} = \Delta R^{-1}_{b_{b_k b_{k+1}}} \bigotimes \Delta R_{c_{b_k b_{k+1}}} \tag{19}$$

where $\Delta R^T_{b_{b_k b_{k+1}}}$ is the pose in the IMU reference, and $\Delta R_{c_{b_k b_{k+1}}}$ stands for the pose in the camera reference.

When there is no bias, Equation (19) must be equal to the identity matrix since the transform between frames computed by the IMU and the camera should be the same. However, during system operation, IMU parameters are affected by noise such as temperature. Therefore, we set a threshold, and when $r_{\Delta T_{b_k b_{k+1}}}$ is within the threshold range, indicating that the IMU parameters do not need to be optimized in this frame, the visual-only MAP estimation (Equation (18)) is performed between the current keyframe and its local keyframes. Otherwise, a joint visual–inertial MAP estimation as Equation (20) is used to combine the inertial residual and visual residual to further refine the solution (Figure 2c). This approach is a finer optimization method for inertial variables and camera poses. Multiple BA methods can be used to reduce the time of back-end optimization and prevent some parameters from being over-optimized.

$$min \left(\sum_{i=1} \| r_{\tau_{k,k+1}} \|^2_{\sum^{-1}_{r_{\tau_{k,k+1}}}} + \sum_{j \in b_i} \sum_{i=1}^{k} \rho_{Hub} \| r_{b_i,j} \|^2_{\sum^{-1}_{r_{\tau_{k,k+1}}}} \right) \tag{20}$$

where b_i is the current keyframe and its local keyframe, $r_{\tau_{k,k+1}}$ stands for the inertial residual, $r_{b_i,j}$ represents the visual residual, ρ_{Hub} is the robust kernel which is used to reduce the influence of wrong matching.

3.2. Inertial and Loop Closing

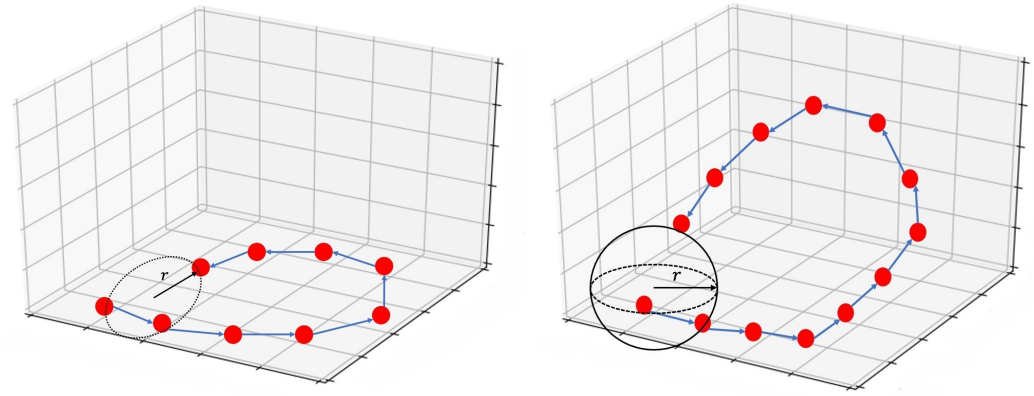
In the loop closing thread, the DBoW2 word-bag library is utilized to solve the position recognition problem, which converts the features in keyframes into bag-of-words vectors and queries the DBoW2 database to retrieve the most similar keyframes

As the SLAM system operates, the number of keyframes gradually increases, leading to an increase in query time and CPU energy consumption. This can be problematic for some embedded systems. To address this issue, we propose a new method for loop closure detection that uses IMU for rough judgment. Our inspiration for this method comes from [22–24], where the incremental consistent measurement set maximization (PCM) method is proposed to check the quality of loop closures and remove outliers after bag-of-words vectors matching. This post-operation effectively reduces the occurrence of false fusion, but it also increases the amount of calculation required. Our method differs in that it performs coarse loop closure detection before bag-of-words database traversal. This reduces the number of queries to the database, thereby reducing the burden on the CPU.

When a loop closure occurs, a portion of the sensor's trajectory must be in close proximity to a ring structure. This means that if the sensor revisits a previous location, it must be moving in the opposite direction of some of its previous trajectory, as shown in Figure 3. Leveraging this feature, we propose a method for coarse loop closure detection that incorporates IMU information. By using the acceleration and angular velocity of the IMU, we can determine the direction of movement of the robot. When the direction aligns

with the positive direction of the IMU reference frame, the value is set to 1; otherwise, if the direction is opposite, the value is set to 0.

