

Article

A GTSAM-Based Monocular Visual-Inertial Odometry for Indoor UAVs: Robust Initialization and Single-Configuration Validation on EuRoC

Gabriel André Araújo ^{1,*} , Ruben Santos ² , João J. Martins ¹ , André Dias ¹  and José Almeida ¹ 

¹ Institute for Systems and Computer Engineering, Technology and Science, Campus da FEUP, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal; joao.p.pires@inesctec.pt (J.J.M.); andre.dias@inesctec.pt (A.D.); jose.m.almeida@inesctec.pt (J.A.)

² ISEP, Instituto Superior de Engenharia do Porto, Rua Dr. António Bernardino de Almeida 431, 4249-015 Porto, Portugal; 1211703@isep.ipp.pt

* Correspondence: gabriel.a.araujo@inesctec.pt

Highlights

What are the main findings?

- A tightly coupled monocular VIO, realized entirely on a GTSAM fixed-lag factor graph with inverse-depth landmarks, on-manifold IMU preintegration, and an online loop-closure pose graph, converges on all eleven EuRoC sequences under one fixed configuration per operating environment, including three that otherwise diverge by tens to thousands of meters.
- Driven by the same feature stream, this GTSAM estimator attains the better pure-odometry accuracy on nine of eleven sequences against two locally run baselines, VINS-Mono and PL-VINS, simultaneously, and on ten of eleven against VINS-Mono alone, a margin that a paired signed-rank test confirms against VINS-Mono ($p = 0.024$) and leaves unconfirmed against PL-VINS ($p = 0.054$).

What are the implications of the main findings?

- A hardened monocular bootstrap, a pre-bundle-adjust conditioning gate and a continuous-window initialization, is what carries the estimator through the near-zero-baseline indoor takeoff: replacing the continuous-window handoff with the conventional frozen single-state seed diverges ten of the eleven sequences.
- The result establishes a characterized point-only VIO baseline on GTSAM on one public benchmark and identifies First-Estimates-Jacobian marginalization within the factor-graph smoother as the route to a more consistent estimator.

Abstract

Reliable localization without GPS is a prerequisite for autonomous unmanned aerial vehicles (UAVs) operating inside warehouses, where a lightweight monocular camera paired with an inertial measurement unit (IMU) and onboard computer are the minimal sensing and processing an onboard platform can carry. This paper presents a tightly coupled monocular point-feature visual-inertial odometry (VIO) system for that setting, realized on a GTSAM fixed-lag factor graph with inverse-depth landmarks, on-manifold IMU preintegration, and an online loop-closure pose graph. The system is developed as the initial estimation stage of an autonomous stock-management UAV under development for indoor logistics warehouses. The decisive design element is the bootstrap: the metric, gravity-aligned initialization of a monocular estimator is well conditioned only under a translation-rich trajectory, a condition the near-zero-baseline pickup and takeoff transient



Academic Editors: Hector Gutierrez and Yang Liu

Received: 24 July 2026

Revised: 4 September 2026

Accepted: 6 September 2026

Published: 9 September 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

that opens every indoor flight violates. Building on the visual-inertial alignment of VINS-Mono, we harden this step with a pre-bundle-adjust conditioning gate and a continuous-window initialization that refines the whole bootstrap window inside the smoother instead of freezing a single seed. On all eleven EuRoC MAV sequences, indoor flight tests recorded onboard a micro air vehicle in an industrial hall and two instrumented rooms, one fixed configuration per operating environment converges on every sequence, including three that otherwise diverge by tens to thousands of meters, and, driven by the same feature stream as locally run VINS-Mono and PL-VINS baselines, attains the better pure-odometry accuracy on nine of the eleven, with ATE RMSE of 0.12–0.37 m on the Machine Hall, a margin a paired signed-rank test confirms against VINS-Mono and leaves unconfirmed against PL-VINS at this sample size. We identify the stock fixed-lag marginalization as the principal consistency limitation and outline First-Estimates-Jacobian marginalization as the route to a more consistent estimator, establishing a characterized point-only baseline on one public benchmark as the starting point for subsequent on-platform work.

Keywords: UAV; visual-inertial odometry; monocular; initialization; bootstrap; takeoff; GTSAM; fixed-lag smoother; GPS-denied navigation

1. Introduction

Estimating the six-degree-of-freedom (6-DoF) ego-motion of a mobile platform from onboard sensing is a foundational capability for autonomous navigation in GPS-denied environments such as warehouses, tunnels, and indoor inspection sites [1]. Among the available sensor suites, the pairing of a single camera with an inertial measurement unit (IMU) is especially attractive for aerial vehicles. It is lightweight, low cost, and low power, and the two modalities are strongly complementary. The IMU provides high-rate motion and observes gravity, which renders roll and pitch observable and fixes the otherwise ambiguous metric scale of a monocular camera. The camera, in turn, bounds the rapid drift of double-integrated inertial measurements [2,3]. This complementarity has made tightly coupled monocular visual-inertial odometry (VIO) one of the most widely deployed localization technologies for resource-constrained robots.

This work is carried out within the DRIVOLUTION project, a research and development effort toward an autonomous UAV for indoor logistics warehouses. The target mission is automated stock management, in which a small aerial vehicle flies the aisles of a warehouse, along shelving and racking, to read and audit inventory without human intervention. Such a warehouse is a canonical GPS-denied environment, enclosed, repetitive in structure, and without reliable satellite reception, so the vehicle must recover its own pose onboard from a lightweight sensor suite. A monocular camera paired with an IMU is the natural choice under the weight, power, and cost budgets of a small UAV, which places the accuracy and robustness of a monocular VIO on the critical path to autonomous flight. The present paper is the initial study of that estimation core, a GTSAM-based monocular point-feature VIO intended to run onboard the project's MANTIS aerial platform [4], characterized and validated here on the EuRoC MAV benchmark as a controlled first step before on-platform trials. Establishing a reproducible, well-understood estimator on public flight data is the foundation on which the project's subsequent warehouse deployment is built.

Estimator design for this problem evolved from recursive filtering—the extended Kalman filter applied jointly to platform pose and landmarks [5,6]—toward graph-based optimization once the field recognized that maximum-likelihood SLAM is a nonlinear least-squares problem over a sparse factor graph [1,7]. Today the field has converged on a tightly

coupled, optimization-based formulation in which camera and inertial measurements are fused as residuals in a single nonlinear least-squares problem over a sliding-window of recent states. VINS-Mono [3] is the canonical example of a sliding-window estimator that jointly optimises keyframe poses, velocities, and IMU biases against reprojection and preintegrated-inertial residuals, seeded by a dedicated visual-inertial initialization. Compared with the filtering approaches (MSCKF [8], OpenVINS [9]), the optimization formulation relinearizes the window each step and therefore better accommodates the nonlinearity of monocular geometry at the cost of bounded but non-trivial per-keyframe computation. Factor-graph libraries such as GTSAM [7], g2o [10], and Ceres [11] have made these back-ends engineering-ready.

The fragile step of a monocular point-feature estimator on a real mission is not online tracking but initialization. Scale, gravity direction, velocity, and gyroscope bias must all be recovered from a short bootstrap window, and the recovery is well conditioned only under a sufficiently excited, translation-rich trajectory [3]. The opening of an indoor UAV flight violates exactly this assumption. The platform is carried, set down, and lifted, and the EuRoC MAV sequences [12] reproduce it, beginning stationary and then picked up by hand, a rotation-dominant, near-zero-baseline phase whose multi-view geometry is degenerate. A bootstrap that accepts such a window inherits a mis-scaled or gravity-tilted seed. On a UAV that closes its control loops on the estimate, that seed is not a poor error statistic but a departure from controlled flight, and the online estimator can correct it only slowly and incompletely. Data association is the other classical fragility of a point-only pipeline: wrong correspondences and tracks that latch onto moving objects can bias an entire optimization window, and robust pipelines guard against this with front-end geometric verification, a soft robust cost (a Cauchy kernel in VINS-Mono [3]), and hard residual gating (ORB-SLAM2 [13], Kimera [14]). The present system incorporates these standard mechanisms (Section 2.2.7). What it hardens for the indoor-UAV setting, and where this paper provides contributions, is the bootstrap.

The system is a tightly coupled monocular point-feature VIO on a single factor graph, hardened at the bootstrap for the degenerate initialization of indoor flight. The estimator is a GTSAM batch fixed-lag smoother, a windowed nonlinear smoother that marginalizes states and landmarks out of the graph by timestamp, bounding per-frame cost for arbitrarily long runs while retaining the accuracy of windowed batch optimization. Inertial measurements enter through on-manifold preintegration [2] and points are inverse-depth landmarks [15]. The specific contributions are:

- **A GTSAM fixed-lag monocular point VIO.** A complete tightly coupled estimator (a custom inverse-depth reprojection factor over body poses, a timestamp-based fixed-lag smoother with marginalization-aware landmark anchoring, and an online 4-DoF loop-closure pose graph) realizing the optimization-based, sliding-window philosophy of VINS-Mono [3] on the GTSAM factor-graph library [7], which does not ship such an estimator.
- **A bootstrap hardened for degenerate takeoff.** Building on the linear visual-inertial alignment of VINS-Mono [3], the initialization adds a pre-bundle-adjust conditioning gate (which exposes an ill-conditioned, near-zero-baseline window by its reprojection residual before the optimizer can mask it by burying error in far landmark depths), and a continuous-window initialization that admits the entire bootstrap sub-graph into the smoother rather than freezing a single seed. These additions are what let the estimator survive the pickup-and-takeoff opening of an indoor flight. We quantify the influence of each separately, and find the continuous-window handoff to be the element that carries convergence. A velocity-consistency check on the recovered metric speed is retained in the pipeline as a safeguard for flight data outside this benchmark, but the

ablation of Section 3.2 finds it inert on all eleven EuRoC sequences, so it is described in Section 2.2.9 and not claimed as a contribution here.

- **Validation on EuRoC flight data under one configuration per environment.** Driven by the same feature stream as locally run VINS-Mono and PL-VINS baselines, one fixed configuration per operating environment converges on all eleven EuRoC MAV sequences (three of which otherwise diverge by tens to thousands of meters) and attains the better pure-odometry accuracy on nine of them, with ATE RMSE of 0.12–0.37 m on the Machine Hall. We further report the influence of each initialization and rejection parameter, the estimator’s covariance calibration, and its computational cost. The system is therefore a competitive point-only baseline that generalizes across the benchmark without per-sequence tuning.
- **A stated consistency limitation and its remedy.** Inverse-depth points use delayed initialization and newest-view anchoring with drop-and-reinitialize, tailored to the smoother’s timestamp-based marginalization. We identify the stock, non-first-estimate-consistent marginalization as the estimator’s principal limitation and outline the First-Estimates-Jacobian remedy as the route to a consistent estimator, the primary direction of the continuing project. We now measure this limitation instead of only asserting it: the median pose-NEES exceeds its 6-DoF expectation on 8 of 11 sequences, and the excess is almost entirely rotational, which is the signature the FEJ remedy targets.

Scope of this paper. Because the motivating application is a flying vehicle, it is worth stating plainly what is and is not delivered here, so that the results are read against the right expectation. This paper contributes an odometry component: it estimates 6-DoF ego-motion, together with velocity and IMU biases, from a single camera and an IMU, and it is intended to be consumed by a larger navigation stack rather than to constitute one. It builds no reusable prior map, performs no relocalization against a previous session, and contains no path planner, obstacle avoidance, or control law. The 4-DoF loop-closure pose graph runs alongside the odometry and never feeds back into it, so it is reported for completeness and is not a claimed contribution. The evaluation is correspondingly offline: recorded benchmark sequences are replayed in timestamp order on a desktop CPU, with the estimate outside any control loop, so that the estimator can be characterized in isolation and reproducibly. No real-time or embedded claim is made, and the computational figures of Section 3.4 should be read as a cost characterization and not as evidence of onboard feasibility. Deployment on the project’s MANTIS platform [4], with the estimate closing the control loop on embedded compute, is the decisive test and remains future work (see the Limitations and Future Directions section).

The remainder of this paper is organized as follows. Section 1.1 reviews monocular VIO, point-feature tracking and robust outlier rejection, factor-graph back-ends, and recent indoor and GPS-denied UAV systems, positioning the present system within that literature. Section 2 presents the approach in two parts: the theoretical background of factor-graph visual-inertial estimation, inverse-depth points, and fixed-lag marginalization (Section 2.1), followed by the concrete system implementation, its point front-end, inverse-depth factors, fixed-lag back-end, outlier rejection, and conditioning-aware initialization (Section 2.2). Section 3 evaluates the system on all eleven EuRoC MAV sequences against two locally run baselines, VINS-Mono and PL-VINS, without and with loop closure and on both back-ends. Section 4 interprets these results, declares the principal limitation of the design, and outlines future directions, and Section 5 concludes.

1.1. Related Work

1.1.1. Filtering-Based SLAM and the Road to Graph Optimization

The earliest SLAM systems posed the problem as recursive Bayesian estimation over a joint state of platform pose and all landmarks. The extended Kalman filter (EKF) formulation [5] maintained this state with a dense covariance matrix whose $\mathcal{O}(n^2)$ growth in the number of landmarks fundamentally limited scale [1]. MonoSLAM [6] carried EKF-SLAM to a single camera in real time, building a sparse map of natural landmarks, and exposed the parametrization difficulties peculiar to monocular vision. The inverse-depth parametrization [15] answered them by encoding a point through its bearing and inverse depth, enabling undelayed initialization of low-parallax and distant features with bounded uncertainty, the representation the present system adopts for its point landmarks. Beyond the EKF, the broader filtering family includes the unscented Kalman filter, which propagates the state through deterministic sigma points to avoid analytic Jacobians, and Rao-Blackwellized particle filters, which factor the posterior so that each particle carries its own map [16,17].

Filtering's recurring weakness is consistency. Linearizing each measurement once, about an estimate that is itself uncertain, injects spurious information along the unobservable directions of the problem and yields overconfident, inconsistent estimates [18]. Invariant-EKF formulations mitigate this by carrying the estimation error on the $SE(3)$ Lie group, where the error dynamics become partly autonomous and consistency is better preserved [19,20]. In the visual-inertial setting, the Multi-State Constraint Kalman Filter (MSCKF) [8] made filtering practical at scale by keeping a sliding window of camera poses and expressing each landmark as a multi-view constraint that is marginalized out and not retained in the state, yielding approximately constant-time updates. OpenVINS [9] is its modern, modular open-source descendant, with multi-camera and rolling-shutter modeling. These estimators remain the embedded-efficiency reference, but their single linearization per measurement is precisely what motivated the move to re-linearizing optimization.

The field's decisive shift was the recognition that, under Gaussian noise, maximum-likelihood SLAM is a nonlinear least-squares problem over a sparse factor graph [1,7], the formulation around which the field's current reference volume is organized [21]. Representing poses and landmarks as variable nodes and measurements as factors makes the sparsity of the information matrix explicit, so loop closures enter simply as additional edges and the whole graph is re-optimized, a structural advantage the EKF lacks, where a loop closure must be folded into a dense covariance [22]. The incremental smoothers iSAM [23] and iSAM2 [24] exploit this by updating only the affected portion of the factorization and relinearizing selectively, achieving near-constant amortised cost on chain-like trajectories. The present system inherits this graph-based, re-linearizing philosophy but bounds its cost through a fixed temporal lag instead of full incremental smoothing.

1.1.2. Optimization-Based Monocular Visual-Inertial Odometry

VINS-Mono [3] is the reference monocular optimization design: a tightly coupled sliding-window optimizer that jointly estimates poses, velocities, and IMU biases against visual reprojection, IMU preintegration, and (optionally) loop-closure residuals, with a hand-rolled marginalization factor that converts states leaving the window into a linear prior. A central enabler of this shift is on-manifold IMU preintegration [2], which compresses the high-rate inertial stream between consecutive keyframes into a single relative-motion factor whose covariance and bias-Jacobians allow re-use across changes in linearization point without re-integration. A second is a dedicated visual-inertial initialization: because metric scale is unobservable from a single camera, VINS-Mono recovers gyroscope bias,

gravity direction, velocity, and scale from a short bootstrap window, a recovery that is well conditioned only under a sufficiently excited, translation-rich trajectory [3,18].

The same tightly coupled philosophy underlies the visual-inertial mode of ORB-SLAM3 [25], which adds multi-map (Atlas) management and feature-based relocalization atop the point-feature SLAM lineage of ORB-SLAM2 [13]—binary ORB descriptors, a covisibility graph, and bag-of-words loop closure. Direct and semi-direct alternatives, DSO [26] and SVO [27], minimize photometric rather than reprojection error and are effective in low-texture scenes, but the feature-based, reprojection-residual formulation remains the dominant substrate for tightly coupled multi-feature fusion. The present system follows the optimization-based, sliding-window philosophy of VINS-Mono, but realises the window as a GTSAM fixed-lag smoother, retains a purely point-based front-end with conventional front-end and back-end outlier rejection, and concentrates on a bootstrap hardened against the degenerate pickup windows that violate the excitation assumption above.

