Monocular Visual-Inertial State Estimation for Mobile Augmented Reality

Peiliang Li[1], Tong Qin[1], Botao Hu[2], Fengyuan Zhu[3] and Shaojie Shen[1]

[1]Robotics Institute, Hong Kong University of Science and Technology *
[2]Amber Garage, Inc †
[3]ITP, New York University ‡

Abstract

Mobile phones equipped with a monocular camera and an inertial measurement unit (IMU) are ideal platforms for augmented reality (AR) applications, but the lack of direct metric distance measurement and the existence of aggressive motions pose significant challenges on the localization of the AR device. In this work, we propose a tightly-coupled, optimization-based, monocular visual-inertial state estimation for robust camera localization in complex indoor and outdoor environments. Our approach does not require any artificial markers, and is able to recover the metric scale using the monocular camera setup. The whole system is capable of online initialization without relying on any assumptions about the environment. Our tightly-coupled formulation makes it naturally robust to aggressive motions. We develop a lightweight loop closure module that is tightly integrated with the state estimator to eliminate drift. The performance of our proposed method is demonstrated via comparison against state-of-the-art visual-inertial state estimators on public datasets and real-time AR applications on mobile devices. We release our implementation on mobile devices as open source software [1].

Index Terms: I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Tracking; H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities

1 Introduction

Augmented Reality (AR) has been drawing increasing attention due to its potential to provide people with immersively interactive experience. Real-time, precise, and drift-free estimation of the camera pose, along with environment representation with metric scale, are required for AR applications. However, the major bottleneck of current AR system is the requirement of artificial markers [14] or extra range sensors to initialize or maintain metric scale while solving odometry incrementally. Purely vision-based approaches are also prone to failure during aggressive motions due to loss of feature tracks or motion blurs. To this end, we propose a monocular visual-inertial state estimation approach for robust camera localization. Our approach runs on common smartphones without relying on any prior information or environment assumptions [2].

The sensor package on a standard smartphone usually consists of a consumer-level camera and a low-cost inertial measurement unit (IMU), which forms the minimum sensor suite to implement a visual-inertial system. On the one hand, the camera provides sufficient environmental perception, but is unable to provide metric scale information. Vision-based motion tracking is also prone to failure during aggressive motions. On the other hand, the IMU gives out outlier-free motion information at high frequency, which is particularly useful during aggressive motions that are frequently encountered in AR applications. However, the low-cost IMU used on smartphones are not accurate enough for using it as a standalone motion sensor. To this end, due to the complementary nature of visual and inertial sensors, the proper fusing these two types of measurements gives the most viable solution for state estimation for AR applications.

Monocular visual-inertial systems (VINS) are highly nonlinear, and to properly fuse the camera and IMU measurements, we need a robust initialization to bootstrap the whole system. For practical AR applications, robustness to aggressive motion and loop awareness are necessary since users may move the camera arbitrarily and visit the same scene from different perspectives. Altogether, we put initialization, motion tracking robustness, and loop closure as the

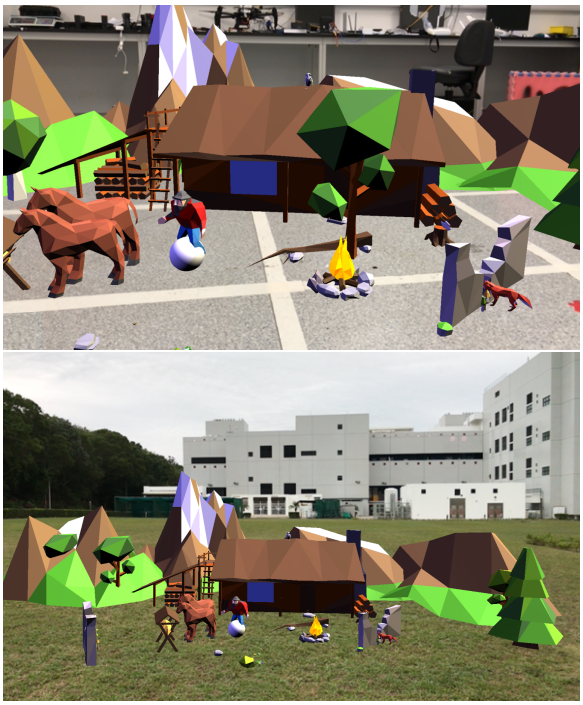


Figure 1: Screenshots showing indoor and outdoor AR demonstrations based on the proposed monocular VINS estimator. Everything runs on a mobile device.

*e-mail: {pliap, tong.qin, eeshaojie}@ust.hk
†e-mail: botao@ambergarage.com
‡e-mail: fz567@nyu.edu

[1]https://github.com/PeiliangLi/VINS-AR

[2]A video showing experimental results can be found at `http://www.ece.ust.hk/~eeshaojie/ismar2017peiliang.mp4`

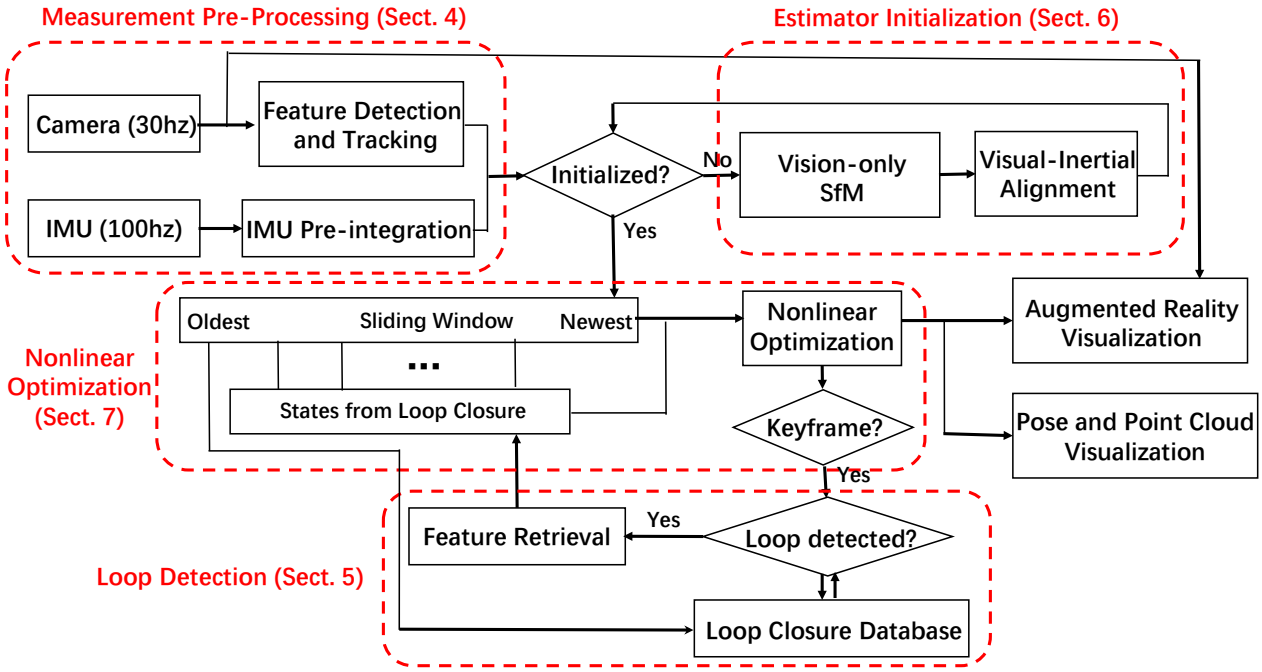


Figure 2: Block diagram illustrating the full pipeline of the proposed visual-inertial AR system.

three most important requirements for our monocular VINS estimator.

