ICE-BA: Incremental, Consistent and Efficient Bundle Adjustment for Visual-Inertial SLAM

Haomin Liu[1] Mingyu Chen[1] Guofeng Zhang[2] Hujun Bao[2] Yingze Bao[1]
[1]Baidu [2]State Key Lab of CAD&CG, Zhejiang University
{liuhaomin,chenmingyu01,baoyingze}@baidu.com {zhangguofeng,bao}@cad.zju.edu.cn

Abstract

Modern visual-inertial SLAM (VI-SLAM) achieves higher accuracy and robustness than pure visual SLAM, thanks to the complementariness of visual features and inertial measurements. However, jointly using visual and inertial measurements to optimize SLAM objective functions is a problem of high computational complexity. In many VI-SLAM applications, the conventional optimization solvers can only use a very limited number of recent measurements for real time pose estimation, at the cost of suboptimal localization accuracy. In this work, we renovate the numerical solver for VI-SLAM. Compared to conventional solvers, our proposal provides an exact solution with significantly higher computational efficiency. Our solver allows us to use remarkably larger number of measurements to achieve higher accuracy and robustness. Furthermore, our method resolves the global consistency problem that is unaddressed by many state-of-the-art SLAM systems: to guarantee the minimization of re-projection function and inertial constraint function during loop closure. Experiments demonstrate our novel formulation renders lower localization error and more than 10x speedup compared to alternatives. We release the source code of our implementation to benefit the community [1].

1. Introduction

Simultaneous localization and mapping (SLAM) is a classic but ongoing research problem in many applications. In recent years, due to the mass availability of imaging and inertial sensors, visual-inertial SLAM (VI-SLAM) is increasingly adopted in products such as mobile augmented reality, drones, autonomous driving, robotics *etc*. Similar to pure visual SLAM, VI-SLAM extracts and establishes feature correspondences across image frames. But it further utilizes inertial measurement (*e.g*. acceleration and angular velocity readings) as constraints in motion estimation. Inertial measurements are very effective for motion estimation especially when the motion is rapid and irregular, which is notoriously challenging for visual feature matching. Given sufficient computation capacity, state-of-the-arts VI-SLAM [22, 26] have shown excellent results in terms of 6 degree of freedom (DOF) accuracy by using a large number of measurements.

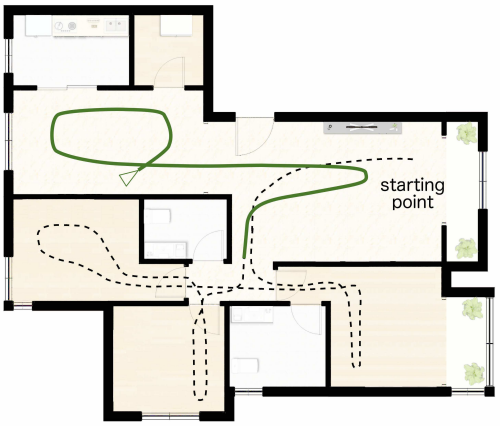


Figure 1: Our SLAM trajectory overlaid with an apartment floor map. The temporal sliding window (green solid line) for our real time pose solver is significantly longer than other methods. Our novel algorithm allows us to use remarkably higher number of measurements without surrendering efficiency. The black dashed line is the full trajectory that is globally optimized and consistent with local optimization.

Since most applications of SLAM are mobile and time critical, the computational complexity of VI-SLAM also deserves great attention. Only a minority of VI-SLAM systems [24, 28, 11] can be deployed onto embedded devices. Improving the efficiency of VI-SLAM computation is inevitably the key to popularizing its applications. There are two major computational tasks in VI-SLAM: front-end task and solver task. Front-end task includes visual feature extraction and matching. Front-end tasks are generally parallelizable, and thus can be accomplished efficiently using modern heterogeneous computing architecture. The goal of

[1]https://github.com/baidu/ICE-BA

the solver task is to optimize the pose parameters by minimizing the VI-SLAM objective functions given a set of visual features and inertial measurements. The solver task is usually the speed bottleneck to VI-SLAM.

Most previous VI-SLAM frameworks simply applied conventional numeric solvers to solve the objective function. Bundle adjustment (BA) is an example of the solver task given only visual measurements. In this work, we generalize the term BA to denote the joint optimization of visual and inertial measurements. These conventional solvers such as Gauss-Newton and Levenberg-Marquardt are designed to provide numerically accurate results without much consideration for real time issues. Consequently, real time VI-SLAM applications [24, 7, 22] based on these solvers are only capable of using the most recent measurements to estimate the latest device pose (*i.e*. apply a very short sliding window in local BA). Theoretically, longer measurement history leads to higher estimation accuracy. The efficiency of BA is apparently one of the most crucial factors to the performance of VI-SLAM.

We renovate the BA process for VI-SLAM, as we considerably improve the local and global optimization efficiency and solve the inconsistency issue during loop closure. In the SLAM problem, the incoming visual and inertial measurements arrive sequentially. We leverage this fact and propose to effectively re-use the intermediate results of previous optimization to avoid redundant new computation. Our generalizable algorithm remarkably increases the solver speed and can be applied to most sliding-window based VI-SLAM.

Furthermore, our method addresses the global consistency problem, which is critical to applications such as AR. A global map is considered to be consistent if loops can be closed and the re-projection error is sufficiently minimized. For visual SLAM, global consistency can be maintained by running global BA or its pose graph approximation [9, 25]. However, the problem is more complicated for VI-SLAM, where the constraints of velocity and IMU bias between frames create many local minimals in the optimization problem. When measurements are removed from a temporal sliding window, naive marginalization accumulates error over time, which would finally conflict with the loop constraint. Previous methods either skip marginalization [26], or apply marginalization without resolving the conflict [28].

This paper proposes a novel solver algorithm for visual-inertial SLAM with the following contributions: a new sliding window based solver that leverages the incremental nature of SLAM measurements to achieve more than 10x efficiency compared to the state-of-the-arts; a new relative marginalization algorithm that resolves the conflicts between sliding window marginalization bias and global loop closure constraints; our experimentally validated implementation will be open sourced.

2. Related Work

Early SLAM are mostly EKF (Extended Kalman Filter) based [6, 8]. The 6 DOF motion parameters and 3D landmarks are probabilistically represented as a single state vector. The complexity of classic EKF grows quadratically with the number of landmarks, restricting its scalability.

Visual SLAM [20, 25, 9] solves the SLAM problem using only visual features. By carefully extracting and matching a very large number of sophisticated visual features, these methods are capable of providing high tracking accuracy.

Visual-inertial SLAM usually does not a require large number of image features to achieve reasonable accuracy, since inertial measurement (angular velocity and acceleration) provides additional constraints. [24, 32] improve early EKF SLAM by excluding 3D landmarks from the state vector. Thereby, they are capable of modeling multiple frames in the state. However, as a common behavior of EKF algorithms, they only maintain the most recent state, and thus they are sensitive to measurement error and difficult to be recovered from unstable tracking status. [7, 22, 28, 26] use a temporally sliding window to select the most recent visual and inertial measurements for optimizing SLAM objective functions. They show that in many cases sliding window based VI-SLAM is more robust and accurate than filter based methods. However, the objective function optimization is of high computational complexity. The performance of sliding window based VI-SLAM highly depends on the computational availability, which is strictly limited on mobile devices and drones. Our proposed novel method intends to address this problem by greatly improving efficiency of the optimization solver.

