Analytic Combined IMU Integration (ACI^2) For Visual Inertial Navigation

Yulin Yang[+], Benzun Pious Wisely Babu[*], Chuchu Chen[+], Guoquan Huang[+] and Liu Ren[*]

Abstract— **Batch optimization based inertial measurement unit (IMU) and visual sensor fusion enables high rate localization for many robotic tasks. However, it remains a challenge to ensure that the batch optimization is computationally efficient while being consistent for high rate IMU measurements without marginalization. In this paper, we derive inspiration from maximum likelihood estimation with partial-fixed estimates to provide a unified approach for handing both IMU pre-integration and time-offset calibration. We present a modularized analytic combined IMU integrator (ACI^2) with elegant derivations for IMU integrations, bias Jabcobians and related covariances. To simplify our derivation, we also prove that the right Jacobians for Hamilton quaterions and SO(3) are equivalent. Finally, we present a time offset calibrator that operates by fixing the linearization point for a given time offset. This reduces re-integration of the IMU measurements and thus improve efficiency. The proposed ACI^2 and time-offset calibration is verified by intensive Monte-Carlo simulations generated from real world datasets. A proof-of-concept real world experiment is also conducted to verify the proposed ACI^2 estimator.**

I. Introduction and Related Work

Precise 6DOF localization is essential for complex manipulation and navigation tasks in robotics. In recent years, robot localization is performed using the high frequency instantaneous rate measurements provided by an Inertial Measurement Unit (IMU). However, due to the incremental nature of the measurements provided by the IMU, the localization estimates accumulate error over time. Hence, measurements from other exteroceptive sensors, such as optical cameras [1–6], RGBD sensors [7, 8], LIDAR [9–11], underwater sonar [12], event cameras [13] etc are combined with IMU measurements to achieve better accuracy and consistency.

Among the different sensors that can be combined with the IMU, camera has gained prominence due to its easy availability and low cost. Hence, algorithms for visual-inertial navigation system (VINS) such as [2, 14, 15] have gained prominence in recent years.

The fusion of IMU and vision sensor is performed using either non-linear filtering [1, 16–18] or non-linear batch optimization [2, 5, 6, 15] approaches. Filter based methods enable faster algorithms but suffer from reduced accuracy due to accumulation of linearization errors, while optimization based methods provide better accuracy at the cost of higher computational requirements.

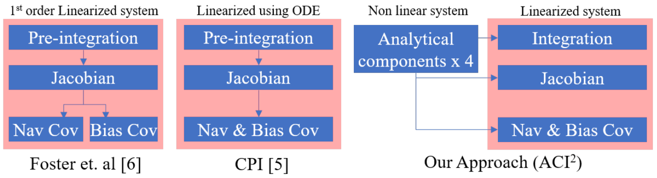

Fig. 1: In contrast to previous approaches, ACI^2 shares analytical components and correctly accounts for correlation between navigation & bias states.

In spite of the recent advances in VINS, there still remain challenges to robustly incorporate high rate IMU measurement into the optimization framework for localization [19, 20]. This is further aggravated in resource constrained systems without dedicated time synchronization capabilities.

In this regard, Leutenegger et al. [2] proposed a keyframe based visual-inertial odometry system (OKVIS). In OKVIS, the IMU measurements were re-integrated between consecutive frames in the local window. This re-integration increased the computational requirements during batch optimization. To tackle the problem of repeated re-integration, Lupton et al. [21] introduced pre-integration using Euler angle representation. Later, Forster et al. [6] applied pre-integration on SE(3) manifold and leveraged the null-space operation based [22] structure-less factor graph [23] to design efficient visual-inertial odometry system. Similarly, Qin et al. [3] derived pre-integration with Hamilton quaternion and build an efficient visual-inertial simultaneous localization and mapping (VI-SLAM) system. Unlike [6], they stored key visual features as anchored depth in the state vector. This enabled their system to handle more features and perform loop closure. However, both [6] and [3] performed pre-integrations based on the first-order approximation of the discrete IMU dynamic model [2, 24], which suffers from errors in integration accuracy. Recently, Eckenhoff et al. [5] proposed a continuous-time IMU pre-integration (CPI) and built two estimator models based on piece-wise constant acceleration measurement assumption and constant true acceleration assumption, respectively. However, their approach requires solving an Ordinary Differential Equation (ODE) based on the IMU dynamic model. This introduced additional complexity in the expressions for IMU integration, bias Jacobians and measurement covariances.

Major part of the work was performed during Yang's internship at Bosch Research, USA. This work was partially supported by the University of Delaware (UD) College of Engineering, the NSF (IIS-1924897), the ARL (W911NF-19-2-0226, JWS 10-051-003) and Bosch Research. Yang was partially supported by the University Doctoral Fellowship.

[+] Y. Yang, C. Chen and G. Huang are with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716, USA. Email: {yuyang,ccchu,ghuang}@udel.edu

[*] Benzun Pious Wisely Babu, and Liu Ren are with Bosch Research and Technology Center, Sunnyvale, USA {benzun.wisely, liu.ren}@us.bosch.com

Also in resource constrained applications, time synchronization is included into the estimation framework. This introduces additional parameters in VINS [10, 25–27]. The time offset parameter introduced, triggers re-integration of the IMU measurements each time a new offset estimate is generated in the batch optimization based VINS. Qin et al. [26] incorporated the time synchronization into their estimator [3] by using a modified visual observation model. The visual observation model introduced a constant-pixel-velocity corresponding to time offset. However, the measurement covariances for the constant-pixel-velocity model is difficult to quantify. In addition, their approach is sensitive to camera frame rates and is not extensible to other sensors like LIDAR or sonar. On the other hand, Gentil et al. [10] incorporated the time synchronization into the IMU pre-integration. The drawback to this approach is that it is IMU dependent and does not scale well to systems with multiple sensors in addition to the IMU.

In this paper, we present an elegant approach to reduce computational complexity while maintaining consistency in batch optimization based VINS. Our approach derives inspiration from MLE with partial-fixed estimates to provide a unified approach for handing both IMU pre-integration and time-offset calibration. Our IMU integrator uses an analytical and modularized method for the integration, computation of state Jacobians and propagation of related measurement covariances. In contrast to the recent pre-integration approach proposed by [5], our approach computes 4 integration components for each IMU time step (Fig. 1). These components are shared to reduce the computation time for Jacobians and measurement covariances calculation. Compared to [6], when we compute the IMU integrated covariances, we combine the IMU navigation states and bias states. Thus, the correlation between these is preserved.

Our online time-offset calibrator, treats the time offset as a random walk similar to [28]. However, to avoid the IMU re-integration, we fix the initial estimate of the time offset in the measurement model and add cost correction terms to the objective function. In contrast to previous approaches, no constant-pixel-velocity assumption is needed and the proposed approach can be easily extended to multiple sensors.

In summary, the contributions of this paper can be listed below:

- A modularized analytic combined IMU integrator (ACI^2), with analytic pre-integrated IMU measurements, bias Jacobians and covariances. We also investigate the equivalence of right Jacobians for Hamilton quaternion and SO(3) to simplify the derivations.
- An online time-offset calibration method for batch optimization based VINS with reduced IMU re-integration.
- Intensive Monte-Carlo simulations to compare different IMU integrators and validate the proposed time offset calibration.

II. Problem Formulation

Occasionally, certain parameters' linearization points in maximum likelihood estimation (MLE) need to be fixed to achieve higher efficiency (e.g., biases in pre-integration [6], iSAM [29]) or better consistency [30]. Following this direction, we introduce the MLE with partial-fixed-estimate estimator. We will provide a unified theoretical foundation for the proposed ACI^2 and time offset calibration.