This work is a significant improvement from our previous works on monocular VINS for aerial robots [23, 24, 28]. To handle the low-performance IMUs on mobile phones, we improve the initialization procedure by taking gyroscope biases into consideration. We add a loop detection module to our system, and propose a lightweight tightly-coupled fusion approach to seamlessly integrate loop information into our sliding window monocular VINS estimator. All these efforts come together as an extremely easy-to-use system that does not require any prior knowledge or assumption about the environment. It works in a drift-free fashion in both indoor and outdoor in large-scale environments. We port our estimator to mobile devices and implement an AR demonstration (Fig. 1). To best benefit the community, we release our implementation as open source software. We summarize our contributions as follows:

- An improved tightly-coupled, optimization-based monocular visual-inertial state estimation framework with robust initialization, IMU bias calibration, and loop closure.
- Real-time implementation of state estimator and AR demonstrations on mobile devices.
- Open source release.

The rest of the paper is organized as follows. In Sect. 2, we discuss the relevant literature. We give an overview of the complete system pipeline in Sect. 3. Measurement pre-processing, including feature tracking front-end, IMU pre-integration, are presented in Sect. 4. The loop detection and feature retrieval module is discussed in Sect. 5. In Sect. 6, we discuss the robust initialization procedure that recovers platform velocity, attitude, and metric scale without any prior knowledge or assumptions about the environment. A tightly-coupled, nonlinear optimization-based, sliding window monocular VINS estimator, which is an extension of our previous works [23, 24], is presented in Sect. 7. We discuss implementation details and present experimental results in Sect. 8, in which our system is compared against state-of-the-art visual-inertial odometry [12] on public datasets. Finally, the paper is concluded with a discussion of possible future research in Sect. 9.

2 Related Work

There are extensive studies on state estimation solutions for AR applications. Most of them utilize cameras as the primary sensor. Pioneering work on vision-based state estimation for AR was proposed in [11], where camera pose estimation and feature map update are decoupled to achieve real-time operations. An improved monocular approach showing remarkable tracking and re-localization ability is presented in [19]. In [14], the author uses an object with known size to initialize the scale, and perform AR functionalities thereafter. [22] focuses on estimating the dense collision mesh of the environment using direct method. However, neither of the above methods fuses visual information with IMU measurements to solve the metric scale, nor do they show the ability to operate in large-scale environments using mobile devices.

More relevant to our work is the family of research on visual-inertial (VINS) state estimation conducted in the computer vision or robotics community. Utilizing stereo [12], RGB-D [7], or only monocular [6, 13] cameras. VINS solutions can be categorized into filtering-based [6, 9, 10, 13, 18], or bundle adjustment/graph optimization-based [8, 12, 23] approaches. Mathematically, both filtering and optimization-based approaches are equivalent, as both of them are realizations of nonlinear maximum likelihood estimators. Towards the implementation side, filtering-based approaches may require fewer computational resources due to the continuous marginalization of past states, but they may have slightly lower performance due to the early fixing of linearization points. On the other hand, graph optimization-based approaches may improve performance via iterative re-linearization at the expense of higher computational demands. Considering from another angle, we can categorize VINS approaches into loosely-coupled [27] or tightly-coupled [6, 8, 9, 12, 13, 23] approaches. Loosely-coupled methods usually separately fuse vision-only pose estimation modules such

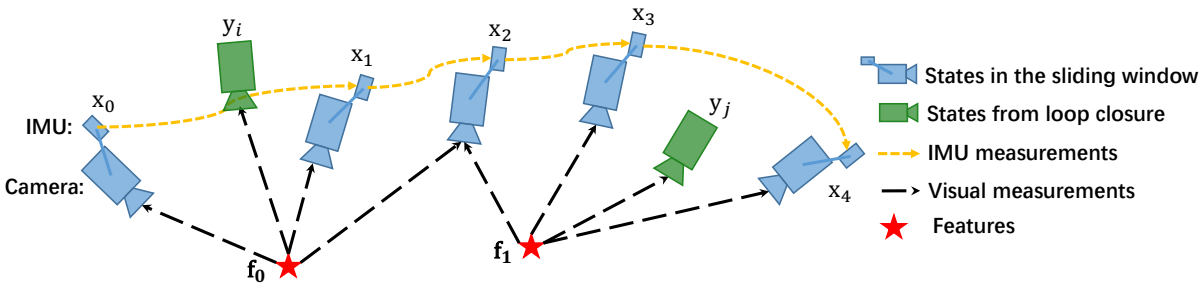


Figure 3: An illustration of our sliding window formulation. All measurements, including pre-integrated IMU quantities, visual observations, and feature correspondences from loop closures, are fused in a tightly-coupled manner.

as PTAM [11] or LSD-SLAM [2] with inertial module [27] to recover the metric scale and velocity. However, loosely-coupled approaches are incapable of eliminating drifts occurred in the vision-only module, which leads to sub-optimal results. Tightly-coupled approaches consider the tight interaction between visual and IMU measurements, which can implicitly incorporate the structural information from the visual measurements into the IMU bias calibration and recover metric scale naturally. It is evidenced through multiple studies that tightly-coupled approaches achieve better accuracy comparing to loosely-coupled ones.

Despite the remarkable results achieved on VINS, we have to acknowledge that the fusion of visual and IMU measurements is a highly nonlinear process, and accurate initial values are required to bootstrap the nonlinear estimator. Initialization is particularly critical for monocular VINS due to the lack of direct scale observations. Recent results suggest that by assuming the orientation is known, VINS may be solved in a linear closed-form [17]. However, these algebraic solutions are sensitive to noisy IMU measurements that are obtained from consumer mobile devices. A probabilistic initialization framework is presented in our previous work [28], and is improved to handle large scene depth and gyroscope bias in [21].

In this work, we study the pros and cons of existing works, and propose a fully integrated estimation solution that includes initialization, nonlinear optimization, IMU bias calibration, and loop closure. Sensor information is fused in a tightly-coupled manner using nonlinear optimization in order to achieve the best accuracy and robustness. Instead of using direct methods for pose estimation, we stay with a feature-based approach due to its better integration with the fusion framework.

3 Overview

Our proposed visual-inertial system consists of four significant modules, as illustrated in Fig. 2. The first module, called the measurements processing front-end, extracts and tracks features for each new image (Sect. 4.1) and pre-integrates all the IMU data between two images (Sect. 4.2). The second module performs estimator initialization. It recovers the metric-scale feature position, platform body velocity, gravity vector and gyroscope bias (Sect. 6.2) by aligning vision-only structure-from-motion (SFM) (Sect. 6.1) with pre-integrated IMU measurements. The third module performs the main nonlinear optimization-based monocular VINS estimator. It solves states in a sliding window formulation by integrating all the visual correspondences, IMU measurements, and loop information (Sect. 7). The fourth module, which runs in the background thread, takes charge of building the keyframe database and detecting loop for each new incoming keyframe (Sect. 5). Finally, we present an AR demonstration by using the VINS outputs to extract a plane from the estimated 3D features and project a 3D model on this plane.

3.1 Notation

We consider $(\cdot)^w$ as world frame, where gravity vector is along with z axis. $(\cdot)^b$ is body frame which is aligned with the IMU. $(\cdot)^c$ is camera frame. We use quaternion $\mathbf{q}$ for rotation representation. b_k is the IMU body frame while taking the k^{th} image. c_k is the camera frame while taking the k^{th} image. We assume that the extrinsic transformation between the camera and the IMU is known.