Optimization Solvers are commonly shared by various SLAM implementations, although their front-end systems and frameworks are very different. BA [30] of visual SLAM utilizes the sparseness structure of re-projection function and Hessian. In this work, the term BA is generalized to denote joint optimization of visual and inertial measurements for VI-SLAM. [2, 4, 31] improve the efficiency of BA in large-scale setup. [17] shows that the block-based preconditioned conjugate gradient (PCG) can be used to solve the Schur complement for efficiency gain. There are also excellently engineered implementations of BA [1, 21] that are commonly used by state-of-the-art SLAM systems. However, all these methods suffer from the fact that its complexity grows quadratically w.r.t the number of cameras. Thus, the SLAM systems built upon these solvers can only use a very limited number of recent measurements for real time pose estimation.

Incremental Solvers are recently being explored by researchers in attempts to exploit the previous optimization

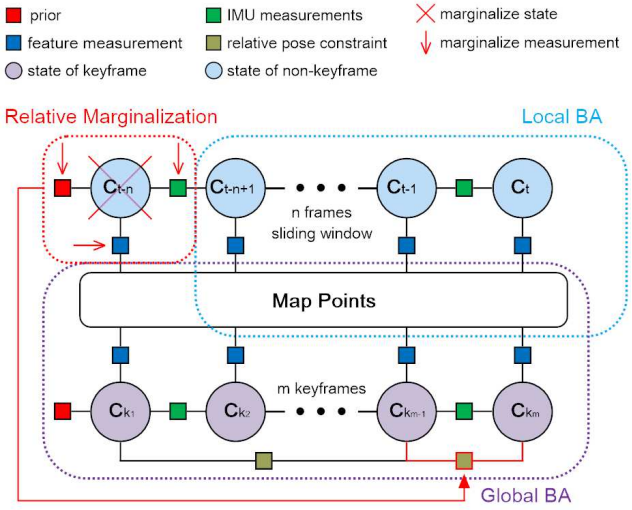


Figure 2: Local and global optimization framework

result to reduce the amount of new computation. Kaess *et al*. [19, 18] propose to solve the optimization by QR factorization of the measurement matrix. Each new optimization iteration only updates a small portion of the factorization results instead of factorizing the entire graph. Similarly, Ila *et al*. [16] propose to incrementally recover the estimate and covariance, and recently propose to update Schur complement incrementally in BA [15]. However, the aforementioned methods are only suitable for solving "sparse" camera problem (*i.e*. most key points are only observable in a small number of cameras). While this is true for large scale structure-from-motion, in SLAM problems, most frames in local sliding window share a large number of common points, which degenerates those incremental solvers into regular BA solver. As a result, they do not show better localization accuracy than other state-of-the-art SLAM. In this work, we propose a novel incremental solver that better leverages the specific block matrix structure in SLAM, and shows superior performance in terms of speed and accuracy. As a major extension to our early work [23], this paper further discusses the acceleration of local BA and the relative marginalization for the global consistency. We also provide substantially more experimental results. To our best knowledge, this paper describes the first BA based VI-SLAM solver, achieving unprecedented efficiency with state-of-the-art accuracy, and simultaneously ensuring global consistency.

3. Framework

We first define the constraint functions, and next explain our local and global optimization framework.

3.1. Constraint Functions

The goal of visual-inertial SLAM is defined as using visual and inertial measurements up to time point t to estimate the motion state $\mathbf{C}_t$, as well as a set of 3D points $\{\mathbf{X}_1, \cdots, \mathbf{X}_{n_t}\}$. $\mathbf{C}_t = (\mathbf{T}_t, \mathbf{M}_t)$, where $\mathbf{T}_t = (\mathbf{R}_t, \mathbf{p}_t)$ is the camera pose, and $\mathbf{M}_t = (\mathbf{v}_t, \mathbf{b}_t)$ is the IMU state including velocity $\mathbf{v}_t$ and sensor bias $\mathbf{b}_t$. A 3D point $\mathbf{X}_j$ is projected onto the i-th image plane corresponding to a 2D feature measurement $\mathbf{x}_{ij} = \pi(\mathbf{T}_i \circ \mathbf{X}_j) + \mathbf{n}_{ij}$, where $\mathbf{n}_{ij}$ is Gaussian noise $\mathbf{n}_{ij} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{ij}^{\text{vis}})$. The 3D point is parametrized using inverse depth [5] as $\mathbf{X}_j = \mathbf{T}_{s_j}^{-1} \circ \frac{1}{\rho_j} \bar{\mathbf{x}}_{s_j j}$, where ρ_j is the inverse depth of j-th point, s_j is the source frame from which j-th point is extracted. $\bar{\mathbf{x}}$ is the homogenous coordinate of $\mathbf{x}$. The visual constraint is defined as

$$f_{ij}^{\text{vis}}(\mathbf{T}_i, \mathbf{T}_{s_j}, \rho_j) = \pi(\mathbf{T}_i \circ \mathbf{T}_{s_j}^{-1} \circ \frac{1}{\rho_j} \bar{\mathbf{x}}_{s_j j}) - \mathbf{x}_{ij} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{ij}^{\text{vis}}). \tag{1}$$

IMU measurements $\mathcal{Z}_{ij}$ obtained between frame i and j provide relative motion constraint. The IMU constraint is defined as

$$\begin{aligned} f_{ij}^{\text{imu}}(\mathbf{C}_i, \mathbf{C}_j) &= (\mathbf{e}_r^T, \mathbf{e}_v^T, \mathbf{e}_p^T, \mathbf{e}_b^T)^T \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{ij}^{\text{imu}}) \\ \mathbf{e}_r &= \text{Log}((\text{Exp}(\Delta\mathbf{J}_{ij}^r(\mathbf{b}_i - \hat{\mathbf{b}}_i))\Delta\mathbf{R}_{ij})^T \mathbf{R}_j \mathbf{R}_i^T) \\ \mathbf{e}_v &= \mathbf{R}_i(\mathbf{v}_j - \mathbf{v}_i - \mathbf{g}\Delta t_{ij}) - (\Delta\mathbf{v}_{ij} + \Delta\mathbf{J}_{ij}^v(\mathbf{b}_i - \hat{\mathbf{b}}_i)) \\ \mathbf{e}_p &= \mathbf{R}_i(\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i\Delta t_{ij} - \frac{1}{2}\mathbf{g}\Delta t_{ij}^2) \\ &\quad - (\Delta\mathbf{p}_{ij} + \Delta\mathbf{J}_{ij}^p(\mathbf{b}_i - \hat{\mathbf{b}}_i)) \\ \mathbf{e}_b &= \mathbf{b}_j - \mathbf{b}_i \end{aligned} \tag{2}$$

The Δ's and the covariance matrix $\mathbf{\Sigma}_{ij}^{\text{imu}}$ depend only on $\mathcal{Z}_{ij}$ and can be pre-integrated before optimization. $\hat{\mathbf{b}}_i$ is the bias estimate at the time of pre-integration. Please refer to [10] for more details.

The absolute position and yaw around the gravity are unobservable in VI-SLAM [14]. A prior is imposed on the first camera $\mathbf{C}_0$, denoted as $f_0^{\text{prior}}(\mathbf{C}_0) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_0^{\text{prior}})$.

3.2. Local and Global Optimization

It is infeasible to only perform global optimization in solving a long-time VI-SLAM problem. Similar to [28, 26], our framework (Fig. 2) includes both a local optimization (local BA) and a global optimization (global BA).