In this way, when a keyframe enters the loop closure detection thread, a change in the direction of motion along one axis indicates that the sensor may be moving in the opposite direction and a loop closure may occur. This is the rough detection method using IMU data. However, it is impractical to match all keyframes based on orientation requirements. To address this, we introduce a threshold radius r to filter out keyframes that meet the orientation requirements. In [25], a search radius of 10 meters was used for lidar SLAM, but this method is sensitive to point cloud density and depth accuracy. Thus, we set the threshold radius r to 20 times the stereo baseline. This prevents the system from missing loops and reduces unnecessary keyframe matching. For 3D scenarios, such as drone flight as shown in Figure 3b, the principle is similar to the 2D case. We simply expand the circle of radius r into a sphere and search for possible matching keyframes within it. If there are other keyframes within the threshold, loop closure detection will begin. Features will be converted into bag-of-words vectors, and the DBoW2 database will be searched within the threshold radius. This method can significantly reduce unnecessary calculations and CPU usage as the number of keyframes increases. Furthermore, the longer the system runs, the more evident this advantage becomes.



(**a**) On a plane (**b**) In a three-dimensional space

Figure 3. Schematic diagram of coarse loop closure detection.

3.3. Main Process

The tracking thread is responsible for feature extraction from sensor input and tracking them. Initially, the system relies on pure visual initialization, which uses stereo disparity to calculate the depth of feature points and backprojects them into the map. With the input of multi-sensor information, the inertial data are used to calculate the transformation $T_{b_k w}$ between b_k and the world reference as the initial value for nonlinear optimization instead of the constant velocity model. Then, map points observed in the previous frame are projected onto the current frame with $T_{b_k w}$. Finally, an optimization procedure is carried out to estimate the orientation $R \in SO(3)$ and translation $t \in R^3$ of the current frame. Once the pose has been estimated, the current frame is evaluated to determine whether it should be considered as a new keyframe in a similar strategy to ORB-SLAM3. In cases of rapid rotation or occlusion where feature points between frames cannot be matched, the pose is calculated by integrating the inertial data, and the system continues to track. If occlusion persists in the two incoming image frames, then the tracking fails and relocalization is performed using the DBoW2 bag-of-words model to increase the robustness of tracking.

The local mapping thread is responsible for performing BA optimization to refine the pose and 3D map points of keyframes in the map. Whenever a keyframe is inserted, it establishes a co-visible graph with a set of connected keyframes, calculates the depth through binocular disparity, back-projects the unmatched feature points into the world reference, and fuses redundant 3D feature points in the map. Other keyframes that observe

the current keyframe points but are not connected are included in the optimization, but their poses remain fixed. Within 2 s of the system running, BA optimization is performed with the pure vision module using Equation (18). Then, the IMU parameters are initialized and refined using the MAP estimation technique shown in Equation (19). Finally, the vision–inertial pose graph is used to judge the quality of variable optimization and guide the system to choose a better nonlinear optimization method, preventing some parameters from being over-optimized.

The loop closing thread is responsible for correcting the accumulated drift, especially over long-term operations. When a new keyframe is inserted, the direction of movement of the current frame relative to the world reference is calculated using the IMU data and stored in a unit-bearing vector. The system then performs a coarse judgment of loop detection by searching in real-time for all keyframes within a radius r and two opposite directions described in Section 3.2. If the coarse judgment passes, the system starts the loop detection approach, and matches features between the current keyframe and other keyframes within the radius in a fine detection process. If the match is successful, the SE(3) transformation between the current frame and the matching keyframe is calculated, correcting the poses of keyframes in the co-visible graph, and the map points through pose propagation to correct the accumulated drift. After a loop correction, a full BA is executed in a new thread to further refine the map without affecting real-time performance.

4. Experimental Results

To demonstrate the performance of our system, we have conducted experiments using the EuRoC [26] dataset. The datasets consist of recordings of aircraft flying in large industrial environments, which are divided into three modes based on illumination, speed, and texture: easy, medium, and hard. We conducted multiple experiments on each of the 11 sequences and compared our system with other state-of-the-art systems. The experiments were divided into three parts: Section 4.1 is the simulation of single-session sequences, Section 4.2 discusses the IMU and loop closure, and Section 4.3 is a comparison of computing time. We aligned the estimated trajectory with ground-truth using SE(3) transformation in stereo-inertial sensor configurations. All experiments were performed on a machine running Ubuntu18.04 with an Intel Core i5-1135CPU, at 2.4 GHz, and 16 Gb memory, using only the CPU.