1.1.3. Point-Feature Tracking and Robust Outlier Rejection

Point features are the dominant primitive of monocular visual-inertial estimation: each corner observation supplies two image constraints and, once triangulated, ties consecutive poses through reprojection. Two front-end families recur. Descriptor-based trackers detect repeatable keypoints and associate them by matching binary descriptors, the ORB detector-descriptor [28] of the ORB-SLAM lineage [13,25] being the canonical choice for its speed and rotation invariance [29]. Sparse-optical-flow trackers instead detect Shi-Tomasi corners once and follow them frame-to-frame with Kanade–Lucas–Tomasi (KLT) tracking, the design VINS-Mono [3] adopts because it avoids per-frame descriptor extraction and yields dense short-baseline tracks well matched to a high-rate optimizer. The present system follows the latter: KLT-tracked corners with forward-backward verification.

The parametrization of these points is itself a design axis. A Euclidean (x, y, z) encoding is ill-conditioned for the low-parallax and distant features that dominate monocular tracking, where depth uncertainty is near-singular and strongly non-Gaussian. The inverse-depth parametrization [15] instead encodes a point by its bearing and inverse range in an anchor frame, bounding depth uncertainty and gracefully representing points near infinity, the encoding the present system adopts, coupled to a marginalization-aware anchoring policy (newest-view anchoring with drop-and-reinitialize) dictated by the timestamp-based marginalization of its fixed-lag smoother.

The accuracy of a point-only estimator is governed less by the choice of detector than by how aggressively it rejects wrong data associations, because a single mistracked correspondence or moving-object point, admitted as a hard reprojection constraint, can bias the entire window. Robustness is therefore layered across the pipeline. At the front-end, geometric verification discards inconsistent matches, forward-backward optical-flow consistency and epipolar RANSAC on the fundamental or essential matrix [30], which can be reduced to a two-point solver when the IMU-predicted rotation is used as a prior. In the back-end, two complementary philosophies coexist. Soft robust cost functions, the Huber and Cauchy M-estimators, down-weight large residuals continuously. VINS-Mono [3] relies on a Cauchy loss over its reprojection factors. Hard rejection instead removes an observation outright once its residual or Mahalanobis distance exceeds a gate: ORB-SLAM2 [13] applies per-observation chi-square tests, and Kimera [14] pairs a two-stage geometric outlier-rejection front-end with a robust back-end. These gates interact subtly with estimator consistency, since down-weighting or removing constraints along the unobservable directions of monocular VIO can silently degrade the solution [18,31]. Recent point-only systems push robustness along orthogonal axes, SP-VIO [32] with a pose-only visual description that removes explicit landmark states, and SL-SLAM [33] with

learned features for more repeatable association, yet the underlying rejection machinery is typically assembled piecemeal from these ingredients. A parallel and more common response to unreliable point association is to change the primitive rather than the gate. RWT-SLAM [34] substitutes learned dense matching for sparse corners in weakly textured scenes, while PLPF-VSLAM [35] and the post-integration point-line design of Liu et al. [36] add line and plane primitives that remain observable where corners fail, the former fusing them adaptively for indoor operation. Both routes buy robustness with a richer front-end. The present work deliberately takes neither, holding the primitive fixed at points so that the contribution of the estimator and of each rejection layer can be attributed without a confounding change in measurement type. The system reported here instead composes a hard reprojection reject, a soft Huber weighting, and a conditioning-aware initialization gate into a single point pipeline, and quantifies the contribution of each layer.

1.1.4. Factor-Graph Back-Ends and Fixed-Lag Smoothing

A factor-graph back-end lets point-based VIO-visual reprojection and preintegrated-inertial residuals share a single optimization substrate, with landmark sparsity exploited through variable elimination [1,7]. The incremental smoothers of the preceding section are realized in GTSAM [7], while the competing libraries g2o [10] and Ceres [11] back ORB-SLAM and VINS-Mono [3]. LIO-SAM [37] likewise builds its LiDAR-inertial graph on GTSAM's iSAM2. The choice of library is not neutral: substituting a GTSAM factor-graph optimizer for the g2o back-end of ORB-SLAM3 has been reported to accelerate the tracking thread by 13–15% on EuRoC while lowering absolute trajectory error [38], which is part of the motivation for building the present estimator on GTSAM.

For bounded-latency operation over unbounded trajectories, full smoothing is replaced by a fixed-lag smoother that retains only a recent time window and marginalizes older states into a dense linear prior [39]. This is the optimization-side analogue of the sliding window and the design point of the present system, which uses the GTSAM batch fixed-lag smoother with a timestamp-based lag: every state and landmark older than the window horizon is Schur-marginalized out. Marginalization, however, interacts subtly with observability. Because relinearizing a marginalized constraint at a moved estimate can inject spurious information along the unobservable directions of monocular VIO (global position and yaw), consistent estimators apply First-Estimates Jacobians (FEJs) to freeze the linearization point of marginalized variables [3,18]. The trade-off between fixed-lag bounded cost and FEJ-consistent marginalization is directly relevant to the residual yaw drift of any monocular VIO without loop closure and frames the position of the system reported here: optimization-based and point-only like VINS-Mono but realised on a GTSAM fixed-lag smoother whose point pipeline layers hard and soft outlier rejection and whose initialization is hardened against the degenerate bootstrap windows of standard benchmarks.

1.1.5. Recent Indoor and GPS-Denied UAV Systems

The application this estimator is built for, i.e., autonomous indoor inventory flight, i.e., has itself been the subject of recent work, and it is worth separating what that literature has settled from what it has left open. Aerial inventory platforms are now demonstrated end to end: the UJI Aerial Librarian [40] performs visual library inventory and book localization from a quadcopter, Alajami et al. [41] fly RFID-based dynamic inventories in mapless indoor environments using stigmergy, and Stanko et al. [42] automate warehouse inspection with an autonomous UAV. What these systems share is that localization is solved by assumption rather than by estimation, through external infrastructure, fiducial markers, or an operator-supplied prior map. Reviews of indoor positioning for mobile robots [43] record the same division: the infrastructure-based families deliver the accuracy, and the infrastructure-free

ones deliver the deployability. The present work sits on the infrastructure-free side, which is why the quality of the onboard estimate, and of its covariance, is the object of study and not an implementation detail.

On the estimation side, recent surveys of visual and visual-inertial SLAM [44,45] and the field's current reference volume [21] confirm that the tightly coupled, optimization-based monocular design remains the reference for this weight and power class, with learning-based components entering at the front-end instead of displacing the geometric back-end. Work specific to indoor drones is consistent with the emphasis of this paper: Firmansyah et al. [46] report that monocular ORB-SLAM for indoor drone localization requires inertial aiding precisely where the visual geometry degenerates, which is the failure mode the bootstrap studied here is hardened against. On feasibility, Matthee et al. [47] profile ORB-SLAM3 on Jetson TX2 and Raspberry Pi class hardware and find that real-time execution on embedded platforms remains the binding constraint and not accuracy, which is the reason the computational cost of the present estimator is measured (Section 3.4) and the reason no embedded claim is made from a desktop measurement.

2. Materials and Methods

2.1. Theoretical Concepts

This section sets out the theoretical background needed to understand the system, drawn from the visual-inertial SLAM literature. Section 2.1.1 casts tightly coupled visual-inertial estimation as maximum-a-posteriori (MAP) inference on a factor graph and shows why this reduces to a sparse nonlinear least-squares problem. Section 2.1.2 then reviews the inverse-depth landmark parametrization used for point features in monocular visual SLAM. Throughout, $\mathcal{W}$, $\mathcal{B}$, and $\mathcal{C}$ denote the world, body (IMU), and camera frames. The pose $T_{WB} \in SE(3)$ is the body pose in the world, and the constant extrinsic T_{BC} places the camera in the body, so the camera pose is $T_{WC} = T_{WB} T_{BC}$. The skew-symmetric matrix of a vector $\mathbf{t}$ is written $[\mathbf{t}]_{\times}$. Throughout the paper, bold lowercase symbols denote column vectors ($\mathbf{v}_i$, $\mathbf{b}_i$, $\mathbf{z}_k$, δ , λ_j), plain italic symbols denote matrices, rigid-body poses, states and scalars (T_{WB} , R_{WB} , H_k , X_i , Λ , λ), and calligraphic symbols denote sets and reference frames ($\mathcal{X}$, $\mathcal{Z}$, $\mathcal{W}$). The typewriter face is reserved for literal identifiers taken from the implementation, such as the GTSAM symbol keys of Section 2.2. Every symbol carries the same face in the text, in the equations and in the figures.

2.1.1. Factor Graphs

Let the full set of unknowns be

$$\mathcal{X} = \{T_{WB}^i, \mathbf{v}_i, \mathbf{b}_i, \lambda_j\}, \quad (1)$$

collecting the body poses, velocities, and IMU biases of every keyframe together with the inverse-depth points λ_j defined below. Given the measurement set $\mathcal{Z} = \{\mathbf{z}_k\}$ and assuming conditionally independent measurements, the posterior factorizes into a prior and one term per measurement,

$$p(\mathcal{X} | \mathcal{Z}) \propto p(\mathcal{X}_0) \prod_k p(\mathbf{z}_k | \mathcal{X}_k), \quad (2)$$

where $\mathcal{X}_k \subseteq \mathcal{X}$ is the (usually small) subset of variables that measurement k involves.

Written this way the factorization is measurement-wise. Its temporal structure, however, is what the estimator exploits, and stating it explicitly also makes precise how the state evolves from one keyframe to the next.

The generative model behind (2) is a Bayes network in which the navigation states form a **first-order Markov chain** and each measurement depends only on the variables it touches. This is the standard decomposition of the SLAM joint density: Dellaert and Kaess [7] separate it into four qualitatively distinct groups of factors, of which the first is “a Markov chain $p(x_1)p(x_2|x_1)p(x_3|x_2)$ on the poses”, the remaining groups being landmark priors and the measurement likelihoods. Writing that decomposition for the present system, with the inertial terms separated from the visual ones,

$$p(\mathcal{X} | \mathcal{Z}) \propto \underbrace{p(\mathcal{X}_0)}_{\text{gauge prior}} \underbrace{\prod_{i=0}^{N-1} p(X_{i+1} | X_i, \mathcal{I}_{i,i+1})}_{\text{Markov chain on the navigation states}} \underbrace{\prod_{(i,j) \in \mathcal{O}} p(\mathbf{z}_{ij} | X_i, \lambda_j)}_{\text{observation likelihoods}}, \quad (3)$$

where $X_i = (T_{WB}^i, \mathbf{v}_i, \mathbf{b}_i)$ is the navigation state of keyframe i , $\mathcal{I}_{i,i+1}$ the inertial measurements accumulated between keyframes i and $i+1$, and $\mathcal{O}$ the set of landmark observations.

The factor graph is obtained from this Bayes network by conditioning on the measurements $\mathcal{Z}$, which are observed and therefore not unknowns (Section 1.7 in [7]): every conditional density becomes a factor over the unknowns alone, and the directionality of the Bayes network is dropped. The Markov-chain terms survive that conversion unchanged, as binary factors between consecutive states. The factor graph optimized here is therefore **the Markov model of the trajectory with the measurements conditioned out**, not an alternative to it, and the transition density $p(X_{i+1} | X_i, \mathcal{I}_{i,i+1})$ is realized concretely by the preintegrated inertial factor developed below, which is the transition kernel of the chain. Equivalently, the estimator is a hidden Markov model whose transition and observation densities are both Gaussian on the manifold [16], with the distinction that it is solved by smoothing over a window of states rather than by recursive filtering of the most recent one.

Two conditional-independence properties follow from (3), and both are load-bearing. First, X_{i+1} is conditionally independent of $X_0, \dots, X_{i-1}$ given X_i and the inertial measurements between them. Second, given the states, the landmarks are conditionally independent of one another, so a landmark couples keyframes only through the states that observe it. It is this sparsity, and not the choice of solver, that makes the variable elimination and bounded-window marginalization of Section 2.1.3 possible. Figure 1 is the graphical statement of this factorization. A factor graph $G = (\mathcal{F}, \Theta, \mathcal{E})$ is the bipartite graph that makes this factorization explicit: each variable in $\mathcal{X}$ is a node $\theta_i \in \Theta$, and each conditional $p(\mathbf{z}_k | \mathcal{X}_k)$ becomes a factor $\phi_k \in \mathcal{F}$ joined by edges $\mathcal{E}$ only to the variables it touches [7]. The MAP estimate is the assignment that maximizes the product of factors,

$$\mathcal{X}^* = \arg \max_{\mathcal{X}} \prod_k \phi_k(\mathcal{X}_k). \quad (4)$$

Every measurement model takes the form $\mathbf{z}_k = h_k(\mathcal{X}_k) \boxplus \boldsymbol{\eta}_k$ with zero-mean Gaussian noise $\boldsymbol{\eta}_k \sim \mathcal{N}(\mathbf{0}, \Sigma_k)$, so each factor is $\phi_k(\mathcal{X}_k) \propto \exp(-\frac{1}{2} \|h_k(\mathcal{X}_k) \boxminus \mathbf{z}_k\|_{\Sigma_k}^2)$, where $\|\mathbf{e}\|_{\Sigma}^2 = \mathbf{e}^T \Sigma^{-1} \mathbf{e}$ is the squared Mahalanobis norm. Taking the negative logarithm of (2) turns the product of exponentials into a sum and the MAP problem into the sparse nonlinear least-squares problem

$$\mathcal{X}^* = \arg \min_{\mathcal{X}} \sum_k \|h_k(\mathcal{X}_k) \boxminus \mathbf{z}_k\|_{\Sigma_k}^2. \quad (5)$$

Because the unknowns live on manifolds (the $SE(3)$ poses in particular), the optimizer operates in the tangent space: an increment δ is mapped back onto the manifold by a retraction $\mathcal{X} \boxplus \delta$, and each residual $\mathbf{r}_k = h_k \boxminus \mathbf{z}_k$ is linearized about the current estimate, $\mathbf{r}_k(\mathcal{X} \boxplus \delta) \approx \mathbf{r}_k(\mathcal{X}) + H_k \delta$, with H_k the Jacobian. How the resulting nonlinear least-squares problem (5) is minimized, by iterated linearization, sparse factorization, in-

cremental smoothing, and the marginalization that bounds the window, is the subject of Section 2.1.3. The factors that populate the graph are of three kinds: the visual reprojection factors detailed in Section 2.1.2, the prior factors that fix the otherwise unobservable gauge of the monocular problem, and the inertial preintegrated factor developed next.

Figure 1 shows the resulting structure, first as the graph the estimator would build over an unbounded trajectory and then as the bounded window it actually maintains.