Local BA optimizes the states within a temporarily sliding window that only contains the latest frames and points. The goal of local BA is to reduce accumulated error and expand the map as fast as possible. The cost function of local BA is to minimize

$$\begin{aligned} &\underset{\{\mathbf{C}_i, \rho_j | i=t_0 \cdots t, j \in \mathcal{V}_i\}}{\arg\min} \sum_{i=t_0}^{t} \sum_{j \in \mathcal{V}_i} ||f_{ij}^{\text{vis}}(\mathbf{T}_i, \mathbf{T}_{s_j}, \rho_j)||_{\mathbf{\Sigma}_{ij}^{\text{vis}}} \\ &+ ||f_{t_0}^{\text{prior}}(\mathbf{C}_{t_0})||_{\mathbf{\Sigma}_{t_0}^{\text{prior}}} + \sum_{i=t_0}^{t-1} ||f_{i,i+1}^{\text{imu}}(\mathbf{C}_i, \mathbf{C}_{i+1})||_{\mathbf{\Sigma}_{i,i+1}^{\text{imu}}} \end{aligned} \tag{3}$$

where $t_0 = t - n + 1$ is the first frame in sliding window and n is the size of sliding window. $\mathcal{V}_i$ denotes the set of points tracked in frame i. As one of our major contributions, Sec. 4 explains how to efficiently solve Eq. (3).

Global BA runs in parallel to local BA at a relatively lower frequency. Global BA optimizes the frames that are removed from local sliding window but selected as key frames in global map. A frame is selected as a key frame in global BA if it carries more than N (*e.g*. 20 in our experiments) features that have not been seen from all other frames. The cost function of global BA is

$$\begin{aligned}&\underset{\{\mathbf{C}_i,\rho_j|i\in\{k_1\cdots k_m\},j\in\mathcal{V}_i\}}{\arg\min}\sum_{i=k_1}^{k_m}\sum_{j\in\mathcal{V}_i}||f_{ij}^{\text{vis}}(\mathbf{T}_i,\mathbf{T}_{s_j},\rho_j)||_{\mathbf{\Sigma}_{ij}^{\text{vis}}}+\\&||f_0^{\text{prior}}(\mathbf{C}_{k_1})||_{\mathbf{\Sigma}_0^{\text{prior}}}+\sum_{i=1}^{m-1}||f_{k_i,k_{i+1}}^{\text{imu}}(\mathbf{C}_{k_i},\mathbf{C}_{k_{i+1}})||_{\mathbf{\Sigma}_{k_i,k_{i+1}}^{\text{imu}}}+\\&\sum_i||f_i^{\text{rel}}(\{\mathbf{T}_{k\in\mathcal{L}_i}\})||_{\mathbf{\Sigma}_i^{\text{rel}}}\end{aligned}\tag{4}$$

where $\mathcal{L}_i$ is the set of keyframes involved in i-th relative pose constraint. Loop closure triggers global BA that should account for map consistency. For a typical loop constraint, $|\mathcal{L}_i|=2$. As one of our major contributions, Sec. 4 explains how to efficiently solve Eq. (4).

Relative Marginalization produces relative pose constraint between the last keyframe in local BA and the latest frame that is removed from local BA (*e.g*. the constraint between $\mathbf{C}_{k_{m-1}}$ and $\mathbf{C}_{k_m}$ in Fig. 2), so that the constraints obtained from global BA (*e.g*. loop closure) can help anchor the camera poses in local BA, preventing drifts caused by accumulation error. More details are discussed in Sec. 5.

4. Efficient Solver for VI-SLAM

Efficiently solving Eq. (3) and Eq. (4) is the key to VI-SLAM speed. Minimizing such formulations can be generalized as $\arg\min_\phi\sum_k||f_k(\phi)||^2$. In a typical Gauss Newton solver, the optimal values of ϕ are obtained by optimization iterations $\phi^+=\phi^-\oplus\delta\phi$ where the subscript $^-/^+$ denotes state before/after iteration, and $\oplus$ is the generalized addition on manifold [10]. At each iteration, the cost function f_k is linearized at current estimate ϕ^- as

$$f_k(\phi^-\oplus\delta\phi)\approx\mathbf{J}_k\delta\phi+\mathbf{e}_k\tag{5}$$

where $\mathbf{J}_k=\frac{\partial f_k(\phi^-\oplus\delta\phi)}{\partial\delta\phi}|_{\delta\phi=\mathbf{0}}$ and $\mathbf{e}_k=f_k(\phi^-)$ are the Jacobian matrix and error vector respectively. $\delta\phi$ is solved by the *normal equation*

$$\begin{aligned}\mathbf{A}\delta\phi&=\mathbf{b}\\[\mathbf{A}|\mathbf{b}]&=\sum_k[\mathbf{A}_k|\mathbf{b}_k]\\[\mathbf{A}_k|\mathbf{b}_k]&=[\mathbf{J}_k^T\mathbf{J}_k|-\mathbf{J}_k\mathbf{e}_k]\end{aligned}\tag{6}$$

4.1. General Incremental BA Solver

In global optimization in VI-SLAM, each cost function f_k involves only a very small subset of variables. For example, f_{ij}^{vis} in (1) only involves 3 types of variables $(\mathbf{T}_i,\mathbf{T}_{s_j},\rho_j)$. Then the corresponding $\mathbf{A}_k$ and $\mathbf{b}_k$ has only 9 and 3 blocks of non-zero entries. Leveraging such sparsity patten [30] and the block structure [17] leads to an efficient construction of (6). Furthermore, due to the nature of SLAM problem, new states and measurements always arrive incrementally. As a result, only a small portion of variables change at each iteration, *i.e*. only an small portion of f_k's need to be re-linearized. This fact can be exploited to significantly accelerate the construction of (6). In our early work [23], instead of computing (6) from scratch in each iteration, we incrementally update $[\mathbf{A}|\mathbf{b}]$ as

$$[\mathbf{A}|\mathbf{b}]^+=[\mathbf{A}|\mathbf{b}]^-+[\ \textstyle\sum_{k\in\mathcal{L}}\delta\mathbf{A}_k\ |\ \sum_{k\in\mathcal{L}}\delta\mathbf{b}_k\ ]\tag{7}$$

where $\mathcal{L}$ is the set of cost functions that need to be re-linearized (*i.e*. involving at least one $|\delta\phi_i|$ exceeding a preset threshold), and $[\delta\mathbf{A}_k|\delta\mathbf{b}_k]\triangleq[\mathbf{A}_k|\mathbf{b}_k]^+-[\mathbf{A}_k|\mathbf{b}_k]^-$.

For BA problem, a common strategy to efficiently solve ((6)) is to marginalize points to obtain a reduced linear system involving only cameras. ϕ is reordered as $\phi=(\phi_c^T,\phi_p^T)^T$, first camera then point parameters. Accordingly, $[\mathbf{A}|\mathbf{b}]$ can be written as

$$[\mathbf{A}|\mathbf{b}]=\left[\begin{array}{cc|c}\mathbf{U}&\mathbf{W}&\mathbf{u}\\\mathbf{W}^T&\mathbf{V}&\mathbf{v}\end{array}\right].\tag{8}$$

The second row is eliminated to obtain the *Schur complement* that involves only $\delta\phi_c$

$$\begin{aligned}\mathbf{S}\delta\phi_c&=\mathbf{s}\\[\mathbf{S}|\mathbf{s}]&=[\ \mathbf{U}-\mathbf{W}\mathbf{V}^{-1}\mathbf{W}^T\ |\ \mathbf{u}-\mathbf{W}\mathbf{V}^{-1}\mathbf{v}\ ]\end{aligned}\tag{9}$$

