

# Relocalization, Global Optimization and Map Merging for Monocular Visual-Inertial SLAM

Tong Qin, Peiliang Li, and Shaojie Shen

**Abstract**—The monocular visual-inertial system (VINS), which consists one camera and one low-cost inertial measurement unit (IMU), is a popular approach to achieve accurate 6-DOF state estimation. However, such locally accurate visual-inertial odometry is prone to drift and cannot provide absolute pose estimation. Leveraging history information to relocalize and correct drift has become a hot topic. In this paper, we propose a monocular visual-inertial SLAM system, which can relocalize camera and get the absolute pose in a previous-built map. Then 4-DOF pose graph optimization is performed to correct drifts and achieve global consistent. The 4-DOF contains  $x$ ,  $y$ ,  $z$ , and yaw angle, which is the actual drifted direction in the visual-inertial system. Furthermore, the proposed system can reuse a map by saving and loading it in an efficient way. Current map and previous map can be merged together by the global pose graph optimization. We validate the accuracy of our system on public datasets and compare against other state-of-the-art algorithms. We also evaluate the map merging ability of our system in the large-scale outdoor environment. The source code of map reuse is integrated into our public code, VINS-Mono<sup>1</sup>.

## I. INTRODUCTION

Accurate state estimation plays an important role in a wide range of applications, such as autonomous navigation, virtual reality (VR), and augmented reality (AR). The camera has become a more and more popular sensor in this area. Approaches [1]–[4] that use a single camera has attracted significant attention. However, the metric scale cannot be directly recovered from one camera, which limits their usage in real world. Recently, assisting the monocular camera with a low-cost inertial measurement unit (IMU) has become a popular trend. IMUs measure acceleration and angular velocity, which render the scale, roll, and pitch angles observable. Furthermore, the integration of IMU measurements can dramatically improve visual tracking performance in the texture-less area and aggressive motion, which extends the range of applications. The monocular camera and IMU form the minimum sensor set for accurate and robust 6-DOF state estimation.

Due to the computation limitation and real-time requirement, visual-inertial odometry approaches only focus on local accuracy. They process measurements collected within a local area or a short period while throw or marginalize past measurements. Therefore, these approaches are prone

All authors are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong, China. {tong.qin, pliap}@connect.ust.hk, eeshaojie@ust.hk. This work was supported by the Hong Kong Research Grants Council Early Career Scheme under project no. 26201616, and HKUST Proof-of-Concept Fund under project no. PCF.009.16/17.

<sup>1</sup><https://github.com/HKUST-Aerial-Robotics/VINS-Mono>

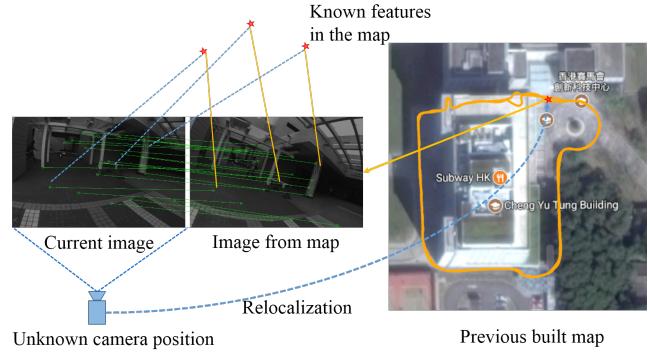


Fig. 1. The proposed visual-inertial system relocalize camera position in real time. The right image is previous-built map aligned with Google Map. The camera starts at an unknown position. We retrieve descriptors on the current image. If a similar image view is found in the map, we relocate camera and get the absolute pose in the previous map.

to drift in a long run. Even IMUs can correct drifts in roll and pitch direction, the visual-inertial system still drifts in other four directions,  $x$ ,  $y$ ,  $z$  and yaw angle. However, a globally drift-free trajectory is required in many tasks, such as robot exploration and navigation, and indoor augmented reality applications. Therefore, it is of crucial importance for a SLAM system to have the ability to relocalize and correct drift smoothly.

Another issue is that the visual-inertial odometry is a relative transformation from the initial frame instead of an absolute position. Every time we launch the system, it sets the start point as the reference frame and outputs the odometry in unfixed reference frames. Therefore, we cannot get poses in a consistent global frame in different trials. However, in some stable environments, we want to get the absolute pose in a fixed frame whenever and wherever we launch the system.

To address all these issues, we propose a real-time monocular visual-inertial system, which can achieve relocalization and global pose graph optimization to eliminate drift. Meanwhile, our system can reuse previous-built map and relocalize current pose in the previous-built map. Therefore, we can get the absolute pose estimation in a known environment. Furthermore, our system can merge current map into previous-built map smoothly. Our system is based on a real-time monocular visual-inertial odometry (VIO) method which provides locally accurate estimation [5]. Loop detection is achieved by a state-of-the-art image retrieval method, DBoW2 [6]. Relocalization is done in a tightly-

coupled feature-level fusion with the monocular VIO. Finally, geometrically verified loops are added into a 4-DOF pose graph optimization to eliminate drift smoothly. The experiments show that the proposed system can improve localization accuracy. Also, the map "evolves" overtime by incrementally merging new sensor data captured at different times.

We highlight that our contribution in twofold:

- A complete SLAM system with relocalization, 4-DOF pose graph optimization, map merging and reuse of previous-built map.
- Open-source code of map reuse.

The rest of the paper is structured as follows. In Sect. II, we discuss the relevant literature. The system overview is discussed in Sect. III. We introduce our algorithm in detail in Sect. IV. Implementation details and experimental evaluations are presented in Sect. V. Finally, the paper is concluded in Sect. VI.