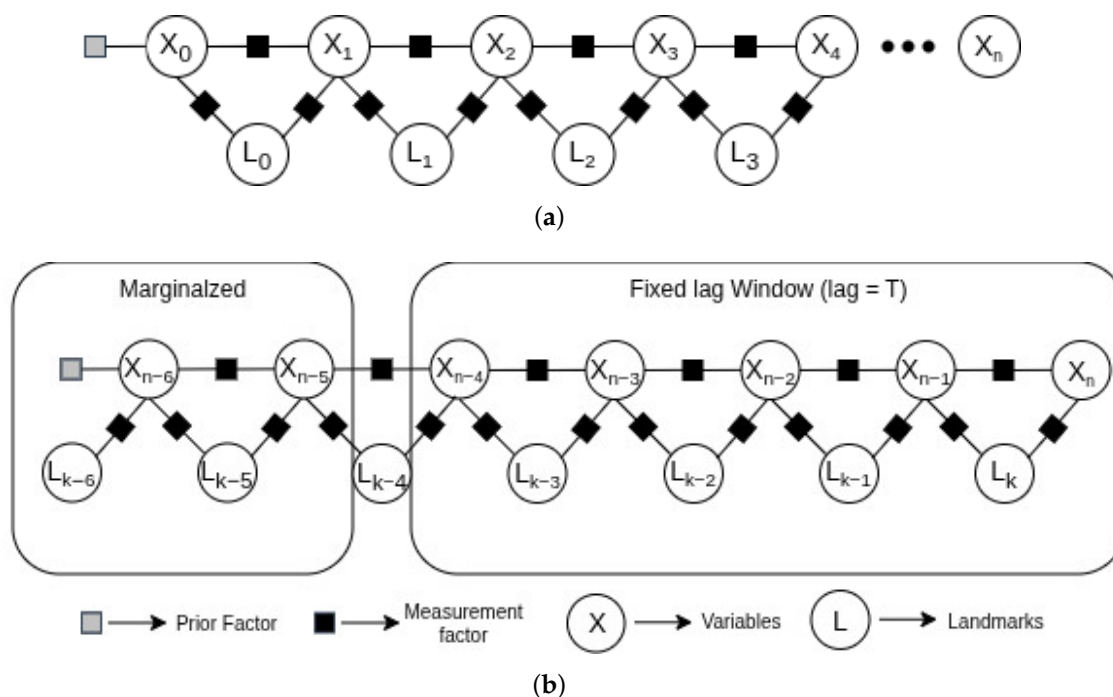


Figure 1. Structure of the estimator's factor graph. (a) The graph over the full trajectory. The ellipsis denotes the keyframes between X_4 and the final keyframe X_n , which repeat the same pattern of states, landmarks and factors and are omitted only to keep the drawing legible. Circles are variables: a navigation state X_i carries the body pose, velocity and IMU bias of keyframe i , and a landmark L_j carries one inverse-depth point. Black markers are measurement factors, and their position says which measurement: a factor joining two consecutive states is a preintegrated inertial measurement, while a factor joining a state to a landmark is a projection of that landmark into that keyframe, one per observation, so a landmark seen from several keyframes couples those keyframes. The grey square is the prior that fixes the otherwise unobservable gauge of the monocular problem. (b) The same graph under a fixed lag: only the variables inside the window of length T are optimized, while everything older is Schur-marginalized out (Section 2.1.3, Equations (12) and (13)). The marginalization does not discard those constraints, it converts them into a dense linear prior on the oldest retained variables, which is what bounds per-keyframe cost independently of trajectory length and is also the source of the consistency limitation quantified in Section 3.3. Note that a landmark whose anchoring keyframe leaves the window (L_{k-4}) must be re-anchored or dropped, the bookkeeping of which is described in Section 2.2.6.

IMU preintegration. A strapdown IMU reports angular velocity $\tilde{\omega}$ and specific force $\tilde{\mathbf{a}}$ at high rate, each corrupted by a slowly varying bias and white noise,

$$\tilde{\omega} = \omega + \mathbf{b}_g + \eta_g, \quad \tilde{\mathbf{a}} = R_{WB}^\top (\mathbf{a}_W - \mathbf{g}) + \mathbf{b}_a + \eta_a, \quad (6)$$

where $\mathbf{g}$ is the gravity vector and $\mathbf{a}_W$ the body acceleration in the world, driving the kinematics $\dot{R}_{WB} = R_{WB}[\omega]_\times$, $\dot{\mathbf{v}} = \mathbf{a}_W$, $\dot{\mathbf{p}} = \mathbf{v}$. Integrating these directly between two keyframes i and j ties the result to the (unknown, still-changing) state at i , so every relinearization of X_i would force a costly re-integration of all the intervening samples. Preintegration [2]

avoids this by accumulating the inertial increments in the body frame of i , independently of the absolute pose and velocity at i :

$$\begin{aligned}\Delta R_{ij} &= \prod_{k=i}^{j-1} \text{Exp}((\tilde{\omega}_k - \mathbf{b}_g) \Delta t), \\ \Delta \mathbf{v}_{ij} &= \sum_{k=i}^{j-1} \Delta R_{ik} (\tilde{\mathbf{a}}_k - \mathbf{b}_a) \Delta t, \\ \Delta \mathbf{p}_{ij} &= \sum_{k=i}^{j-1} [\Delta \mathbf{v}_{ik} \Delta t + \frac{1}{2} \Delta R_{ik} (\tilde{\mathbf{a}}_k - \mathbf{b}_a) \Delta t^2],\end{aligned}\quad (7)$$

where $\text{Exp}(\cdot)$ is the $SO(3)$ exponential map. The increments $(\Delta R_{ij}, \Delta \mathbf{v}_{ij}, \Delta \mathbf{p}_{ij})$ are evaluated once, at a fixed bias estimate, together with the covariance Σ_{ij} obtained by propagating the linearized IMU noise of (6). A first-order expansion in the bias then corrects them for an updated estimate $\mathbf{b} + \delta \mathbf{b}$ without re-integration, e.g., $\Delta R_{ij}(\mathbf{b}_g + \delta \mathbf{b}_g) \approx \Delta R_{ij}(\mathbf{b}_g) \text{Exp}(\frac{\partial \Delta R_{ij}}{\partial \mathbf{b}_g} \delta \mathbf{b}_g)$, which is exactly what makes the factor cheap to reuse as the estimate is relinearized. The factor couples the two navigation states through the residual

$$\mathbf{r}_{ij} = \begin{bmatrix} \text{Log}(\Delta R_{ij}^\top R_{WB_i}^\top R_{WB_j}) \\ R_{WB_i}^\top (\mathbf{v}_j - \mathbf{v}_i - \mathbf{g} \Delta t_{ij}) - \Delta \mathbf{v}_{ij} \\ R_{WB_i}^\top (\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i \Delta t_{ij} - \frac{1}{2} \mathbf{g} \Delta t_{ij}^2) - \Delta \mathbf{p}_{ij} \end{bmatrix}, \quad (8)$$

weighted by Σ_{ij} , plus a random-walk term $\mathbf{b}_j - \mathbf{b}_i$ penalizing bias drift. GTSAM realizes precisely this construction as a combined preintegrated IMU factor over (X_i, X_j) , so the entire inter-keyframe inertial stream enters the graph as a single 15-dimensional residual block.

2.1.2. Inverse Depth Points

A Euclidean point $\mathbf{p}^W \in \mathbb{R}^3$ is a poor variable for a monocular estimator: features at large or unknown depth have near-singular, strongly non-Gaussian uncertainty in (x, y, z) , and a point first seen with negligible parallax cannot be initialized at all without an arbitrary depth guess. The inverse-depth parametrization [15] removes both problems by encoding a point relative to the camera that first observed it, its anchor, through a bearing and an inverse range.

In this parametrization a point is encoded by the six-vector $\mathbf{y} = (x_0, y_0, z_0, \theta, \phi, \rho)$ [15]: the optical center $\mathbf{c}_0 = (x_0, y_0, z_0)^\top$ of the camera from which the point was first observed, the azimuth θ and elevation ϕ of the ray pointing toward it, and the inverse depth ρ along that ray. The corresponding Euclidean point is

$$\mathbf{P}(\mathbf{y}) = \mathbf{c}_0 + \frac{1}{\rho} \mathbf{m}(\theta, \phi), \quad \mathbf{m}(\theta, \phi) = (\cos \phi \sin \theta, \sin \phi, \cos \phi \cos \theta)^\top, \quad (9)$$

with $\mathbf{m}$ a unit bearing vector. Encoding the inverse depth ρ in place of the depth itself keeps the representation finite and well conditioned as the point recedes toward infinity ($\rho \rightarrow 0$), where a Euclidean encoding would diverge.

A landmark is linked to a camera by reprojection. For a camera with pose T_{WC} and intrinsics K , the predicted projection of $\mathbf{P}(\mathbf{y})$ is $\hat{\mathbf{z}} = \pi(K, T_{WC}^{-1} \mathbf{P}(\mathbf{y}))$, where $\pi(K, \cdot)$ is the pinhole projection, and the visual residual is the pixel error $\mathbf{r} = \hat{\mathbf{z}} - \mathbf{z}$ against the observed corner $\mathbf{z}$. Referring the parametrization to the first observing camera has two consequences that are valuable for monocular estimation: the bearing (θ, ϕ) is well determined from a single view, so only the inverse depth ρ remains to be refined as parallax accumulates, and a feature can be initialized immediately, with bounded uncertainty, instead of being delayed until its depth is well constrained, including features near infinity and those seen with little parallax, which a Euclidean (x, y, z) encoding represents poorly [6,15].

2.1.3. Optimization and Incremental Smoothing

The factor graph of Section 2.1.1 defines the nonlinear least-squares objective (5). This subsection reviews how that objective is minimized, and how the GTSAM library [7] that realizes these methods solves it efficiently and incrementally.

Gauss–Newton and Levenberg–Marquardt. Because (5) is nonconvex, it is minimized iteratively. At the current estimate each residual is linearized on the manifold, and stacking the whitened Jacobians and residuals, $A = [\dots; \Sigma_k^{-1/2} H_k; \dots]$ and $\mathbf{b} = -[\dots; \Sigma_k^{-1/2} \mathbf{r}_k; \dots]$, reduces each iteration to the linear least-squares problem $\min_{\delta} \|A\delta - \mathbf{b}\|^2$. The Gauss–Newton step solves the corresponding normal equations

$$\Lambda \delta^* = A^\top \mathbf{b}, \quad \Lambda = A^\top A = \sum_k H_k^\top \Sigma_k^{-1} H_k, \quad (10)$$

after which the estimate is updated by retraction, $\mathcal{X} \leftarrow \mathcal{X} \boxplus \delta^*$, and the linearization is repeated to convergence. Gauss–Newton converges quickly near the optimum but can diverge when the linear model is a poor fit far from it. Levenberg–Marquardt guards against this with an adaptive damping term,

$$(\Lambda + \lambda \operatorname{diag}(\Lambda)) \delta^* = A^\top \mathbf{b}, \quad (11)$$

where λ is raised when a trial step fails to decrease the cost, shrinking the step toward a robust gradient-descent direction, and lowered when it succeeds, recovering the fast Gauss–Newton step. GTSAM provides both, with Levenberg–Marquardt the default batch optimizer.

Variable elimination: QR and Cholesky. The linear system (10) is solved by sparse matrix factorization in one of two algebraically equivalent ways. The Cholesky route forms the information matrix Λ explicitly and factors it as $\Lambda = R^\top R$ with R upper-triangular, then recovers δ^* by forward and back substitution. The QR route factors the Jacobian directly, $A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$ with Q orthogonal, giving $R\delta^* = Q^\top \mathbf{b}$ without ever forming Λ . Because it operates on the square root R rather than on $\Lambda = A^\top A$, it avoids squaring the condition number and is the numerically preferable choice, the square-root view of smoothing-and-mapping [7]. In either case the triangular factor is computed not densely but by variable elimination: eliminating a variable $\mathbf{x}_j$ gathers every factor incident on it into one dense subproblem, factors $\mathbf{x}_j$ out as a conditional $p(\mathbf{x}_j \mid \mathcal{S}_j)$ on its separator $\mathcal{S}_j$ (the remaining variables that share those factors), and leaves a new factor on $\mathcal{S}_j$ alone. Sweeping the elimination over the graph produces the sparse R , whose cost is governed by its fill-in. The elimination ordering, typically an approximate-minimum-degree heuristic such as COLAMD, is chosen to minimize that fill-in, and for the chain-like trajectories of odometry it is near-linear, which is what makes batch smoothing computationally efficient on such trajectories. Eliminating a variable and marginalizing it are the same algebraic operation, the factor induced on $\mathcal{S}_j$ being a Schur complement, which is the link to the fixed-lag marginalization below.

Incremental smoothing: iSAM and iSAM2. Re-eliminating the entire graph at every new keyframe is wasteful, because a new measurement perturbs only a small part of the factorization. iSAM [23] exploits this by updating the triangular factor R incrementally with Givens rotations as new factors arrive, periodically re-ordering the variables and re-linearizing to keep the fill-in and the linearization point current. iSAM2 [24] recasts the factorization as a Bayes tree, a clique tree equivalent to R , and edits only the cliques touched by the incoming factors, while fluid relinearization re-linearizes a variable only once its estimate has moved past a threshold. Together these give a near-constant amortized cost

per update while remaining algebraically equivalent to full batch smoothing, which is why iSAM2 underlies many modern smoothing back-ends.

Fixed-lag smoothing and marginalization. Even with incremental updates, full smoothing keeps every state, so the graph grows without bound. A fixed-lag smoother retains only the variables inside a temporal window and removes the rest, but, rather than simply discarding them, it marginalizes them, preserving their accumulated information. Partition the active variables into a block $\mathbf{x}_m$ to be marginalized (those whose timestamp has fallen behind the lag horizon, $t < t_{\text{now}} - \tau$) and a retained block $\mathbf{x}_k$, and collect the factors touching $\mathbf{x}_m$ into a Gaussian with

$$\Lambda = \begin{bmatrix} \Lambda_{mm} & \Lambda_{mk} \\ \Lambda_{km} & \Lambda_{kk} \end{bmatrix}, \quad \boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_m \\ \boldsymbol{\eta}_k \end{bmatrix}. \quad (12)$$

Integrating $\mathbf{x}_m$ out of this Gaussian leaves a reduced quadratic on $\mathbf{x}_k$ whose information is the Schur complement

$$\Lambda_{kk}^* = \Lambda_{kk} - \Lambda_{km} \Lambda_{mm}^{-1} \Lambda_{mk}, \quad \boldsymbol{\eta}_k^* = \boldsymbol{\eta}_k - \Lambda_{km} \Lambda_{mm}^{-1} \boldsymbol{\eta}_m. \quad (13)$$

The pair $(\Lambda_{kk}^*, \boldsymbol{\eta}_k^*)$ defines a single dense marginalization prior that replaces all the eliminated factors, so the window thereafter optimizes only $\mathbf{x}_k$ together with this prior, at constant per-step cost. The same reduction can be carried out directly on the square-root factor R by a QR update that never forms Λ_{mm} , algebraically equivalent to (13) but numerically far better behaved, which matters acutely in single precision [39].

Marginalization is not without cost to consistency. The prior is dense, it couples every variable that shared a factor with $\mathbf{x}_m$, and, being a fixed Gaussian, it freezes the linearization point of the marginalized constraint. Re-evaluating the rest of the graph at a later, displaced estimate while this prior stays anchored injects spurious information along the directions a monocular system cannot observe, namely global position and yaw, which is why consistent fixed-lag estimators apply First-Estimates Jacobians, holding each variable's Jacobian at the value it had when first marginalized [3]. This bounded-cost-versus-consistency trade-off is intrinsic to fixed-lag smoothing and shapes the behavior of any monocular VIO operated without loop closure.

2.2. System Implementation

The methods of Section 2.1 are realized as a single tightly coupled estimator built on the GTSAM factor-graph library [7]. This section details the concrete point pipeline: the front-end that produces feature observations, the inverse-depth factors that bind them into the graph, the fixed-lag smoother that optimizes it, the outlier rejection that guards it, and the conditioning-aware bootstrap that seeds it. Throughout, the navigation state X_i of Section 2.1.1 is realized in GTSAM by three separate symbol keys: a pose key X_i carrying $T_{WB}^i \in SE(3)$, a velocity key V_i carrying the world-frame velocity $\mathbf{v}_i \in \mathbb{R}^3$, and a bias key B_i carrying the 6-vector IMU bias $\mathbf{b}_i = (\mathbf{b}_a, \mathbf{b}_g)$; each inverse-depth point λ_j is carried by a landmark key L_j . The typewriter face marks these as the implementation's literal symbol keys and distinguishes them from the mathematical variables they hold. The camera-in-body extrinsic T_{BC} is a fixed calibration constant.

2.2.1. Pipeline Overview

Figure 2 shows the arrangement of the running system. The system executes as a sequence of phases rather than a single interleaved loop, a structure chosen so that the front-end, the bootstrap, and the smoother can each be reasoned about and tested in isolation. (A) A front-end pre-pass produces, for every frame, the set of point observations together

with the inter-frame inertial stream. (B) A bootstrap loop consumes the opening frames until a well-conditioned window yields a metric, gravity-aligned seed. (C) That entire window is handed to the smoother as an initial sub-graph. Online processing then resumes from the last bootstrap state, integrating IMU on every frame but instantiating a graph state only on keyframes. The estimated keyframe trajectory is aligned to ground truth by an $SE(3)$ Umeyama fit and its absolute trajectory error reported. Because only keyframes become graph variables, the number of optimized poses is far smaller than the frame count, which is what keeps the windowed batch solve tractable in bounded time per step.