A. MLE with Partial-Fixed Estimates

Given an observation model containing state $\mathbf{x}_a$ and $\mathbf{x}_b$ as:

$$\mathbf{z} = \mathbf{h}(\mathbf{x}_a, \mathbf{x}_b) + \mathbf{n} \tag{1}$$

with $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$. Note that $\mathbf{x}_b$ is fixed with certain linearization point during linearization. A weighted least-squares cost function can be formulated as:

$$\mathbb{C}(\mathbf{x}_a, \mathbf{x}_b) = \left\| \mathbf{z} - \mathbf{h}(\mathbf{x}_a, \mathbf{x}_b) \right\|^2_{\mathbf{R}^{-1}} \tag{2}$$

$$\mathbf{x}_a^*, \mathbf{x}_b^* = \arg\min_{\mathbf{x}_a, \mathbf{x}_b} \mathbb{C}(\mathbf{x}_a, \mathbf{x}_b) \tag{3}$$

If we define $\hat{\mathbf{x}}$ as current estimate and $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$ as error states, then we have:

$$\mathbf{x}_b = \hat{\mathbf{x}}_b + \tilde{\mathbf{x}}_b = \hat{\mathbf{x}}_b^{(0)} + \tilde{\mathbf{x}}_b^{(0)} \quad \Rightarrow \quad \tilde{\mathbf{x}}_b^{(0)} = \Delta\mathbf{x}_b + \tilde{\mathbf{x}}_b \tag{4}$$

where we have defined $\Delta\mathbf{x}_b = \hat{\mathbf{x}}_b - \hat{\mathbf{x}}_b^{(0)}$ and $\hat{\mathbf{x}}_b^{(0)}$ is the fixed linearization point. If using iterative algorithms (e.g., Gaussian-Newton method) to solve (3), we can linearize the observation (1) with current best estimate of $\hat{\mathbf{x}}_a$ and the fixed estimate of $\hat{\mathbf{x}}_b^{(0)}$. If we denote $\mathbf{H}_a \triangleq \mathbf{H}_a|_{\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}}$ and $\mathbf{H}_b \triangleq \mathbf{H}_b|_{\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}}$ for simplicity, we can arrive at:

$$\mathbf{z} \simeq \mathbf{h}(\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}) + \mathbf{H}_a\tilde{\mathbf{x}}_a + \mathbf{H}_b\tilde{\mathbf{x}}_b^{(0)} + \mathbf{n} \tag{5}$$

$$= \mathbf{h}(\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}) + \mathbf{H}_b\Delta\mathbf{x}_b + \mathbf{H}_a\tilde{\mathbf{x}}_a + \mathbf{H}_b\tilde{\mathbf{x}}_b + \mathbf{n} \tag{6}$$

The cost function (2) can be approximated by:

$$\mathbb{C} \simeq \left\| \mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}) - \mathbf{H}_b\Delta\mathbf{x}_b - \mathbf{H}_a\tilde{\mathbf{x}}_a - \mathbf{H}_b\tilde{\mathbf{x}}_b \right\|^2_{\mathbf{R}^{-1}} \tag{7}$$

By iteratively solving the linear system of (7), we can get solutions to (3). But compared with standard MLE algorithms, we have the following remarks:

1) Case 1: Since states $\mathbf{x}_b$ can only be linearized with fixed estimates ($\hat{\mathbf{x}}_b^{(0)}$), a correction term $\mathbf{H}_b\Delta\mathbf{x}_b$ appears in the linearized cost function (7).

2) Case 2: If the Jacobians $\mathbf{H}_b|_{\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}}$ for $\mathbf{x}_b$ only relates to $\hat{\mathbf{x}}_b^{(0)}$, then we have $\mathbf{H}_b|_{\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}} = \mathbf{H}_b|_{\hat{\mathbf{x}}_b^{(0)}}$. Hence, when solving the (3) iteratively, $\mathbf{H}_b$ only needs to be computed once and can be used repeatedly (since $\hat{\mathbf{x}}_b^{(0)}$ is fixed).

3) Case 3: If the noise $\mathbf{n}$ is not additive, instead, it is included in the nonlinear observation model:

$$\mathbf{z} = \mathbf{h}(\mathbf{x}_a, \mathbf{x}_b, \mathbf{n}) \tag{8}$$

Accordingly, the linearzied model and cost function can be rewritten as:

$$\mathbf{z} \simeq \mathbf{h}(\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}, \mathbf{0}) + \mathbf{H}_b\Delta\mathbf{x}_b + \mathbf{H}_a\tilde{\mathbf{x}}_a + \mathbf{H}_b\tilde{\mathbf{x}}_b + \mathbf{H}_n\mathbf{n} \tag{9}$$

$$\mathbb{C} \simeq \left\| \mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}, \mathbf{0}) - \mathbf{H}_b\Delta\mathbf{x}_b - \mathbf{H}_a\tilde{\mathbf{x}}_a - \mathbf{H}_b\tilde{\mathbf{x}}_b \right\|^2_{\mathbf{R}_0^{-1}} \tag{10}$$

where $\mathbf{H}_n = \mathbf{H}_n|_{\hat{\mathbf{x}}_a, \hat{\mathbf{x}}_b^{(0)}}$ and $\mathbf{R}_0 = \mathbf{H}_n\mathbf{R}\mathbf{H}_n^\top$. Similarly, if the $\mathbf{H}_n$ only relates to $\hat{\mathbf{x}}_b^{(0)}$, $\mathbf{H}_n$ and $\mathbf{R}_0$ are also fixed, which both will be used repeatedly for solving (3). In contrast to iSAM, which relinearize the states after every few steps, our approach relinearizes $\mathbf{x}_a$ every step but keeps $\hat{\mathbf{x}}_b^{(0)}$ fixed throughout. Compared to first-estimates Jacobians (FEJ [31]) which uses initial estimates to improve filter consistency, the partial-fixed-estimate estimator iteratively solves MLE with fixed linearization constraints for certain parameters.

Our formulation enables a unified approach for representing both IMU optimization costs and time offset optimization cost. While formulating the IMU optimization cost, the bias linearization is fixed and Case II-A.2 and II-A.3 is applied. Similarly, while formulating the offset optimization cost, the time offset is fixed and Case II-A.1 is applied.

III. Analytic Combined IMU Integration (ACI^2)

In this section, we will detail the IMU model and the proposed modularized IMU integration.