3.2 Sliding Window Formulation

As illustrated in Fig. 3, we use a sliding window formulation throughout the whole system, including estimation initialization, nonlinear optimization, and loop closure. The full state vector in the sliding window is defined as (the transpose is ignored for simplicity of presentation):

$$\begin{aligned} \mathcal{X} &= [\mathbf{x}_0,\, \mathbf{x}_1,\, \cdots \mathbf{x}_n,\, \lambda_0,\, \lambda_1,\, \cdots \lambda_m] \\ \mathbf{x}_k &= \left[\mathbf{p}^w_{b_k},\, \mathbf{v}^w_{b_k},\, \mathbf{q}^w_{b_k},\, \mathbf{b}^b_a,\, \mathbf{b}^b_g\right], k \in [0, n] \end{aligned} \tag{1}$$

where $\mathbf{x}_k$ is the k^{th} frame state that consists of position $\mathbf{p}^w_{b_k}$, velocity $\mathbf{v}^w_{b_k}$, and orientation $\mathbf{q}^w_{b_k}$ in the world frame, and acceleration bias $\mathbf{b}^b_a$ and gyroscope bias $\mathbf{b}^b_g$ in the IMU body frame. n is the number of keyframes in the sliding window. m is the number of features observed by all frames in the sliding window. λ_l is the inverse depth of the l^{th} feature from its first observed keyframe.

4 Measurement Pre-Processing

All raw camera images and IMU samples are processed through a pre-processing module before they are used for estimator initialization or nonlinear optimization. For visual measurements, we detect and track features in consecutive frames, and we retrieve features from old frames after loop detection. For the IMU measurements, we pre-integrate them between two consecutive frames. Note that since IMU measurements are affected by both bias and noise, we particularly take bias into consideration in IMU pre-integration and in the following optimization. This is essential to enable the usage of low-cost IMU chips in mobile devices.

4.1 Feature Detection and Tracking

For each new image, existing features are tracked using the KLT tracker [15] at 30 Hz. Meanwhile, new corner features are detected [25] to maintain a minimum feature number in every image. The detector enforces uniform feature distribution by setting a minimum separation of 30 pixels between two closed features. An image-level outlier rejection is performed using RANSAC with fundamental matrix test. After this, the temporal connections between frames are represented by feature correspondences.

Keyframes are also selected in this step. We have two criteria for keyframe selection. The first is the average parallax. If the average parallax of the tracked features is beyond a certain threshold,

we treat this image as a keyframe. Note that rotation-only motions cause large pixel displacement, but it does not contribute to the parallax that is required for feature triangulation. In fact, the rotation-only motion is the degenerate motion pattern for monocular VINS. To this end, we use the gyroscope measurements to roughly remove the rotational components when calculating the parallax. We also use tracking quality as the criteria for keyframe switching. In a particular frame, if many tracked features are lost while many new features are detected, beyond the certain threshold, we will treat this frame as a new keyframe.

4.2 IMU Pre-integration

It is often the case that the IMU sends out data at a much higher frequency than the camera. IMU pre-integration is a technique to summarize multiple IMU measurements into a single measurement "block". This avoids excessive evaluation of IMU measurements and saves significant amount of computation power. It also enables the use of IMU measurements without knowing the initial velocity and global orientation. The IMU pre-integration technique was first proposed in [16], and was advanced to consider on-manifold uncertainty propagation in our previous work [23]. Further improvements to incorporate IMU biases and integrate them with a full SLAM framework were proposed in [3].

We denote the real angular velocity and acceleration as $\boldsymbol{\omega}^b$, $\mathbf{a}^b$ and raw IMU measurements as $\hat{\boldsymbol{\omega}}^b$, $\hat{\mathbf{a}}^b$, which are affected by bias $\mathbf{b}$ and noise $\boldsymbol{\eta}$:,

$$\begin{aligned} \hat{\boldsymbol{\omega}}^b &= \boldsymbol{\omega}^b + \mathbf{b}_g + \boldsymbol{\eta}_g \\ \hat{\mathbf{a}}^b &= \mathbf{q}_w^b(\mathbf{a}^w + \mathbf{g}^w) + \mathbf{b}_a + \boldsymbol{\eta}_a, \end{aligned} \tag{2}$$

where we use the quaternion $\mathbf{q}_w^b$ to represent the rotation matrix, which transforms a vector from the world frame to the body frame. Given two time instants k and $k+1$ that correspond to two images, we can pre-integrate the linear acceleration and angular velocity in the local frame b_k:

$$\begin{aligned} \boldsymbol{\alpha}_{b_{k+1}}^{b_k} &= \iint_{t\in[k,k+1]} \boldsymbol{\gamma}_{b_t}^{b_k} \hat{\mathbf{a}}^{b_t} dt^2 \\ \boldsymbol{\beta}_{b_{k+1}}^{b_k} &= \int_{t\in[k,k+1]} \boldsymbol{\gamma}_{b_t}^{b_k} \hat{\mathbf{a}}^{b_t} dt \\ \boldsymbol{\gamma}_{b_{k+1}}^{b_k} &= \int_{t\in[k,k+1]} \boldsymbol{\gamma}_{b_t}^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\hat{\boldsymbol{\omega}}^{b_t} \end{bmatrix} dt, \end{aligned} \tag{3}$$

In the above equations, $\otimes$ denotes the quaternion multiplication operation. It can be seen that the pre-integrated measurement block can be obtained solely with IMU measurements within $[k, k+1]$ since it is independent of any initial conditions (position, velocity, attitude).

5 Loop Detection and Feature Retrieval

In order to eliminate the drift caused by incremental state estimation, we utilize DBoW2 [4], a state-of-the-art implementation of bag-of-words (BOW) place recognition. The loop detection runs in a separate thread to minimize the impact to real-time state estimation. The construction of the loop closure database follows our sliding window estimator structure, as illustrated in Fig. 4. A keyframe, along with its latest estimated pose, is added into the database whenever it slides out of the window and is marginalized out from the estimator.

In the loop detection process, we try to find possible matches between the newest keyframe and frames in the loop closure database, again illustrated in Fig. 4. If an eligible loop ($\mathbf{y}_i$) is found, the corresponding loop closure frame will be used to constrain states in the sliding window in a tightly-coupled fashion.

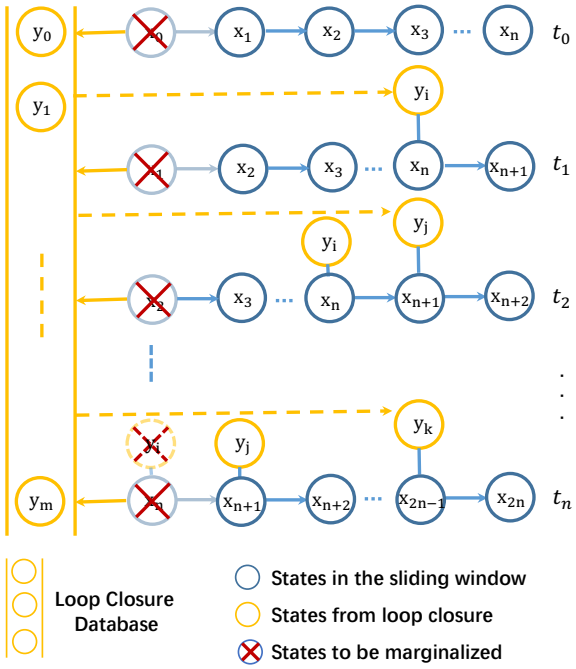


Figure 4: An illustration of our sliding window formulation with loop fusion. $\mathbf{x}_n$ represent keyframes. Larger indices indicate newer keyframes. At t_0, we query the loop closure database to search loop for the newest keyframe $\mathbf{x}_n$. If loop $\mathbf{y}_i$ is found in the database, it will be put into our window and optimized together at t_1. We repeat this process at t_2, and the earlier loop closure frame $\mathbf{y}_i$ slides backward along with the window. Note that all keyframes that are marginalized out ($\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, all the way to $\mathbf{x}_n$ in sequential order) will be added to the loop closure database. In particular, at t_n, $\mathbf{x}_n$ is marginalized out and added into the database, and $\mathbf{y}_i$ is simply dropped since it is already in the database.