Histogram equalization is applied to each input image to enhance its contrast, which is highly significant for solving under- or over-exposed environments. FAST feature points are extracted from an 8-scale pyramid model with a scale factor of 1.2. Additionally, the system uses the quadtree algorithm to recursively search for groups of points and employs the point with the highest corresponding value of the FAST corner point in the neighborhood of local feature points for non-maximum suppression and fast screening. The descriptor is calculated using the BRIEF algorithm to enable the matching of feature points.

4.1. EuRoC Sequences

We select the most representative SLAM systems for each method. BASALT relies on optical flow and BA optimization. OKVIS is considered a pioneering work in the field of visual–inertial SLAM, based on nonlinear optimization. ORB-SLAM3 is considered the most advanced system currently available. KIMERA is an open-source SLAM library based on metric semantics. It consists of multiple modules and introduces semantic segmentation alongside SLAM. We use the Absolute Trajectory Error (ATE) to compare our system with other state-of-the-art SLAM systems, as shown in Table 1. The data in the table are obtained from the corresponding papers, and some missing data are from our reproduction of their papers.

$$ATE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \left\| trans(T_{gt,i}^{-1} T_{esti,i}) \right\|^2} \tag{21}$$

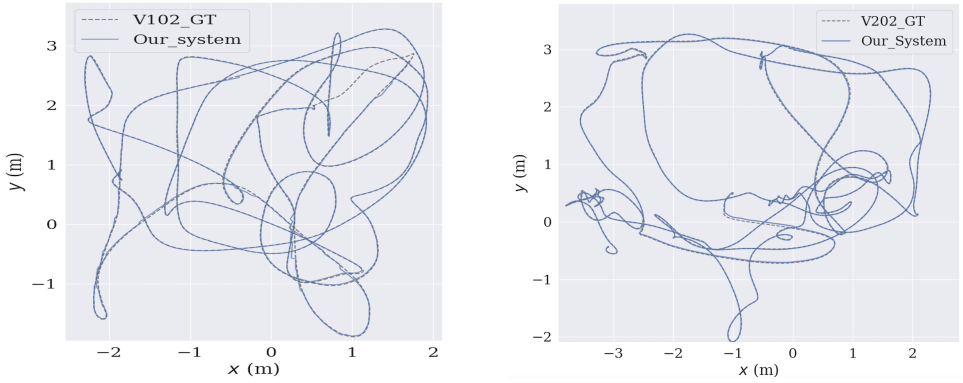
where N represents the number of keyframes, *trans* stands for the translation of the transformation, $T_{esti,i}$ represents the estimated trajectory, and $T_{gt,i}$ is the real trajectory.

Table 1. Comparison of various state-of-the-art systems on the EuRoC dataset.

EuRoC	Mono-Inertial				Stereo-Inertial			
Sequences	**OKVIS [27]**	**VI-DSO [28]**	**ORB-SLAM3 [6]**	**VINS-Fusion [27]**	**BASALT [8]**	**KIMERA [29]**	**ORB-SLAM3 [6]**	**Ours**
MH01	0.16	0.062	0.071	0.166	0.08	0.08	0.042	0.04
MH02	0.22	0.044	0.041	0.152	0.06	0.09	0.037	0.035
MH03	0.24	0.117	0.068	0.125	0.05	0.11	0.041	0.043
MH04	0.34	0.132	0.088	0.28	0.131	0.15	0.079	0.088
MH05	0.47	0.121	0.054	0.284	0.08	0.24	0.08	0.075
V101	0.09	0.059	0.042	0.076	0.04	0.05	0.043	0.045
V102	0.2	0.067	0.031	0.069	0.02	0.11	0.018	0.014
V103	0.24	0.096	0.042	0.114	0.03	0.12	0.033	0.042
V201	0.13	0.04	0.051	0.066	0.03	0.07	0.028	0.023
V202	0.16	0.062	0.02	0.091	0.02	0.1	0.024	0.022
V203	0.29	0.174	0.041	0.096	-	0.19	0.038	0.056
Avg	0.231	0.089	0.05	0.138	0.051	0.119	0.041	0.044

It can be concluded that the ATE of the stereo-inertial system is slightly better than that of the monocular-inertial system, and has a lower dependence on scale optimization. At the beginning of the system, the stereo system only requires one frame to initialize, while the monocular system requires multiple inputs to triangulate. Additionally, for the calculation of the depth of feature points, the stereo-inertial system can choose between stereo triangulation or using the left (or right) camera from the previous frame to the current frame. These advantages are the reasons why the stereo system is more robust.