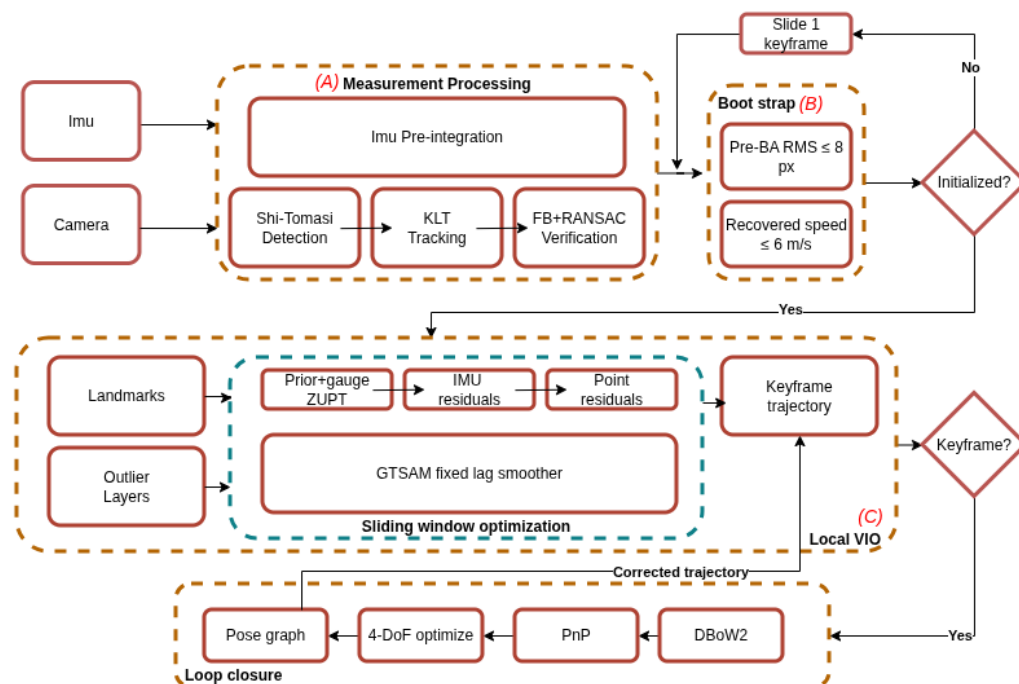


Figure 2. System architecture of the points-only VIO pipeline. Rounded boxes are processing blocks and diamonds are decision points; arrows give the direction of data flow, with the branch taken labeled Yes or No. The dashed boxes group the blocks into the stages of the pipeline, and their colors carry no meaning beyond that grouping, every box being labeled in the figure itself: the four orange boxes are the front-end measurement pre-pass, the bootstrap loop, the local VIO and the loop-closure pose graph, whose first three are the phases (A), (B) and (C) described in the text below, while the single teal box marks the fixed-lag optimization nested inside the local VIO. The corrected trajectory returned by the pose graph updates the published keyframe trajectory and is never fed back into the smoother, so loop closure does not affect the odometry results of Section 3.

2.2.2. Point Front-End and Observation Model

The point front-end is not a contribution of this work and is deliberately held fixed. The system consumes the point channel of the PL-VINS front-end [48], taken directly from its publicly released implementation and run at its native 20 Hz with the default parameters, with no modification. That front-end publishes two channels from the same image: a point channel, which is the VINS-Mono point tracker [3] that PL-VINS inherits unchanged, and a line channel, which PL-VINS adds. Only the point channel is used here. The point tracker detects Shi-Tomasi corners and tracks them frame to frame with Kanade–Lucas–Tomasi optical flow, enforcing forward-backward consistency and a fundamental-matrix RANSAC check, and for each surviving track it emits a persistent identifier together with the track’s raw and calibrated undistorted-normalized image coordinates [3,30]. Reusing this front-end unchanged is a deliberate methodological choice. The present system and both baselines of Section 3 are then driven by the same point tracks and the same inertial stream. Against VINS-Mono, which consumes exactly the same point channel and nothing

else, every difference in the results is therefore attributable to the back-end alone and not to feature extraction. PL-VINS also consumes the line channel of that stream, so it is compared here while carrying strictly more measurement information than the present system. Each frame thus contributes a set of observations $\{(\ell, \mathbf{z}_\ell)\}$ pairing a track identifier ℓ with its pixel measurement $\mathbf{z}_\ell$. The back-end re-projects the associated undistorted-normalized coordinate through the linear intrinsics K to form the reprojection residual. The front-end's own geometric verification (the forward-backward and epipolar tests) is the first of the three outlier-rejection layers evaluated in Section 2.2.7.

2.2.3. Inverse-Depth Points as GTSAM Factors

The estimator departs from the six-vector encoding of Section 2.1.2 in a way dictated by the choice of optimization variables. Because the graph optimizes body poses, not camera poses, and because the anchor's optical centre $\mathbf{c}_0$ is already represented by the anchor keyframe's pose variable, the landmark itself need not carry it. Each point is therefore a three-vector

$$\lambda_j = (\theta_j, \phi_j, \rho_j), \quad (14)$$

the azimuth, elevation, and inverse depth of the ray in the frame of its anchor camera a . With the unit bearing $\mathbf{m}(\theta, \phi)$ of (9), the corresponding world point and its predicted projection into any observing keyframe i are

$$\mathbf{p}_j^W = T_{WB}^a T_{BC} \frac{1}{\rho_j} \mathbf{m}(\theta_j, \phi_j), \quad \hat{\mathbf{z}}_{ij} = \pi(K, (T_{WB}^i T_{BC})^{-1} \mathbf{p}_j^W), \quad (15)$$

with reprojection residual $\mathbf{r}_{ij} = \hat{\mathbf{z}}_{ij} - \mathbf{z}_{ij}$. The composition $T_{WB} T_{BC}$ inside (15) is what lets the factor operate on body poses while the geometry lives in the camera. Its analytic form makes the factor's Jacobians with respect to both poses and the landmark available in closed form. Two factor types realize (15). The anchor's own observation is a binary factor over the anchor pose and the landmark, (x_a, L_j) , in which the anchor pose plays both roles and the residual reduces to the bearing consistency of the ray. Every other observing keyframe adds a ternary factor over (x_a, x_i, L_j) , coupling the anchor pose, the observing pose, and the landmark. Each factor carries the fixed extrinsic T_{BC} and the intrinsics K .

2.2.4. Delayed Initialization

A newly seen track is not inserted immediately. Its observations accumulate in a buffer until two views span a translation baseline of at least 0.10 m, at which point a two-view linear (DLT) triangulation recovers a 3-D point. The baseline gate self-delays the landmark under the low-parallax motion where triangulation is ill-posed. The triangulated point is expressed in the anchor camera frame and converted to (θ, ϕ, ρ) , and the track is promoted into the graph at its anchor keyframe with one anchor factor, one projection factor per subsequent view, and a weak isotropic prior $\mathcal{N}(\lambda_j, \sigma_\lambda^2 I)$ with $\sigma_\lambda = 1.0$ on (θ, ϕ, ρ) . This floor prior is deliberately loose: it prevents a landmark from being momentarily gauge-free when it enters the window with too few views to constrain its inverse depth, yet is weak enough to leave a well-triangulated point essentially unbiased. Once promoted, later observations simply append a further projection factor. The parametrization is marginalization-aware: because the fixed-lag smoother removes states by timestamp and cannot perform the joint depth-transfer re-anchoring of a full-window estimator, a landmark whose anchor keyframe leaves the window is dropped and, if still tracked, re-triangulated against a newer anchor, and not re-parametrized in place.

2.2.5. Inertial Factor, Navigation States, and Gauge

Between two keyframes the high-rate inertial samples are accumulated, one at a time, into a single preintegrated measurement following the on-manifold construction of (7). Gravity is oriented along $-Z$, consistent with the world frame produced by the realignment of Section 2.2.9. The accelerometer and gyroscope white-noise densities, the bias random-walk covariances, and the initial-bias cross-term required by the combined preintegration model are all supplied from configuration. Consecutive keyframes are then joined by a single combined preintegrated inertial factor, a 15-dimensional residual over $(X_i, V_i, B_i, X_j, V_j, B_j)$ that couples the two navigation states through (8) and simultaneously penalizes bias drift. A non-keyframe frame does not create a state, but preintegration continues across it, so the factor built at the next keyframe correctly spans every skipped frame.

A monocular visual-inertial problem has four unobservable degrees of freedom, global position and the yaw about gravity. The bootstrap fixes this gauge explicitly: priors are placed on the first state (X_0, V_0, B_0) , with a tight pose prior ($\sigma = 10^{-3}$) that anchors the global frame and looser velocity ($\sigma = 0.3$ m/s) and bias priors that merely regularize the seed. These are the only absolute constraints in the graph. All other factors are relative.

2.2.6. Fixed-Lag Smoother and Marginalization

The graph is optimized by a GTSAM batch fixed-lag smoother, configured with a timestamp for every inserted variable and a temporal lag of $\tau = 3.0$ s. At each update the smoother relinearizes and re-solves the active window with Levenberg–Marquardt (11), then Schur-marginalizes every state and landmark whose timestamp has fallen behind $t_{\text{now}} - \tau$ into the dense prior of (13). Point landmarks marginalize out on the same rule, keyed by the timestamp at which they were last observed, so a track that leaves the field of view ages out of the window together with the poses that saw it. An incremental fixed-lag smoother built on iSAM2 (Section 2.1.3) is available as an alternative back-end selected by configuration. The batch smoother is the paper’s reference arm because it re-optimizes the whole window each step and is the simpler of the two to reason about, while the incremental smoother attains statistically indistinguishable accuracy (Section 3.1.7) at roughly half the per-keyframe cost and a larger resident memory footprint (Section 3.4). Every result in this paper is reported on both, and Section 3.2 states which conclusions depend on the choice. Both are evaluated in Section 3.

Two implementation contracts follow from the marginalization. First, a landmark’s anchor pose must remain in the window for as long as the landmark does. The drop-and-re-triangulate policy of Section 2.2.3 enforces this. Second, the smoother caches a single pre-update estimate per keyframe for state prediction and triangulation and reads the refreshed estimate only once after the solve to reset the preintegration bias. Queries for a just-inserted state must fall back to the pending initial values, since that state is not yet in the cached estimate.

2.2.7. Point Outlier Rejection

The point measurements are guarded by three mechanisms drawn from two robustness lineages, applied in series.

1. **Front-end geometric verification.** Forward-backward optical-flow consistency and fundamental-matrix RANSAC remove grossly inconsistent correspondences before they ever reach the graph (Section 2.2.2).
2. **Hard reprojection rejection (Kimera-style).** When a new observation of an existing landmark arrives, its pixel is predicted from the cached estimate via (15). If the point falls behind the observing camera (a cheirality violation, predicted depth ≤ 0) or the reprojection error exceeds a gate of 20 px, the observation is discarded and no factor

is added [13,14]. This removes gross data-association errors that a continuous robust kernel would still admit with non-zero weight.

3. **Soft M-estimator weighting.** The observations that survive the gate enter the graph through a robust noise model: a Huber M-estimator with threshold $k = 0.1$ wrapping an isotropic pixel noise of $\sigma = 1.0$ px on the reprojection factor. Residuals that pass the hard gate but remain large are thereby down-weighted continuously instead of trusted at full strength. This occupies the same soft-robust role that VINS-Mono fills with a Cauchy loss on its reprojection residuals [3]. The present system uses a Huber kernel, whose bounded (rather than redescending) influence we found the more stable of the two inside the fixed-lag window.

The hard reject and the soft kernel play complementary roles: the reject removes the gross tail that a bounded influence function cannot suppress, while the Huber kernel governs the continuum of mildly inconsistent survivors that a hard threshold would either admit at full weight or, if tightened, begin to reject en masse. Both are standard back-end robustness mechanisms, used here as in VINS-Mono [3] and Kimera-VIO [14].

2.2.8. Keyframe Gating and Zero-Velocity Updates

Not every frame becomes a state. A frame is promoted to a keyframe when any of three conditions holds: the average parallax of its tracks against the last keyframe exceeds 30 px, fewer than 40 tracks carry over from the last keyframe, or 10 frames have elapsed since the last keyframe (a hard bound that keeps the preintegration interval, and hence its covariance, from growing without limit). A non-keyframe adds no graph node, only inertial accumulation. When the platform is stationary, a condition detected in the front-end from the IMU acceleration and angular-rate variance together with the median feature displacement, a zero-velocity update is applied: a velocity prior of $\sigma = 0.01$ m/s pins V_i , and between two consecutive stationary keyframes the relative translation is additionally locked. This is the only information constraining the pose through the stationary hand-pickup phase that opens the benchmark sequences, where monocular vision produces no parallax and the inertial channel sees only gravity.

2.2.9. Conditioning-Aware Initialization

The bootstrap recovers the metric, gravity-aligned seed from a short opening window. Its backbone is the robust visual-inertial initialization of VINS-Mono [3]: a vision-only structure-from-motion over a short keyframe window establishes an up-to-scale geometry, which a linear visual-inertial alignment then reconciles with the preintegrated inertial measurements to recover the gyroscope bias, the gravity direction, the per-keyframe velocity, and the metric scale. The world frame is finally rotated so that gravity points along $-Z$ and the scale is baked in. Frames are admitted to the window by a parallax gate (10 keyframes, parallax 10 px, at least 20 common tracks), and the attempt is triggered only once an accelerometer-variance excitation check confirms the platform is sufficiently excited (threshold 0.25). A further consistency gate rejects any window whose aligned speed exceeds 6 m/s. On any failure the window slides forward by one keyframe and the attempt repeats.

The bootstrap is summarized in Algorithm 1 and proceeds in seven stages.

1. **Window accumulation.** Frames are admitted to a sliding buffer by a parallax gate: 10 keyframes, at least 10 px median parallax, and at least 20 tracks common to the window. A window that cannot meet these is never presented to the solver.
2. **Excitation check.** The attempt is triggered only once the accelerometer variance over the window exceeds 0.25, which suppresses attempts while the platform is still at

rest. Without it the estimator repeatedly attempts initialization on stationary windows whose structure-from-motion is degenerate by construction.

3. **Structure from motion.** Relative pose from the essential matrix (five-point with RANSAC), then PnP for the remaining frames, then DLT triangulation, yielding an up-to-scale geometry over the window.
4. **Vision-only bundle adjustment.** Refinement of the geometry before any inertial information is introduced.
5. **Defer-until-well-conditioned gate.** The window is accepted only if the pre-bundle-adjust per-observation reprojection RMS is at most 8 px. This is the element on which the robustness of the bootstrap rests, and the choice of the pre-BA residual is deliberate. A rotation-dominant, near-zero-baseline pickup window is geometrically inconsistent before optimization. Afterwards the optimizer has hidden that inconsistency by driving the offending landmarks to large depths, so the post-BA residual looks healthy while the recovered scale and gravity are wrong. Measuring the residual before the optimizer can mask it is what makes a single threshold generalize across sequences whose opening motions differ. In practice, the discrimination is wide: a well-conditioned Machine-Hall window resolves at ≈ 2.5 px, while the rotation-dominant openings sit at 12–18 px.
6. **Linear visual-inertial alignment.** Recovering in closed form what monocular structure cannot: (a) the gyroscope bias, by requiring the visual-relative rotation to match the preintegrated $\Delta \tilde{R}_{ij}$ after a first-order correction through the stored Jacobian, (b) the per-keyframe velocities, the gravity direction and the metric scale s jointly, from the inertial kinematic identity of each consecutive pair, in which s multiplies the up-to-scale visual translation, and (c) a gravity refinement that re-solves on the 2-DoF sphere tangent at fixed magnitude $\|\mathbf{g}\| = g$. The solution is retained only if $0.1 < s < 10$ and $|\|\mathbf{g}_w\| - g| < 1.5$.
7. **Visual-inertial consistency gate.** A window can be visually well conditioned and still inertially inconsistent. When that happens the online optimizer reconciles vision against the IMU only by inventing a large accelerometer bias, which integrates into a runaway trajectory while feature tracking remains healthy, so the failure is invisible to any front-end diagnostic. The gate rejects any window whose maximum recovered body velocity exceeds 6 m/s and slides forward.

On any rejection the window slides forward by one keyframe and the attempt repeats. The world is finally rotated so that gravity points along $-Z$ and the scale is baked in, which is the convention the preintegration factor expects.

The thresholds are $N_w = 10$, $\sigma_{\text{exc}}^2 = 0.25$, $r_{\text{max}} = 8$ px and $v_{\text{max}} = 6$ m/s and are identical across both operating environments.

The accepted window is not frozen into a single prior. Instead, the entire bootstrap sub-graph is admitted into the smoother: every keyframe's (X, V, B) , a combined preintegrated inertial factor between each consecutive pair, the gauge priors on state 0 of Section 2.2.5, and every bootstrap landmark as an inverse-depth factor group. The smoother then re-optimizes this sub-graph jointly with the first online keyframes as they arrive, so the seed is continuously refined within the lag instead of fixed at handoff, removing the metric offset that a single-state seed would inject at the boundary between initialization and online operation. This element, rather than either threshold, is what carries convergence: replacing it with the conventional frozen single-state handoff diverges 10 of the 11 sequences (Section 3.2). The mechanism is that a bootstrap admitted as a sub-graph keeps re-optimizing against the first online keyframes until it slides out of the lag, so an imperfect

seed is corrected while it can still be corrected, whereas a frozen seed commits its scale and gravity error permanently.