## II. RELATED WORK

Tremendous research works on visual SLAM have appeared in the last few years. Current state-of-the-art monocular approaches include SVO [1], LSD-SLAM [2], DSO [4], ORB-SLAM [3] and so on. They use monocular vision to track camera pose and map the environment at the same time. Some of them are based on sparse features and some of them are based on the dense image. They achieve convincing results of localization and mapping in an up-to-scale structure.

In order to recover the real scale, IMU is often used to assist camera in the visual system. IMU-aided visual odometry has attracted significant attention recently. Some simple but effective works [7, 8] loosely fuse IMU and camera by Kalman Filter (KF). The visual result is independent of IMU. The camera depicts the up-to-scale structure firstly, then the IMU complement the scale. Tightly-coupled visual-inertial fusion can achieve higher accuracy. One popular EKF based VIO approach is MSCKF [9, 10]. Several camera poses are maintained in the state vector. Therefore, the observations of the same features crossing multiple camera views form the multi-constraint update. The camera poses, velocity and IMU bias are jointly updated. SR-ISWF [11, 12] is a similar work with MSCKF. The improvement is that it uses square-root form [13] to achieve single-precision representation and avoid poor numerical properties, which can run on computation-limited platforms, such as mobile devices. Another trend uses graph optimization [5, 14]–[16] to tightly solve the visual-inertial problem. They usually keep multiple camera measurements and IMU measurements in a bundle and jointly optimize them to obtain the optimal state estimates. The graph optimization framework usually requires high computation resource. To bound computation complexity and achieve real-time performance, some visual-inertial odometry methods [5, 14, 16] keep a limited window size of recent states, while marginalize out past states. Therefore, these approaches focus on local accuracy. Accumulating drift is unavoidable in a long run.

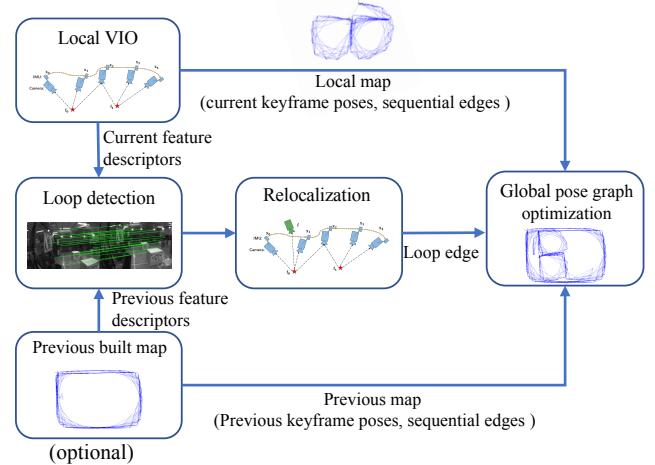


Fig. 2. A block diagram illustrating the full pipeline of the proposed monocular visual-inertial system.

Relocalization algorithms can be divided into two categories based on the type of map. One is the offline-built map, and the other one is the online-built map. Noticeable works based on the offline-built map include [17]–[19]. [18, 19] build an offline map in geometric configuration while [17] build the offline map by learning method. [20]–[22] are algorithms which can achieve relocalization in the visual-inertial system with an online-built map. [20] and [22] retrieve previous images by BRISK [23] features. Raul et al. [21] retrieves previous images by ORB [24] features. Burri et al. [20] and Raul et al [21] achieve global consistency by a global bundle adjustment (BA) in the background, while Kasyanov et al. [22] achieves this by 6-DOF pose graph optimization. However, these online map building algorithms lack the ability to load and reuse a previous build map. Also, in contrast with Kasyanov et al. [22], we perform 4-DOF pose graph optimization on 3D translation and the rotation around the gravity direction (yaw angle), which are minimum unobservable directions.

## III. SYSTEM OVERVIEW

The pipeline of the proposed visual-inertial system is depicted in Fig. 2. The system starts with a state-of-the-art monocular visual-inertial odometry [5], which achieves high accuracy local estimation. The VIO method keeps several keyframes in the local window and marginalized past frames. The keyframe poses are added into a global pose graph, which runs in another thread. Meanwhile, we process relocalization procedure on every keyframe in the third thread. The relocalization process starts with a loop detection module that identifies places that have already been visited. Once current keyframe detects loop with the previous keyframe, relocalization is processed immediately by jointly optimizing previous keyframe with the local window in raw feature-level. Then the loop information will be added to pose graph as a loop edge connecting current keyframe with the loop closure frame. If we have a map that is previously built, we can directly load it into the pose graph. The global

pose graph thoroughly optimizes all edges from current and previous-built map to achieve global consistency. In the consideration of large-scale environment, we only maintain a sparse pose graph in our map, instead of the full bundle.

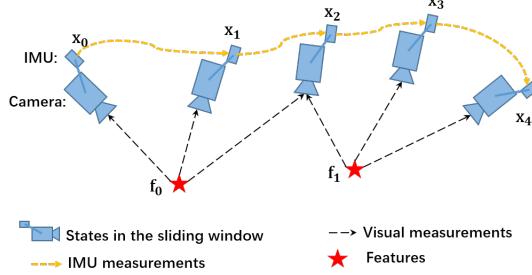


Fig. 3. An illustration of sliding-window based monocular VIO. The local window keeps several keyframes and IMU measurements between consecutive keyframes. A local bundle adjustment (BA) jointly optimization keyframes poses, velocity, IMU bias as well as feature depths.