The block corresponding to (i_1,i_2) camera pair in $\mathbf{S}$ and i-th camera in $\mathbf{s}$ can be efficiently computed as

$$\begin{aligned}[\mathbf{S}_{i_1i_2}|\mathbf{s}_i]&=\left[\ \mathbf{U}_{i_1i_2}-\textstyle\sum_{j\in\mathcal{V}_{i_1}\cup\mathcal{V}_{i_2}}\mathbf{S}_{i_1i_2}^j\ \middle|\ \mathbf{u}_i-\sum_{j\in\mathcal{V}_i}\mathbf{s}_i^j\ \right]\\[\mathbf{S}_{i_1i_2}^j|\mathbf{s}_i^j]&=[\ \mathbf{W}_{i_1j}\mathbf{V}_{jj}^{-1}\mathbf{W}_{i_2j}^T\ |\ \mathbf{W}_{ij}\mathbf{V}_{jj}^{-1}\mathbf{v}_j\ ]\end{aligned}\tag{10}$$

As introduced in [23], the incremental arrival of SLAM measurements can be exploited to accelerate the construction of (10), by incrementally update $[\mathbf{S}_{i_1i_2}|\mathbf{s}_i]$ as

$$\begin{aligned}[\mathbf{S}_{i_1i_2}|\mathbf{s}_i]^+&=[\mathbf{S}_{i_1i_2}|\mathbf{s}_i]^-+\left[\ \textstyle\sum_{j\in\mathcal{P}_{i_1i_2}}\delta\mathbf{S}_{i_1i_2}^j\ \middle|\ \sum_{j\in\mathcal{P}_{ii}}\delta\mathbf{s}_i^j\ \right]\\\mathcal{P}_{i_1i_2}&=\mathcal{P}\cup\mathcal{V}_{i_1}\cup\mathcal{V}_{i_2}\end{aligned}\tag{11}$$

where $\mathcal{P}$ is the set of points involved in cost functions need to be re-linearized.

$\mathbf{S}_{i_1i_2}$ is nonzero if and only if (i_1,i_2) share common points or have constraint between them. This particular sparseness structure can be specifically leveraged by preconditioned conjugated gradient (PCG) to efficiently solve ((9)) [2, 4, 17]. After solving $\delta\phi_c$, point variable $\delta\phi_p$ can be solved by back-substituting $\delta\phi_c$ to the second row of (8), for each point j separately

$$\delta\phi_{p_j}=\mathbf{V}_{jj}^{-1}\left(\mathbf{v}_j-\sum_{i\in\mathcal{X}_j}\mathbf{W}_{ij}^T\delta\phi_{c_i}\right)\tag{12}$$

where $\mathcal{X}_j$ denotes the set of cameras seeing point j.

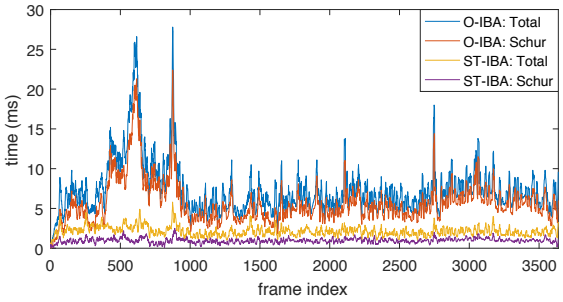


Figure 3: The total runtime and the Schur complement time for each frame in MH_01_easy sequence [3]. O-IBA is original IBA introduced in Sec. 4.1; ST-IBA is the sub-track based IBA introduced in Sec. 4.2. Between frame 400 to 900, O-IBA time significantly increases since higher number of frames share the same feature points during this period of time. ST-IBA does not suffer from this as expected.

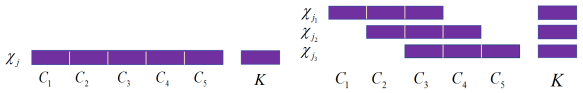
4.2. Improvement for Local BA

The incremental BA (IBA) introduced in Section 4.1 can significantly accelerate global BA where most keyframes do not share common points. However, in local BA most points can be observed by most frames in the sliding window. As a result, a large portion of $[\mathbf{S}^j_{i_1 i_2}|\mathbf{s}^j_i]$ defined in (10) has to be re-evaluated, and the incremental update of Schur complement degrades to the standard process. Fig. 3 shows the runtime for this IBA process (original IBA, or O-IBA). The update of Schur complement dominates the total runtime.

We propose an improved incremental BA solver to address the Schur complement problem in local BA. We name it Sub-Track based IBA (ST-IBA). The key idea is to split the origin long feature track $\mathcal{X}_j$ into several short overlapping sub-tracks $\mathcal{X}_{j_1}, \mathcal{X}_{j_2}, \cdots$, as illustrated in Fig. 4. Each sub-track $\mathcal{X}_{j_k}$ spans over l neighboring frames with $l < |\mathcal{X}_j|$. We set $l = 5$ in our experiments. Sub-tracks also include key frames in local BA. The corresponding inverse depth ρ_j becomes several identical duplicates $\rho_{j_1}, \rho_{j_2}, \cdots$. Instead of marginalizing ρ_j that introduces nonzero block $\mathbf{S}_{i_1 i_2}$ for each pairs of $(i_1, i_2) \in \mathcal{X}_j \times \mathcal{X}_j$, we marginalize ρ_{j_k} that introduces $\mathbf{S}_{i_1 i_2}$ for a much smaller set of pairs $(i_1, i_2) \in \mathcal{X}_{j_k} \times \mathcal{X}_{j_k}$. Consequently, $\mathbf{S}$ becomes from a dense full matrix - as long as there is one $|\mathcal{X}_j|$ reaches the size of sliding window n - to a diagonal band matrix. Furthermore, the incremental update of $[\mathbf{S}_{i_1 i_2}|\mathbf{s}_i]$ (11) becomes

$$
\begin{aligned}
[\mathbf{S}_{i_1 i_2}|\mathbf{s}_i]^+ &= [\mathbf{S}_{i_1 i_2}|\mathbf{s}_i]^- + \left[\begin{array}{c|c} \sum_{j\in\bar{\mathcal{P}}_{i_1 i_2}} \delta\bar{\mathbf{S}}^j_{i_1 i_2} & \sum_{j\in\bar{\mathcal{P}}_{ii}} \delta\bar{\mathbf{s}}^j_i \end{array}\right] \\
[\bar{\mathbf{S}}^j_{i_1 i_2}|\bar{\mathbf{s}}^j_i] &= \left[\begin{array}{c|c} \mathbf{W}_{i_1 j}\bar{\mathbf{Q}}^j_{i_1 i_2}\mathbf{W}^T_{i_2 j} & \mathbf{W}_{i_1 j}\bar{\mathbf{q}}^j_i \end{array}\right] \\
[\bar{\mathbf{Q}}^j_{i_1 i_2}|\bar{\mathbf{q}}^j_i] &= \left[\begin{array}{c|c} \sum_{j_k\in\bar{\mathcal{V}}^j_{i_1 i_2}} \mathbf{V}^{-1}_{j_k j_k} & \sum_{j_k\in\bar{\mathcal{V}}^j_{ii}} \mathbf{V}^{-1}_{j_k j_k}\mathbf{v}_{j_k} \end{array}\right] \\
\bar{\mathcal{P}}_{i_1 i_2} &= \{j|\exists k : j_k \in \bar{\mathcal{P}} \cup \bar{\mathcal{V}}^j_{i_1 i_2}\}
\end{aligned}
\tag{13}
$$

where $\bar{\mathcal{P}}$ is the set of sub-track points involved in cost functions that need to be re-linearized, and $\bar{\mathcal{V}}^j_{i_1 i_2}$ denotes the set of common sub-track points of frame (i_1, i_2) corresponding to j-th point. Comparing to (11), (13) is more efficient not only because $\mathbf{S}$ becomes sparser, but also because $\bar{\mathcal{P}}_{i_1 i_2}$ is