Algorithm 1 Conditioning-aware visual-inertial bootstrap.

```

1: Accumulate a window of  $N_w$  keyframes with median parallax  $\geq 10$  px and  $\geq 20$  com-
   common tracks
2: repeat
3:   if accelerometer variance  $< \sigma_{\text{exc}}^2$  then
4:     slide window and continue ▷ platform still at rest
5:   recover up-to-scale poses and structure by SfM, then refine by vision-only BA
6:    $r \leftarrow$  pre-BA per-observation reprojection RMS
7:   if  $r > r_{\text{max}}$  then
8:     slide window and continue ▷ degenerate, rotation-dominant window
9:   solve the linear VI alignment  $\rightarrow \mathbf{b}_g, \{\mathbf{v}_i\}, \mathbf{g}_w, s$ 
10:  refine  $\mathbf{g}_w$  on the sphere at fixed  $\|\mathbf{g}\| = g$ 
11:  if  $s \notin (0.1, 10)$  or  $\max_i \|\mathbf{v}_i\| > v_{\text{max}}$  then
12:    slide window and continue ▷ inertially inconsistent seed
13: until window accepted
14: rotate all estimates so  $\hat{\mathbf{g}} \mapsto -\mathbf{Z}$  and apply the scale  $s$ 
15: push the entire bootstrap window into the smoother (states, IMU factors, landmarks)
16: continue online ▷ the seed keeps re-optimizing until it leaves the lag

```

2.2.10. Configuration

Every tunable parameter—the noise densities, the gate thresholds, the keyframe and initialization parameters—is a field of one configuration structure loaded at startup from a single YAML file per operating environment (one for the EuRoC Machine Hall sequences, one for the Vicon-room sequences). A missing field falls back to its committed default, so a partial file reproduces the baseline behavior exactly. By design, every gate is expressed so that disabling it, a threshold set to zero or to a value it can never exceed, is bit-identical to removing the mechanism from the code path, which keeps the configuration reproducible and each mechanism independently switchable. The shipped configuration retains the front-end verification, the hard reprojection reject, the soft Huber weighting, the floor prior, the zero-velocity update, and the initialization gate, while disabling the triangulation-depth and chirality-acceptance gates and the inertial-bias clamp.

Two environment files are used, one for the Machine Hall and one for the Vicon rooms, and it is worth stating precisely what separates them. They differ in exactly one scalar: the inertial integration covariance (10^{-6} for the Machine Hall against 10^{-4} for the Vicon rooms), which absorbs the different sensor-to-scene geometry of an industrial hall and a small motion-capture room. Every gate threshold, every noise multiplier, the window length, the keyframe criterion, the rejection bounds and all initialization parameters are identical across the two environments, and no parameter is tuned per sequence. This is what the phrase “one configuration per operating environment” means throughout this paper.

All reported runs are bit-reproducible. The pipeline pins its task scheduler to a single worker, so no result depends on thread interleaving, and seeds every RANSAC estimator (front-end epipolar verification, bootstrap structure-from-motion and loop-closure geometric verification) from a fixed value, so a given configuration and sequence produce identical outputs on every execution. We verified this over an 88-run validation campaign, executed across three independently compiled builds, which agreed to six decimal places. The values reported below are therefore exact for their configuration and not samples of a stochastic procedure, so a repeat-and-average protocol would add no information and every difference between two configurations is attributable to the configuration alone.

3. Results

The proposed method is evaluated on all eleven indoor field tests of the EuRoC MAV benchmark [12] (five Machine-Hall, six Vicon-Room), in both of its back-end configurations, incremental (iSAM2) and batch. The benchmark is onboard flight data: every sequence was recorded by a visual-inertial sensor carried on a micro air vehicle in real flight. These recordings are therefore indoor flight tests in the target regime of this work, GPS-denied indoor flight, and not a synthetic or a hand-carried benchmark, which is what makes them the appropriate public evidence for the estimator. They are replayed offline in timestamp order, so the estimator never sees the future, and it is validated outside the control loop ahead of trials on the project's own platform.

The sequences and their ground truth. The two environments differ in more than difficulty. The Machine-Hall sequences were flown in an industrial hall, a cluttered and unstructured space, and the Vicon-Room sequences in a motion-capture room of roughly $8 \times 8.4 \times 4$ m under two different obstacle configurations. Ground truth is obtained differently in each, which matters for how rotation errors should be read: the Machine Hall is tracked by a Leica Nova MS50 laser tracker giving millimeter-accurate position only, so attitude there is observed by the inertial measurements alone, whereas the rooms are tracked by a Vicon system giving full six-degree-of-freedom pose. Both are released as `state_groundtruth_estimate0`, a batch solution expressed at 200 Hz in the body frame, which is the quantity the estimator itself estimates and the reference every error in this paper is measured against. The authors grade the sequences easy, medium and difficult on three stated factors, illumination, visual texture and motion dynamics, so the labels reflect a combination of darkness, motion blur and flight speed rather than speed alone.

How the runs are configured and scored. All systems are driven by the same recorded front-end output at its native 20 Hz (Section 2.2.2), so that no result here depends on feature extraction. That output is produced by the PL-VINS front-end, which publishes a point channel, the VINS-Mono point tracker it inherits unchanged, and a line channel of its own. The present system and VINS-Mono consume the point channel only, and PL-VINS consumes both. The comparison against VINS-Mono is therefore back-end against back-end on an identical measurement set, and what differs is the inverse-depth points, the fixed-lag smoother and the 4-DoF loop-closure pose graph. PL-VINS is compared while carrying line measurements the present system does not use. All of our results use the per-observation rejection gate (20 px) active. The Absolute Trajectory Error (ATE) RMSE is then computed identically for every system, as the evo APE-translation RMSE after an SE(3) Umeyama alignment of the estimated trajectory to the EuRoC IMU-frame ground truth. For every method we report both the odometry result without loop closure ("no LC", isolating pure drift) and the full result with loop closure ("LC"), so that the comparison is column-aligned: no-LC against no-LC, and LC against LC.

Selection of the baselines. The proposed method is compared against two systems from the literature. A system qualifies as a baseline here if it is an optimization-based monocular visual-inertial estimator with a public implementation that can be driven by the same recorded front-end output as ours, so that no measured difference is attributable to feature extraction, to tuning, or to reporting conditions. VINS-Mono [3] and PL-VINS [48] meet this criterion. Neither is quoted from its paper. Both are built and run locally on the same machine and scored by the procedure above, and both post stronger no-loop figures locally than their published tables report, so the comparison is a demanding one. The same criterion fixes what is excluded and why. Stereo and stereo-inertial systems (the visual-inertial mode of ORB-SLAM3 [25], Kimera [14]) and LiDAR-inertial systems (LIO-SAM [37]) recover metric scale from a second sensor, which is the difficulty this paper addresses, so a comparison against them would confound the estimator with the sensor suite. Learned-

feature systems such as SL-SLAM [33] replace the shared front-end and therefore break the controlled comparison. Filtering-based estimators such as OpenVINS [9] belong to a different estimator class and are discussed in Section 1.1 as the embedded-efficiency reference and are not tabulated here. No qualifying system was omitted.

Figure 3 shows what the evaluation consumes: a representative Machine-Hall frame as recorded, and the micro air vehicle that flew the benchmark.

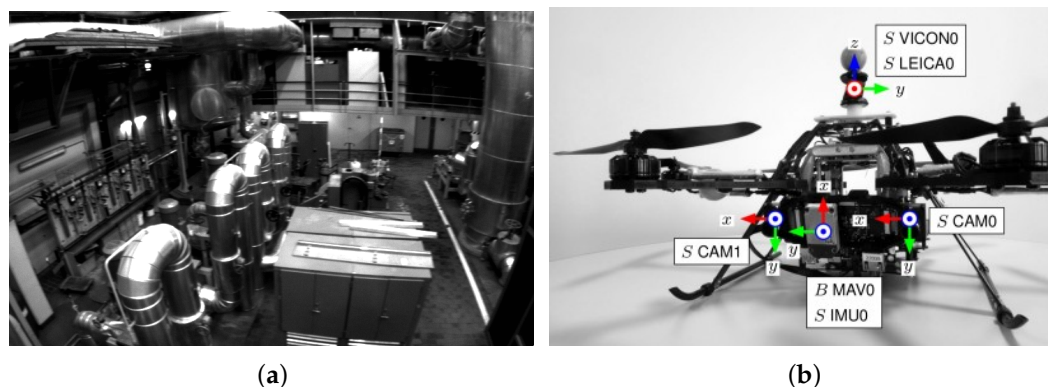


Figure 3. Experimental data source. (a) A representative Machine-Hall frame as the sensor delivers it (MH_01_easy, cam0, no contrast enhancement): an industrial interior of pipework, racking and walkways under uneven natural and artificial light, which is the raw input the estimator consumes. (b) The recording platform: the EuRoC AscTec Firefly hexrotor micro air vehicle, carrying the visual-inertial sensor (cam0/cam1 and imu0) whose onboard flight data constitute the benchmark, with the body frame MAV0 and the Vicon/Leica ground-truth frames indicated [12].

Figure 4 makes the difference between this work and the richer of its two baselines concrete on that same frame. Both systems consume the identical image. This work uses the point channel only, whereas PL-VINS also uses the line channel that the same front-end publishes and carries a reprojection residual for each segment. The lines fall predominantly on the structural boundaries of the scene, the racking uprights, the pipe runs and the panel edges, which is the geometry a point detector under-samples because it responds to corners rather than to extended edges. That is the argument for adding them, and Section 3.1 reports what it is worth in practice on this benchmark.

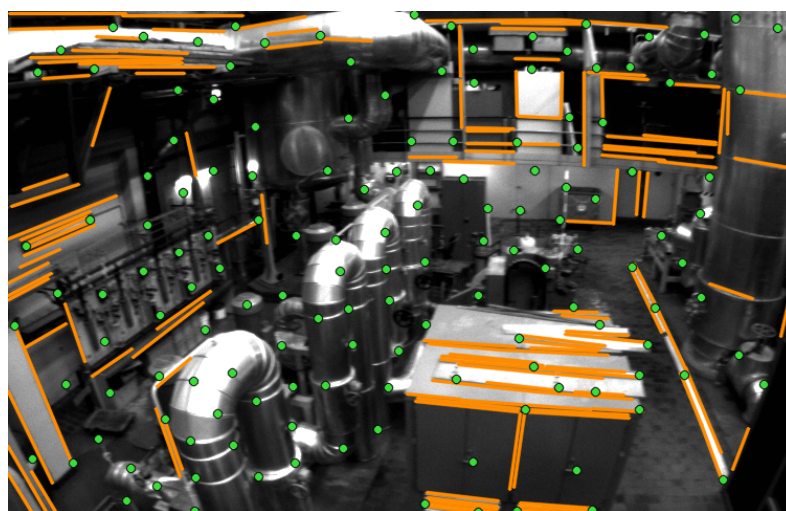


Figure 4. What each front-end extracts from one identical frame (MH_01_easy, cam0, shown without contrast enhancement). Green markers are the 150 Shi-Tomasi point features of the point channel, which this work and VINS-Mono both consume. Orange segments are the line features of the line channel that PL-VINS [48] additionally consumes, detected by LSD and filtered to the segments longer

than 40 px that it retains. The lines concentrate on the structural edges of the interior, which a corner detector samples only sparsely, and this complementarity is the reason a point-line system is the stronger of the two baselines used here.

The evaluation is organized in four parts: accuracy against the literature (Section 3.1), the influence of the initialization and rejection parameters (Section 3.2), the calibration of the estimator's covariance (Section 3.3), and its computational cost (Section 3.4).

3.1. Accuracy: Comparison with the Literature

Table 1 reports the points-only ATE against two locally run baselines, VINS-Mono and PL-VINS. The headline is the odometry column: without loop closure, taking the better of our two back-ends per sequence, our estimator is the most accurate on **9 of the 11** sequences against both baselines simultaneously, losing only MH_05 and V1_01. With loop closure the margin narrows, because the baselines pair their odometry with mature global pose graphs while the pose graph reported here is a prototype. Loop closure is outside the scope of this evaluation and the +LC columns are given for completeness only. The following paragraphs read the table column by column.

Table 1. Points-only ATE RMSE [m] versus two locally run baselines, VINS-Mono and PL-VINS, for both back-ends without (“no LC”) and with (“+LC”) loop closure. Per row, **bold** marks the best odometry (no-LC) result and underlined the best loop-closed (+LC) result, taken across all four estimators. On pure odometry our back-ends are the most accurate on **9 of 11** sequences against both baselines simultaneously, losing only MH_05 and V1_01.

Sequence	VINS-Mono (Local)		PL-VINS (Local)		Ours-Inc (iSAM2)		Ours-Batch	
	no LC	+LC	no LC	+LC	no LC	+LC	no LC	+LC
Machine Hall								
MH_01 easy	0.202	<u>0.075</u>	0.216	0.086	0.190	0.088	0.183	0.097
MH_02 easy	0.188	0.081	0.193	0.092	0.135	<u>0.049</u>	0.175	0.113
MH_03 medium	0.228	0.064	0.218	<u>0.061</u>	0.117	0.074	0.120	0.070
MH_04 difficult	0.376	0.111	0.249	<u>0.090</u>	0.225	0.168	0.198	0.156
MH_05 difficult	0.315	0.179	0.278	<u>0.139</u>	0.396	0.206	0.369	0.170
Vicon Room								
V1_01 easy	0.097	<u>0.042</u>	0.068	0.044	0.079	0.070	0.077	0.078
V1_02 medium	0.091	0.057	0.108	<u>0.038</u>	0.081	0.046	0.078	0.048
V1_03 difficult	0.177	0.196	0.154	0.141	0.132	0.125	0.146	<u>0.122</u>
V2_01 easy	0.079	0.059	0.090	<u>0.052</u>	0.050	0.057	0.059	0.064
V2_02 medium	0.119	0.187	0.123	0.100	0.074	0.083	0.066	<u>0.071</u>
V2_03 difficult	0.306	0.197	0.219	0.165	0.128	0.122	0.120	<u>0.116</u>
Mean	0.198	0.113	0.174	0.092	0.146	0.099	0.145	0.100

Both baselines were run locally on the same hardware, and not quoted from their papers, so the comparison reflects a single environment. All three systems are driven by the same recorded PL-VINS front-end output: VINS-Mono [3] and this work consume its point channel only, PL-VINS [48] consumes its point and line channels. All values are evo APE-translation RMSE against the EuRoC IMU-frame ground truth (state_groundtruth_estimate0). No difference is attributable to feature extraction.

Table 1 is per sequence, which is the resolution at which the mechanisms of Section 3.2 act. Aggregated by scene it reads as in Table 2.

Two things are visible here that the per-sequence table does not make obvious. On the Vicon rooms our odometry improves both the mean and the dispersion (0.091 ± 0.032 against 0.145 ± 0.086 for VINS-Mono): it is not only more accurate on average but more uniform across six sequences of differing difficulty, which is the property a single fixed parameter set has to have. On the Machine Hall the mean also improves (0.209 against

0.262 m), but our dispersion is the larger of the four (± 0.094 against ± 0.033 for PL-VINS), and the cause is a single sequence: MH_05 is the only Machine-Hall cell we lose, and it dominates the spread. We report it in full and do not present only the favorable aggregate. The loop-closed rows show the reverse ordering on the Machine Hall, consistent with the pose graph being a prototype (Section 4).

Table 2. Scene-wise ATE RMSE [m], mean $\pm$ sample standard deviation over the sequences of each scene. **Bold** marks the lowest mean in each row. The standard deviation is reported because dispersion across a scene, and not its average alone, is what determines whether one parameter set holds across the whole scene.

	Scene (<i>n</i>)	VINS-Mono	PL-VINS	Ours-Inc	Ours-Batch
no LC	Machine Hall (5)	0.262 $\pm$ 0.081	0.231 $\pm$ 0.033	0.213 $\pm$ 0.111	0.209 $\pm$ 0.094
	Vicon Room (6)	0.145 $\pm$ 0.086	0.127 $\pm$ 0.054	0.091 $\pm$ 0.032	0.091 $\pm$ 0.034
+LC	Machine Hall (5)	0.102 $\pm$ 0.046	0.094 $\pm$ 0.028	0.117 $\pm$ 0.067	0.121 $\pm$ 0.041
	Vicon Room (6)	0.123 $\pm$ 0.077	0.090 $\pm$ 0.054	0.084 $\pm$ 0.033	0.083 $\pm$ 0.030

3.1.1. The Relative-Pose Error Is Dominated by a Cadence Artifact

The RPE columns of Tables 3 and 4 are worse than both baselines in every cell, and the reason is largely a property of the metric rather than of the estimator. Those columns are computed at a delta of one frame, that is, between consecutive rows of the trajectory file. Both baselines emit a fixed 10 Hz stream, whereas our output is keyframe-gated with a median inter-row interval that varies five-fold across the suite (0.05–0.25 s), so a one-frame pair spans up to $2.5\times$ more path for our system than for theirs. The two columns therefore do not measure the same displacement.