#### IV. ALGORITHM

##### A. Visual-inertial Odometry

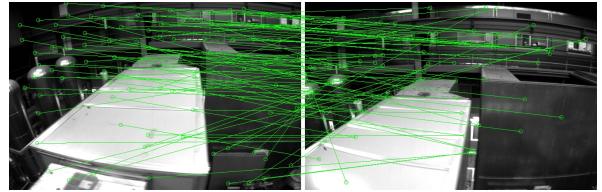
We adopt the algorithm proposed in [5] for monocular visual-inertial odometry. As depicted in the Fig. 3, The sliding window based nonlinear optimization framework processes visual and inertial measurements in a tightly-coupled way. Conner features are detected [25] and tracked [26], while IMU measurements are locally integrated. The VIO starts with a robust initialization procedure to guarantee the system can launch under any unknown state or motion. Poses, velocities, IMU bias of several keyframes as well as feature depths are optimized in a local bundle adjustment. Only keyframes, which contain sufficient feature parallax with its neighbors, are temporarily kept in the local window. Previous keyframes are marginalized out of the window in order to bound computation complexity.

The definition of full states in a sliding window with  $n$  frames and  $m$  features are (the transpose is ignored):

$$\begin{aligned} \mathcal{X} &= [\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_n, \mathbf{x}_c^b, \lambda_0, \lambda_1, \dots, \lambda_m] \\ \mathbf{x}_k &= [\mathbf{p}_{b_k}^w, \mathbf{v}_{b_k}^w, \mathbf{q}_{b_k}^w, \mathbf{b}_a, \mathbf{b}_g], k \in [0, n] \\ \mathbf{x}_c^b &= [\mathbf{p}_c^b, \mathbf{q}_c^b], \end{aligned} \quad (1)$$

where the  $k$ -th IMU state consists of the position  $\mathbf{p}_{b_k}^w$ , velocity  $\mathbf{v}_{b_k}^w$ , orientation  $\mathbf{q}_{b_k}^w$  in the world frame, and IMU bias  $\mathbf{b}_a, \mathbf{b}_g$  in body frame. 3D features are parameterized by their inverse depth  $\lambda$  when first observed in camera frame, and  $\mathbf{x}_c^b$  is the extrinsic transformation from camera frame  $c$  to body frame  $b$ . The estimation is formulated as a nonlinear least-square problem:

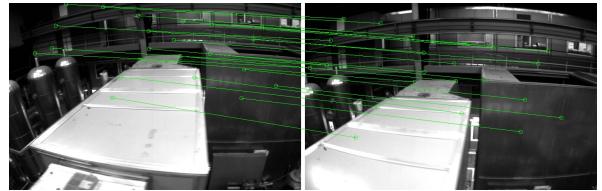
$$\begin{aligned} \min_{\mathcal{X}} \left\{ \|\mathbf{r}_p - \mathbf{H}_p \mathcal{X}\|^2 + \sum_{k \in \mathcal{B}} \left\| \mathbf{r}_B(\hat{\mathbf{z}}_{b_{k+1}}^{b_k}, \mathcal{X}) \right\|_{\mathbf{P}_{b_{k+1}}^{b_k}}^2 + \right. \\ \left. \sum_{(l,j) \in \mathcal{C}} \rho(\|\mathbf{r}_C(\hat{\mathbf{z}}_l^{c_j}, \mathcal{X})\|_{\mathbf{P}_l^{c_j}}^2) \right\}, \end{aligned} \quad (2)$$



(a) BRIEF descriptor matching results



(b) First step: 2D-2D outlier rejection results



(c) Second step: 3D-2D outlier rejection results.

Fig. 4. Descriptor matching and outlier removal for feature retrieval during loop detection.

where  $r_B(\hat{\mathbf{z}}_{b_{k+1}}^{b_k}, \mathcal{X})$  and  $r_C(\hat{\mathbf{z}}_l^{c_j}, \mathcal{X})$  are nonlinear residual functions for inertial and visual measurements.  $\|\cdot\|$  is the Mahalanobis distance weighted by covariance  $\mathbf{P}$ . To be specific,  $r_B$  is the residual of IMU factor which constrains pair of consecutive frames  $b_k$  and  $b_{k+1}$  by the integration of inertial measurements  $\hat{\mathbf{z}}_{b_{k+1}}^{b_k}$ .  $r_C$  is the residual of vision factor which represents the reprojection error by reprojecting feature  $l$  into frame  $j$  and comparing against raw measurements  $\hat{\mathbf{z}}_l^{c_j}$ .  $\rho(\cdot)$  is the robust huber norm [27] to relieve outliers. Past states are marginalized and converted to the prior information,  $\{\mathbf{r}_p, \mathbf{H}_p\}$ .

Only a small set of recent frames is optimized in the window, and the past states are linearized and fixed into marginalization factor. Therefore, accumulating drift is inevitable in a long run.

##### B. Loop Detection

To achieve relocalization, we need to identify places that have already been visited. We follow a state-of-the-art approach DBOW2 [6] for loop detection. For every keyframe, we detect 500 FAST features [28] and describe them by the BRIEF descriptors [29]. The descriptors are converted to a visual vector to query the visual database. We get the best loop closure candidate from DBOW2. The descriptors are also used for feature retrieving. Raw images are discarded to reduce memory.

1) *Feature Retrieval*: After loop detection, we establish the connection between the current frame and loop closure frame in feature level. The feature matching is performed by the BRIEF descriptors matching. We choose the matching pairs by finding the minimum Hamming distance. Directly

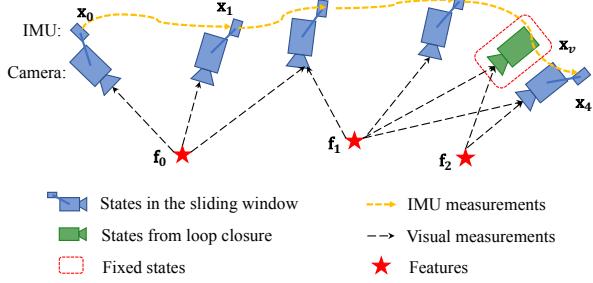


Fig. 5. An illustration of sliding-window based monocular VIO with a loop closure frame. The loop closure frame serves as an additional camera view with the fixed pose in the local window. A local bundle adjustment (BA) jointly optimization keyframes poses, velocity, IMU bias as well as feature depths.

descriptor matching may cause a lot of outliers, as shown in Fig. 4. To remove outliers and verify the loop detection, we perform two-step geometric outlier rejection procedure.