A. IMU Model

The IMU state can be described as:

$$\mathbf{x}_I := \begin{bmatrix} \mathbf{x}_n^\top & \mathbf{x}_b^\top \end{bmatrix}^\top \tag{11}$$

where $\mathbf{x}_n := \begin{bmatrix} {}^G_I\bar{q}^\top & {}^G\mathbf{p}_I^\top & {}^G\mathbf{v}_I^\top \end{bmatrix}^\top$ and $\mathbf{x}_b := \begin{bmatrix} \mathbf{b}_g^\top & \mathbf{b}_a^\top \end{bmatrix}^\top$ represent the IMU navigation state and the bias state, respectively. Hamilton quaternion [32] ${}^G_I\bar{q}$ represents the rotation ${}^G_I\mathbf{R}$ from frame $\{I\}$ to frame $\{G\}$, with error states $\delta\boldsymbol{\theta}$ defined as $\delta\bar{q} = \begin{bmatrix} 1 & \delta\boldsymbol{\theta}^\top \end{bmatrix}^\top = \hat{\bar{q}}^{-1} \otimes \bar{q}$. ${}^G\mathbf{p}_I$ and ${}^G\mathbf{v}_I$ represent the IMU position and velocity in the global frame, respectively. $\mathbf{b}_g$ and $\mathbf{b}_a$ are gyroscope and accelerometer biases. The IMU dynamic model can thus be defined as [24, 33]:

$$\begin{aligned} &{}^G_I\dot{\bar{q}} = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega})^G_I\bar{q}, \quad {}^G\dot{\mathbf{p}}_I = {}^G\mathbf{v}_I \\ &{}^G\dot{\mathbf{v}}_I = {}^G_I\mathbf{R}^I\mathbf{a} + {}^G\mathbf{g}, \quad \dot{\mathbf{b}}_g = \mathbf{n}_{wg}, \quad \dot{\mathbf{b}}_a = \mathbf{n}_{wa} \end{aligned} \tag{12}$$

where $\boldsymbol{\omega}$ and $\mathbf{a}$ denote the local angular velocity and linear acceleration, $\mathbf{n}_{wg}$ and $\mathbf{n}_{wa}$ are the white Gaussian noises driving the gyroscope and accelerometer biases. $\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\boldsymbol{\omega}^\top \\ \boldsymbol{\omega} & -\lfloor\boldsymbol{\omega}\rfloor \end{bmatrix}$ and $\lfloor\cdot\rfloor$ represents a skew symmetric matrix. The IMU readings can be described as:

$$\boldsymbol{\omega}_m = \boldsymbol{\omega} + \mathbf{b}_g + \mathbf{n}_g, \quad \mathbf{a}_m = \mathbf{a} + \mathbf{b}_a + \mathbf{n}_a - {}^I_G\mathbf{R}^G\mathbf{g} \tag{13}$$

where ${}^G\mathbf{g} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}^\top$. $\mathbf{n}_g$ and $\mathbf{n}_a$ are continuous-time Gaussian noises that contaminate the IMU readings. For simplicity, we also define the IMU noise vector $\mathbf{n}_I(t) = \begin{bmatrix} \mathbf{n}_g^\top & \mathbf{n}_a^\top & \mathbf{n}_{wg}^\top & \mathbf{n}_{wa}^\top \end{bmatrix}^\top$, which means that the system noises are modeled as zero-mean white Gaussian process with autocorrelation $\mathbb{E}\left[\mathbf{n}_I(t)\mathbf{n}_I^\top(\tau)\right] = \mathbf{Q}_c\delta(t-\tau)$.

B. Interval-based IMU Measurement

In senor fusion, IMU operates at a much higher rate than complementary sensors such as cameras. In order to avoid adding too many parameters into the optimization, we would like to collect and pre-process the IMU measurements between camera frame interval and add only the starting and ending IMU states (i.e, IMU states at time step k and j, see Fig. 2) into the state vector.

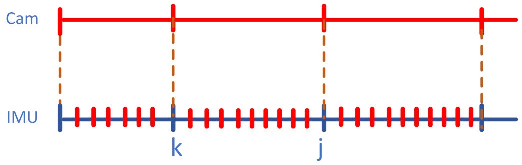

Fig. 2: Aligned camera and IMU time line.

To simplify notation, we define the index of all the IMU measurements between time step k and j as from 0 to $i+1$, $j \triangleq k+i+1$ and $t_{k+i} \triangleq t_i$. Hence, starting from frame k, multiple $(i+1)$ steps integration can be described as:

$$\Delta\bar{q}_{i+1} = \Delta\bar{q}_i \otimes \bar{q}(\int_{t_i}^{t_{i+1}} \boldsymbol{\omega}_i d\tau) \tag{14}$$

$$\Delta\mathbf{p}_{i+1} = \Delta\mathbf{p}_i + \Delta\mathbf{v}_i\delta t_i + \Delta\mathbf{R}_i \int_{t_i}^{t_{i+1}} \int_{t_i}^{t_s} {}^{I_{k+i}}_{I_\tau}\mathbf{R} \cdot \mathbf{a}_i d\tau ds \tag{15}$$

$$\Delta\mathbf{v}_{i+1} = \Delta\mathbf{v}_i + \Delta\mathbf{R}_i \int_{t_i}^{t_{i+1}} {}^{I_{k+i}}_{I_\tau}\mathbf{R} \cdot \mathbf{a}_i d\tau \tag{16}$$

$$\Delta\mathbf{b}_{g_{i+1}} = \Delta\mathbf{b}_{g_i} + \int_{t_i}^{t_{i+1}} \mathbf{n}_{wg} d\tau, \ \Delta\mathbf{b}_{a_{i+1}} = \Delta\mathbf{b}_{a_i} + \int_{t_i}^{t_{i+1}} \mathbf{n}_{wa} d\tau \tag{17}$$

where $\Delta\bar{q}_i$, $\Delta\mathbf{v}_i$, $\Delta\mathbf{p}_i$, $\Delta\mathbf{b}_{gi}$ and $\Delta\mathbf{b}_{ai}$ are the integrated IMU state measurements from time step k to $k+i$. $\Delta\mathbf{R}_i \triangleq {}^{I_k}_{I_{k+i}}\mathbf{R}$ is the corresponding rotation matrix for $\Delta\bar{q}_i$. $\delta t_i = t_{k+i+1} - t_{k+i}$. $\boldsymbol{\omega}_i$ and $\mathbf{a}_i$ represents the angular velocity and linear acceleration between $k+i$ and $k+i+1$. For simplicity, based on the assumption that the IMU reading are constant in each small time interval (e.g., between $k+i$ and $k+i+1$), we introduce 4 integration components:

$$\boldsymbol{\Xi}_1 = \int_{t_i}^{t_{i+1}} {}^{I_{k+i}}_{I_\tau}\hat{\mathbf{R}} d\tau, \ \boldsymbol{\Xi}_2 = \int_{t_i}^{t_{i+1}} \int_{t_i}^{t_s} {}^{I_{k+i}}_{I_\tau}\hat{\mathbf{R}} d\tau ds \tag{18}$$

$$\boldsymbol{\Xi}_3 = \int_{t_i}^{t_{i+1}} {}^{I_{k+i}}_{I_\tau}\hat{\mathbf{R}} \lfloor\hat{\mathbf{a}}_i\rfloor \mathbf{J}_r(\boldsymbol{\omega}_i\delta\tau)\delta\tau d\tau \tag{19}$$

$$\boldsymbol{\Xi}_4 = \int_{t_i}^{t_{i+1}} \int_{t_i}^{t_s} {}^{I_{k+i}}_{I_\tau}\hat{\mathbf{R}} \lfloor\hat{\mathbf{a}}_i\rfloor \mathbf{J}_r(\boldsymbol{\omega}_i\delta\tau)\delta\tau d\tau ds \tag{20}$$

where $\mathbf{J}_r(\cdot)$ denotes the right Jacobians (see Appendix I). These 4 components can be used repeatedly for the integrated measurements, Jacobians and covariances. Hence, the integrated measurements can be computed as:

$$\Delta\bar{q}_{i+1} \simeq \Delta\bar{q}_i \otimes \bar{q}(\hat{\boldsymbol{\omega}}_i\delta t_i) \tag{21}$$

$$\Delta\hat{\mathbf{p}}_{i+1} \simeq \Delta\hat{\mathbf{p}}_i + \Delta\hat{\mathbf{v}}_i\delta t_i + \Delta\hat{\mathbf{R}}_i\boldsymbol{\Xi}_2\hat{\mathbf{a}}_i \tag{22}$$

$$\Delta\hat{\mathbf{v}}_{i+1} \simeq \Delta\hat{\mathbf{v}}_i + \Delta\hat{\mathbf{R}}_i\boldsymbol{\Xi}_1\hat{\mathbf{a}}_i \tag{23}$$