(a) Original feature track $\mathcal{X}_j$ (b) Overlapping sub-tracks

Figure 4: We split the original feature track $\mathcal{X}_j$ in (a) into 3 overlapping sub-tracks $\mathcal{X}_{j_1}$, $\mathcal{X}_{j_2}$ and $\mathcal{X}_{j_3}$ in (b), each spans $l = 3$ neighboring frames and the keyframes K

generally much smaller than $\mathcal{P}_{i_1 i_2}$, as the probability that a short sub-track involved in re-linearization is very low. $[\bar{\mathbf{Q}}^j_{i_1 i_2}|\bar{\mathbf{q}}^j_i]$ defined in (13) can also be incrementally updated for further speedup:

$$
\begin{aligned}
\left(\bar{\mathbf{Q}}^j_{i_1 i_2}\right)^+ &= \left(\bar{\mathbf{Q}}^j_{i_1 i_2}\right)^- + \sum_{j_k\in\bar{\mathcal{P}}^j_{i_1 i_2}} \delta\left(\mathbf{V}^{-1}_{j_k j_k}\right) \\
\left(\bar{\mathbf{q}}^j_i\right)^+ &= \left(\bar{\mathbf{q}}^j_i\right)^- + \sum_{j_k\in\bar{\mathcal{P}}^j_{ii}} \delta\left(\mathbf{V}^{-1}_{j_k j_k}\mathbf{v}_{j_k}\right) \\
\bar{\mathcal{P}}^j_{i_1 i_2} &= \bar{\mathcal{P}} \cup \bar{\mathcal{V}}^j_{i_1 i_2}
\end{aligned}
\tag{14}
$$

Note that the sub-track process is only used for the update of Schur complement. After solving Schur complement, we update 3D points by (12) for each original point j rather than the sub-track points j_k. Compared to the traditional method, since the objective function is exactly the same, especially the point substitution still uses the original normal equation without any approximation, a few more iterations can make the solution converge and the final accuracy does not decrease. As shown in Fig. 3 and Tab. 1, the proposed ST-IBA is faster than the original IBA by $2 \sim 10$ times without any noticeable loss of accuracy.

4.3. Incremental PCG for IBA

In order to solve (9), we renovated the original PCG algorithm [17]. In standard PCG, $\delta\phi_c$ is initialized as zero then iteratively updated toward the optimal values. In the case of IBA, the minimizer $\delta\phi_{c_i}$ will not actually update the state of camera i if $\delta\phi_{c_i}$ is not large enough (Sec. 4.1). For such camera i, the result of the next iteration $\delta\phi^+_{c_i}$ will be very close to the previous one $\delta\phi^-_{c_i}$, because both results are obtained by updating the same $\phi^-_{c_i}$ towards the similar optimal values. This observation helps us to better initialize $\delta\phi_c$ and accelerate convergence of PCG. Specifically, we initialize $\delta\phi^+_{c_i} = \delta\phi^-_{c_i}$ for those camera i whose state was not changed in the last iteration, and $\delta\phi^+_{c_i} = \mathbf{0}$ for the rest. We name this algorithm as incremental PCG (I-PCG) as it also utilizes the incremental nature of SLAM measurements. As shown in Tab. 1, I-PCG improves the accuracy by approx. 20% due to better convergence.

5. Relative Marginalization

If the number of frames in the sliding window of local BA surpasses a threshold (*e.g.* 50 in our experiments), the earliest frame t_0 in the sliding window needs to be eliminated. Instead of neglecting the information carried in this eliminated frame, marginalization converts it