- 2D-2D: We perform fundamental matrix test with RANSAC [30] on 2D observation of matched pairs.
- 3D-2D: We perform PnP test with RANSAC [31] between 3D positions of features (from VIO) and 2D observations on the loop closure frame.

When we find enough inliers, we treat this candidate as a correct loop detection.

### C. Tightly-Coupled Relocalization

Instead of calculating relative pose just between two matched frames, we solve it by jointly optimize loop closure frame within the local window. The loop closure frame is treated as an additional frame with the fixed pose in the local sliding window of VIO, as shown in the Fig. 5. The connection is established by retrieved features.

We use  $v$  to denote loop closure frame. During relocalization, we treat previous pose estimation  $(\hat{\mathbf{q}}_v^w, \hat{\mathbf{p}}_v^w)$  of frame  $v$  as constant. The loop closure frame connects local window by retrieved features. We jointly optimize the sliding window using all IMU measurements, local visual measurement measurements, and retrieved feature correspondences from loop closure. We can easily write the same visual measurement model for retrieved features observed by a loop closure frame  $v$ . The nonlinear cost function in (2) only need to add visual reprojection error term of loop closure frame:

$$\min_{\mathcal{X}} \left\{ \|\mathbf{r}_p - \mathbf{H}_p \mathcal{X}\|^2 + \sum_{k \in \mathcal{B}} \left\| \mathbf{r}_B(\hat{\mathbf{z}}_{b_{k+1}}^{b_k}, \mathcal{X}) \right\|_{\mathbf{P}_{b_{k+1}}^{b_k}}^2 + \sum_{(l,j) \in \mathcal{C}} \rho(\|\mathbf{r}_C(\hat{\mathbf{z}}_l^{c_j}, \mathcal{X})\|_{\mathbf{P}_l^{c_j}}^2) + \underbrace{\sum_{(l,v) \in \mathcal{L}} \rho(\|\mathbf{r}_C(\hat{\mathbf{z}}_l^v, \mathcal{X}, \hat{\mathbf{q}}_v^w, \hat{\mathbf{p}}_v^w)\|_{\mathbf{P}_l^v}^2)}_{\text{reprojection error in loop closure frame}} \right\}, \quad (3)$$

where  $\mathcal{L}$  is the set of the observation of retrieved features.  $(l, v)$  means  $l^{th}$  feature observed in the loop closure frame

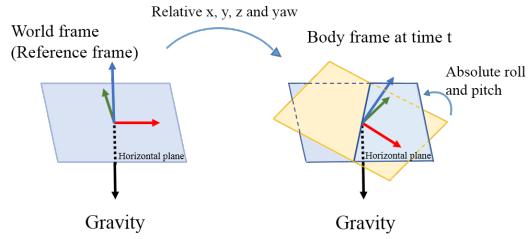


Fig. 6. An illustration of four drifted direction. The measurement of gravity renders roll and pitch angle fully observable. With the movement of the object, the  $x$ ,  $y$ ,  $z$  and yaw angle change relatively with respect to the reference frame. The absolute roll and pitch angle can be determined by the horizontal plane from the gravity vector.

$v$ .

This joint optimization framework results in higher accuracy. The relocalization process effectively shifts local window to "drift-free" location.

### D. Global Pose Graph Optimization

Since we fix past states in the relocalization procedure, the local window shifts to a "drift-free" place immediately. A jumping change will appear on the trajectory. To make the whole trajectory consistent and smooth, we perform a light-weight 4-DOF global pose graph optimization.

1) *Four Accumulated Drift Direction*: Since IMU measures gravity vector, the roll and pitch angles are fully observable in the visual-inertial system. As depicted in the Fig. 6, the gravity is always in the vertical direction. With the movement of the object, the 3D position and rotation change relatively with respect to the reference frame. However, we can determinate horizontal plane by the gravity vectors, that means we can observable the absolute roll and pitch angles all the time. Therefore, the roll and pitch are absolute values in the world frame, while the  $x$ ,  $y$ ,  $z$  and yaw are relative estimates with respect to the reference frame. The accumulated drift only occurs in four degrees-of-freedom ( $x$ ,  $y$ ,  $z$  and yaw angle). To take full advantage of valid information and correct drift efficiently, we fix the drift-free roll and pitch, and only perform pose graph optimization in 4-DOF.