And their related bias Jacobians can be computed by using these 4 components repeatedly as:

$$\frac{\partial\Delta\boldsymbol{\theta}_{i+1}}{\partial\tilde{\mathbf{b}}_{gk}} \simeq {}^{I_{k+i}}_{I_{k+i+1}}\hat{\mathbf{R}}^\top \frac{\partial\Delta\boldsymbol{\theta}_i}{\partial\tilde{\mathbf{b}}_{gk}} - \mathbf{J}_r\left(\hat{\boldsymbol{\omega}}_i\delta t_i\right)\delta t_i \tag{24}$$

$$\frac{\partial\Delta\tilde{\mathbf{p}}_{i+1}}{\partial\tilde{\mathbf{b}}_{gk}} \simeq \frac{\partial\Delta\tilde{\mathbf{p}}_i}{\partial\tilde{\mathbf{b}}_{gk}} + \frac{\partial\Delta\tilde{\mathbf{v}}_i}{\partial\tilde{\mathbf{b}}_{gk}}\delta t_i - \Delta\hat{\mathbf{R}}_i(\lfloor\boldsymbol{\Xi}_2\hat{\mathbf{a}}_i\rfloor\frac{\partial\Delta\boldsymbol{\theta}_i}{\partial\tilde{\mathbf{b}}_{gk}} - \boldsymbol{\Xi}_4) \tag{25}$$

$$\frac{\partial\Delta\tilde{\mathbf{p}}_{i+1}}{\partial\tilde{\mathbf{b}}_{ak}} \simeq \frac{\partial\Delta\tilde{\mathbf{p}}_i}{\partial\tilde{\mathbf{b}}_{ak}} + \frac{\partial\Delta\tilde{\mathbf{v}}_i}{\partial\tilde{\mathbf{b}}_{ak}}\delta t_i - \Delta\hat{\mathbf{R}}_i\boldsymbol{\Xi}_2 \tag{26}$$

$$\frac{\partial\Delta\tilde{\mathbf{v}}_{i+1}}{\partial\tilde{\mathbf{b}}_{gk}} \simeq \frac{\partial\Delta\tilde{\mathbf{v}}_i}{\partial\tilde{\mathbf{b}}_{gk}} - \Delta\hat{\mathbf{R}}_i\lfloor\boldsymbol{\Xi}_1\hat{\mathbf{a}}_i\rfloor\frac{\partial\Delta\boldsymbol{\theta}_i}{\partial\tilde{\mathbf{b}}_{gk}} + \Delta\hat{\mathbf{R}}_i\boldsymbol{\Xi}_3 \tag{27}$$

$$\frac{\partial\Delta\tilde{\mathbf{v}}_{i+1}}{\partial\tilde{\mathbf{b}}_{ak}} \simeq \frac{\partial\Delta\tilde{\mathbf{v}}_i}{\partial\tilde{\mathbf{b}}_{ak}} - \Delta\hat{\mathbf{R}}_i\boldsymbol{\Xi}_1 \tag{28}$$

The derivation for other IMU state measurements and analytic Jacobians to $\Delta\mathbf{b}_{gi}$ and $\Delta\mathbf{b}_{ai}$ can be found in technical report [33]). Thus, we can construct the linearized measurement system as:

$$\tilde{\mathbf{z}}_{I,i+1} \simeq \mathbf{\Phi}(i+1,i)\tilde{\mathbf{z}}_{I,i} + \mathbf{G}_i\mathbf{n}_{Idi} \tag{29}$$

$$\tilde{\mathbf{z}}_{I,i} = \begin{bmatrix}\Delta\boldsymbol{\theta}_i^\top & \Delta\tilde{\mathbf{p}}_i^\top & \Delta\tilde{\mathbf{v}}_i^\top & \Delta\tilde{\mathbf{b}}_{gi}^\top & \Delta\tilde{\mathbf{b}}_{ai}^\top\end{bmatrix}^\top \tag{30}$$

where $\mathbf{n}_{Idi} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{di})$ is the discrete noise [24, 33] for IMU noise $\mathbf{n}_I$ at time step $k+i$ and $\mathbf{G}_i$ is the corresponding noise Jacobians. The state transition matrix $\mathbf{\Phi}(i+1,i)$ can be written as:

$$\mathbf{\Phi}(i+1,i) = \begin{bmatrix}\mathbf{\Phi}_{11} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{\Phi}_{14} & \mathbf{0}_3\\ \mathbf{\Phi}_{21} & \mathbf{I}_3 & \mathbf{I}_3\delta t_i & \mathbf{\Phi}_{24} & \mathbf{\Phi}_{25}\\ \mathbf{\Phi}_{31} & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{\Phi}_{34} & \mathbf{\Phi}_{35}\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3\end{bmatrix} \tag{31}$$

where we have:

$$\mathbf{\Phi}_{11} = {}^{I_{k+i}}_{I_{k+i+1}}\hat{\mathbf{R}}^\top,\ \mathbf{\Phi}_{14} = -\mathbf{J}_r(\hat{\boldsymbol{\omega}}_i\delta t_i)\,\delta t_i \tag{32}$$

$$\mathbf{\Phi}_{21} = -\Delta\hat{\mathbf{R}}_i\lfloor\mathbf{\Xi}_2\hat{\mathbf{a}}_i\rfloor,\ \mathbf{\Phi}_{24} = \Delta\hat{\mathbf{R}}_i\mathbf{\Xi}_4,\ \mathbf{\Phi}_{25} = -\Delta\hat{\mathbf{R}}_i\mathbf{\Xi}_2 \tag{33}$$

$$\mathbf{\Phi}_{31} = -\Delta\hat{\mathbf{R}}_i\lfloor\mathbf{\Xi}_1\hat{\mathbf{a}}_i\rfloor,\ \mathbf{\Phi}_{34} = \Delta\hat{\mathbf{R}}_i\mathbf{\Xi}_3,\ \mathbf{\Phi}_{35} = -\Delta\hat{\mathbf{R}}_i\mathbf{\Xi}_1 \tag{34}$$

Then, the propagated IMU measurement covariance $\mathbf{Q}_{i+1}$ can thus be computed iteratively as:

$$\mathbf{Q}_{i+1} = \mathbf{\Phi}(i+1,i)\mathbf{Q}_i\mathbf{\Phi}^\top(i+1,i) + \mathbf{G}_i\mathbf{Q}_{di}\mathbf{G}_i^\top \tag{35}$$

The detailed derivation for $\mathbf{G}_i$ can be found in technical report [33]. For the proposed ACI^2, we have the following remarks:

- Compared to [5], we provide a modularized and elegant way to integrate the IMU measurements. For each integration step, we need to analytically solve only once the 4 integration components $\mathbf{\Xi}_i, i \in 1\ldots 4$. Once solved, they are repeatedly used for state transition $\mathbf{\Phi}(i+1,i)$ and noise Jacobians $\mathbf{G}_i$.
- Compared to [6] which only integrates the measurements of navigation state (IMU orientation, position and velocity), we integrate the full IMU state (combining the IMU navigation and bias states). Therefore, $\mathbf{Q}_i$ will include the correlation between biases and the navigation state (which are ignored in [6]). Thus ACI^2 is a combined integration approach that accounts for all correlations in navigation state.

IV. Visual Inertial Navigation

In this section, we will formulate the VINS cost functions including IMU measurements, visual measurements and time offset calibration.