Table 3. Batch back-end versus two locally run baselines, VINS-Mono and PL-VINS, points: absolute (APE) and 1-frame relative (RPE) translation RMSE [m], without (no LC) and with (+LC) loop closure, against the EuRoC IMU-frame ground truth (`state_groundtruth_estimate0`). All three systems share the VINS-Mono front-end (Section 2.2.2).

Sequence	Method	APE RMSE [m]		RPE RMSE [m]	
		no LC	+LC	no LC	+LC
Machine Hall					
MH_01 easy	Ours	0.181	0.097	0.0118	0.0116
	VINS-Mono	0.202	0.075	0.0040	0.0065
	PL-VINS	0.216	0.086	0.0036	0.0061
MH_02 easy	Ours	0.174	0.114	0.0134	0.0133
	VINS-Mono	0.188	0.081	0.0031	0.0050
	PL-VINS	0.193	0.092	0.0033	0.0050
MH_03 medium	Ours	0.120	0.070	0.0144	0.0144
	VINS-Mono	0.228	0.064	0.0069	0.0083
	PL-VINS	0.218	0.061	0.0063	0.0078
MH_04 difficult	Ours	0.197	0.155	0.0214	0.0212
	VINS-Mono	0.376	0.111	0.0073	0.0090
	PL-VINS	0.249	0.090	0.0081	0.0095
MH_05 difficult	Ours	0.369	0.170	0.0268	0.0266
	VINS-Mono	0.315	0.179	0.0065	0.0102
	PL-VINS	0.278	0.139	0.0066	0.0110

Table 3. Cont.

Sequence	Method	APE RMSE [m]		RPE RMSE [m]	
		no LC	+LC	no LC	+LC
Vicon Room					
V1_01 easy	Ours	0.077	0.078	0.0141	0.0140
	VINS-Mono	0.097	0.042	0.0052	0.0072
	PL-VINS	0.068	0.044	0.0049	0.0071
V1_02 medium	Ours	0.078	0.048	0.0073	0.0075
	VINS-Mono	0.091	0.057	0.0057	0.0060
	PL-VINS	0.108	0.038	0.0046	0.0045
V1_03 difficult	Ours	0.146	0.122	0.0087	0.0087
	VINS-Mono	0.177	0.196	0.0064	0.0064
	PL-VINS	0.154	0.141	0.0060	0.0061
V2_01 easy	Ours	0.059	0.064	0.0073	0.0075
	VINS-Mono	0.079	0.059	0.0031	0.0039
	PL-VINS	0.090	0.052	0.0028	0.0035
V2_02 medium	Ours	0.066	0.071	0.0071	0.0070
	VINS-Mono	0.119	0.187	0.0047	0.0051
	PL-VINS	0.123	0.100	0.0042	0.0044
V2_03 difficult	Ours	0.120	0.116	0.0129	0.0128
	VINS-Mono	0.306	0.197	0.0065	0.0062
	PL-VINS	0.219	0.165	0.0068	0.0063

Table 4. Incremental (iSAM2) back-end versus two locally run baselines, VINS-Mono and PL-VINS, points: absolute (APE) and 1-frame relative (RPE) translation RMSE [m], without (no LC) and with (+LC) loop closure, against the EuRoC IMU-frame ground truth (`state_groundtruth_estimate0`). All three systems share the VINS-Mono front-end (Section 2.2.2).

Sequence	Method	APE RMSE [m]		RPE RMSE [m]	
		no LC	+LC	no LC	+LC
Machine Hall					
MH_01 easy	Ours	0.189	0.089	0.0130	0.0127
	VINS-Mono	0.202	0.075	0.0040	0.0065
	PL-VINS	0.216	0.086	0.0036	0.0061
MH_02 easy	Ours	0.133	0.049	0.0145	0.0143
	VINS-Mono	0.188	0.081	0.0031	0.0050
	PL-VINS	0.193	0.092	0.0033	0.0050
MH_03 medium	Ours	0.117	0.074	0.0240	0.0240
	VINS-Mono	0.228	0.064	0.0069	0.0083
	PL-VINS	0.218	0.061	0.0063	0.0078
MH_04 difficult	Ours	0.225	0.169	0.0354	0.0354
	VINS-Mono	0.376	0.111	0.0073	0.0090
	PL-VINS	0.249	0.090	0.0081	0.0095
MH_05 difficult	Ours	0.395	0.206	0.0340	0.0336
	VINS-Mono	0.315	0.179	0.0065	0.0102
	PL-VINS	0.278	0.139	0.0066	0.0110

Table 4. Cont.

Sequence	Method	APE RMSE [m]		RPE RMSE [m]	
		no LC	+LC	no LC	+LC
Vicon Room					
V1_01 easy	Ours	0.079	0.070	0.0141	0.0140
	VINS-Mono	0.097	0.042	0.0052	0.0072
	PL-VINS	0.068	0.044	0.0049	0.0071
V1_02 medium	Ours	0.081	0.046	0.0075	0.0076
	VINS-Mono	0.091	0.057	0.0057	0.0060
	PL-VINS	0.108	0.038	0.0046	0.0045
V1_03 difficult	Ours	0.132	0.125	0.0089	0.0089
	VINS-Mono	0.177	0.196	0.0064	0.0064
	PL-VINS	0.154	0.141	0.0060	0.0061
V2_01 easy	Ours	0.050	0.057	0.0077	0.0080
	VINS-Mono	0.079	0.059	0.0031	0.0039
	PL-VINS	0.090	0.052	0.0028	0.0035
V2_02 medium	Ours	0.074	0.082	0.0075	0.0075
	VINS-Mono	0.119	0.187	0.0047	0.0051
	PL-VINS	0.123	0.100	0.0042	0.0044
V2_03 difficult	Ours	0.128	0.122	0.0134	0.0134
	VINS-Mono	0.306	0.197	0.0065	0.0062
	PL-VINS	0.219	0.165	0.0068	0.0063

Recomputing the identical metric at a cadence-invariant delta of one meter, under an otherwise unchanged protocol, separates the artifact from the residual (Table 5).

Table 5. Relative-pose error at the published one-frame delta and at a cadence-invariant one-meter delta, over all eleven sequences and under one identical protocol (state_groundtruth_estimate0, SE(3) Umeyama alignment, translation part). The last two columns are medians of the per-sequence ratio, which is the paired statistic, and are not ratios of the medians to their left. Bold marks the two ratios at the cadence-invariant one-meter delta, which are the corrected figures the text discusses.

Delta	Median RPE Translation [m]			Median Paired Ratio	
	Ours	VINS-Mono	PL-VINS	vs. VINS-Mono	vs. PL-VINS
1 frame (as published)	0.0129	0.0057	0.0049	2.35×	2.64×
1 meter (cadence-invariant)	0.0490	0.0481	0.0408	1.13×	1.31×

Normalizing the delta also makes it worth separating the two channels of the metric, because a single translation figure cannot show which degree of freedom the local error occupies. Table 6 reports translation and rotation per meter travelled, aggregated by scene, on both back-ends.

The asymmetry is the point. Translation drift agrees between the two environments to within 5% on either back-end, so the estimator translates equally well in both, while rotation drift is $3.1\times$ worse in the rooms on the batch smoother and $2.7\times$ worse on the incremental one. The same imbalance appears in the spread across the eleven sequences, where translation drift varies by a factor of 2.8 and rotation drift by a factor of 6.8. Whatever makes the harder sequences hard is rotational.

Table 6. Relative-pose error per meter travelled, separated into translation and rotation and aggregated by scene, for both back-ends, loop closure off. Translation drift is the same between the two environments to within 5% on either solver. Rotation drift is roughly three times worse in the rooms, so the entire difference in difficulty between the two environments lies in attitude. Bold marks the rotation columns and their ratio, which is where that difference lies.

Scene (<i>n</i>)	Batch		Incremental	
	trans. [m/m]	rot. [°/m]	trans. [m/m]	rot. [°/m]
Machine Hall (5)	0.058	0.195	0.065	0.230
Vicon Room (6)	0.061	0.604	0.062	0.629
Ratio, rooms over hall	1.05×	3.10×	0.95×	2.73×

This connects the local metric to the consistency measurement of Section 3.3, and the two are best read as one finding rather than two. V1_01 carries the worst rotation drift of the suite, $1.143^\circ/\text{m}$ on batch and $1.151^\circ/\text{m}$ on incremental, nearly five times the Machine-Hall level, and it is also the sequence with the worst rotational NEES, 137.6 against a median expectation of 2.37. The error and the over-confidence occupy the same channel. We add the qualifier that keeps this in proportion: over a single meter of travel half a degree of attitude error induces about nine millimeters of position error, so rotation is not what makes the local translation figure large. It is that attitude error does not stay local, because every meter travelled afterwards is rotated by it, and it is the one channel in which the covariance does not report the problem.

The paired penalty against VINS-Mono falls from $2.35\times$ to $1.13\times$, and against PL-VINS from $2.64\times$ to $1.31\times$. On V2_02 the ordering reverses outright and our system is the more accurate of the two at one meter (0.037 against 0.042 m). Most of the apparent deterioration is thus an artifact of comparing a keyframe-rate output against a fixed-rate one at a frame-indexed delta.

What survives normalization is real, and it is a design trade and not a defect. Two mechanisms account for it. Our trajectory rows are independently optimized keyframe states, whereas the dense output of both baselines is IMU-propagated between optimized keyframes and is therefore locally smooth by construction. And the hard reprojection reject removes local constraints, which is the same mechanism that prevents three divergences in the ablation of Section 3.2. The design wins on the global axis and pays a bounded price on local smoothness. Emitting a fixed-rate IMU-propagated stream between keyframes would close most of the remainder without altering the estimator, which is noted in the Limitations and Future Directions section.

3.1.2. Machine Hall: Back-End Outlier Handling Drives the Odometry Margin

Even without loop closure, the Machine-Hall no-LC column tracks to 0.12–0.37 m and beats the local VINS-Mono’s own odometry on four of the five sequences (MH_01 0.183 vs. 0.202, MH_02 0.135 vs. 0.188, MH_03 0.117 vs. 0.228, MH_04 0.198 vs. 0.376 m). Only MH_05 (0.369 vs. 0.315 m) trails. This is the direct signature of the per-observation rejection gate: the few hundred grossly wrong feature associations per sequence (tracker id switches whose reprojection error exceeds 20 px) are removed before they become factors, and are not merely down-weighted by the Huber kernel. MH_02 is the clearest example: it settles at 0.135 m (incremental, no-LC) and reaches the suite-best 0.049 m once loop closure is enabled.

Adding loop closure then sharpens every Machine-Hall sequence, because all five revisit earlier places. The batch back-end improves monotonically (MH_01 0.183 $\rightarrow$ 0.097 m, MH_03 0.120 $\rightarrow$ 0.070 m, MH_05 0.369 $\rightarrow$ 0.170 m), and the incremental back-end improves further where drift is largest (MH_02 0.135 $\rightarrow$ 0.049 m, MH_05 0.396 $\rightarrow$ 0.206 m).

MH_04 closes more modestly on both back-ends (batch 0.198 $\rightarrow$ 0.156 m, incremental 0.225 $\rightarrow$ 0.168 m) and is the one Machine-Hall sequence where the local VINS-Mono's loop-closed result (0.111 m) stays ahead of ours, with the baseline's mature global pose graph doing its work.

3.1.3. Vicon Room: Generalization Under a Single Parameter Set

The Vicon-Room block is the headline robustness result. Three of these sequences (V1_02, V2_01, V2_02) previously diverged catastrophically—by tens to thousands of meters—because a visually well-conditioned but visual-inertially inconsistent bootstrap window seeded a runaway accelerometer bias. With the visual-inertial consistency gate rejecting those windows, the same three now track to sub-decimeter accuracy (V1_02 0.078 m, V2_01 0.059 m, V2_02 0.066 m, batch no-LC), and all eleven EuRoC sequences converge under one fixed parameter set. On the Vicon Rooms our no-LC odometry beats the local VINS-Mono on all six sequences: V1_03 0.132–0.146 m against 0.177 m, V2_03 0.120–0.128 m against 0.306 m, and even the low-drift easy rooms (V2_01 0.050–0.059 m against 0.079 m, V2_02 0.066–0.074 m against 0.119 m). The gain here is in the odometry and initialization, not the loop graph.

3.1.4. Loop Closure: Signal Where Drift Exists, Noise Where It Does Not

The +LC columns make the precision/recall stance of the loop closure visible. Where genuine drift has accumulated, on the Machine Hall and the higher-drift rooms (V1_02, V1_03, V2_03), closing loops removes a large fraction of it. Where drift is already below the loop-edge (PnP) noise floor, on the two lowest-drift rooms V2_01 and V2_02, closing a geometrically valid loop can only re-inject that noise, so ATE rises slightly (V2_01 0.059 $\rightarrow$ 0.064 m, V2_02 0.066 $\rightarrow$ 0.071 m, batch). This is the unobservable regime, and the local VINS-Mono is affected far more strongly by it: its own loop closure regresses on the low-drift cells (V1_03 0.177 $\rightarrow$ 0.196 m, V2_02 0.119 $\rightarrow$ 0.187 m). We keep these loops active and disclosed instead of suppressing them, and a signal-vs-noise gate (rejecting any loop whose implied correction exceeds 3% of the traversed path) removes the gross-wrong edges without touching the genuine ones.

3.1.5. Incremental Versus Batch

The two back-ends do not bracket an accuracy trade-off. They are statistically indistinguishable on ATE (Section 3.1.7, $p = 0.765$), and what separates them is cost: the incremental back-end reaches the same accuracy at 37.8 ms per keyframe against 85.5 ms for the batch smoother, and pays for it in peak memory (Section 3.4). The incremental (iSAM2) back-end updates only the affected part of the factorization and, with loop closure, produces several of the best cells in the table (MH_02 0.049 m, V1_02 0.046 m, V2_01 0.057 m). The batch back-end re-linearizes the whole window each step, and both back-ends converge on all eleven sequences under the same configuration, so neither is the more robust in the sense of surviving cases the other does not. Across the suite the two agree closely—within a few millimeters on most cells—with MH_04 the widest gap, where batch (0.198 m no-LC, 0.156 m +LC) sits a few centimeters ahead of incremental (0.225 m, 0.168 m). Neither back-end diverges on any of the eleven sequences under the single shared configuration.

3.1.6. Error over Time Across the Benchmark

The tables above compress each run to a single scalar. Because a given ATE can arise either from steady drift or from one bad interval, Figure 5 reports the absolute position error against time for all eleven sequences, for both of our back-ends and both baselines, under the whole-trajectory alignment that produces the tabulated RMSE. Each curve therefore integrates to the number in Tables 1, 3 and 4.