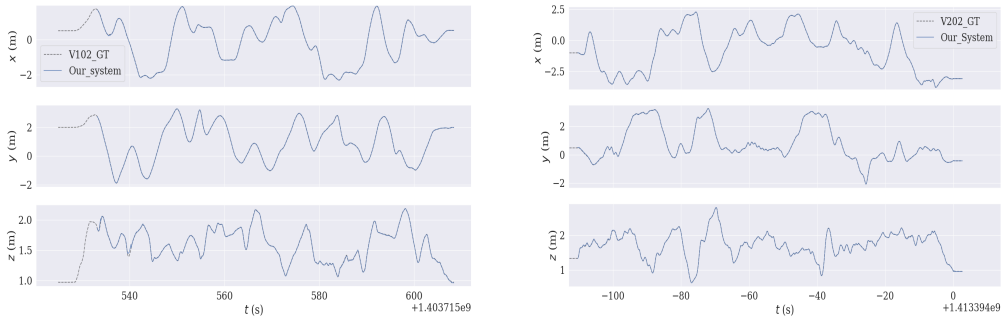
To summarize the system's performance, we presented the median of ten executions for each sequence. KIMERA emphasizes SLAM with semantic segmentation, while VINS-Fusion utilizes keypoint extraction and optical flow for tracking. However, the rapid movement of unmanned aerial vehicles may result in blurriness. As a result, our system demonstrated significantly higher accuracy than VINS-Fusion and KIMERA in certain scenarios, highlighting the advantages of medium-term and long-term data association. Our system achieves comparable accuracy to BASALT and ORB-SLAM3, except in the case of the V203 sequence in EuRoC where BASALT fails due to missing frames and severe motion blur. In contrast to BASALT, which relies on optical flow and cannot track pixel intensity variations during motion blur, our tightly-coupled approach, combined with MAP estimation, allows for rapid IMU initialization and the utilization of optimized inertial variables, resulting in improved accuracy. The local mapping thread's joint visual–inertial optimization further refines the solution, making the system more accurate and faster. Our coarse-to-fine optimization method's advantage is the two cooperating threads that make the system more robust. Every time a loop is detected, a full BA is performed to optimize all camera poses in the map, ensuring the localization accuracy of the entire system. Our system achieved the highest accuracy in most of the Machine Hall (MH), V102, and V201 sequences. In complex flight scenarios, our system's trajectory closely follows the ground truth from start to finish, as shown in Figure 4, demonstrating our system's advantages in short-term, medium-term, and long-term data association.



(**a**) Motion trajectory in V102 (**b**) Motion trajectory in V202

Figure 4. Comparison of our system with ground truth.

Figure 5 illustrates the tracking errors in the x, y, and z directions over time. As shown, the relative error of our system does not increase with distance, nor does the drift suffer from drifting in any of the three directions, remaining consistently low. This indicates that our system effectively suppresses noise. The tightly coupled approach and coarse-to-fine optimization we proposed, combining the characteristics of the tracking and local mapping threads, enable the system to run for extended periods without accumulating drift, achieving global consistency and building a reliable map. In Figure 6, our system and ORB-SLAM3 show a similar trend in ATE error, with both systems remaining within a constant range over time. Our system exhibits slightly superior positioning accuracy, which aligns with our objective of proposing a lightweight solution while upholding high precision.



(**a**) (**b**)

Figure 5. Motion track in X, Y, and Z directions. (**a**) Motion track in three directions for V102. (**b**) Motion track in three directions for V202.

4.2. IMU and Loop Closure

As shown in the partial map in Figure 7, our loop closure detection method effectively identifies loop closures and forms a nearly closed trajectory shape each time a loop is detected. The use of IMU data to calculate the sensor's displacement direction and a radius threshold of 20 times the stereo baseline from the current keyframe's camera optical center are key factors in achieving accurate and reliable loop closure detection. Our system then utilizes the DBoW2 database to identify three consecutive keyframes with temporal consistency to ensure higher recall. Once a feature matching is successful, we project the feature points of the three co-visible keyframes into the matching frame and check if the number of feature points is sufficient to close the loop.