A. IMU Cost Function

The IMU measurement model can be defined as:

$$\Delta\bar{q}_{i+1} \triangleq \mathbf{h}_q(\mathbf{x}_{\mathbf{n}_k}, \mathbf{x}_{\mathbf{n}_j}) = {}^{I_k}_{G}\bar{q} \otimes {}^{G}_{I_j}\bar{q}$$

$$\Delta\mathbf{p}_{i+1} \triangleq \mathbf{h}_p(\mathbf{x}_{\mathbf{n}_k}, \mathbf{x}_{\mathbf{n}_j}) = {}^{G}_{I_k}\mathbf{R}^\top\left({}^G\mathbf{p}_{I_j} - {}^G\mathbf{p}_{I_k} - {}^G\mathbf{v}_{I_k}\Delta t_i - \frac{1}{2}{}^G\mathbf{g}\Delta t_i^2\right)$$

$$\Delta\mathbf{v}_{i+1} \triangleq \mathbf{h}_v(\mathbf{x}_{\mathbf{n}_k}, \mathbf{x}_{\mathbf{n}_j}) = {}^{G}_{I_k}\mathbf{R}^\top\left({}^G\mathbf{v}_{I_j} - {}^G\mathbf{v}_{I_k} - {}^G\mathbf{g}\Delta t_i\right)$$

$$\Delta\mathbf{b}_{g_{i+1}} = \mathbf{b}_{g_j} - \mathbf{b}_{g_k},\ \Delta\mathbf{b}_{a_{i+1}} = \mathbf{b}_{a_j} - \mathbf{b}_{a_k}$$

where $\Delta t_i = t_{k+i} - t_k$. In order to avoid the re-integration of IMU measurements, we fix the IMU bias ($\hat{\mathbf{x}}_{\mathbf{b}_k}^{(0)}$ [6]). Then, following Eq.(7), the IMU measurements with fixed bias states can be written as:

$$\underbrace{\begin{bmatrix}\Delta\bar{q}_{i+1}(\hat{\mathbf{x}}^0_{\mathbf{b}_k}, \mathbf{n}_I)\\ \Delta\mathbf{p}_{i+1}(\hat{\mathbf{x}}^0_{\mathbf{b}_k}, \mathbf{n}_I)\\ \Delta\mathbf{v}_{i+1}(\hat{\mathbf{x}}^0_{\mathbf{b}_k}, \mathbf{n}_I)\\ \Delta\mathbf{b}_{g_{i+1}}(\mathbf{n}_I)\\ \Delta\mathbf{b}_{a_{i+1}}(\mathbf{n}_I)\end{bmatrix}}_{\mathbf{z}_{I_{kj}}(\hat{\mathbf{x}}^0_{\mathbf{b}_k}, \mathbf{n}_I)} \simeq \underbrace{\begin{bmatrix}\mathbf{h}_q(\mathbf{x}_{\mathbf{n}_k}, \mathbf{x}_{\mathbf{n}_j}) \otimes \bar{q}^{-1}(\mathbf{H}_b^q\tilde{\mathbf{x}}_{\mathbf{b}_k}) \otimes \bar{q}^{-1}(\mathbf{H}_b^q\Delta\mathbf{x}_{\mathbf{b}_k})\\ \mathbf{h}_p(\mathbf{x}_{\mathbf{n}_k}, \mathbf{x}_{\mathbf{n}_j}) - \mathbf{H}_b^p\Delta\mathbf{x}_{\mathbf{b}_k} - \mathbf{H}_b^p\tilde{\mathbf{x}}_{\mathbf{b}_k}\\ \mathbf{h}_v(\mathbf{x}_{\mathbf{n}_k}, \mathbf{x}_{\mathbf{n}_j}) - \mathbf{H}_b^v\Delta\mathbf{x}_{\mathbf{b}_k} - \mathbf{H}_b^v\tilde{\mathbf{x}}_{\mathbf{b}_k}\\ \mathbf{b}_{g_j} - \mathbf{b}_{g_k}\\ \mathbf{b}_{a_j} - \mathbf{b}_{a_k}\end{bmatrix}}_{\mathbf{h}(\mathbf{x}_{\mathbf{n}_k}, \mathbf{x}_{\mathbf{n}_j}, \mathbf{x}_{\mathbf{b}_k}, \mathbf{x}_{\mathbf{b}_j})} \tag{36}$$

where $\mathbf{z}_{I_{kj}}$ is the integrated IMU measurements connecting state $\mathbf{x}_{I_k}$ and state $\mathbf{x}_{I_j}$. $\mathbf{H}_y^z$ represents the Jacobians of state z to variable y. $\mathbf{n}_I$ describes all the stacked IMU noises between t_k and t_j. This formulation is similar to the II-A.2 and II-A.3, where $\mathbf{x}_{\mathbf{b}_k}$ is the fixed states. Please refer to our companion technical report [33] for detailed derivation of Jacobians and covariance. Finally, the corresponding IMU cost function can be written as:

$$\mathbb{C}_{I_{kj}} \triangleq \left\|\tilde{\mathbf{z}}_{I_{kj}}\right\|^2_{\mathbf{Q}_{kj}^{-1}} \tag{37}$$

where $\mathbf{Q}_{kj}$ is the integrated IMU measurement covariance between k and j following (35).

B. Point Measurement Model

As the camera moves through an environment, visual point feature measurements can be extracted and tracked between images. These camera measurements are described by:

$$\mathbf{z}_f = \mathbf{\Pi}({}^C\mathbf{p}_f) + \mathbf{n}_f,\quad \mathbf{\Pi}([x\ y\ z]^\top) = \begin{bmatrix}\frac{x}{z} & \frac{y}{z}\end{bmatrix}^\top \tag{38}$$

where $\mathbf{n}_f \sim (\mathbf{0}, \mathbf{R}_f)$ and ${}^C\mathbf{p}_f$ represents the 3D position of the point feature expressed in the camera frame. In this work, we treat the IMU clock as the true time and estimate the offset of the aiding sensor relative to this base clock [26, 34]. Similarly to [28], we model the time offset t_d as a time-varying value:

$$\dot{t}_d = \mathbf{n}_t,\ t_{d,k} = t_{C,k} - t_{I,k} \tag{39}$$

where $\mathbf{n}_t \sim \mathcal{N}(\mathbf{0}, \sigma_t)$. $t_{C,k}$ is the time recorded on the camera measurements while $t_{I,k}$ is the corresponding true IMU time. According to our time offset definition (39), the feature ${}^C\mathbf{p}_f$ in the sensor frame with reported time stamp $t_{C,k}$ corresponds to the time $t_{C,k} - t_{d,k}$ in the IMU base clock. Hence, we have:

$${}^C\mathbf{p}_f = {}^C_I\mathbf{R}^I_G\mathbf{R}(t_{C,k} - t_{d,k})\left({}^G\mathbf{p}_f - {}^G\mathbf{p}_I(t_{C,k} - t_{d,k})\right) + {}^C\mathbf{p}_I \tag{40}$$

where ${}^I_G\mathbf{R}(t_{C,k} - t_{d,k})$ and ${}^G\mathbf{p}_I(t_{C,k} - t_{d,k})$ represent the IMU pose at time $t_{C,k} - t_{d,k}$, which will be denoted as time step k for simplicity in the ensuing derivations.