2) *Adding Keyframes into the Pose Graph*: Every keyframe is added into the pose graph after it is marginalized out from VIO local window. One keyframe serves as one vertex in the pose graph. Every vertex connects others by two types of edges, sequential edge and loop edge, as shown in Fig. 7:

- Sequential Edge: a keyframe will connect several previous keyframes with sequential edges. The sequential edge represents the relative transformation between two vertexes, which is taken directly from VIO result. Considering a keyframe  $i$  and one of its previous keyframes  $j$ , the sequential edge contains relative position  $\hat{\mathbf{p}}_{ij}^i$  in local frame and relative yaw angle  $\hat{\psi}_{ij}$ ,

$$\begin{aligned} \hat{\mathbf{p}}_{ij}^i &= \hat{\mathbf{R}}_i^{w-1}(\hat{\mathbf{p}}_j^w - \hat{\mathbf{p}}_i^w) \\ \hat{\psi}_{ij} &= \hat{\psi}_j - \hat{\psi}_i. \end{aligned} \quad (4)$$

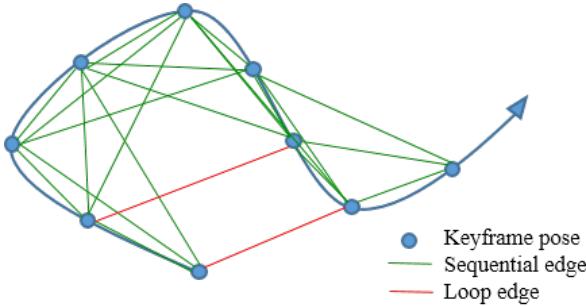


Fig. 7. An illustration of pose graph. The keyframe serves as a vertex in the pose graph and it connects other vertexes by sequential edges and loop edges. Every edge represents relative translation and relative yaw angle.

- Loop Edge: if loop detection happens, the keyframe will connect the loop closure frame by a loop edge. Similar with the sequential edge, the loop edge also contains 4-DOF relative transformation. The value of the loop closure edge is obtained from relocalization result.

3) *4-DOF Pose Graph Optimization*: The main idea of the pose graph optimization is that we adjust the poses of vertexes, such that the configuration matches the edges as much as possible. In our framework, we only adjust 3D position  $\mathbf{p}^w$  and yaw angle  $\psi$  of vertexes, and set their roll and pitch angles as constant variables. Such that loop correction will not occur in drift-free direction. We define the residual of the edge between frame  $i$  and  $j$  as:

$$\mathbf{r}_{i,j}(\mathbf{p}_i^w, \psi_i, \mathbf{p}_j^w, \psi_j) = \begin{bmatrix} \mathbf{R}(\hat{\phi}_i, \hat{\theta}_i, \psi_i)^{-1}(\mathbf{p}_j^w - \mathbf{p}_i^w) - \hat{\mathbf{p}}_{ij}^i \\ \psi_j - \psi_i - \hat{\psi}_{ij} \end{bmatrix}, \quad (5)$$

where  $\mathbf{p}_i^w, \psi_i, \mathbf{p}_j^w, \psi_j$  are variables of  $i$  and  $j$  frame.  $\hat{\phi}_i, \hat{\theta}_i$  are the fixed roll and pitch angles obtained from VIO.  $\hat{\mathbf{p}}_{ij}^i, \hat{\psi}_{ij}$  are relative transformation from edge.

The residual of all sequential edges and loop closure edges are formed into following least squares problem:

$$\min_{\mathbf{p}, \psi} \left\{ \sum_{(i,j) \in \mathcal{S}} \|\mathbf{r}_{i,j}\|^2 + \sum_{(i,j) \in \mathcal{L}} \rho(\|\mathbf{r}_{i,j}\|^2) \right\}, \quad (6)$$

where  $\mathcal{S}$  is the set of all sequential edges and  $\mathcal{L}$  is the set of all loop closure edges. Huber norm  $\rho(\cdot)$  is used to further reduce the impact of any possible wrong loops.

4) *Map Merging*: The pose graph can not only optimize current map, but also merge current map with a previous-built map. If we have loaded a previous-built map and detected loop connections between two map, we can merge them together. Since all edges are relative constraints, the pose graph optimization automatically merges two maps together by the loop connections. As shown in the Fig. 8, the current map is pulled into the previous map by loop edges. Every vertex and every edge are relative variables, therefore, we only need to fix the first vertex in the pose graph.

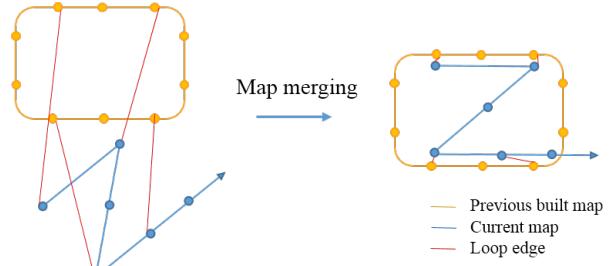


Fig. 8. An illustration of map merging. The yellow figure is previous-built map. The blue figure is the current map. Two maps are merged according to the loop connections.

5) *Pose Graph Management*: When the travel distance increases, the size of the pose graph may grow unbounded, limiting the real-time performance of the system. To this end, we implement a downsample process to maintain the pose graph database to a limited size. All keyframes with loop closure constraints will be kept, while other keyframes that are either too close or have very similar orientations to its neighbors are removed.

#### E. Map Reuse

In a stable environment, we can first build and save the map. Then we load and reuse the map for the next time. Current pose is relocated in the previous map and the current map is merged into the previous map. Not only we achieve map reuse, but also we can always get the absolute odometry in this known environment.

1) *Pose Graph Saving*: The structure of our pose graph is very simple. We only need to save vertexes and edges, as well as descriptors of every keyframe (vertex). Raw images are discarded to reduce memory consumption. To be specific, the states we save for  $i^{th}$  keyframe are:

$$[i, \hat{\mathbf{p}}_i^w, \hat{\mathbf{q}}_i^w, v, \hat{\mathbf{p}}_{iv}^i, \hat{\psi}_{iv}, \mathbf{D}(u, v, des)], \quad (7)$$

where  $i$  is frame index,  $\hat{\mathbf{p}}_i^w$  and  $\hat{\mathbf{q}}_i^w$  are position and orientation from VIO. If this frame has a loop closure frame,  $v$  is the loop closure frame's index.  $\hat{\mathbf{p}}_{iv}^i$  and  $\hat{\psi}_{iv}$  are the relative position and yaw angle between these two frames, which is obtained from relocalization.  $\mathbf{D}(u, v, des)$  is feature set. Each feature contains 2D observation and its BRIEF descriptor (32 Byte). The feature descriptors cost the most memory, which equals a  $500 \times 32$  resolution image for 500 features in one keyframe. Therefore, it takes approximately 17kB for one keyframe.