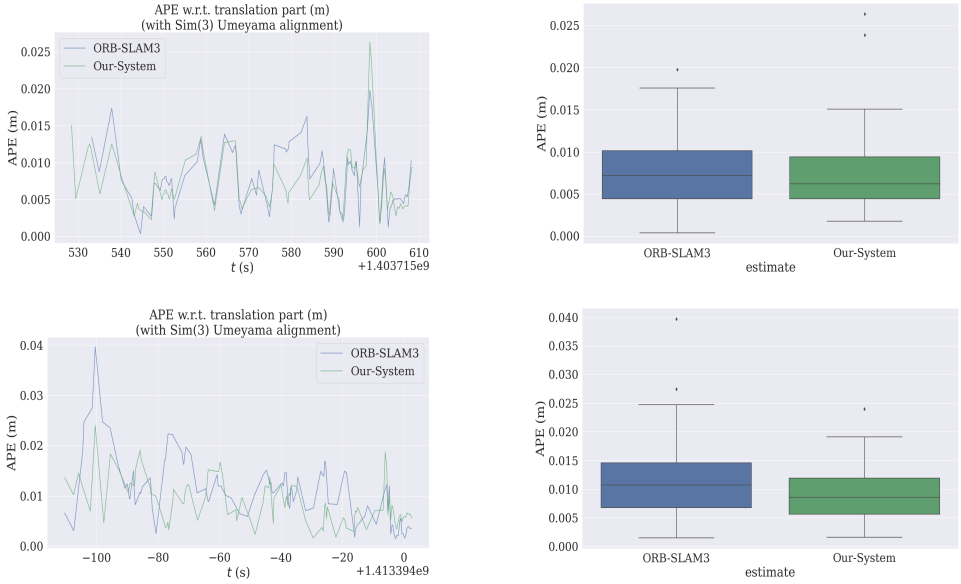
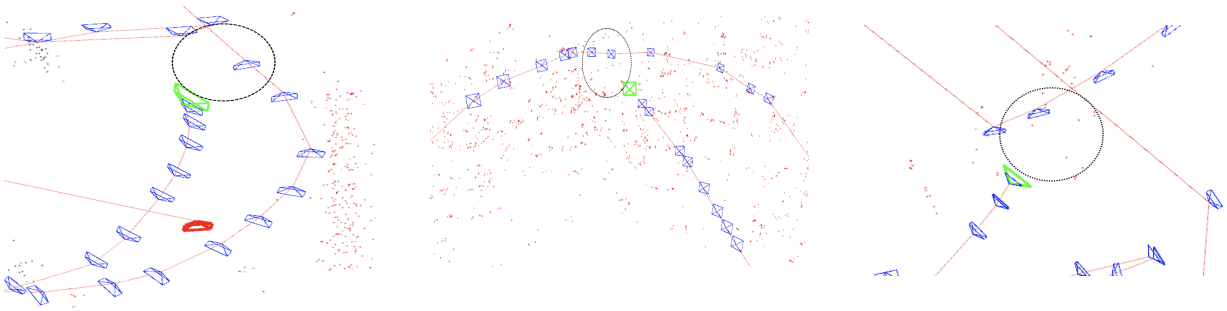


Figure 6. ATE comparison with ORB-SLAM3 on EuRoC. The top of this group of pictures is the ATE comparison of V102, and the bottom is ATE comparison for V202.

In order to filter out unnecessary keyframes and reduce computational costs, we use a radius r to separate possible keyframes for loop detection, as shown in Figure 7. However, it is important to ensure that this approach does not miss any loops. To evaluate the effect of different radii, we conducted experiments and plotted the results in Figure 8. As SLAM is a probabilistic event, the mode was used as the evaluation criterion to reflect the effect of different radii on loop closure. We found that a radius of 20 times the stereo baseline was optimal for the EuRoC dataset, as it ensured both accuracy and reduced CPU utilization. A larger radius would be closer to the original DBoW2 calculation method, while a smaller radius would negatively impact loop detection. Overall, our approach strikes a good balance between accuracy and computational efficiency.



(**a**) Loop for MH05 (**b**) Loop for V203 (**c**) Loop for V102

Figure 7. Fusion of IMU data and loop closure. The red line represents the motion trajectory of the camera, the green frame is the current keyframe, the blue frame is the keyframe in the map, and the red point represents the 3D map point.