C. Visual Inertial Navigation with Time offset Calibration

In order to avoid re-integration of IMU readings when the time offset estimates change, we propose to fix the time offset linearization point, which is similar to II-A.1. We have the point feature measurement at time step k as:

$$\mathbf{z}_{f,k} = \mathbf{h}_C(\mathbf{x}_{I_k}, \mathbf{p}_f, t_{d,k}) + \mathbf{n}_f \tag{41}$$

If we fix the linearization $t_{d,k}^{(0)}$, we get:

$$\mathbf{z}_{f,k} = \mathbf{h}_C(\hat{\mathbf{x}}_{I_k}, \hat{\mathbf{p}}_f, \hat{t}_{d,k}^{(0)}) + \mathbf{H}_t\Delta t_{d,k} + \mathbf{H}_{I_k}\tilde{\mathbf{x}}_{I_k} + \mathbf{H}_f\tilde{\mathbf{p}}_f + \mathbf{H}_t\tilde{t}_{d,k} + \mathbf{n}_f$$

where $\mathbf{H}_x$ is the Jacobian to state x and can be found in our companion technical report [33]. Note that $\mathbf{H}_t \Delta t_{d,k}$ is the time offset correction term due to the fixed time offset linearization point. Therefore, the related cost for vision model can be written as:

$$\mathbb{C}_{C_k} \triangleq \left\|\tilde{\mathbf{z}}_{f,k}\right\|^2_{\mathbf{R}_f^{-1}} \tag{42}$$

Taking into account the random walk of time-offset, we have:

$$\mathbb{C}_{t_{kj}} \triangleq \left\|\tilde{\mathbf{z}}_{t_{kj}}\right\|^2_{\mathbf{R}_t^{-1}}, \; z_{t_{kj}} = t_{d,j} - t_{d,k} + \int_{t_k}^{t_j} n_t d\tau \tag{43}$$

where $\mathbf{R}_t = \sigma_t^2(t_j - t_k)$ and $z_{t_{kj}}$ is defined as the time offset measurement between time step k and j. Finally, the overall time-offset calibration cost function for the ACI^2 based VINS system is described as:

$$\mathbb{C} = \sum_{k,j} \mathbb{C}_{I_{kj}} + \sum_{k} \mathbb{C}_{C_k} + \sum_{k,j} \mathbb{C}_{t_{kj}} \tag{44}$$

Where $\mathbb{C}_{I_{k,j}}$, $\mathbb{C}_{C_k}$ and $\mathbb{C}_{t_{kj}}$ are from (37), (42) and (43), respectively. By minimizing (44), we can optimize IMU states, landmarks and IMU-cam time offset.

V. Experimental Verification

A. IMU Integrator Comparison

In the VI-SLAM simulator [5], an MAV with an IMU and stereo camera is simulated to follow a 3D trajectory shown in Fig. 3. The camera was simulated to operate at 10Hz while the IMU was simulated at four different rates from 100Hz to 800Hz. At each camera frame, 80 visual feature measurements are generated. We implement the proposed ACI^2 using the fixed-lag smoother within GTSAM [23] and integrated it into the simulator. For a fair comparison, we ran over 30 Monte-Carlo iterations to compare the proposed ACI^2 against the state-of-art discrete pre-integration [6] and CPI [5] side by side. Note that for complete comparison with CPI, we implement two ACI^2 models: MODEL 1 (based on piece-wise constant acceleration measurement assumption) and MODEL 2 (based on piece-wise constant true acceleration assumption, reference to [5, 33] for detail comparisons and implementations.)

Table I presents the root mean square error (RMSE) in IMU poses from 30 Monte-Carlo runs for different IMU integrators. We observe that both ACI^2 and CPI have better accuracy than Forster's integrator [6]. It is also observed that at lower frequency, ACI^2 has better performance compared to both CPI and Forster's. Overall, ACI^2 has improved accuracy than CPI MODEL 1 and MODEL 2.

B. Time Offset Calibration

With Open VINS [18] simulator, we simulated two trajectories (see Fig. 4) from real world datasets: UD Gore Hall dataset [5] and TUM Room1 dataset [35]. In both simulations, the IMU operated at 400Hz while the camera operated at 10Hz. When the simulated visual-inertial sensor rig was moving along the trajectories, 50 point feature measurements are generated and one pixel white Gaussian noise is added to each of the measurement. The true time offset between IMU and camera is set to be zero.

When running our system, the IMU state is initialized with ground truth and the initial time offset is perturbed by 0.01s. 10 Monte-Carlo runs are performed with these two simulated datasets. The relative pose error (RPE) [36] was used to evaluate the estimation accuracy for the proposed ACI^2 (shown in Fig. 5). For comparison, we evaluated the proposed system with and without time offset calibration. The pose estimation results compared to online time offset calibration are both shown in Fig. 5. The estimated trajectories and time offset of a typical run for both datasets are shown in Fig. 4. It can be seen that even (when) the system is perturbed with 0.01s, the time offset converges to the true value with the proposed algorithm.

C. VI-SLAM Experiments

We also conducted a proof-of-concept experiment with real world data collected by Intel RealSense $T265$[1] observing ArUco tags [37] to verify the proposed ACI^2 estimator. The collected data has a total trajectory length of approx. 66m and has 24 tags in front of the sensor. Note that when collecting the data, the trajectory started and ended at approximately the same point. The estimated trajectory and tags are shown in Fig. 6. We observe that in the resultant trajectory the starting and ending points are close. Additionally, most of the tags are in a planar surface similar to the experimental setup.

VI. Conclusions and Future Work

In this paper, we use MLE with partial-fixed estimates to formulate a unified derivation for analytic combined IMU integrator (ACI^2) and time-offset calibrator. This enables us to incorporate linearization constraints on parameters in MLE. In contrast to all previous approaches, our method enables a modularized analytic combined IMU integrator (ACI^2), in which 4 integration components are shared between IMU integration, Jacobians and covariances. Next, we have introduced a novel batch optimization based time offset calibration between the IMU and camera, which has the potential to be extended to multiple types of sensors. Finally, we have performed intensive Monte-Carlo simulations to verify that our ACI^2 is comparable to the state-of-the-art and validate the time offset calibration. A proof-of-concept real world experiment is also conducted to validate the proposed ACI^2. Overall, we have presented computationally efficient approaches for handling both re-integration and online time offset calibration in VINS. This paves the way for more efficient resource constrained implementation on hardware. In the future, we will improve the system design (feature tracking and outliers rejection) to verify the time offset calibration with real world datasets and compare the efficiency of these integrators. The proposed time offset calibration will also be evaluated and compared to other approaches [10, 26]. Additionally, graph sparsification and marginalization [38] will be leveraged to improve the system efficiency.

[1] https://www.intelrealsense.com/tracking-camera-t265/

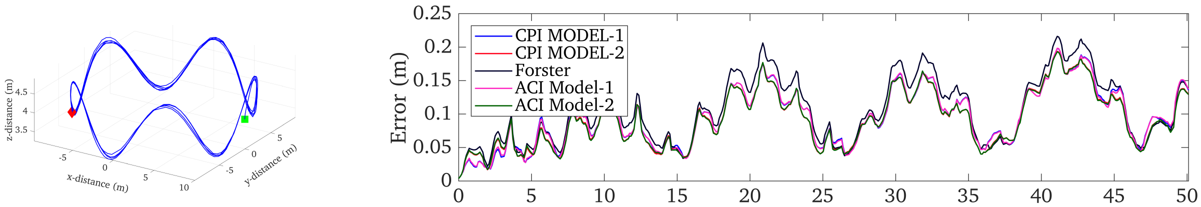

Fig. 3: Simulated trajectory (about 290m) for visual-inertial SLAM (left). Average position RMSE for visual-inertial SLAM with different IMU integrator at 100Hz over 30 Monte-Carlo simulation runs (right).