We leverage DBoW2 to find potential candidates for loop closure frames. Instead of directly choosing the one that has the best similarity score with the current keyframe, we choose the earliest among the top 25 percent of the candidates to avoid loops that are temporally nearby. Because the earlier keyframes can eliminate the accumulated drift more effectively. After getting the loop closure frame, as Fig. 5 shows, we search for feature correspondences between the current keyframe and the loop closure candidate frame using the following two feature retrieval strategies.

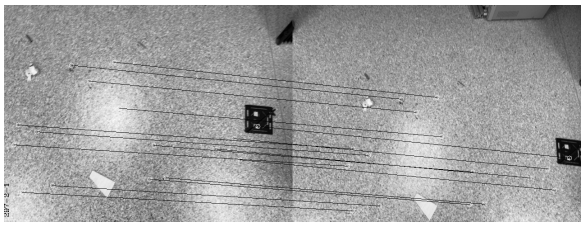
5.1 Correspondence by Feature Tracking

The first strategy applies to scenarios with small pose drift. We back-project all 3D features observed in the current keyframe to the loop closure candidate frame to serve as an initial guess. We then use the KLT tracker to find correspondences in the loop closure frame. Outliers are rejected through a multi-step process. The tracking score in KLT first rejects features that cannot be tracked in the loop closure frame. RANSAC with fundamental matrix test is then applied to find a consistent set of feature correspondences. When the number of inliers is sufficient, we declare successful loop closure using this strategy. Small drift can guarantee accurate initial guesses, so this method is able to efficiently recover feature correspondences that can be used in tightly-coupled optimization (Sect. 7.4).

5.2 Sub-sampling from BoW Vector Matches

In extreme cases where the pose drift is large, which leads to large pixel displacement between the re-projected initial guess and the

(a)



(b)

Figure 5: Loop closure feature retrieval results using two different strategies. Fig. 5(a) illustrates loop closure correspondence by feature tracking (Sect. 5.1). Since we use projection as the initial guess for tracking, in the case of small pose drift, this method can return many inliers even under long baseline and image blur. Fig. 5(b) represents the sub-sampling from BoW vectors(Sect. 5.2). This method will be activated if the drift is large. Although it will generate fewer correspondences, it is still enough to correct the drift due to the tightly-coupled loop fusion.

actual feature correspondence, feature tracking-based approaches may not be able to find sufficient matches. In such cases, we find feature correspondences using BoW vector pairs given by DBoW2. We need to select pairs which correspond to features observed in the current keyframe and reject outliers. We sub-sample the BoW pairs in a small neighborhood around current keyframe features, then apply RANSAC for outlier rejection. We choose the most possible match in each neighborhood as the retrieved feature correspondence.

6 Estimator Initialization

Monocular VINS is a highly nonlinear system with states that are not directly observable from raw measurements. We know that metric scale, velocity, and gravity-aligned attitude can only be indirectly estimated from acceleration, angular velocity, and feature measurements. A good initial guess is required to bootstrap the whole estimator. Vision-only structure from motion (SfM) has a good initialization property by deriving the initial guess from a relative motion method, such as the eight-point [5] and five-point [20] algorithms. Inspired by this, we employ a loosely-coupled alignment strategy between the visual-only SfM and pre-integrated IMU measurements to find initial values for the metric scale (feature depth), gravity vector, body velocity, and gyroscope bias.

This is a two-step process. We first construct a vision-only SfM using a window of keyframes and feature correspondences. The second step is the visual-inertial alignment where we scale the SfM result to metric scale, and align it with the gravity direction.

6.1 Vision-Only Structure from Motion

We first choose two keyframes which contain sufficient feature correspondences and parallax in the sliding window. Five-point method [20] is then used to recover the relative rotation and up-to-scale translation between these two keyframes. We fix the scale of translation to some arbitrary number, and triangulate features observed in these two frames. Based on these triangulated features, the Perspective-n-Point (PnP) method is performed to estimate poses of all keyframes in the sliding window.

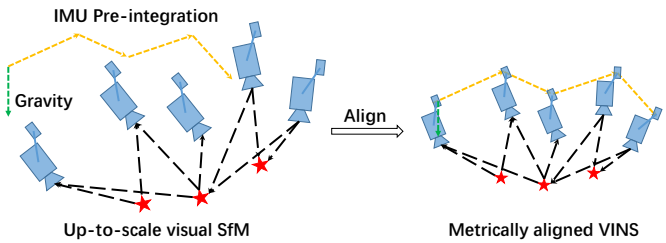


Figure 6: An illustration of the visual-inertial alignment process.Both metric scale and gravity direction are aligned.

Bundle adjustment [26] is then applied to minimize the total reprojection error of all feature observations between all frames. We get all keyframe poses ($\bar{\mathbf{p}}^v_{c_k}$,$\mathbf{q}^v_{c_k}$) and feature positions. Here, $(\cdot)^v$ is an arbitrarily fixed visual base frame. Note that translation components and feature depths are all up-to-scale. Assume that we know the extrinsic parameter ($\mathbf{p}^c_b, \mathbf{q}^c_b$) between the camera and the IMU, we can transform all poses to the IMU frame:

$$\begin{aligned} \mathbf{q}^v_{b_k} &= \mathbf{q}^v_{c_k} \otimes \mathbf{q}^c_b \\ s\bar{\mathbf{p}}^v_{b_k} &= s\bar{\mathbf{p}}^v_{c_k} + \mathbf{q}^v_{c_k}\mathbf{p}^c_b \end{aligned} \tag{4}$$

where s is an unknown scale, which will be solved in the following visual-inertial alignment step.

6.2 Visual-Inertial Alignment

We note that the gyroscope bias that often exists in low-cost IMUs can pose serve negative impact on the estimator performance. Therefore, in the visual-inertial alignment process, we first recover the gyroscope bias, then initialize velocity, gravity vector, and metric scale.

6.2.1 Gyroscope Bias

Considering two consecutive frames b_k and b_{k+1} in the window, we have the relative rotation $\mathbf{q}^v_{b_k}$ and $\mathbf{q}^v_{b_{k+1}}$ from the visual SfM (Sect. 6.1), as well as relative constraint $\hat{\boldsymbol{\gamma}}^{b_k}_{b_{k+1}}$ from IMU pre-integration (Sect. 4.2). We estimate the gyroscope bias by minimizing the error between these two terms:

$$\begin{aligned} &\min_{b_g} \sum_{k\in\mathcal{B}} \left\| \mathbf{q}^{b_k}_v \otimes \mathbf{q}^v_{b_{k+1}} - \boldsymbol{\gamma}^{b_k}_{b_{k+1}} \right\|^2 \\ &\boldsymbol{\gamma}^{b_k}_{b_{k+1}} \approx \hat{\boldsymbol{\gamma}}^{b_k}_{b_{k+1}} + \frac{\partial \boldsymbol{\gamma}^{b_k}_{b_{k+1}}}{\partial \mathbf{b}_g}\delta\mathbf{b}_g, \end{aligned} \tag{5}$$

where $\mathcal{B}$ indexes the all frames in the window. In the second equation, we linearize the rotation constraint with respect to gyroscope bias. Aligning rotation from visual SfM with the pre-integrated IMU constraint $\boldsymbol{\gamma}$, we can get the estimation of $\mathbf{b}_g$.

After the gyroscope bias is updated, we re-evaluate $\hat{\boldsymbol{\alpha}}^{b_k}_{b_{k+1}}, \hat{\boldsymbol{\beta}}^{b_k}_{b_{k+1}}$ with respect to $\mathbf{b}_g$, following the IMU pre-integration equations (2) and (3).