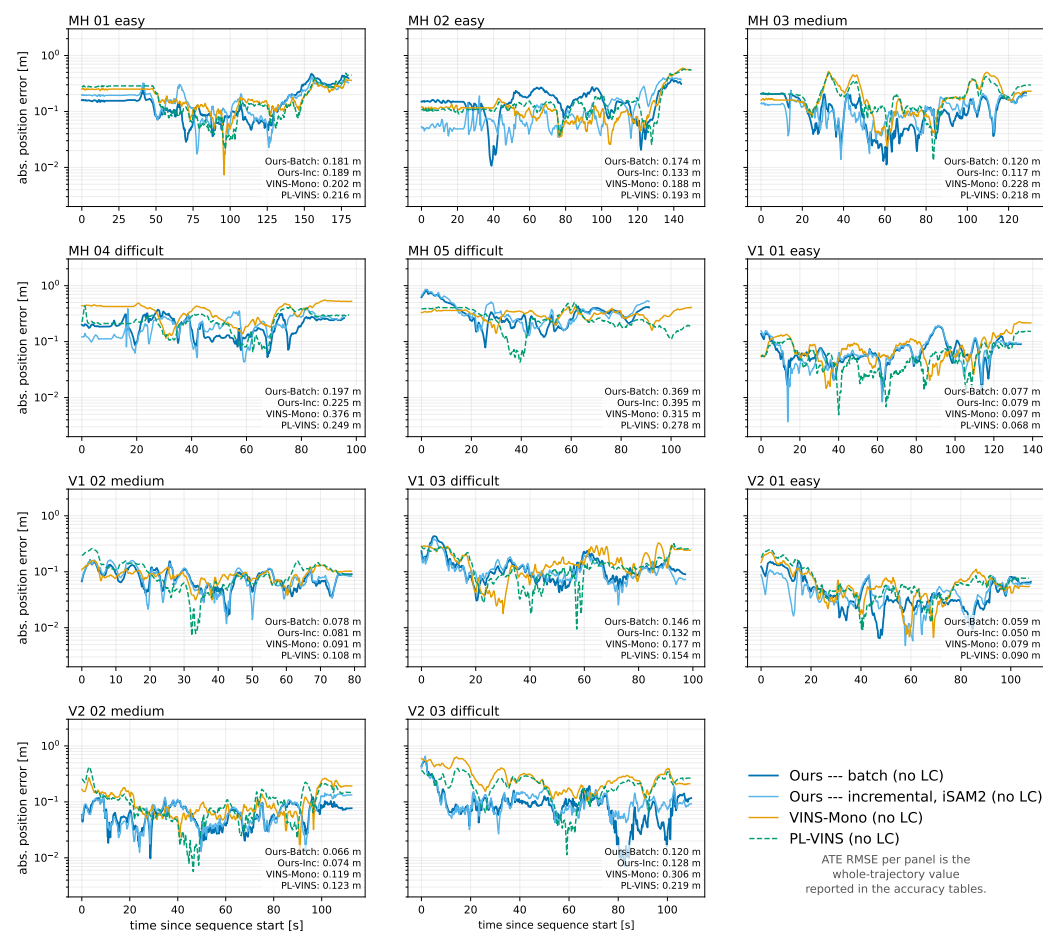


Figure 5. Absolute position error against time for all eleven EuRoC sequences, logarithmic ordinate on a common scale, for the four systems that converge: our batch and incremental back-ends and the two locally run baselines. Each trajectory is Umeyama-aligned to the ground truth over its whole length, so the RMSE annotated in each panel is the value reported in the accuracy tables. The diverging configuration of Section 3.2.1 is deliberately omitted here: its excursion spans five decades and would compress the comparison between the converged systems to invisibility.

Three things are visible here that a scalar RMSE cannot show.

The error oscillates inside a bounded band and does not grow monotonically. On every sequence and for all four systems, the curve fluctuates within roughly one decade instead of climbing steadily. Over the one-to-three-minute span of these sequences the inertial-visual constraint holds the estimate. The unbounded growth that motivates a global layer is a longer-horizon phenomenon than EuRoC exercises. The common rise in the closing seconds of MH_01, MH_02 and MH_03, shared by all three systems, is the non-revisiting final leg of those trajectories, and is the interval a loop closure removes.

The margin holds across the whole run and does not come from a lucky interval. On the nine sequences we lead, the blue curve sits below both baselines for most of the flight, instead of winning on average through a small number of favourable segments. This is the per-sequence form of the aggregate in Table 2.

The one Machine-Hall sequence we lose is a bootstrap failure, not a drift failure. On MH_05 our error starts near 0.8 m, an order of magnitude above its own steady-state level and above both baselines at the same instant, and, although it decays as the window re-optimizes, it never recovers the margin. The sequence is the longest and dimmest of the Machine Hall, and its opening is the hardest to condition. This single run is what raises our Machine-Hall standard deviation to the largest of the four columns in Table 2, and it

identifies where the remaining headroom in this system is: the acceptance decision at the bootstrap, not the estimator that follows it.

3.1.7. Paired Significance of the Accuracy Margin

Tables 1 and 2 report counts and aggregates. Neither of them establishes that the margin is larger than the variation an eleven-sequence sample can produce by chance, and that question has two distinct halves.

The first half is variation between repetitions of the same configuration, and here there is nothing to test. Every reported run is bit-reproducible (Section 2.2.10), so a repeated execution returns the identical value and the within-configuration variance is exactly zero rather than merely small. Each difference between two entries of Table 1 is therefore attributable to the systems being compared and not to run-to-run noise. What this does not establish is that the estimator is insensitive to perturbation, which is measured separately by the component study of Section 3.2.

The second half is inference between systems, which the eleven sequences do bound. The sequences are the natural blocking factor, because all four estimators consume the identical feature stream on each one, and the per-sequence ATE distribution is small and skewed. The appropriate test is therefore a paired non-parametric one. We apply the two-sided Wilcoxon signed-rank test to the paired per-sequence differences, evaluated against the exact null distribution rather than a normal approximation, and report an exact sign test alongside it as the weaker but assumption-free companion. The outcome is collected in Table 7.

Table 7. Paired significance of the odometry (no loop closure) margin over the eleven sequences of Table 1. Wins counts sequences on which the first system has the lower ATE. The median paired Δ is the median of the per-sequence differences, negative where the first system is the more accurate, with the median of the per-sequence relative differences in parentheses. Both tests are two-sided, and the Wilcoxon p is exact. Bold marks significance at the 5% level. Both tests rank the paired differences, so they are computed from the full-precision result files and not from the three-decimal values printed in Table 1. The distinction is immaterial for the baseline comparisons, whose paired differences are tens of millimeters, and it is not immaterial for the batch-against-incremental row, whose differences are a few millimeters and whose ranks rounding would reorder.

Comparison (Odometry, No LC, $n = 11$)	Wins	Median Paired Δ	Wilcoxon p	Sign p
Ours-Batch vs. VINS-Mono	10/11	−0.0203 m (−20.5%)	0.024	0.012
Ours-Inc vs. VINS-Mono	10/11	−0.0444 m (−27.9%)	0.024	0.012
Ours-Batch vs. PL-VINS	9/11	−0.0312 m (−20.7%)	0.054	0.065
Ours-Inc vs. PL-VINS	9/11	−0.0268 m (−24.8%)	0.067	0.065
Ours-Batch vs. Ours-Inc	7/11	−0.0029 m (−3.6%)	0.765	0.549

Three outcomes follow, and two of them qualify claims made earlier in this paper.

The margin over VINS-Mono is significant. Ten of the eleven sequences favor this estimator on both back-ends, at $p = 0.024$ (Wilcoxon) and $p = 0.012$ (sign). This is the claim the paper rests on and it survives the test.

The margin over PL-VINS is not significant at the 5% level. Nine of eleven favor this estimator, at $p = 0.054$ for the batch back-end. The count of nine sequences in Table 1 is factually correct and is retained, but as an inference about which of the two systems is the more accurate it is not supported at this sample size, and the Abstract, the Discussion and the Conclusions of this paper state the comparison against PL-VINS in those terms.

The two back-ends are statistically indistinguishable on accuracy. At $p = 0.765$ there is no evidence that the batch smoother is the more accurate of the two. What separates them is computational cost, not accuracy, and that separation is reported in Section 3.4.

Two limitations of this analysis belong with it. The first concerns the power of the test. With eleven paired sequences the smallest attainable two-sided sign-test p is $2^{-10} \approx 0.001$, and a 9/11 split yields $p = 0.065$ irrespective of how large the per-sequence margins are, so the test cannot declare significance below 10/11 wins. The borderline PL-VINS result is accordingly a limit of the sample size and is not evidence of equivalence. The second concerns effect size, which a rank test discards. A percentile bootstrap of the median-paired relative difference (20,000 resamples) gives -20.5% with a 95% interval of $[-47.3, -9.4]\%$ against VINS-Mono and -20.7% with $[-44.8, -5.4]\%$ against PL-VINS, so both intervals exclude zero and the effect-size evidence is the more favorable of the two views. We report the rank test as the primary result because it is the conservative one, and we state that the two disagree at the margin rather than quoting whichever is convenient.

Finally, the population over which these tests license an inference is the eleven sequences of this benchmark, recorded by one sensor rig on one airframe in two environments. They bound the comparison on EuRoC and they do not extend it to indoor flight in general, a limitation returned to the Limitations and Future Directions section.

The best-of-two-back-ends marking used in Table 1 is a presentation of results and a post-hoc selection, so it is not used as the basis of any test above. Every p -value is computed for one fixed back-end against one fixed baseline.

3.2. Influence of the Initialization and Rejection Parameters

Each mechanism of Sections 2.2.7 and 2.2.9 is switchable from the configuration file, and every gate is expressed so that disabling it is bit-identical to removing it from the code path. We exploit this to measure each contribution separately: six configurations, each differing from the full system in exactly one setting, run over all eleven sequences and repeated on both back-ends (132 runs, no loop closure, deterministic). Repeating the whole study on the incremental smoother is what separates a property of the system from a property of the solver it was first measured on, and it is reported here for every arm rather than for a selected one. Configuration A0 is the full system and reproduces the per-sequence results of Table 3 to a median of 0.3%, so every delta below is measured against the reported baseline itself. Comparisons are paired per sequence. A median of absolute ATEs across sequences is not a paired statistic and can reverse the sign of a verdict.

Five results follow from Table 8, two of which correct the design intuition stated in Section 2.2.9, and the last of which concerns whether any of the other four depend on the solver.

Convergence is produced by the continuous window, not by either threshold. Replacing the continuous-window handoff with the conventional frozen single-state seed (A5) diverges ten of the eleven sequences, by 1.4 m (MH_01) to 6265 m (V1_02). Only MH_03 stays bounded, and degrades by +326%. Neither initialization threshold diverges any sequence (A1, A2: 0/11). The “tens to thousands of meters” behavior that motivates a hardened bootstrap is therefore attributable to the architecture of the handoff, admitting the bootstrap as a re-optimizable sub-graph rather than a fixed prior, and not to a tuned rejection threshold.

The two outlier-rejection layers are complementary, not redundant. Removing the hard reprojection reject (A3) diverges MH_03 (6.33 m) and MH_04 (2.27 m) and aborts V2_03 with an indeterminate linear system. Removing the soft Huber weighting (A4) aborts V1_03 and V2_03 with the same exception and costs +130% median ATE on the nine sequences that still run. The two layers fail on different sequences, so neither subsumes the other: V2_03 requires both. This is the quantitative form of the layered-rejection argument of Section 2.2.7. The hard gate is not free, it is mildly harmful on the two easiest cells

(MH_01 −2.1%, V2_01 −6.1%), so it trades a little accuracy on well-conditioned sequences for the difference between operating and failing on ill-conditioned ones.

Table 8. Influence of each mechanism, measured by disabling it in isolation and pairing against the full system (configuration A0) over all eleven sequences, on both back-ends. Median is the paired per-sequence ATE delta over the runs that remained bounded, each back-end paired against its own A0 baseline and never against the other. Failed counts sequences that diverged (>1 m) or aborted. Configurations are ordered by severity. A2 is bit-identical to A0 on all eleven sequences on both back-ends. The batch and incremental medians of A3 are not directly comparable, the first being taken over eight survivors and the second over all eleven. Bold marks every non-zero failure count, separating the arms that cost convergence from those that cost only accuracy.

Config.	Mechanism Disabled	Batch		Incremental	
		Median Δ ATE	Failed	Median Δ ATE	Failed
A5	Continuous-window initialization	+326%	10	+416%	10
A4	Soft Huber weighting	+130%	2	+191%	3
A6	Inverse depth ($\rightarrow$ world-frame points)	+131%	2	+105%	3
A3	Hard 20 px reprojection reject	+3.5%	3	+31.9%	0
A1	Pre-BA conditioning gate	+6.7%	0	+0.0%	0
A2	Velocity-consistency gate	+0.0%	0	+0.0%	0

The conditioning gate contributes accuracy, not convergence, and does so only on the batch back-end. Disabling it (A1) costs +6.7% median and +14.6% mean ATE on the batch smoother, with the largest single effect on V1_01 (+127%), but diverges nothing. On two sequences (V1_02, V1_03) it is bit-identical to the baseline: the gate never changes which window is accepted there. On the incremental smoother the same arm is inert, at +0.0% median and no divergence, so on that back-end neither initialization threshold has a measurable effect. We state the accuracy contribution of this gate as back-end-specific rather than as a property of the system, because the measurement does not support the stronger reading. The mechanism is consistent with the divergence result below: the gate changes which opening window is accepted, and a solver that re-optimizes that window more aggressively recovers from a marginally worse choice of window, so the value of the gate is largest where the solver behind it does the least.

The velocity-consistency gate is inert on this benchmark, on both back-ends. Configuration A2 is bit-identical to A0 on all eleven sequences and on both solvers, and the initialization logs (Section 3.2.2) record zero velocity rejections against 319 conditioning rejections. The gate is fully shadowed: every window that reaches it has already been accepted by the conditioning gate, whose residual test is the stricter of the two on EuRoC. We retain the check as a safeguard for flight data outside this benchmark, but it cannot be credited with any measured contribution here.

Every one of these conclusions survives the change of solver. The four results above were first measured on the batch smoother, and the natural objection is that they are properties of that solver rather than of the system. Repeating all six arms on the incremental smoother answers it directly, and Table 9 collects the comparison across the whole evaluation rather than the ablation alone.

The replication is close to exact where it matters most. Removing the continuous-window handoff diverges ten of eleven sequences on both solvers, and not merely the same count: the same ten sequences fail, with MH_03 the sole survivor in both campaigns. The velocity-consistency gate is inert on eleven of eleven on both. Removing the Huber weighting or the inverse-depth parametrization is catastrophic on both. The ordering of the arms by severity is unchanged. Since the ablation is where the contribution of this work

is argued, this is the evidence that the argument is about the estimator and not about the solver chosen to measure it.

Table 9. Every substantive result of this evaluation, measured on both back-ends. The lag, the factors, the feature stream and the configuration are identical throughout and only the solver differs, so a row that replicates is a property of the estimator and a row that does not is a property of the solver. Paired comparisons are per sequence. The two rows in the third block are the only place the solvers genuinely diverge, and they point in opposite directions: the incremental smoother carries the larger attitude error and reports the larger attitude uncertainty.

Result	Batch	Incremental	Paired Verdict
Better than both baselines	9/11	9/11	Replicates, and the same two sequences are lost
Mean ATE [m]	0.145	0.146	Indistinguishable (Table 7, $p = 0.765$)
Frozen seed (A5) diverges	10/11	10/11	Replicates, and on the same ten sequences
Velocity gate (A2) inert	11/11	11/11	Replicates exactly
Huber (A4), inverse depth (A6)	+130%, +131%	+191%, +105%	Replicates
Conditioning gate (A1)	+6.7%	+0.0%	Does not replicate, batch only
RPE rotation [$^{\circ}$ /m]	0.418	0.448	Incremental worse on 10/11 ($p = 0.002$)
Median pose NEES	13.2	8.6	Incremental better on 10/11 ($p = 0.032$)
Per-keyframe cost [ms]	85.5	37.8	2.3 $\times$ faster (Section 3.4)
Peak resident memory [MB]	101.5	145.4	1.4 $\times$ the memory

Two rows do move, and both are informative rather than inconvenient. The conditioning gate (A1) is worth +6.7% on batch and nothing on incremental, which is why its contribution is now stated as back-end-specific. And the hard reprojection reject (A3) changes the form of its failure rather than its importance: on the batch smoother removing it diverges three sequences while the survivors barely change, whereas on the incremental smoother nothing diverges but every sequence degrades by about a third. The mechanism is the same and the consequence is expressed differently, which is why the two medians in Table 8 are not directly comparable.

3.2.1. Case Study: The Degenerate Takeoff on V1_02

The aggregate of Table 8 says the continuous window carries convergence. It is worth seeing that claim on a single sequence, because the failure it prevents is abrupt rather than gradual. V1_02 is the extreme case of the suite: 0.078 m with the continuous window and a divergence with the frozen seed.

Figures 6 and 7 compare the full system on both back-ends, the identical binary with the continuous window replaced by the conventional frozen single-state seed (arm A5), and the locally run VINS-Mono baseline, all against the same ground truth. Both of our back-ends survive the opening and settle at the same accuracy (0.078 and 0.081 m), which places the failure below the smoother: it is the handoff, not the solver behind it, that decides whether the run survives. We stress what A5 is and is not. It is our estimator with the initialization handoff of VINS-Mono substituted, everything else held bit-identical, which is what isolates the mechanism. It is not the VINS-Mono system, which converges on this sequence (0.091 m) and carries failure-detection and re-initialization machinery we do not replicate. The comparison to draw is therefore between the two handoffs, with the baseline present to show that the sequence is tractable.