TABLE I: Comparisons of pose RMSE of ACI, CPI and Forster IMU integration algorithms with different IMU frequencies over 30 Monte-Carlo runs.

Algorithm	100Hz		200Hz		400Hz		800Hz	
	Position	Orientation	Position	Orientation	Position	Orientation	Position	Orientation
FORSTER	0.10693	0.33638	0.05723	0.21008	0.03478	0.11395	0.02887	0.08586
CPI-Model 1	0.09609	0.33562	0.04992	0.20712	0.03351	0.11276	0.02838	**0.08525**
ACI^2-Model 1	0.09580	0.33503	0.04968	0.20741	0.03346	0.11286	**0.02813**	0.08526
CPI-Model 2	0.09282	0.30895	0.04749	0.18084	0.03286	0.10568	0.02838	0.08545
ACI^2-Model 2	**0.09244**	**0.30569**	**0.04732**	**0.17995**	**0.03284**	**0.10557**	0.02838	0.08545

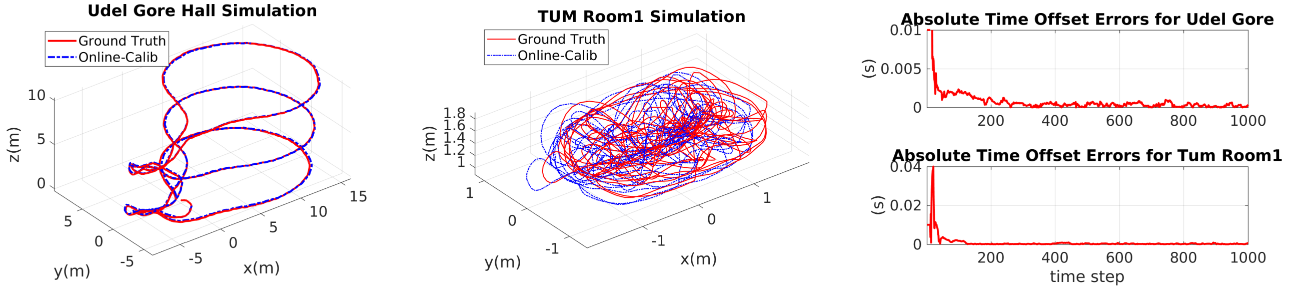

Fig. 4: A typical run for proposed system with online time offset calibration on UDel Gore Hall dataset (smooth trajectory about 249m with 2437 map points, shown in left) and TUM Room1 dataset (dynamic trajectory about 207m with 898 map points, middle). The absolute errors of the time offset estimates for these two datasets are shown in right. The time offset errors starts from 0.01s and converge to near 0.0 finally.

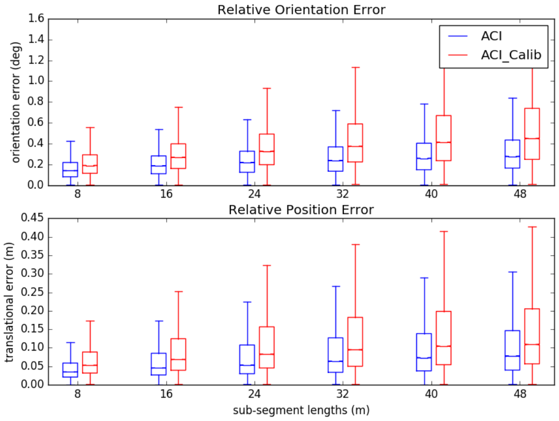

Fig. 5: Boxplot of the relative trajectory error statistics for proposed ACI with and without online calibration over 10 Monte-Carlo runs. Note that ACI denotes the system without online calibration but initialized with true time offset. ACI_Calib denotes the online calibration system with initial time offset as 0.01s while the true time offset is 0s.

APPENDIX I

The right Jacobians of Hamilton quaterion ($\mathbf{J}_r^{(H)}$[33]) and SO(3) ($\mathbf{J}_r^{(S)}$[6]), and the left Jacobians of JPL quaternion ($\mathbf{J}_l^{(J)}$[24]) can be defined as:

$$\bar{q}(\boldsymbol{\phi}+\delta\boldsymbol{\theta}) = \bar{q}(\boldsymbol{\phi})\otimes\bar{q}(\mathbf{J}_r^{(H)}\delta\boldsymbol{\theta}) \tag{45}$$

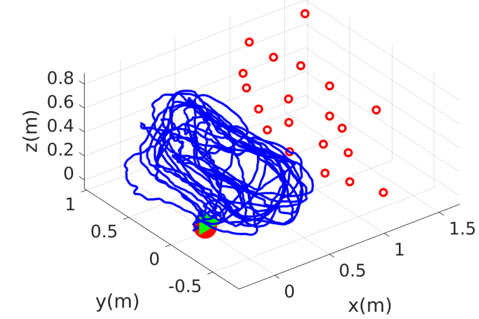

Fig. 6: ACI^2 based VI-SLAM with ArUco tags. The blue line is estimated trajectory (around 66m) for the IMU and the red circles are estimated tags. The green triangular and the red filled circle are the starting and ending points, respectively.

$$\bar{q}_J(\boldsymbol{\phi}+\delta\boldsymbol{\theta}) = \bar{q}_J(\mathbf{J}_l^{(J)}\delta\boldsymbol{\theta})\otimes\bar{q}_J(\boldsymbol{\phi}) \tag{46}$$

$$\mathbf{exp}(\boldsymbol{\phi}+\delta\boldsymbol{\theta}) = \mathbf{exp}(\boldsymbol{\phi})\mathbf{exp}(\mathbf{J}_r^{(S)}\delta\boldsymbol{\theta}) \tag{47}$$

Proposition 1. *The right Jacobians of Hamilton quaternion and SO(3), and the left Jacobians of JPL quaterion are equivalent:* $\mathbf{J}_r^{(H)} = \mathbf{J}_r^{(S)} = \mathbf{J}_l^{(J)}$ *with* $\phi = \|\boldsymbol{\phi}\|$, $\mathbf{k} = \boldsymbol{\phi}/\phi$ *and*

$$\mathbf{J}_r^{(H)}(\boldsymbol{\phi}) = \mathbf{I}_3 - \frac{1-\cos\phi}{\phi}\lfloor\mathbf{k}\rfloor + \frac{\phi-\sin\phi}{\phi}\lfloor\mathbf{k}\rfloor^2 \tag{48}$$

Proof. See our companion technical report [33]. □

References

[1] A. I. Mourikis and S. I. Roumeliotis. “A multi-state constraint Kalman filter for vision-aided inertial navigation”. In: *International Conference on Robotics and Automation*. Rome, Italy, Apr. 2007, pp. 3565–3572.

[2] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. “Keyframe-based visual-inertial odometry using nonlinear optimization”. In: *International Journal of Robotics Research* (Dec. 2014).

[3] Tong Qin, Peiliang Li, and Shaojie Shen. “VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator”. In: *arXiv preprint arXiv:1708.03852* (2017).

[4] Zheng Huai and Guoquan Huang. “Robocentric Visual-Inertial Odometry”. In: *International Journal of Robotics Research* (Apr. 2018).

[5] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. “Closed-form preintegration methods for graph-based visual–inertial navigation”. In: *The International Journal of Robotics Research* 38.5 (2019), pp. 563–586.

[6] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. “On-Manifold Preintegration for Real-Time Visual–Inertial Odometry”. In: *IEEE Transactions on Robotics* 33.1 (Feb. 2017), pp. 1–21. ISSN: 1552-3098. DOI: 10.1109/TRO.2016.2597321.