Compared to ORB-SLAM3, our system has clear advantages in terms of CPU load. In ORB-SLAM3, the loop detection is triggered every time a new keyframe is added, and the DBoW2 database is searched to check if the robot revisits a previous location. As shown in Figure 9, the CPU usage of ORB-SLAM3 fluctuates regularly. When the matching between the current frame and the database is successful, the CPU needs to process the similar transformations between them and use graph optimization to approximate the optimal solution, resulting in the highest CPU usage. Our algorithm has a similar trend

to ORB-SLAM3, but due to the coarse-to-fine optimization using IMU data, it is no longer necessary to search the DBoW2 database every time, leading to lower CPU usage. Figure 9 demonstrates that ORB-SLAM3 and our system have the same number of peaks (once the loop merge occurs successfully, a new thread will be started, so that the CPU usage will reach its peak), which indicates that our system does not miss any loops despite using additional constraints. Not every successful coarse judgment will trigger a loop closure. Nevertheless, our system is already very friendly to some low-end processors compared to real-time loop detection.

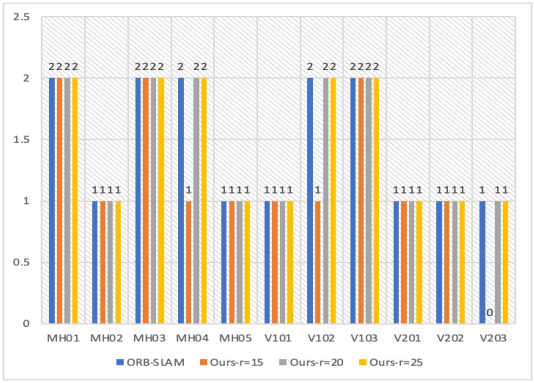
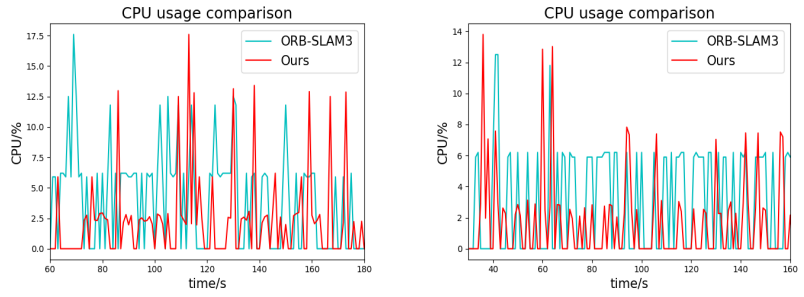


Figure 8. The number of loops per sequence. Each sequence is performed 5 times, with the mode as the statistical standard.



(**a**) MH05 sequence (**b**) V103 sequence

Figure 9. CPU comparison in loop closure.

4.3. Computing Time

Table 2 shows the running time of the main operations in the tracking and local mapping thread between the stereo-inertial ORB-SLAM3 and our system on V102 and V202. As for loop closure thread, this paper relies on the IMU data to analyze the motion trajectory of the camera, while ORB-SLAM3 performs place recognition on incoming keyframes every time, so it is not fair to compare the time of loop closure thread between them.

Table 2. Time comparison of the systems on EuRoC (ms).

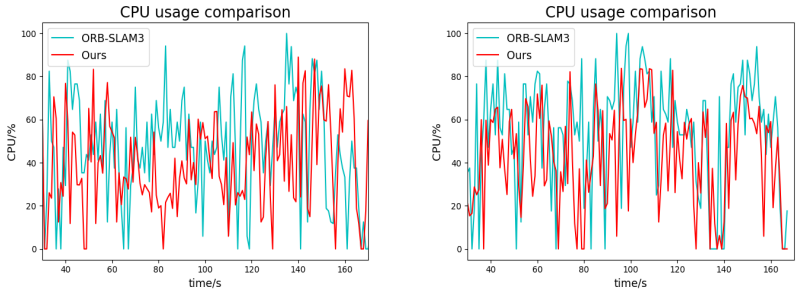
Sequence	System	Tracking			Local Mapping		
		ORB Extract	Stereo Match	Track	KF Insert	IMU Init	Local BA
V102	ORBSLAM3	12.66	2.84	12.11	6.49	30.33	153.3
	Ours	14.24	2.75	5.94	6.71	32.62	103.88
V202	ORBSLAM3	16.37	3.26	13.53	8.51	35.79	181.62
	Ours	15.98	2.79	7.65	9.13	33.85	118.48

Compared to ORB-SLAM3, our system boasts a significant advantage by operating in real-time, processing up to 50 frames with around 8 keyframes per second. The system

maintains a consistent range of time differences in operations such as feature extraction and stereo matching, ensuring a reliable and efficient performance. While ORB-SLAM3 employs a constant velocity motion model for feature point tracking, our system uses the optimized IMU variables to generate a more accurate initial value for the pose. This approach enables us to reduce the number of iterations required for pose optimization, resulting in a higher frame rate.