6.2.2 Initialization of Velocity, Gravity, and Metric Scale

The final step of the initialization process recovers all metric quantities by aligning all IMU pre-integrated measurement blocks with visual SfM in a sliding window fashion, as Fig. 6 shows. We define the variables that we would like to estimate as

$$\mathcal{X}_I = [\mathbf{v}^v_{b_0}, \mathbf{v}^v_{b_1}, \cdots \mathbf{v}^v_{b_n}, \mathbf{g}^v, s], \tag{6}$$

where s is the scale parameter that aligns the visual SfM to the metric scale implicitly provided by IMU measurements. The following linear measurement model describes the relationship between the variables that we want to initialize with respect to the pre-integrated IMU measurements.

$$\hat{\mathbf{z}}^{b_k}_{b_{k+1}} = \begin{bmatrix} \hat{\boldsymbol{\alpha}}^{b_k}_{b_{k+1}} \\ \hat{\boldsymbol{\beta}}^{b_k}_{b_{k+1}} \end{bmatrix} = \mathbf{H}^{b_k}_{b_{k+1}} \mathcal{X}_I + \mathbf{n}^{b_k}_{b_{k+1}}$$

$$\approx \begin{bmatrix} \mathbf{q}^{b_k}_v \Delta t_k & \mathbf{0} & \frac{1}{2}\mathbf{q}^{b_k}_v \Delta t_k^2 & \mathbf{q}^{b_k}_v (\bar{\mathbf{p}}^v_{b_{k+1}} - \bar{\mathbf{p}}^v_{b_k}) \\ -\mathbf{q}^{b_k}_v & \mathbf{q}^{b_k}_v & \mathbf{q}^{b_k}_v \Delta t_k & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}^v_{b_k} \\ \mathbf{v}^v_{b_{k+1}} \\ \mathbf{g}^v \\ s \end{bmatrix}. \tag{7}$$

The measurement matrix $\mathbf{H}^{b_k}_{b_{k+1}}$ between every pairs of consecutive keyframes can be constructed using the up-to-scale pose from visual SfM. $\mathbf{q}^v_{b_k}$, $\bar{\mathbf{p}}^v_{b_k}$ and $\bar{\mathbf{p}}^v_{b_{k+1}}$ are obtained from visual SfM. $\mathbf{q}^{b_k}_v$ is the inverse rotation of $\mathbf{q}^v_{b_k}$, and Δt_k is the time interval between two consecutive keyframes. $\mathbf{n}^{b_k}_{b_{k+1}}$ is modeled as zero mean additive Gaussian noise.

Solving the following least square problem:

$$\min_{\mathcal{X}_I} \sum_{k \in \mathcal{B}} \left\| \hat{\mathbf{z}}^{b_k}_{b_{k+1}} - \mathbf{H}^{b_k}_{b_{k+1}} \mathcal{X}_I \right\|^2, \tag{8}$$

we can get velocities and the gravity vector in the visual base frame $(\cdot)^v$, as well as the scale parameter s. The translational components $\bar{\mathbf{p}}^v$ from the visual SfM will be scaled to metric units. All variables are rotated from the frame $(\cdot)^v$ to the the world frame $(\cdot)^w$ according to the gravity vector which is vertical in world frame.

6.3 Termination Criteria

The initialization process continues in a memoryless sliding window manner. The visual SfM is considered as successful once all reprojection errors fall below a certain threshold. The visual-inertial alignment process is terminated if the norm of the recovered gravity vector is close to the nominal gravity value ($\sim 9.8\,\mathrm{m/s^2}$). Note that the alignment process does not use any prior knowledge about the gravity vector, and the convergence of the initializer towards a known nominal value indicates good metric initialization. Once the termination criteria is achieved, the initialization procedure is completed and these metric values will be fed into a tightly-coupled nonlinear visual-inertial estimator.

7 Tightly-Coupled Nonlinear Optimization

After state initialization, we proceed with a nonlinear estimator for high-accuracy state estimation. This extends our previous work [23, 28] by adding IMU bias calibration and loop fusion into the nonlinear optimization framework. We combine IMU pre-integration, visual correspondences and loop information together in a unified formulation.

7.1 Formulation

We minimize the sum of the Mahalanobis norm of all measurement residuals to obtain a maximum posteriori estimation:

$$\min_{\mathcal{X}} \left\{ \|\mathbf{r}_p - \mathbf{H}_p \mathcal{X}\|^2 + \sum_{k \in \mathcal{B}} \left\| r_{\mathcal{B}}(\hat{\mathbf{z}}^{b_k}_{b_{k+1}}, \mathcal{X}) \right\|^2_{\mathbf{P}^{b_k}_{b_{k+1}}} + \sum_{(l,j) \in \mathcal{C}} \left\| r_{\mathcal{C}}(\hat{\mathbf{z}}^{c_j}_l, \mathcal{X}) \right\|^2_{\mathbf{P}^{c_j}_l} + \sum_{(l,i) \in \mathcal{L}} \| r_{\mathcal{L}}(\hat{\mathbf{z}}^{c_i}_l, \mathcal{X}) \|^2_{\mathbf{P}^{c_i}_l} \right\}, \tag{9}$$

where $r_{\mathcal{B}}(\hat{\mathbf{z}}^{b_k}_{bk+1}, \mathcal{X})$, $r_{\mathcal{C}}(\hat{\mathbf{z}}^{c_j}_l, \mathcal{X})$ and $r_{\mathcal{L}}(\hat{\mathbf{z}}^{c_i}_l, \mathcal{X})$ are residuals for the IMU, visual and loop closure models respectively. $\mathcal{B}$ is the set of all pre-integrated IMU measurements. $\mathcal{C}$ is the set of features that have been observed for at least two times in the sliding window. $\mathcal{L}$ is the set of features that are observed both by keyframes in the current sliding window and some frames in the loop closure database. Detailed measurement models are defined in Sect. 7.2, Sect. 7.3, and Sect. 7.4 respectively. $\{\mathbf{r}_p, \mathbf{H}_p\}$ is the prior information from marginalization, which is discussed in Sect. 7.5.

The nonlinear cost function in (9) is linearized and solved iteratively using Gauss-Newton or Levenberg-Marquardt methods. Thanks to the explicit initialization process (Sect. 6), our estimator is always able to converge. We use the error state formulation by handling rotation errors on the tangent space of the rotation manifold.

7.2 IMU Measurement Model

Following the kinematics theory, the residual of a pre-integrated IMU measurement can be defined as

$$r_{\mathcal{B}}(\hat{\mathbf{z}}^{b_k}_{b_{k+1}}, \mathcal{X}) = \begin{bmatrix} \delta\boldsymbol{\alpha}^{b_k}_{b_{k+1}} \\ \delta\boldsymbol{\beta}^{b_k}_{b_{k+1}} \\ \delta\boldsymbol{\gamma}^{b_k}_{b_{k+1}} \\ \delta\mathbf{b}_a{}^{b_k}_{b_{k+1}} \\ \delta\mathbf{b}_g{}^{b_k}_{b_{k+1}} \end{bmatrix} = \begin{bmatrix} \mathbf{q}^{b_k}_w (\mathbf{p}^w_{b_{k+1}} - \mathbf{p}^w_{b_k} + \mathbf{g}^w \Delta t^2/2) - \mathbf{v}^{b_k}_{b_k} \Delta t - \hat{\boldsymbol{\alpha}}^{b_k}_{b_{k+1}} \\ \mathbf{q}^{b_k}_w (\mathbf{q}^w_{b_{k+1}} \mathbf{v}^{b_{k+1}}_{b_{k+1}} + \mathbf{g}^w \Delta t) - \mathbf{v}^{b_k}_{b_k} - \hat{\boldsymbol{\beta}}^{b_k}_{b_{k+1}} \\ 2 \left[\hat{\boldsymbol{\gamma}}^{b_k^{-1}}_{b_{k+1}} \otimes \mathbf{q}^{b_w^{-1}}_{b_k} \otimes \mathbf{q}^{b_w}_{b_{k+1}} \right]_{xyz} \\ \mathbf{b}_a{}^{b_{k+1}}_{b_{k+1}} - \mathbf{b}_a{}^{b_k}_{b_k} \\ \mathbf{b}_g{}^{b_{k+1}}_{b_{k+1}} - \mathbf{b}_g{}^{b_k}_{b_k} \end{bmatrix}, \tag{10}$$