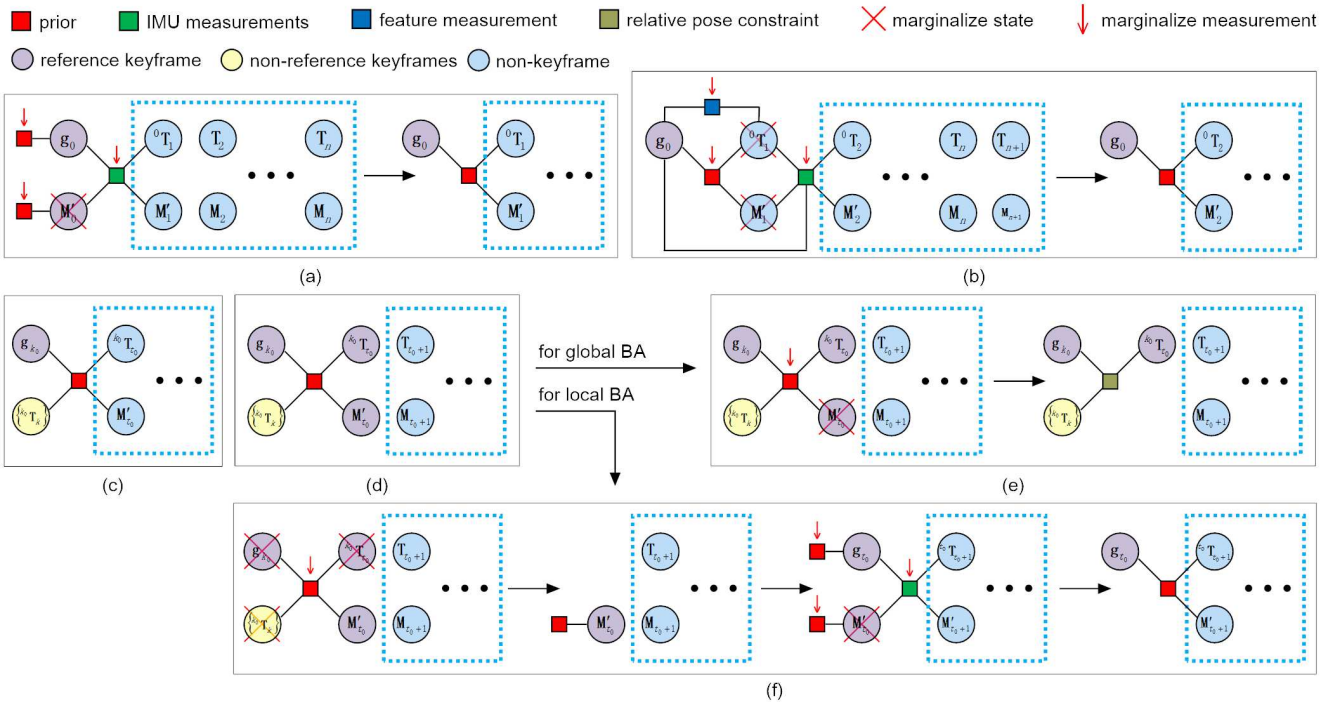


Figure 5: Relative marginalization. Let h^{vis}, h^{imu}, h^{prior} denote the visual, inertial, and prior factors, respectively. (a) For the first frame $t_0 = 0$, we add a weak prior factor $h_0^{\text{prior}}(\mathbf{g}_0, \mathbf{M}'_0) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_0^{\text{prior}})$. The state $\mathbf{M}'_0$ connected to the prior factor $h_0^{\text{prior}}(\mathbf{g}_0, \mathbf{M}'_0)$ and the inertial factor $h_{01}^{\text{imu}}(\mathbf{g}_0, \mathbf{M}'_0, \mathbf{C}'_1)$ are marginalized out, which results in a prior factor $h_1^{\text{prior}}(\mathbf{g}_0, \mathbf{C}'_1) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_1^{\text{prior}})$. (b) For the next frame $t_0 = 1$, the process is similar except that the visual factor $h_{\mathcal{V}_1}^{\text{vis}}({}^0\mathbf{T}_1)$ is involved, and both ${}^0\mathbf{T}_1$ and $\mathbf{M}'_1$ are marginalized. (c) In general, more keyframes other than k_0 are involved in the visual factor $h_{\mathcal{V}_{t_0}}^{\text{vis}}({}^{k_0}\mathbf{T}_{t_0}, \{{}^{k_0}\mathbf{T}_{s_j \neq k_0 | j \in \mathcal{V}_{t_0}}\})$. Marginalizing such a factor will introduce correlation among all the involved keyframes (yellow circles). Repeat this process until the marginalized frame t_0 is a new keyframe as in (d). Then the process for local and global BA goes in different ways. (e) For global BA, we marginalize the prior factor $h_{t_0}^{\text{prior}}(\mathbf{g}_{k_0}, \mathbf{M}'_{t_0}, \{{}^{k_0}\mathbf{T}_{k \in \mathcal{K}_{t_0}}\})$ and the IMU state $\mathbf{M}'_{t_0}$. A relative constraint is submitted to global BA as shown in Fig. 2. (f) For local BA, we first marginalize the prior factor $h_{t_0}^{\text{prior}}(\mathbf{g}_{k_0}, \mathbf{M}'_{t_0}, \{{}^{k_0}\mathbf{T}_{k \in \mathcal{K}_{t_0}}\})$. All involved states except $\mathbf{M}'_{t_0}$ are marginalized , producing a prior on $\mathbf{M}'_{t_0}$. At this point, t_0 becomes the new reference keyframe. The new state $\mathbf{g}_{t_0}$ appears, along with a weak prior on it, and the pose of the next frame $t_0 + 1$ is represented in the reference of frame t_0, *i.e.* ${}^{t_0}\mathbf{T}_{t_0+1}$. We then marginalize the prior factor and the inertial factor $h_{t_0,t_0+1}^{\text{imu}}(\mathbf{g}_{t_0}, \mathbf{M}'_{t_0}, \mathbf{C}'_{t_0+1})$. $\mathbf{M}'_{t_0}$ is marginalized out, producing a prior on $\mathbf{g}_{t_0}$ and $\mathbf{C}'_{t_0+1}$. After (e) and (f) are done, the system goes back to a state similar to (b).

into a linear prior applied onto the remaining variables. Marginalization is commonly used in visual inertial odometry (VIO) [24, 22, 11] that does not maintain a global map. Nevertheless, in the case of VI-SLAM, error accumulation will gradually corrupt the prior produced by marginalization. The corrupted prior generated from the sliding window will eventually conflict with the global map and loop closure constraints, and degrade the overall accuracy.

One of our main contributions is maintaining the consistency between marginalization prior and global BA with the proposed relative marginalization. The key idea is to formulate the prior relative to the reference keyframe coordinate system instead of the global coordinate system. It is similar to the relative BA [29] for visual SLAM, in which all parameters are represented in the relative coordinate to avoid adjusting all parameters at loop closure. By contrast, we use the relative representation for marginalization. In addition, the relative representation is more complicated for VI-SLAM since the gravity direction becomes observable.

Before explaining details, we first recap the notations. $\mathbf{C}_i$ is the motion state of frame i, which comprises a pose $\mathbf{T}_i = (\mathbf{R}_i, \mathbf{p}_i)$ and an IMU state $\mathbf{M}_i = (\mathbf{v}_i, \mathbf{b}_i)$. We can represent the global pose $\mathbf{T}_i$ and the gravity direction in reference of frame i's closest keyframe k_0 as follows: ${}^{k_0}\mathbf{T}_i = \mathbf{T}_i \circ \mathbf{T}_{k_0}^{-1}$ and $\mathbf{g}_{k_0} = \mathbf{R}_{k_0}\mathbf{g}$. The velocity $\mathbf{v}_i$ is represented in its own reference as ${}^i\mathbf{v}_i = \mathbf{R}_i\mathbf{v}_i$. The motion state can be represented locally as $\mathbf{C}'_i = ({}^{k_0}\mathbf{T}_i, \mathbf{M}'_i)$ and $\mathbf{M}'_i = ({}^i\mathbf{v}_i, \mathbf{b}_i)$. Accordingly, $f_{ij}^{\text{vis}}(\mathbf{T}_i, \mathbf{T}_{s_j}, \rho_j)$ becomes

$$h_{ij}^{\text{vis}}({}^{k_0}\mathbf{T}_i, {}^{k_0}\mathbf{T}_{s_j}, \rho_j) = \pi({}^{k_0}\mathbf{T}_i \circ {}^{k_0}\mathbf{T}_{s_j}^{-1} \circ \frac{1}{\rho_j}\bar{\mathbf{x}}_{s_j j}) - \mathbf{x}_{ij}. \tag{15}$$

Marginalizing ${}^{k_0}\mathbf{T}_i$ will result in full correlation among $\{\rho_j | j \in \mathcal{V}_i\}$, invalidating the sparseness of BA. Inspired by [27], we maintain the sparseness by duplicating each ρ_j as $\rho'_j = \rho_j$, and discard all measurements except $\mathbf{x}_{ij}$. Then the duplicated points are marginalized out, producing

a Gaussian factor

$$h^{\text{vis}}_{\mathcal{V}_i}({}^{k_0}\mathbf{T}_i, \{{}^{k_0}\mathbf{T}_{s_j \neq k_0 | j \in \mathcal{V}_i}\}) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{\mathcal{V}_i}). \quad (16)$$

Similarly, $f^{\text{imu}}_{ij}(\mathbf{C}_i, \mathbf{C}_j)$ becomes

$$\begin{aligned} h^{\text{imu}}_{ij}(\mathbf{g}_{k_0}, \mathbf{C}'_i, \mathbf{C}'_j) &= ((\mathbf{e}'_r)^T, (\mathbf{e}'_v)^T, (\mathbf{e}'_p)^T, \mathbf{e}_b^T)^T \\ \mathbf{e}'_r &= \text{Log}((\text{Exp}(\Delta\mathbf{J}^r_{ij}(\mathbf{b}_i - \hat{\mathbf{b}}_i))\Delta\mathbf{R}_{ij})^T {}^{k_0}\mathbf{R}_j {}^{k_0}\mathbf{R}_i^T) \\ \mathbf{e}'_v &= {}^{k_0}\mathbf{R}_i({}^{k_0}\mathbf{R}_j^T {}^{j}\mathbf{v}_j - \mathbf{g}_{k_0}\Delta t_{ij}) - {}^{i}\mathbf{v}_i \\ &\quad - (\Delta\mathbf{v}_{ij} + \Delta\mathbf{J}^v_{ij}(\mathbf{b}_i - \hat{\mathbf{b}}_i)) \\ \mathbf{e}'_p &= {}^{k_0}\mathbf{R}_i({}^{k_0}\mathbf{p}_j - {}^{k_0}\mathbf{p}_i - \frac{1}{2}\mathbf{g}_{k_0}\Delta t_{ij}^2) - {}^{i}\mathbf{v}_i\Delta t_{ij} \\ &\quad - (\Delta\mathbf{p}_{ij} + \Delta\mathbf{J}^p_{ij}(\mathbf{b}_i - \hat{\mathbf{b}}_i)) \end{aligned} \quad (17)$$