Our local mapping thread incorporates three graph optimization techniques: inertial-only, vision-only, and vision–inertial. However, we have an additional variable state observer that selects the appropriate graph optimization method based on the joint visual–inertial error. This ensures that certain variables are not over-optimized and enhances the accuracy of the system.

Furthermore, we compared the CPU load of our system with that of ORB-SLAM3 in the local mapping thread. The results demonstrated the effectiveness of our visual–inertial pose graph as an observer in reducing CPU load, as depicted in Figure 10. Overall, our system presents a significant improvement over ORB-SLAM3, achieving real-time operation and offering notable advantages in accuracy and efficiency.



(**a**) V203 sequence (**b**) V103 sequence

Figure 10. CPU comparison in local mapping.

5. Conclusions

In this paper, we present a lightweight stereo-inertial SLAM system that employs nonlinear optimization and feature-based techniques while leveraging IMU information throughout the pipeline. To optimize the inertial bias and noise efficiently, we propose a coarse-to-fine variable optimization method that enables the use of IMU data for tracking from the outset, thus reducing the number of BA iterations. Moreover, we incorporate optimized inertial data as a temporary replacement for feature point tracking in challenging scenarios such as fast rotation, occlusion, or poor texture, thereby enhancing the system's robustness. In the local mapping thread, we enhance the solution through joint visual–inertial optimization. We propose a novel visual–inertial pose graph that effectively identifies the variables requiring optimization, preventing continuous over-optimization of IMU parameters and reducing CPU load. In the loop closure thread, we integrate inertial data and sensor motion information as a prerequisite for loop detection. Additionally, we introduce a threshold radius to selectively filter out keyframes that satisfy the orientation requirements. This ensures that the system avoids missing potential loops and reduces the frequency of unnecessary DBoW2 database retrieval, thereby further alleviating CPU pressure on the system. Experimental results demonstrate that our system achieves improved accuracy with low CPU consumption and enhanced robustness compared to other state-of-the-art approaches on benchmark datasets.

However, our system has higher requirements for IMU data as they are utilized for initializing the pose between consecutive frames. Additionally, the selection of the threshold value for keyframe filtering is fixed, limiting its generalization to other cameras or datasets. Our future research direction will focus on setting the threshold as an adaptive value to accommodate a wider range of scenarios.

Author Contributions: H.Z. (Hongwei Zhu) is the primary contributor to this manuscript, including innovation, visualization, writing, and code editing. G.Z. is the corresponding author and proofreader of the paper, providing funding support. Z.Y. and H.Z. (Hongyi Zhou) are responsible for editing a portion of the code and performing statistical analysis on experimental data. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by Jiangsu Provincial Social Developing Project under Grant BE2020116 and BE2021750.

Data Availability Statement: All data generated or analysed during this study are included in this published article: The EuRoC micro aerial vehicle datasets. This dataset is available for download at any time. The address to download the data is here https://projects.asl.ethz.ch/datasets/doku.php?id=kmavvisualinertialdatasets (accessed on 19 May 2023). Additionally, we have made the method for computing CPU load ratios publicly available and open source https://github.com/zhuhongwei123/CPU-usage-statistics (accessed on 19 May 2023).