where $\left[\cdot\right]_{xyz}$ extracts the vector part of the quaternion $\mathbf{q}$, and $[\hat{\boldsymbol{\alpha}}^{b_k}_{b_{k+1}}, \hat{\boldsymbol{\beta}}^{b_k}_{b_{k+1}}, \hat{\boldsymbol{\gamma}}^{b_k}_{b_{k+1}}]^{\mathrm{T}}$ is the pre-integrated IMU measurements using only noisy accelerometer and gyroscope measurements. These pre-integrated measurements are correlated with accelerometer and gyroscope bias, but are independent of the initial velocity and attitude. Unlike in the initialization phase where we do not estimate the accelerometer bias, we do estimate both accelerometer and gyroscope biases in the nonlinear optimization. IMU biases are updated in each nonlinear optimization. If these bias values significantly deviate from their previous estimates, we will re-evaluate the IMU pre-integration quantities.

The covariance matrix $\mathbf{P}^{b_k}_{b_{k+1}}$ can then be calculated by first-order discrete-time propagation of the linearized IMU pre-integration model within the time interval $[k, k+1]$. For brevity, we omit the derivation, and refer interested readers to [28] for details about on-manifold uncertainty propagation.

7.3 Visual Measurement Model

The camera measurement model can be formulated as the reprojection error with covariance matrix $\mathbf{P}^{c_j}_l$. The camera measurement residual for the observation of the l^{th} feature in the j^{th} image is defined as

$$\mathbf{f}^{c_j}_l = \begin{bmatrix} fx^{c_j}_l \\ fy^{c_j}_l \\ fz^{c_j}_l \end{bmatrix} = \mathbf{T}^{b^{-1}}_c \cdot \mathbf{T}^{w^{-1}}_{b_j} \cdot \mathbf{T}^w_{b_i} \cdot \mathbf{T}^b_c \cdot \frac{1}{\lambda_l} \cdot \left(\begin{bmatrix} u^{c_i}_l \\ v^{c_i}_l \\ 1 \end{bmatrix} \right)$$
$$r_{\mathcal{C}}(\hat{\mathbf{z}}^{c_j}_l, \mathcal{X}) = \begin{bmatrix} \frac{fx^{c_j}_l}{fz^{c_j}_l} - \hat{u}^{c_j}_l \\ \frac{fy^{c_j}_l}{fz^{c_j}_l} - \hat{v}^{c_j}_l \end{bmatrix} \tag{11}$$

where $\mathbf{f}_l^{c_j}$ is the 3D location of the l^{th} feature in the j^{th} image. This is obtained by back-projecting the feature from its first observation in the i^{th} image using its inverse depth λ_l. $[\hat{u}_l^{c_j}, \hat{v}_l^{c_j}]$ is the noisy observation of the same feature in the j^{th} image using a normalized pixel model. $\mathbf{T} = \left(\begin{smallmatrix}\mathbf{R} & \mathbf{p} \\ \mathbf{0} & 1\end{smallmatrix}\right)$ is the homogeneous representation of frame transformation. $\mathbf{T}_c^b$ transforms a 3D point from the camera frame to the IMU (body) frame. $\mathbf{T}_{b_i}^w$ represents the location of the i^{th} IMU frame in the world frame, which can be derived directly using the position and orientation components in the IMU state $\mathbf{x}_i$.

7.4 Loop Closure Model

When loop closure is detected, we use feature correspondences from loop closure as additional measurements for the tightly-coupled nonlinear optimization. In contrast to [19], which perform global camera pose optimization using all frames and all loop closure constraints in a jointly manner, we treat loop closure as a mechanism for camera re-localization, and do not perform global optimization. This saves significantly amount of computing power. Specifically, every time a keyframe is added into the loop closure database (Sect. 5), we treat its pose as a constant that will not be further optimized. When loops are detected, the whole sliding window is anchored to the constant loop closure frames, in a least square setting, through the integration of feature correspondences. Probabilistically, our loop closure mechanism *conditions* the sliding window on older frames. This way, although loop closures introduces additional visual measurements, it does not change the state vector, thus bounding the computational complexity of the sliding window estimator.

The residual for the observation of the l^{th} feature in the m^{th} looped closure frame is formulated as

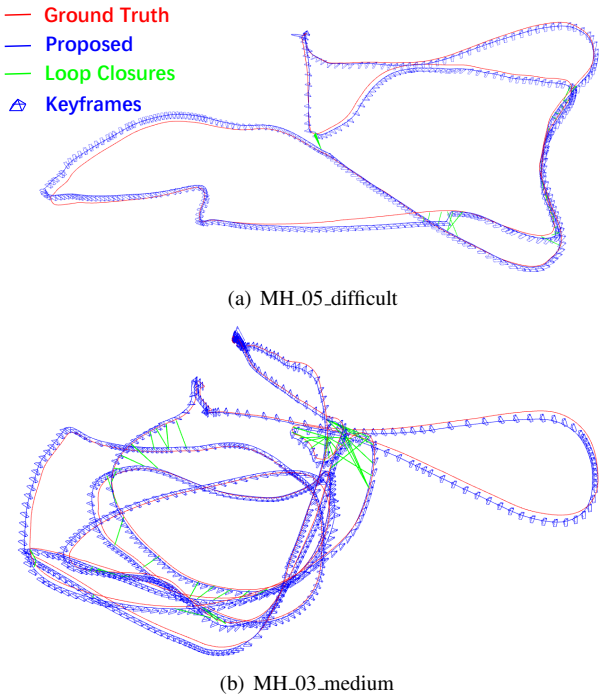
$$\mathbf{f}_l^{c_m} = \begin{bmatrix} fx_l^{c_m} \\ fy_l^{c_m} \\ fz_l^{c_m} \end{bmatrix} = \mathbf{T}_{c_m}^{w^{-1}} \cdot \mathbf{T}_{b_i}^w \cdot \mathbf{T}_c^b \cdot \frac{1}{\lambda_l} \cdot \left(\begin{bmatrix} u_l^{c_i} \\ v_l^{c_i} \\ 1 \end{bmatrix} \right)$$
$$r_{\mathcal{L}}(\hat{\mathbf{z}}_l^{c_m}, \mathcal{X}) = \begin{bmatrix} \frac{fx_l^{c_m}}{fz_l^{c_m}} - \hat{u}_l^{c_m} \\ \frac{fy_l^{c_m}}{fz_l^{c_m}} - \hat{v}_l^{c_m} \end{bmatrix} \tag{12}$$

The formulation is similar to visual measurement model in Sect. 7.3. The set $\mathcal{L}$ are retrieved features correspondences for the m^{th} loop closure frame the database. $\mathbf{T}_{c_m}^{w^{-1}}$ fixed at the time that a keyframe is added into the database, and it is treated as a known constant in the loop closure model.

Fig. 4 shows the loop fuse process. After getting sufficient feature correspondences between the newest keyframe $\mathbf{x}_n$ and the loop closure candidate $\mathbf{y}_i$, we optimize by adding all visual measurements in $\mathbf{y}_i$. Note that $\mathbf{y}_i$ slides together with $\mathbf{x}_n$. When $\mathbf{x}_n$ is removed from the window using marginalization (Sect 7.5), it will be added into the loop closure database. This ensures that we always have the best estimate of $\mathbf{x}_n$ before it is added into the database. Meanwhile, measurements corresponding to $\mathbf{y}_i$ will be simply removed, as $\mathbf{y}_i$ is already in the database. A frame in the database may be detected again when the sliding window moves back to a similar place in a future time.