We illustrate the relative marginalization process with detailed descriptions in Fig. 5. After marginalizing the earliest frame t_0, the process will result in a prior on the next frame $t_0 + 1$, denoted as

$$h^{\text{prior}}_{t_0+1}(\mathbf{g}_{k_0}, \mathbf{C}'_{t_0+1}, \{{}^{k_0}\mathbf{T}_{k \in \mathcal{K}_{t_0}}\}) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}^{\text{prior}}_{t_0+1}) \quad (18)$$

where $\mathcal{K}_{t_0}$ is the set of involved keyframes that is evolving as $\mathcal{K}_{t_0} = \mathcal{K}_{t_0-1} \cup \{s_j | j \in \mathcal{V}_{t_0}\} \backslash \{k_0\}$. Note that these relative representation of states is only used in marginalization. During optimization, states and priors need to be converted to the global frame. We convert the prior factor (18) into the global frame, denoted as

$$f^{\text{prior}}_{t_0+1}(\mathbf{C}_{t_0+1}) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}^{\text{prior}}_{t_0+1}). \quad (19)$$

Note that keyframe poses are only adjusted in global BA, thus eliminated from the prior factor for local BA. If the marginalized frame t_0 is a new keyframe, the marginalization process will submit a relative constraint to global BA (Fig. 5e), denoted as

$$h^{\text{rel}}_{t_0}(\mathbf{g}_{k_0}, \{{}^{k_0}\mathbf{T}_{k \in \mathcal{K}'_{t_0}}\}) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}^{\text{rel}}_{t_0}) \quad (20)$$

where $\mathcal{K}'_{t_0} = \mathcal{K}_{t_0-1} \cup \{t_0\}$. Similarly, the relative constraint is converted from the reference frame k_0 to the global frame, denoted as

$$f^{\text{rel}}_{t_0}(\{\mathbf{T}_{k \in \mathcal{L}_{t_0}}\}) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}^{\text{rel}}_{t_0}) \quad (21)$$

where $\mathcal{L}_{t_0} = \mathcal{K}'_{t_0} \cup \{k_0\}$.

6. Evaluation

To evaluate our proposed solver, we build a SLAM system that consists of the proposed solver, a frontend for visual measurements, and a loop closure detector. The frontend detects Harris features [13], establish inter-frame feature tracks using optical-flow [33], and match features across stereo frames using direct-matching [11]. Our loop closure detector stores bag-of-words features from keyframes for loop detection [12]. Once a loop closure is detected, we use the relative pose and covariance between the matched frames as a relative constraint in global BA.

We perform quantitative evaluation using EuRoC [3] dataset, and qualitative comparison against Google Tango in a number of challenging environments. The sliding window size is set to 50 in all experiments. Larger sliding window does not increase accuracy but decreases efficiency.

Configuration	RMSE (m)	LBA time (ms)	GBA time (ms)
Proposed	0.120792	2.45	12.90
w/o fix. linear.	0.117973	10.3	103.94
w/o ST-IBA	0.123548	7.03	-
w/o I-PCG	0.152073	-	12.91
w/o rel. marg.	0.179655	-	13.50

Table 1: Average RMSE and runtime of proposed methods for the whole EuRoC dataset. Fixing linearization points and ST-IBA significantly improves efficiency without sacrificing accuracy. I-PCG reduces RMSE due to better convergence, but not the computation time because we set a minimal iteration number. Relative marginalization improves both the accuracy as expected, and efficiency because the additional constraints accelerate convergence.

Seq.	Ours w/ loop	Ours w/o loop	OKVIS	SVO	iSAM2
MH_01	0.11	0.09	0.22	**0.06**	0.07
MH_02	0.08	**0.07**	0.16	0.08	0.11
MH_03	**0.05**	0.11	0.12	0.16	0.12
MH_04	**0.13**	0.16	0.18	-	0.16
MH_05	**0.11**	0.27	0.29	0.63	0.25
V1_01	0.07	0.05	**0.03**	0.06	0.07
V1_02	0.08	**0.05**	0.06	0.12	0.08
V1_03	**0.06**	0.11	0.12	0.21	0.12
V2_01	0.06	0.12	**0.05**	0.22	0.10
V2_02	**0.04**	0.09	0.07	0.16	0.13
V2_03	**0.11**	0.17	0.14	-	0.20
Avg	**0.08**	0.12	0.14	0.20	0.13

Table 2: Translation RMSE (m) with EuRoC dataset. Note that the spatial alignment of estimated and ground-truth trajectories is performed without scale adjustment for stereo algorithms. The results of other methods are generated from our own experiments based on their released codes, which are slightly different from the reported numbers in their papers.

6.1. Algorithm Validation

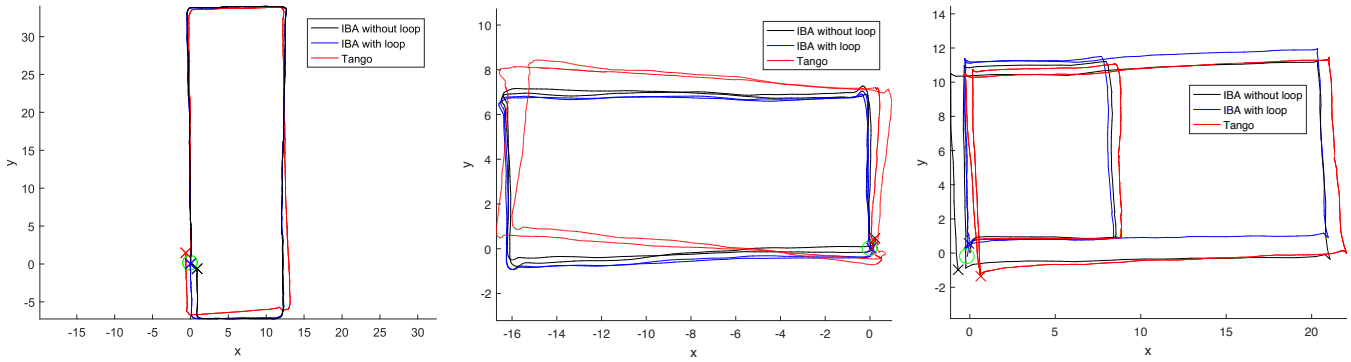
We validate each step of our algorithm introduced in each sub-section. Tab. 1 shows the performance of the full system, as well as disabling fixation of linearization point, ST-IBA, I-PCG and relative marginalization, respectively. All tests are run on a desktop PC with an i7 CPU @ 3.6GHz.