2) *Pose Graph Loading*: We use the same saving format to load keyframe. Every keyframe is a vertex in the pose graph. The initial pose of vertex is  $\hat{\mathbf{p}}_i^w, \hat{\mathbf{q}}_i^w$ . The loop edge is established directly by the loop information  $\hat{\mathbf{p}}_{iv}^i, \hat{\psi}_{iv}$ . Every keyframe establishes several sequential edges with its neighbor keyframes, as eq. (4). After loading the pose graph, we perform global 4-DOF pose graph once immediately. The speed of pose graph saving and loading is in linear correlation with pose graph's size.

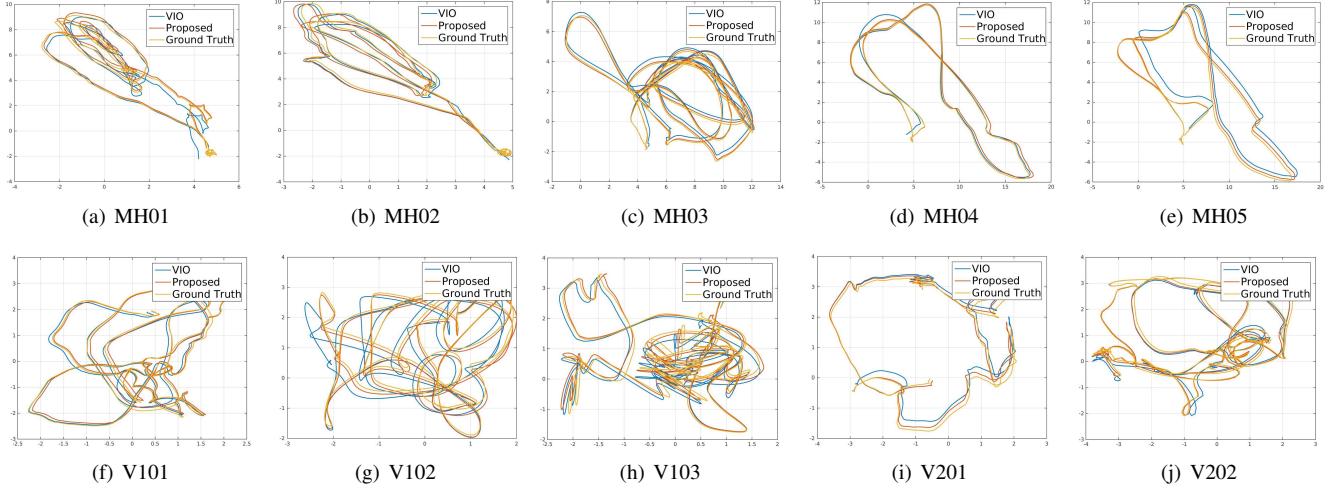


Fig. 9. Trajectory of proposed system in EuRoC dataset. Our system compares with ground truth and pure VIO without loop closure.

TABLE I  
ABSOLUTE TRAJECTORY ERROR, ATE [32] IN EUROC DATASETS IN METERS.

Sequence	MH01	MH02	MH03	MH04	MH05	V101	V102	V103	V201	V201	MH.all_merged
VIO without loop	0.15	0.15	0.22	0.32	0.30	0.079	0.11	0.18	0.080	0.16	
VI SLAM [22]	0.25	0.18	0.21	0.30	0.35	0.11	0.13	0.20	0.12	0.20	
proposed	<b>0.12</b>	<b>0.12</b>	<b>0.13</b>	<b>0.18</b>	<b>0.21</b>	<b>0.068</b>	<b>0.084</b>	<b>0.19</b>	<b>0.081</b>	<b>0.16</b>	<b>0.21</b>

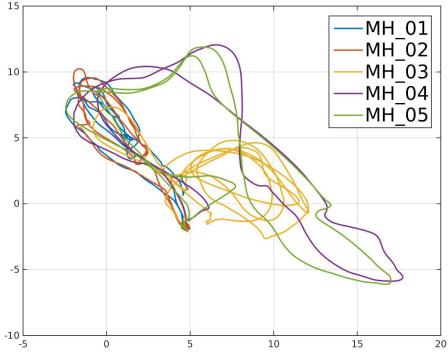


Fig. 10. Trajectories of all Machine hall sequences in a global map.

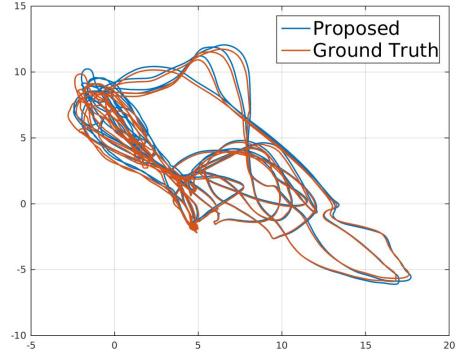


Fig. 11. Trajectories merging results of our system compare against with ground truth.

## V. EXPERIMENT RESULTS

We validate proposed system on a public dataset and outdoor environment. In the public dataset experiments, we compare the proposed algorithm with another state-of-the-art algorithm [22]. A numerical analysis shows the accuracy of our proposed system. We also merge different sequences into a global pose graph. The outdoor experiment is performed to illustrate the large-scale practicability of our system.