7.5 Marginalization

In order to bound the computational complexity of the nonlinear optimization, marginalization is usually used. We use a two-way-marginalization scheme [23, 28] to selectively marginalize out IMU states $\mathbf{x}_k$ and features λ_l from the sliding window, and convert the measurements corresponding to the marginalized states into a prior.



(a) MH_05_difficult

(b) MH_03_medium

Figure 7: The estimated trajectory of proposed method and its comparison against the ground truth in MH_05_difficult 7(a) and MH_03_medium 7(b) datasets. Green line denotes loop closures, after which the accumulated drift is eliminated.

8 Experimental Results

We evaluate the performance of the proposed approach on both public visual-inertial datasets and on mobile devices. The experimental results show our estimator is drift-free in long term operations due to the tightly-coupled loop fusion. Thanks to the fusion of the always-available IMU measurements, our approach shows robust performance against aggressive movements, motion blur and illumination changes.

8.1 Quantitative Analysis Using Public Datasets

We test our approach on the public visual-inertial datasets captured from an aerial robot [1]. This dataset contains 752 $\times$ 480 stereo images at 20 Hz captured by an Aptina MT9V034 global shutter camera, synchronized IMU data at 200 Hz from the ADIS16448 IMU, and ground truth states extracted by Leica MS50 and motion capture system. We only use one camera from the stereo image set. The experiments are performed on a laptop computer with Intel Core i7-6500U 2.50GHz CPU and an 8GB RAM. Our proposed method runs successfully on all datasets sequences. To this end, instead of simply putting down results from all sequences, we use two sequences, MH_05_difficult and MH_03_medium, for performance evaluation. These two long term sequences contain significant rotational motions and illumination changes. These results show that our monocular VINS approach is able to eliminate drift after loop closures are detected.

The qualitative illustration of the estimated trajectories of our approach comparing against the ground truth is shown in Fig. 7. It can be seen that the errors between our approach and ground truth are small and the drift is further eliminated after loop closures.

Quantitative comparisons between our approach, the ground truth, and OKVIS [12], a state-of-the-art visual-inertial odometry, are shown in Fig. 8 and Fig. 9 respectively. Since roll and pitch angles are observable in the VINS formulation, we only compare

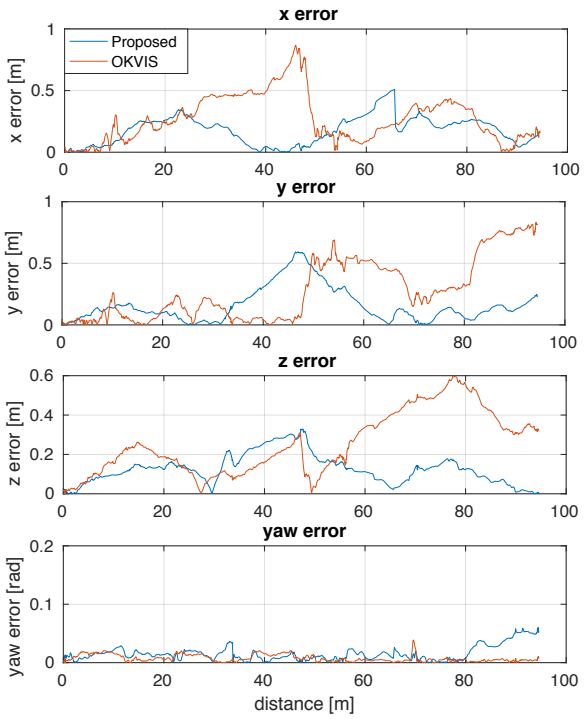


Figure 8: Performance comparison for the MH_05_difficult dataset. Since roll and pitch are observable in the VINS setting, we only compare errors in x, y, z, and yaw. It can be seen that our approach demonstrates higher accuracy comparing to OKVIS. The first loop was detected after traveling for approximately 66 meters. Accumulated drift is eliminated after loop closure. Towards the end of the sequence, we see slight increase in rotation error for our method, but the final error is still small and is sufficient for camera localization.

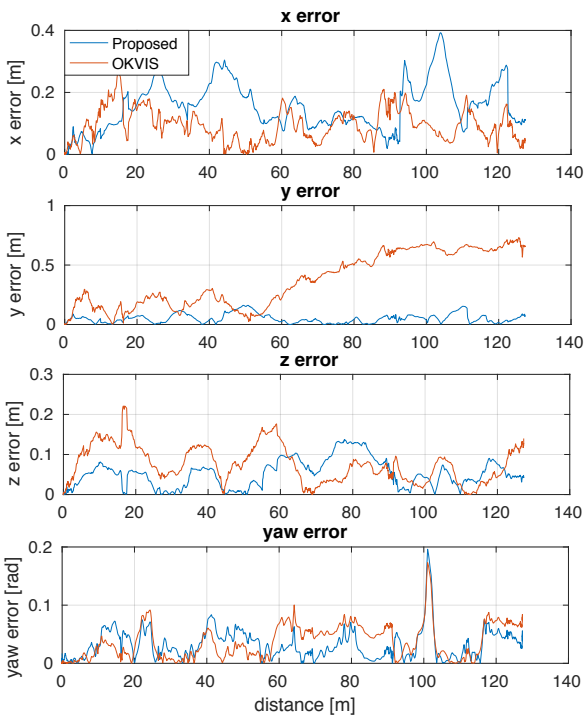


Figure 9: Performance comparison for the MH_03_medium dataset. Plots can be interpreted in the same way as Fig. 8.

errors in x, y, z, and yaw. In both figures, blue lines and red lines represent errors of our approach and OKVIS respectively. For the MH_05_difficult dataset, the average translation and yaw errors for our approach are 0.28 meters and 0.016 rads respectively. In comparison, the error statistics are 0.49 meters and 0.007 rads for OKVIS. Both methods show negligible rotation errors, and our method shows fewer translation errors obviously. Our method shows more accurate results in the MH_03_medium dataset due to loop closure, with errors of 0.15 meters in translation, and 0.029 rads in yaw. The OKVIS statistics are 0.39 meters in translation, and 0.037 rads in yaw. In both dataset, we demonstration drift-free property, while OKVIS suffers from mainly translational drifting. The local accuracy is roughly the same for both methods. However, we stress that our method is capable of running on standard mobile phones, which will be shown in the next experiment.

8.2 Performance on Mobile Devices

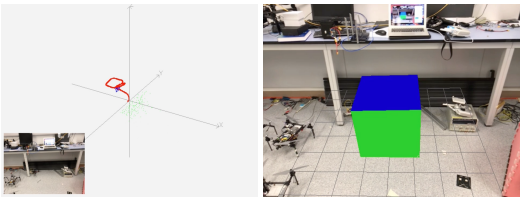
In this experiment, we port our monocular VINS estimator to a mobile device and present an simple AR application for demonstration purpose. We implement all the system as an iOS app that is able to run in real time on iPhone devices. We use 30 Hz images with 640×480 resolution captured by the mobile phone, and IMU data at 100 Hz obtained by the built-in InvenSense MP67B 6-axis gyroscope and accelerometer. The total computation is divided into three parallel threads. The first thread performs corner detection at 10 Hz and feature tracking at 30 Hz. We detect a maximum of 60 new features with uniform distribution for new incoming images. The second thread performs estimator initialization and nonlinear optimization at 10 Hz. We keep 10 frames in the sliding window. The third thread performs loop detection and feature retrieval at 3 Hz. Loop closure candidate frames are fused in the nonlinear optimization if more than 10 feature correspondences are found.