Conflicts of Interest: We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

References

1. Company-Corcoles, J.P.; Garcia-Fidalgo, E.; Ortiz, A. Lipo-lcd: Combining lines and points for appearance-based loop closure detection. In Proceedings of the British Machine Vision Conference (BMVC), Virtual Event, UK, 7–10 September 2020.
2. Zhang, X.; Wang, W.; Qi, X.; Liao, Z.; Wei, R. Point-plane slam using supposed planes for indoor environments. *Sensors* **2019**, *19*, 3795. [CrossRef] [PubMed]
3. Coughlan, J.M.; Yuille, A.L. Manhattan world: Compass direction from a single image by bayesian inference. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–27 September 1999; IEEE: Piscataway, NJ, USA, 1999; Volume 2, pp. 941–947.
4. Song, C.; Zeng, B.; Su, T.; Zhang, K.; Cheng, J. Data association and loop closure in semantic dynamic slam using the table retrieval method. *Appl. Intell.* **2022**, *52*, 11472–11488. [CrossRef]
5. Qin, T.; Li, P.; Shen, S. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
6. Campos, C.; Elvira, R.; Rodriguez, J.J.G.; Montiel, J.M.; Tards, J.D. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
7. Geneva, P.; Eckenhoff, K.; Lee, W.; Yang, Y.; Huang, G. Openvins: A research platform for visual-inertial estimation. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Piscataway, NJ, USA, 2020.
8. Usenko, V.; Demmel, N.; Schubert, D.; Stuckler, J.; Cremers, D. Visual-inertial mapping with non-linear factor recovery. *IEEE Robot. Autom. Lett.* **2019**, *5*, 422–429. [CrossRef]
9. Cao, S.; Lu, X.; Shen, S. Gvins: Tightly coupled gnss–visual–inertial fusion for smooth and consistent state estimation. *IEEE Trans. Robot.* **2022**, *38*, 2004–2021. [CrossRef]
10. Gao, B.; Wang, D.; Lian, B.; Tang, C. Lovins: Lightweight omnidirectional visual-inertial navigation system. In Proceedings of the 2021 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), Virtual, 17–19 August 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
11. Fu, Q.; Yu, H.; Wang, X.; Yang, Z.; He, Y.; Zhang, H.; Mian, A. Fast orb-slam without keypoint descriptors. *IEEE Trans. Image Process.* **2021**, *31*, 1433–1446. [CrossRef] [PubMed]
12. Diao, Y.; Cen, R.; Xue, F.; Su, X. Orb-slam2s: A fast orb-slam2 system with sparse optical flow tracking. In Proceedings of the 2021 13th International Conference on Advanced Computational Intelligence (ICACI), Chongqing, China, 14–16 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 160–165.
13. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
14. Klein, G.; Murray, D. Parallel tracking and mapping for small ar workspaces. In Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Nara, Japan, 13–16 November 2007; IEEE: Piscataway, NJ, USA, 2007; pp. 225–234.
15. Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. Orb-slam: A versatile and accurate monocular slam system. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
16. Mur-Artal, R.; Tardos, J.D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
17. Galvez-Lopez, D.; Tardos, J.D. Bags of binary words for fast place recognition in image sequences. *IEEE Trans. Robot.* **2012**, *28*, 1188–1197. [CrossRef]

18. Mourikis, A.I.; Roumeliotis, S.I. A multi-state constraint kalman filter for vision-aided inertial navigation. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation (ICRA), Rome, Italy, 10–14 April 2007; Volume 2, p. 6.
19. Leutenegger, S.; Furgale, P.; Rabaud, V.; Chli, M.; Konolige, K.; Siegwart, R. Keyframe-based visual-inertial slam using nonlinear optimization. In Proceedings of the Robotis Science and Systems (RSS) 2013, Berlin, Germany, 24–28 June 2013.
20. Kaiser, J.; Martinelli, A.; Fontana, F.; Scaramuzza, D. Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation. *IEEE Robot. Autom. Lett.* **2016**, *2*, 18–25. [CrossRef]
21. Huang, W.; Liu, H. Online initialization and automatic camera-imu extrinsic calibration for monocular visual-inertial slam. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 5182–5189.
22. Li, G.; Hou, J.; Chen, Z.; Yu, L.; Fei, S. Robust stereo inertial odometry based on self-supervised feature points. *Appl. Intell.* **2022**, *53*, 7093–7107. [CrossRef]
23. Martinelli, A. Closed-form solution of visual-inertial structure from motion. *Int. J. Comput. Vis.* **2014**, *106*, 138–152. [CrossRef]
24. Burri, M.; Nikolic, J.; Gohl, P.; Schneider, T.; Rehder, J.; Omari, S.; Achtelik, M.W.; Siegwart, R. The euroc micro aerial vehicle datasets. *Int. J. Robot. Res.* **2016**, *35*, 1157–1163. [CrossRef]
25. Leutenegger, S.; Lynen, S.; Bosse, M.; Siegwart, R.; Furgale, P. Keyframe-based visual–inertial odometry using nonlinear optimization. *Int. J. Robot. Res.* **2015**, *34*, 314–334. [CrossRef]
26. Von Stumberg, L.; Usenko, V.; Cremers, D. Direct sparse visual-inertial odometry using dynamic marginalization. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 2510–2517.
27. Rosinol, A.; Abate, M.; Chang, Y.; Carlone, L. Kimera: An open-source library for real-time metric-semantic localization and mapping. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1689–1696.
28. Mangelson, J.G.; Dominic, D.; Eustice, R.M.; Vasudevan, R. Pairwise consistent measurement set maximization for robust multi-robot map merging. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; IEEE: Piscataway, NJ, USA, 2018.
29. Qin, T.; Pan, J.; Cao, S.; Shen, S. A general optimization-based framework for local odometry estimation with multiple sensors. *arXiv* **2019**, arXiv:1901.03638.