### A. Public Dataset

We evaluate proposed system using the EuRoC MAV Visual-Inertial Datasets [33]. The datasets are collected on-board a micro aerial vehicle, which contains stereo images (Aptina MT9V034 global shutter, WVGA monochrome, 20

FPS), synchronized IMU measurements (ADIS16448, 200 Hz), and ground truth states (VICON and Leica MS50). The datasets contain 5 sequences in the machine hall and 6 sequences in the Vicon room. The ground truth is provided by the laser tracker and motion capture system respectively. Only images from the left camera are used in experiments.

The trajectory of all the sequences is shown in Fig. 9. We compare the proposed system with ground truth and VIO without loop closure [5] in one figure. It can be seen that our relocalization and global pose graph optimization greatly increase the accuracy of pure VIO. For quantitative analysis, we compare our system against another state-of-the-art SLAM work, VI SLAM [22], which is built on



Fig. 12. The device we used for the indoor experiment. It contains one global shutter camera (MatrixVision mvBlueFOX-MLC200w) with  $752 \times 480$  resolution. We use the built-in IMU (ADXL278 and ADXRS290, 400Hz) from the DJI A3 flight controller.

the top of OKVIS [14]. VI SLAM relocalize camera pose only between two frames. It performs 6-DOF pose graph optimization after relocalization. We compare quantitative results in terms of absolute trajectory error (ATE, [32]). As shown in Table. I, our relocalization and global pose graph optimization improve pure VIO result obviously. Furthermore, our proposed system outperforms VI SLAM [22] in the most of sequences. Because our relocalization performs in a tightly-coupled local window instead of only two frames, our relocalization results are more accurate. In addition, our 4-DOF optimization seizes the actual drifted direction, ignoring drift-free directions, which corrects drift more effectively and accurately.

We also merge five Machine hall sequences into one map. To the best of our knowledge, this is the first work trying to splice different visual-inertial sequences together on EuRoC dataset. These sequences start at different poses and different times. We do relocalization and pose graph optimization based on similar camera views in diffident sequences. We only fix the first frame in the first sequence, whose position and yaw angle is set to zero. Then we merge new sequences into previous map one by one. The trajectory is shown in Fig. 10. We also compare the whole trajectory with ground truth. The ATE is 0.34m, which is an impressive result in a 500-meter-long run in total. This experiment shows that the map "evolves" overtime by incrementally merging new sensor data captured at different times.

### B. Large-scale Outdoor Environment

TABLE II  
TIMING TABLE IN OUTDOOR EXPERIMENT.

Thread	Modules	Time(ms)	Rate
1	VIO	40	10 Hz
2	Loop Detection	8	every keyframe
2	Relocalization	40	every loop
2	Pose Graph Optimization	186	every loop
	Pose Graph Save	907	once
	Pose Graph Load	4464	once

\* This table represents maximum time cost in the outdoor experiment, which has 2747 keyframes in the pose graph.

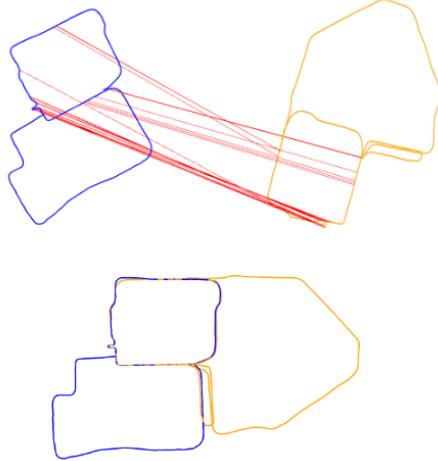


Fig. 13. The left picture shows the trajectories of two sequences respectively. The yellow line is the trajectory of sequence 1 and the blue line is the trajectory of sequence 2. The red lines, connecting two trajectories, draw loop detection places. The right picture shows the merging results.



Fig. 14. The whole trajectory aligns with Google Map.

This experiment valid proposed relocalization and pose graph optimization in the large-scale outdoor environment.

The sensor suite used in this experiment is shown in Fig. 12. It contains a monocular camera (MatrixVision mvBlueFOX-MLC200w, 20Hz,  $752 \times 480$  resolution) and an IMU (ADXL278 and ADXRS290, 400Hz) inside the DJI A3 controller<sup>2</sup>. The camera and IMU are hardware-synchronized. The intrinsic parameter of the camera is calibrated offline. The extrinsic parameter between camera and IMU is calibrated online. The two outdoor sequences are collected by a person walking on the campus at different times. The first sequence is around 740 m, and the second sequence is 540m. We first build and save the map of sequence one. Then we load this map into memory. The sequence two starts with an arbitrary unknown position. Every keyframe is used to detect loop with the previous-built map. Once a loop is detected, we do relocalization and global pose graph optimization to fuse this two map together.

We process data on a desk computer equipped with an Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz. The timing

<sup>2</sup><http://www.dji.com/a3>

table is shown in the II. The whole system runs in the real time. We do loop detection for every new-coming keyframes. Only when new loop is detected, we perform relocalization and pose graph optimization,

The trajectory is shown in Fig. 13. The top picture in the Fig. 13 shows the trajectories of two sequences respectively. The yellow line is the trajectory of sequence 1, which serves as a previous-built map. The blue line is the trajectory of sequence 2, which serves as a current local map. The red lines, connecting two trajectories, represent loop connection between two maps. The bottom picture in the Fig. 13 shows the merging results. The two map tightly integrated together by global pose graph optimization. For intuitive visualization purpose, we align the whole trajectory with Google Map in Fig. 14. The trajectory matches Google Map well, which validates proposed system.