Table 1 details the timing statistics on the iPhone 7 Plus for all modules. We estimate the metric-scale feature position, velocity, gravity vector, and gyroscope bias in the initialization module. After initialization is successful, camera pose, velocity, feature position, and IMU biases are continuously estimated using nonlinear optimization at 10 Hz. Beside pose, we also have accurate velocity and IMU bias estimation, which enable to propagate the camera pose with IMU data up to 100 Hz. This ensures low-latency AR experience.

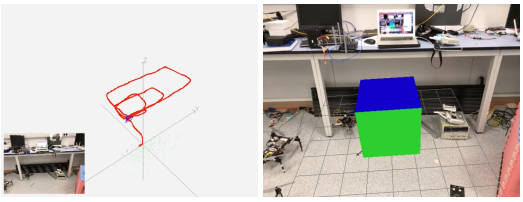
In this experiment, we demonstrate the robustness of our approach against aggressive motions, and show its ability on real-time drift elimination. Firstly, we insert a virtual cube with a size of 0.5 m on each side on the plane extracted from estimated visual features. We then hold the mobile phone and walk around the room in a normal pace. As shown in Fig. 10, we record screeshots at each time we return to the starting location. Due to our tightly-coupled

Table 1: Timing Statistics

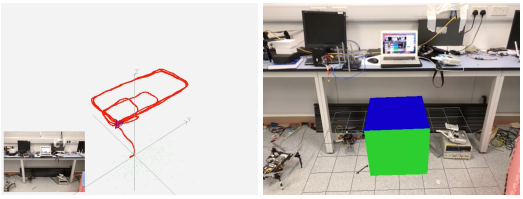
Thread	Module	Time (ms)	Rates (Hz)
1	Feature detection	17	10
	Feature tracking	3	30
	IMU data	1	100
2	Initialization	80	once
	Non-Linear optimization	60	10
	Propagation with IMU	1	100
3	Loop detection	60	3
	Feature retrieval	10	3



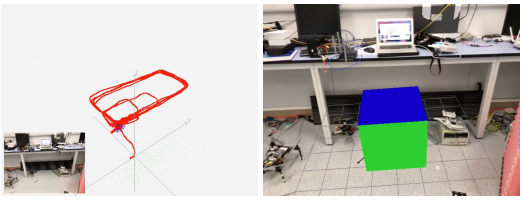
(a)



(b)



(c)



(d)

Figure 10: An AR example showing long term drift elimination. Left figures show the estimated, right figures shows the registered virtual cube in the image. The total trajectory length is about 100 meters. It can be seen that the virtual cube remains in the same location even after long travel distances.

loop fusion, accumulated drift can be eliminated effectively. This is evidenced by the fact that the cube is registered to the same place on the image. The total length of the trajectory is about 100 meters.

In another trial shown in Fig. 11, we shake and move the phone aggressively. In this case, the visual connection between consecutive frames will decrease significantly because of the insufficient frame overlapping and motion blur. However, our system still works reliably with the help of the tightly-coupled IMU factors and loop closure. This is again evidenced by the properly registered cube on the image.

Finally, we present an AR demo with rendered animation for qualitative evaluation as shown in Fig. 12. It can be seen that our approach works both indoor and outdoor, and is capable of handling large-depth features observed in outdoor scenes. The multi-view screenshots show accurate camera pose registration with respect to the global frame.

9 Conclusion and Future Work

In this paper, we propose a tightly-coupled monocular visual-inertial state estimator for mobile AR applications. Our estimator is equipped with robust online metric initialization and loop closure. The nonlinear optimization-based estimator is able to provide accurate metric state estimates both indoor and outdoor without requiring any prior knowledge about the environment. We experimentally show that the proposed method achieves better results comparing to the state-of-the-art approach. The proposed method is implemented on a mobile device. Simple AR demonstrations are presented to show robustness against both long term drifting and aggressive motions. We make our software implementation open source.

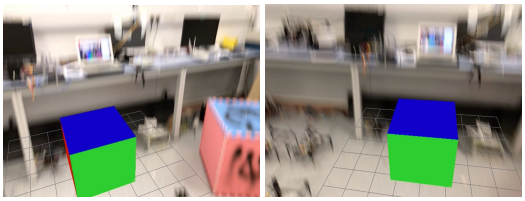


Figure 11: Estimator robustness evaluation under aggressive motions and motion blur. This is again evidenced by the proper registration of the virtual box against the captured images.

In the future, we will improve the estimator accuracy by taking rolling shutter and camera intrinsic parameters into consideration. We also aim to move towards dense 3D reconstruction on mobile devices in order to assist 3D environment understanding, and achieve AR rendering that is aware of occlusions from physical objects.

References

[1] 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.

[2] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.

[3] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. In *Proc. of Robot.: Sci. and Syst.*, Rome, Italy, July 2015.

[4] D. Gálvez-López and J. D. Tardós. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, 2012.

[5] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[6] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis. Consistency analysis and improvement of vision-aided inertial navigation. *IEEE Trans. Robot.*, 30(1):158–176, Feb. 2014.

[7] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy. Visual odometry and mapping for autonomous flight using an RGB-D camera. In *Proc. of the Int. Sym. of Robot. Research*, Flagstaff, AZ, Aug. 2011.

[8] V. Indelman, S. Williams, M. Kaess, and F. Dellaert. Information fusion in navigation systems via factor graph based incremental smoothing. *Robot. and Auton. Syst.*, 61(8):721–738, 2013.

[9] E. S. Jones and S. Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *Int. J. Robot. Research*, 30(4):407–430, Apr. 2011.

[10] J. Kelly and G. S. Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *Int. J. Robot. Research*, 30(1):56–79, Jan. 2011.

Figure 12: Screenshots showing our indoor and outdoor mobile AR demonstrations in which the AR scene is shown from different perspective.

[11] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[12] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *Int. J. Robot. Research*, 34(3):314–334, 2015.

[13] M. Li and A. Mourikis. High-precision, consistent EKF-based visual-inertial odometry. *Int. J. Robot. Research*, 32(6):690–711, May 2013.

[14] H. Liu, G. Zhang, and H. Bao. Robust keyframe-based monocular slam for augmented reality. In *Mixed and Augmented Reality (ISMAR), 2016 IEEE International Symposium on*, pages 1–10. IEEE, 2016.

[15] 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*, pages 24–28, Vancouver, Canada, Aug. 1981.

[16] T. Lupton and S. Sukkarieh. Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Trans. Robot.*, 28(1):61–76, Feb. 2012.

[17] A. Martinelli. Closed-form solution of visual-inertial structure from motion. *Int. J. Comput. Vis.*, 106(2):138–152, 2014.

[18] 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.*, pages 3565–3572, Roma, Italy, Apr. 2007.

[19] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular slam system. *IEEE Trans. Robot.*, 31(5):1147–1163, 2015.

[20] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004.

[21] T. Qin and S. Shen. Robust initialization of monocular visual-inertial estimation on aeral robots (under review). In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Vancouver, Canada, Sept. 2017. URL: `http://www.ece.ust.hk/~eeshaojie/iros2017tong.pdf`.

[22] T. Schöps, J. Engel, and D. Cremers. Semi-dense visual odometry for ar on a smartphone. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 145–150. IEEE, 2014.

[23] 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.

[24] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar. Initialization-free monocular visual-inertial estimation with application to autonomous MAVs. In *Proc. of the Int. Sym. on Exp. Robot.*, Marrakech, Morocco, June 2014.

[25] J. Shi and C. Tomasi. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 593–600. IEEE, 1994.

[26] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustmenta modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999.

[27] 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.*, pages 957–964, Saint Paul, MN, May 2012.

[28] Z. Yang and S. Shen. Monocular visual-inertial state estimation with online initialization and camera-IMU extrinsic calibration. *IEEE Trans. Autom. Sci. and Engineering*, 14(1):39–51, 2017.