[7] Chao X Guo and Stergios I Roumeliotis. “IMU-RGBD camera navigation using point and plane features”. In: *International Conference on Intelligent Robots and Systems*. Tokyo, Japan, Nov. 2013, pp. 3164–3171.

[8] Yulin Yang, Patrick Geneva, Xingxing Zuo, Kevin Eckenhoff, Yong Liu, and Guoquan Huang. “Tightly-Coupled Aided Inertial Navigation with Point and Plane Features”. In: *Proc. IEEE International Conference on Robotics and Automation*. Montreal, Canada, May 2019.

[9] Patrick Geneva, Kevin Eckenhoff, Yulin Yang, and Guoquan Huang. “LIPS: LiDAR-Inertial 3D Plane SLAM”. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Madrid, Spain, Oct. 2018.

[10] Cedric Le Gentil, Teresa Vidal-Calleja, and Shoudong Huang. “IN2LAAMA: INertial Lidar Localisation Autocalibration And MApping”. In: *arXiv preprint arXiv:1905.09517* (2019).

[11] Haoyang Ye, Yuying Chen, and Ming Liu. “Tightly Coupled 3D Lidar Inertial Odometry and Mapping”. In: *arXiv preprint arXiv:1904.06993* (2019).

[12] Yulin Yang and Guoquan Huang. “Acoustic-Inertial Underwater Navigation”. In: *Proc. of the IEEE International Conference on Robotics and Automation*. Singapore, May 2017, pp. 4927–4933.

[13] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. “Event-based visual inertial odometry”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. Honolulu, Hi, USA, July 2017, pp. 5391–5399.

[14] A. I. Mourikis, S. I. Roumeliotis, and J. W. Burdick. “SC-KF Mobile Robot Localization: A Stochastic Cloning-Kalman Filter for Processing Relative-State Measurements”. In: *IEEE Transactions on Robotics* 23.3 (Aug. 2007), pp. 717–730.

[15] T. Qin, P. Li, and S. Shen. “VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator”. In: *IEEE Transactions on Robotics* 34.4 (Aug. 2018), pp. 1004–1020.

[16] Mingyang Li and Anastasios I. Mourikis. “Vision-aided inertial navigation with rolling-shutter cameras”. In: *International Journal of Robotics Research* 33.11 (Sept. 2014), pp. 1490–1507.

[17] Kejian Wu, Ahmed Ahmed, Georgios A. Georgiou, and Stergios I. Roumeliotis. “A Square Root Inverse Filter for Efficient Vision-aided Inertial Navigation on Mobile Devices”. In: *Robotics: Science and Systems*. 2015.

[18] Patrick Geneva, Kevin Eckenhoff, Woosik Lee, Yulin Yang, and Guoquan Huang. “OpenVINS: A Research Platform for Visual-Inertial Estimation”. In: *IROS 2019 Workshop on Visual-Inertial Navigation: Challenges and Applications*. Macau, China, Nov. 2019. URL: https://github.com/rpng/open_vins.

[19] Guoquan Huang. “Visual-Inertial Navigation: A Concise Review”. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Montreal, Canada, May 2019.

[20] Benzun Pious Wisely Babu, David Cyganski, James Duckworth, and Soohwan Kim. “Detection and Resolution of Motion Conflict in Visual Inertial Odometry”. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, May 2018.

[21] T. Lupton and S. Sukkarieh. “Visual-Inertial-Aided Navigation for High-Dynamic Motion in Built Environments Without Initial Conditions”. In: *IEEE Transactions on Robotics* 28.1 (Feb. 2012), pp. 61–76.

[22] Yulin Yang, James Maley, and Guoquan Huang. “Null-Space-based Marginalization: Analysis and Algorithm”. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vancouver, Canada, Sept. 2017.

[23] Frank Dellaert. “Factor graphs and GTSAM: A hands-on introduction”. In: *Technical Report* (2012).

[24] Nikolas Trawny and Stergios I. Roumeliotis. *Indirect Kalman Filter for 3D Attitude Estimation*. Tech. rep. University of Minnesota, Dept. of Comp. Sci. & Eng., Mar. 2005.

[25] Mingyang Li and Anastasios I. Mourikis. “Vision-aided inertial navigation with rolling-shutter cameras”. In: *International Journal of Robotics Research* 33.11 (Sept. 2014), pp. 1490–1507.

[26] Tong Qin and Shaojie Shen. “Online Temporal Calibration for Monocular Visual-Inertial Systems”. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Madrid, Spain, Oct. 2018.

[27] Y. Yang, P. Geneva, K. Eckenhoff, and G. Huang. “Degenerate Motion Analysis for Aided INS With Online Spatial and Temporal Sensor Calibration”. In: *IEEE Robotics and Automation Letters* 4.2 (Apr. 2019), pp. 2070–2077. ISSN: 2377-3774. DOI: 10.1109/LRA.2019.2893803.

[28] Chao Guo, Dimitrios Kottas, Ryan DuToit, Ahmed Ahmed, Ruipeng Li, and Stergios Roumeliotis. “Efficient Visual-Inertial Navigation using a Rolling-Shutter Camera with Inaccurate Timestamps”. In: *Proc. of the Robotics: Science and Systems Conference*. Berkeley, CA, July 2014.

[29] M. Kaess, A. Ranganathan, and F. Dellaert. “iSAM: Incremental Smoothing and Mapping”. In: *IEEE Transactions on Robotics* 24.6 (Dec. 2008), pp. 1365–1378.

[30] Guoquan Huang, Anastasios I. Mourikis, and Stergios I. Roumeliotis. “A First-Estimates Jacobian EKF for Improving SLAM Consistency”. In: *Proc. of the 11th International Symposium on Experimental Robotics*. Athens, Greece, July 2008.

[31] Albert S Huang, Abraham Bachrach, Peter Henry, Michael Krainin, Daniel Maturana, Dieter Fox, and Nicholas Roy. “Visual odometry and mapping for autonomous flight using an RGB-D camera”. In: *Proc. of the International Symposium on Robotics Research*. Flagstaff, AZ, Aug. 2011, pp. 1–16.

[32] Joan Sola. “Quaternion kinematics for the error-state Kalman filter”. In: *arXiv preprint arXiv:1711.02508* (2017).

[33] Yulin Yang, Chuchu Chen, and Guoquan Huang. *Supplementary Material: Analytic Combined IMU Integration (ACI^2) For Visual Inertial Navigation*. Tech. rep. RPNG, University of Delaware, 2019. URL: http://udel.edu/~yuyang/downloads/supplementary_aci.pdf.

[34] M. Li and A. I. Mourikis. “Online Temporal Calibration for Camera-IMU Systems: Theory and Algorithms”. In: *International Journal of Robotics Research* 33.7 (June 2014), pp. 947–964.

[35] David Schubert, Thore Goll, Nikolaus Demmel, Vladyslav Usenko, Jörg Stückler, and Daniel Cremers. “The TUM VI benchmark for evaluating visual-inertial odometry”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1680–1687.

[36] Zichao Zhang and Davide Scaramuzza. “A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 7244–7251.

[37] OpenCV Developers Team. *Open Source Computer Vision (OpenCV) Library*. Available: http://opencv.org.

[38] Kevin Eckenhoff, Liam Paull, and Guoquan Huang. “Decoupled, Consistent Node Removal and Edge Sparsification for Graph-based SLAM”. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Daejeon, Korea, Oct. 2016, pp. 3275–3282.