6.2. Localization Accuracy

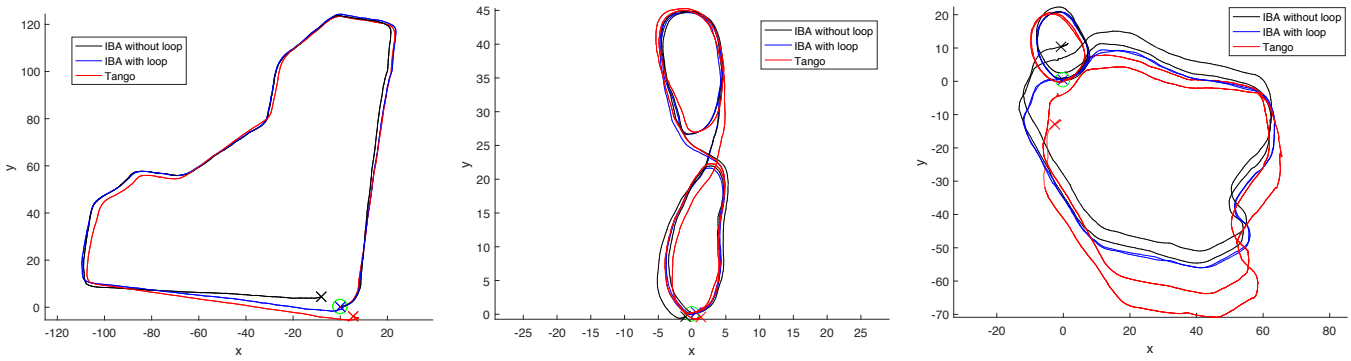
We compare the end-to-end accuracy of different stereo SLAM systems in Tab. 2. OKVIS [22] and SVO [11] are both visual inertial odometry (VIO). We run iSAM2 [18] by feeding the same feature tracks as ours, without providing loop constraints so it runs as a VIO. For a fair comparison, we show both our results with and without loop closure. Without loop closure, our system already achieves better localization accuracy than state-of-the-art alternatives since we use 50 frames in our local sliding window. With loop closure relative constraints provided to our solver, the RMSE considerably decreases for most sequences.

6.3. Solver Efficiency

The efficiency of our solver is a key contribution of this work. We measure the optimization time of different SLAM systems as shown in Tab. 3. We also measure the speed of our solver using an oct-core ARM CPU (A9 x 4 + A15 x 4). We configure the solver to run on A15 in single thread



(a) Indoor office (b) Indoor office (c) Indoor office



(d) Outdoor road (e) Outdoor road (f) Outdoor road

Figure 6: Trajectories of our system and Google tango. Ideally the final position of the trajectory should be identical to the initial position.

mode. The optimization time is 12.18ms, 78.14ms, and 193.72ms for local BA, global BA without and with loop, respectively. Our solver shows great potential to be applied to mobile and power-constraint applications.

6.4. Qualitatively Comparison with Google Tango

Google tango is a commercial device that is highly optimized for robust and accurate motion tracking. We compare our stereo SLAM system with a Tango Phab 2 as shown in Fig. 6. Without loop closure, our system shows comparative trajectories and more accurate scale than Tango. With loop closure, our system consistently outperforms Tango.

	Ours w/o loop	Ours w/ loop	OKVIS	iSAM2	ORB-SLAM
LBA	2.45	2.45	26.83	-	99
GBA	12.90	24.67	-	225.87	3515

Table 3: Comparison of runtime (ms) for local/global BA (LBA/GBA) with EuRoC dataset using an Intel i7 CPU. Multi-threading is disabled. The runtime does not include the frontend process (feature detection and matching). OKVIS [22] uses 5 keyframes plus 3 IMU frames in sliding window, whereas our system uses 50 frames and still achieves 10x speedup. Note that the optimization time of SVO [11] cannot be measured directly. We feed our frontend results to iSAM2 [18] to emulate the optimization time of SVO. iSAM2 is the solver used by SVO and also a state-of-the-arts incremental solver. We also measure the optimization time of ORB-SLAM [25] which uses g2o [21] as its solver. The runtime for LBA/GBA is approximately 40/140 times slower than ours. Note that ORB-SLAM requires more features for robust tracking, which is also a reason for the low efficiency. If we reduce the number of extracted features from default 1200 to 490, tracking fails on 3/11 sequences on EuRoC dataset.

7. Conclusion

In this paper, we have proposed a novel optimization algorithm for VI-SLAM that leverages the sparseness and the unique matrix structure for the optimization of sliding window based bundle adjustment. In addition, a novel relative marginalization is proposed to improve global consistency. Experiments demonstrate our approach can not only substantially accelerate the optimization process but also provide lower pose estimation error than other state-of-the-art SLAM approaches as well as a commercial system.

Acknowledgement

We would like to thank Bangbang Yang and Quanhan Qian for their kind help in producing results of OKVIS, SVO, iSAM2 and ORB-SLAM in Tab. 2 and 3. Hujun Bao is partially supported by 973 program of China (No. 2015CB352503), and Guofeng Zhang is partially supported by NSF of China (No. 61672457).

References

[1] S. Agarwal, K. Mierle, et al. Ceres solver, 2012.

[2] S. Agarwal, N. Snavely, S. M. Seitz, and R. Szeliski. Bundle adjustment in the large. In *European Conference on Computer Vision*, pages 29–42. Springer, 2010.

[3] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research*, 2016.

[4] M. Byröd and K. Åström. Conjugate gradient bundle adjustment. In *European Conference on Computer Vision*, pages 114–127. Springer, 2010.

[5] J. Civera, A. J. Davison, and J. M. Montiel. Inverse depth parametrization for monocular SLAM. *IEEE transactions on robotics*, 24(5):932–945, 2008.

[6] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007.

[7] T.-C. Dong-Si and A. I. Mourikis. Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis. In *International Conference on Robotics and Automation*, pages 5655–5662. IEEE, 2011.

[8] E. Eade and T. Drummond. Monocular SLAM as a graph of coalesced observations. In *International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[9] 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.

[10] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017.

[11] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza. SVO: Semidirect visual odometry for monocular and multicamera systems. *IEEE Transactions on Robotics*, 33(2):249–265, 2017.

[12] 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, October 2012.

[13] C. Harris and M. Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.

[14] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis. Camera-IMU-based localization: Observability analysis and consistency improvement. *The International Journal of Robotics Research*, 33(1):182–201, 2014.

[15] V. Ila, L. Polok, M. Solony, and K. Istenic. Fast incremental bundle adjustment with covariance recovery. In *International Conference on 3D Vision*, pages 4321–4330, 2017.

[16] V. Ila, L. Polok, M. Solony, and P. Svoboda. SLAM++ 1-a highly efficient and temporally scalable incremental slam framework. *The International Journal of Robotics Research*, 36(2):210–230, 2017.

[17] Y. Jeong, D. Nister, D. Steedly, R. Szeliski, and I.-S. Kweon. Pushing the envelope of modern methods for bundle adjustment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1605–1617, 2012.

[18] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the bayes tree. *The International Journal of Robotics Research*, 31(2):216–235, 2012.

[19] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

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

[21] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g 2 o: A general framework for graph optimization. In *International Conference on Robotics and Automation*, pages 3607–3613. IEEE, 2011.

[22] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[23] H. Liu, C. Li, G. Chen, G. Zhang, M. Kaess, and H. Bao. Robust keyframe-based dense SLAM with an RGB-D camera. *arXiv preprint arXiv:1711.05166*, 2017.

[24] 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*, pages 3565–3572. IEEE, 2007.

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

[26] R. Mur-Artal and J. D. Tardós. Visual-inertial monocular SLAM with map reuse. *Robotics and Automation Letters*, 2(2):796–803, 2017.

[27] E. D. Nerurkar, K. J. Wu, and S. I. Roumeliotis. C-KLAM: Constrained keyframe-based localization and mapping. In *International Conference on Robotics and Automation*, pages 3638–3643. IEEE, 2014.

[28] T. Qin, P. Li, and S. Shen. VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *arXiv preprint arXiv:1708.03852*, 2017.

[29] D. Sibley, C. Mei, I. D. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Robotics: Science and Systems*, volume 32, page 33, 2009.

[30] 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.

[31] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *Computer Vision and Pattern Recognition*, pages 3057–3064. IEEE, 2011.

[32] K. Wu, A. Ahmed, G. A. Georgiou, and S. I. Roumeliotis. A square root inverse filter for efficient vision-aided inertial navigation on mobile devices. In *Robotics: Science and Systems*, 2015.

[33] J. yves Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000.