## VI. CONCLUSION

In this paper, we propose a monocular visual-inertial SLAM system which has the capability of relocalization and pose graph optimization to achieve global consistency in real-time when loop closure happens. Our system can relocalize camera position in the previous-built map and merge current map with the previous map by pose graph optimization. The whole system saves and loads a pose graph efficiently, which has the ability to reuse previous results.

Our system has the potential ability of building map for a huge city. In the future, we want to collect data and built local maps in multi distributed devices. Then we merge all the local maps into a huge global map together. Finally, we can relocalize and get the absolute pose in this global map wherever you look at.

## REFERENCES

- [1] C. Forster, M. Pizzoli, and D. Scaramuzza, “SVO: Fast semi-direct monocular visual odometry,” in *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, Hong Kong, China, May 2014.
- [2] J. Engel, T. Schöps, and D. Cremers, “Lsd-slam: Large-scale direct monocular slam,” in *European Conference on Computer Vision*. Springer International Publishing, 2014, pp. 834–849.
- [3] R. Mur-Artal, J. Montiel, and J. D. Tardos, “Orb-slam: a versatile and accurate monocular slam system,” *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.
- [4] J. Engel, V. Koltun, and D. Cremers, “Direct sparse odometry,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.
- [5] T. Qin and S. Shen, “Robust initialization of monocular visual-inertial estimation on aerial robots,” in *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Vancouver, Canada, 2017.
- [6] D. Gálvez-López and J. D. Tardós, “Bags of binary words for fast place recognition in image sequences,” *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, October 2012.
- [7] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, “Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments,” in *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, 2012, pp. 957–964.
- [8] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, “A robust and modular multi-sensor fusion approach applied to mav navigation,” in *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, 2013, pp. 3923–3929.
- [9] A. I. Mourikis and S. I. Roumeliotis, “A multi-state constraint Kalman filter for vision-aided inertial navigation,” in *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, Roma, Italy, Apr. 2007, pp. 3565–3572.
- [10] M. Li and A. Mourikis, “High-precision, consistent EKF-based visual-inertial odometry,” *Int. J. Robot. Research*, vol. 32, no. 6, pp. 690–711, May 2013.
- [11] K. Wu, A. Ahmed, G. A. Georgiou, and S. I. Roumeliotis, “A square root inverse filter for efficient vision-aided inertial navigation on mobile devices,” in *Robotics: Science and Systems*, 2015.
- [12] M. K. Paul, K. Wu, J. A. Hesch, E. D. Nerurkar, and S. I. Roumeliotis, “A comparative analysis of tightly-coupled monocular, binocular, and stereo vins,” in *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, Singapore, May 2017.
- [13] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, “isam2: Incremental smoothing and mapping using the bayes tree,” *Int. J. Robot. Research*, vol. 31, no. 2, pp. 216–235, 2012.
- [14] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, “Keyframe-based visual-inertial odometry using nonlinear optimization,” *Int. J. Robot. Research*, vol. 34, no. 3, pp. 314–334, Mar. 2014.
- [15] S. Shen, N. Michael, and V. Kumar, “Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft MAVs,” in *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, Seattle, WA, May 2015.
- [16] Z. Yang and S. Shen, “Monocular visual-inertial state estimation with online initialization and camera-imu extrinsic calibration,” *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 39–51, 2017.
- [17] A. Kendall, M. Grimes, and R. Cipolla, “Posenet: A convolutional network for real-time 6-dof camera relocalization,” in *IEEE international conference on computer vision*, 2015, pp. 2938–2946.
- [18] S. Lynen, T. Sattler, M. Bosse, J. A. Hesch, M. Pollefeys, and R. Siegwart, “Get out of my lab: Large-scale, real-time visual-inertial localization,” in *Robotics: Science and Systems*, 2015.
- [19] K. Qiu, T. Liu, and S. Shen, “Model-based global localization for aerial robots using edge alignment,” *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1256–1263, 2017.
- [20] M. Burri, H. Oleynikova, M. W. Achtelik, and R. Siegwart, “Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments,” in *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, 2015, pp. 1872–1878.
- [21] R. Mur-Artal and J. D. Tardós, “Visual-inertial monocular slam with map reuse,” *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.
- [22] A. Kasyanov, F. Engelmann, J. Stückler, and B. Leibe, “Keyframe-based visual-inertial online slam with relocalization,” *arXiv preprint arXiv:1702.02175*, 2017.
- [23] S. Leutenegger, M. Chli, and R. Y. Siegwart, “Brisk: Binary robust invariant scalable keypoints,” in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2548–2555.
- [24] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *2011 IEEE international conference on Computer Vision (ICCV)*, 2011, pp. 2564–2571.
- [25] J. Shi and C. Tomasi, “Good features to track,” in *Proc. of the IEEE Int. Conf. on Pattern Recognition*, Seattle, WA, Jun. 1994, pp. 593–600.
- [26] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proc. of the Intl. Joint Conf. on Artificial Intelligence*, Vancouver, Canada, Aug. 1981, pp. 24–28.
- [27] P. Huber, “Robust estimation of a location parameter,” *Annals of Mathematical Statistics*, vol. 35, no. 2, pp. 73–101, 1964.
- [28] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” *Computer vision–ECCV 2006*, pp. 430–443, 2006.
- [29] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” *Computer Vision–ECCV 2010*, pp. 778–792, 2010.
- [30] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [31] V. Lepetit, F. Moreno-Noguer, and P. Fua, “Epnp: An accurate o (n) solution to the pnp problem,” *International journal of computer vision*, vol. 81, no. 2, pp. 155–166, 2009.
- [32] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of rgb-d slam systems,” in *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, 2012, pp. 573–580.
- [33] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, “The euroc micro aerial vehicle datasets,” *Int. J. Robot. Research*, 2016.