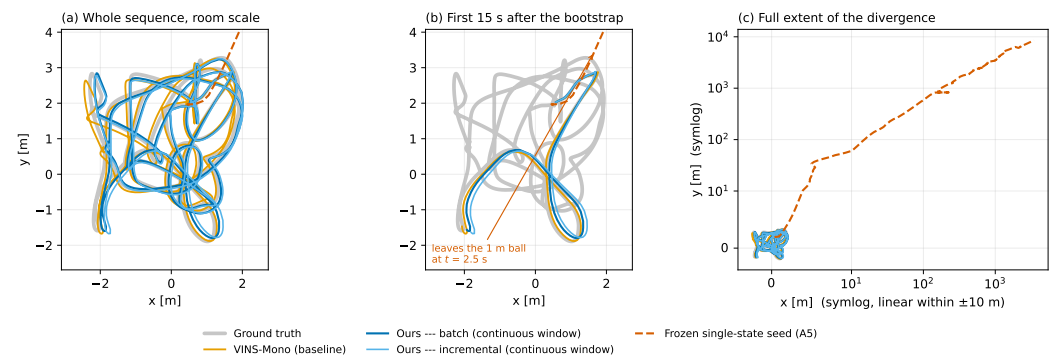


Figure 6. Trajectory overlay on V1_02_medium. (a) The whole sequence at room scale: both of our back-ends and the VINS-Mono baseline track the reference through the flight. (b) The first 15 s: all four start together, and the frozen-seed arm departs almost immediately. (c) The full extent, on a symmetric-log axis so that a five-meter room and a multi-kilometer excursion are visible at once. Every trajectory is Umeyama-aligned to the ground truth over the first 20 keyframes (the ≈ 2 s following acceptance of the bootstrap) instead of over the whole trajectory, so that the plot shows where each estimate departs from a common start. The whole-trajectory alignment used for the ATE of Table 8 minimizes total error instead and therefore reports a smaller displacement for the diverging arm.

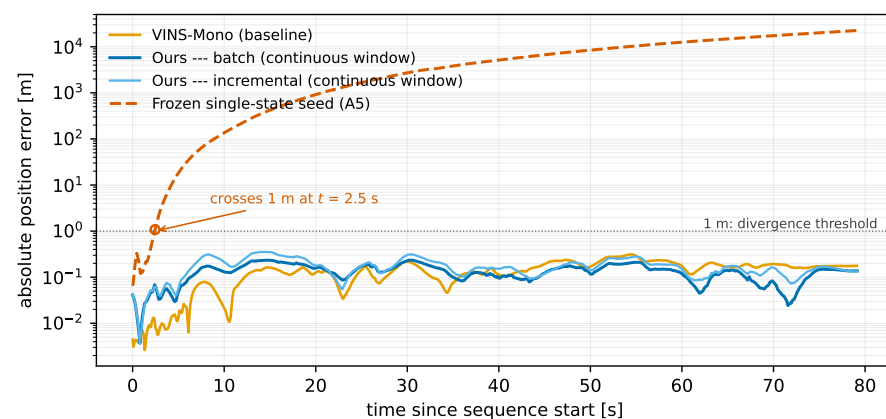


Figure 7. Absolute position error against time on V1_02_medium, logarithmic ordinate, same alignment as Figure 6. Both of our back-ends and the baseline stay between 0.01 and 0.3 m for the whole flight. The frozen-seed arm leaves the one-meter ball 2.5 s after the bootstrap is accepted and grows monotonically thereafter, by four orders of magnitude.

The error-over-time plot is the more diagnostic of the two. The frozen-seed arm does not drift away: it crosses the one-meter divergence threshold 2.5 s after the bootstrap is accepted and then grows monotonically, which is the signature of a mis-scaled, gravity-tilted seed being integrated forward and not of accumulating estimation error. Because that seed enters the graph as a fixed prior, no subsequent measurement can correct it. Admitting the whole bootstrap window as a re-optimizable sub-graph removes exactly this failure mode: the same window that produced the seed keeps being re-optimized while it remains in the smoother, so an opening that was merely acceptable can still be corrected before it slides out. This is the mechanism behind the 10/11 column of Table 8.

3.2.2. Initialization Behavior

Because the bootstrap is where a monocular estimator is most fragile, we report its behavior directly and not only through downstream ATE (Table 10).

Table 10. Initialization statistics per sequence: candidate windows evaluated, windows rejected by the pre-BA conditioning gate and by the velocity-consistency gate, the frame at which a window was finally accepted, and the metric scale of the accepted seed. Initialization succeeds on all eleven sequences. Bold marks the Total row, which aggregates the eleven per-sequence rows above it.

Sequence	Windows Tried	Rejected (Cond.)	Rejected (Vel.)	Accepted @ Frame	Seed Scale
MH_01 easy	49	48	0	79	0.245
MH_02 easy	93	92	0	144	0.121
MH_03 medium	36	35	0	113	0.140
MH_04 difficult	10	9	0	111	0.186
MH_05 difficult	68	67	0	455	0.285
V1_01 easy	35	34	0	239	0.246
V1_02 medium	11	10	0	109	0.149
V1_03 difficult	17	16	0	199	0.155
V2_01 easy	3	2	0	134	0.644
V2_02 medium	5	4	0	93	0.299
V2_03 difficult	3	2	0	146	0.637
Total	330	319	0	—	—

Initialization succeeds on 11/11 sequences under the single configuration. The number of rejected windows varies by more than an order of magnitude (2 to 92), which is a direct measure of how much the opening motion of each sequence violates the excitation assumption: MH_02 must slide 92 windows before finding a translation-rich one, whereas V2_01 and V2_03 find one almost immediately. The accepted seed scale spans 0.12–0.64, a five-fold spread, confirming that the metric information available in an opening window is highly sequence-dependent and that a fixed acceptance threshold, rather than a fixed window, is the appropriate design.

3.3. Estimator Consistency

Absolute error alone does not establish that an estimator's covariance is usable downstream, which matters here because the intended role of this odometry is to feed a larger factor graph that must weight it correctly. We therefore evaluate the Normalized Estimation Error Squared (NEES) per keyframe, using the 6-DoF pose marginal covariance and the body-frame residual $e = \text{Log}(T_{\text{est}}^{-1}T_{\text{gt}})$, which is invariant to the global alignment gauge. A consistent estimator has median NEES near the median of the corresponding χ^2 distribution. We compare against that median and not against the dimension, because the statistic reported here is itself a median and the two targets differ: the median of χ_6^2 is 5.35 against a mean of 6, and the median of χ_3^2 is 2.37 against a mean of 3. Using the dimension as the target would make the estimator appear better calibrated than it is. Values above indicate over-confidence, values below indicate under-confidence. Table 11 reports the result per sequence.

Three sequences (MH_03, V2_01, MH_04) are close to calibrated, with 74–90% of keyframes inside the 95% band. The remaining eight are over-confident, severely so on V1_01 (pose NEES 176, no keyframe inside the band). Measured against the median of χ_6^2 , only MH_03 sits below expectation, so median pose-NEES exceeds it on ten of the eleven sequences. Two observations follow. First, the excess is almost entirely rotational: on V1_01 the rotation component is 137.6 against a median expectation of 2.37, while translation is 1.5, i.e., below expectation. Second, over-confidence does not track ATE. V1_01 has the second best ATE in the suite (0.077 m) and the worst consistency. Accuracy and calibration are therefore separate axes, and reporting only ATE would hide this entirely.

Table 11. Per-keyframe NEES for the full system (configuration A0), sorted by pose NEES. Median over keyframes. The last column is the fraction of keyframes inside the 95% χ^2 band. Sequences are consistent where pose NEES ≈ 5.35 , the median of χ^2_6 . MH_01 and MH_02 were inadvertently omitted from this table in the previous revision and are restored here, and both exceed the expectation, so their inclusion makes the reported consistency worse and not better.

Sequence	Pose NEES (Dof 6)	Rotation (3)	Translation (3)	Within 95% Band
MH_03 medium	4.75	4.18	0.51	74%
V2_01 easy	5.48	4.70	0.22	90%
MH_04 difficult	6.93	5.66	0.49	85%
V1_02 medium	7.49	6.22	0.85	60%
MH_01 easy	11.68	5.56	1.38	59%
V2_02 medium	13.23	11.17	0.61	54%
V2_03 difficult	13.59	11.67	0.87	53%
MH_02 easy	26.10	24.00	1.35	16%
V1_03 difficult	43.10	35.54	1.71	3%
MH_05 difficult	46.56	39.88	3.19	6%
V1_01 easy	176.29	137.57	1.49	0%

The deficit is a property of how linearization points are managed, not of the measurement model. Repeating the covariance run on the incremental smoother, with the lag, the factors and the feature stream held identical so that only the solver differs, improves median pose-NEES from 13.2 to 8.6, better on ten of the eleven sequences (Wilcoxon $p = 0.032$), and raises the fraction of keyframes inside the 95% band from 54% to 65%. The mechanism is relinearization policy rather than marginalization: both back-ends Schur-complement the aged-out variables into a linear prior frozen at one linearization point, but the batch smoother relinearizes every retained variable at every step, whereas the incremental smoother relinearizes only those whose update exceeds its threshold and therefore leaves many variables closer to the point at which the prior was formed. That a change confined to the linearization schedule moves consistency by a quarter, while the measurement model, the noise densities and the observations are untouched, is what identifies the linearization point of marginalized information as the responsible quantity.

Three things stop this being a fix, and we report them together because separately each is misleading. The improvement is bought with accuracy in the same channel, the incremental smoother being the worse of the two on rotation drift on ten of eleven sequences (Table 6, $p = 0.002$), so the covariance grew faster than the error did and part of the gain is inflation rather than corrected information flow. Four incremental sequences overshoot below the median expectation of 5.35 and are under-confident, which is the signature inflation leaves. And the improvement does not reach the cases that matter: V1_01 falls only from 176 to 105, seven sequences remain over-confident, and MH_02 moves 41% the wrong way for reasons we cannot presently explain. The limitation is real on both solvers. What an uncontrolled relinearization schedule shifts by a quarter, a controlled treatment of the marginalized linearization point should address deliberately and without paying for it in accuracy, which is the argument for the direction named in the Limitations and Future Directions section and not evidence that it has been validated here.

This is the quantitative form of the limitation declared in Section 4: relinearizing a marginalized constraint at a moved estimate injects spurious information along the directions monocular visual-inertial estimation cannot observe—global position and yaw—and the rotational NEES is where that appears. It is also the reason First-Estimates Jacobians, which freeze the linearization point of marginalized variables, are the appropriate remedy and not a re-tuning of the measurement noise.

3.4. Computational Cost

All timings were taken serially on a single machine, one process at a time. The front-end cost is excluded because all systems consume the same pre-computed feature stream. The figures in this table were re-measured for this revision, on the current build and over a single recorded campaign covering both back-ends, and they replace those of the previous revision. Two differences account for the change and we state both rather than presenting the new numbers silently. The previous table reported the configuration with covariance dumping enabled, which is the configuration the consistency evaluation of Section 3.3 requires and not the one the odometry runs in. On this campaign that option costs 24% of the per-keyframe budget, a mean of 106.2 ms against 85.5 ms on the identical binary and sequences, so the table now reports the plain odometry configuration and Section 3.3 carries the cost of the instrumentation it needs. The remainder of the difference is that the whole campaign was re-run, so the absolute timings are those of the current build and machine rather than of the earlier one. Table 12 reports the mean per-keyframe cost and the mean peak resident memory over the eleven sequences, for both back-ends, with loop closure off.

Table 12. Per-keyframe back-end cost and peak resident memory for the full system (configuration A0), for both back-ends, with loop closure off, measured serially with one process on the machine at a time. The keyframe count is identical for the two back-ends because keyframe selection is a front-end gate that does not depend on the solver, so the two pairs of columns are measured on the same workload. Peak RSS is the whole-process maximum, so it includes the dataset index and the trajectory buffers as well as the factor graph. Bold marks the Mean row, which aggregates the eleven per-sequence rows above it.

Sequence	Keyframes	Batch		Incremental	
		[ms/KF]	Peak RSS [MB]	[ms/KF]	Peak RSS [MB]
MH_01 easy	634	88.6	126.2	39.7	163.4
MH_02 easy	534	89.9	114.1	38.4	142.9
MH_03 medium	646	109.0	110.6	45.0	162.3
MH_04 difficult	446	86.1	95.3	38.4	130.4
MH_05 difficult	361	90.2	99.8	37.4	141.6
V1_01 easy	565	80.7	105.8	39.0	156.1
V1_02 medium	715	111.5	93.6	47.9	162.6
V1_03 difficult	1131	79.6	90.1	34.9	135.8
V2_01 easy	430	69.4	96.7	34.8	138.0
V2_02 medium	889	75.7	96.3	33.6	140.1
V2_03 difficult	1063	60.0	87.8	26.6	126.5
Mean	674	85.5	101.5	37.8	145.4

The two back-ends do not trade accuracy against cost. Section 3.1.7 finds them statistically indistinguishable on ATE ($p = 0.765$), so the incremental smoother reaches the same accuracy as the batch smoother at 37.8 ms per keyframe against 85.5 ms, which is a factor of 2.3 in per-keyframe cost. What it pays for that is memory: its peak resident set is 145.4 MB against 101.5 MB, some 43% more, because iSAM2 retains the Bayes tree and its cached linear factors across updates whereas the batch smoother rebuilds the window each step and keeps nothing between them. The trade between the two back-ends is therefore time against memory, and the choice between them is a deployment budget question rather than an accuracy question.

Cost is bounded by the fixed lag and not by trajectory length, which is the property the fixed-lag design exists to provide. V1_03 and V2_03 carry roughly 2.4 times the keyframes of MH_04 at a lower per-keyframe cost on both back-ends (79.6 and 60.0 ms against 86.1

on the batch smoother, 34.9 and 26.6 against 38.4 on the incremental one), and their peak resident sets are the smallest in the table rather than the largest. Across the suite the keyframe count varies by a factor of 3.1 while per-keyframe cost varies by a factor of 1.9 and moves in the opposite direction, because the longer sequences are the ones whose keyframes are more widely spaced in a window of fixed length.

Neither back-end is recommended over the other here, because the evidence is a trade rather than a win and the right side of it depends on the consumer. A downstream global factor graph, which is the consumer this odometry is built for, has to weight the odometry by its covariance, and on that axis the incremental smoother is the better behaved of the two (Section 3.3) as well as the cheaper. An application that instead needs the smallest attitude drift, or the smaller resident footprint, is better served by the batch smoother. We report both and state the trade rather than declaring a default.

Both figures leave margin at the ≈ 4 Hz keyframe rate of these sequences, and both peak resident sets are within the memory of the companion computers typical of small aerial platforms. We emphasize, as in Section 1, that these are offline measurements on a desktop CPU with the estimate outside any control loop, so they characterize the cost of the algorithm and are not an on-platform timing result.

The study also prices robustness. The Huber-weighted arm costs 48% more wall clock than the pure-L2 arm (1129 s versus 590 s over the eleven sequences), which is the price of the +130% accuracy and two avoided failures reported in Table 8.

4. Discussion

This preliminary study asks whether a monocular VIO built on a GTSAM fixed-lag smoother, and hardened at its bootstrap, can generalize across an entire indoor-flight benchmark under one configuration per operating environment, and whether the resulting estimator is competitive with a mature reference. We read the results of Section 3 against those two questions, in the context of the monocular-VIO literature the system builds on.

Hardening the bootstrap yields single-configuration convergence. Three sequences (V1_02, V2_01, V2_02) previously diverged by tens to thousands of meters, not because tracking failed but because a visually plausible yet visual-inertially inconsistent opening window seeded a runaway accelerometer bias. The conditioning- and consistency-aware initialization (a pre-bundle-adjust reprojection test and a check that the recovered metric velocity is physically plausible) brings all three to sub-decimeter accuracy and lets one fixed configuration per operating environment converge on all eleven sequences. This turns the well-known requirement that monocular visual-inertial alignment is well conditioned only under a translation-rich trajectory [3,18] from an implicit assumption into an explicit admission test. The window it rejects is not a benchmark artifact but the pre-takeoff handling and takeoff transient that opens every flight, in which the camera sees rotation without baseline. Since a UAV closes its control loops on the estimate it is given, a mis-scaled or gravity-tilted seed is a departure from controlled flight and not a poor error statistic. Because the gate acts before any state enters the graph, and because a single threshold set holds across eleven differing openings, it is the element of this study we expect to transfer most readily, independent of the smoother behind it. Transfer is a hypothesis and not a measurement here, because all eleven openings were recorded by one sensor rig on one airframe, and the Limitations and Future Directions section states what would be required to test it.

The seeded system is a competitive point-only baseline. Driving the present system and both baselines from the same recorded PL-VINS front-end output [48] over the same inertial stream holds feature extraction constant. Against VINS-Mono, which consumes the same point channel, the odometry column differs only through the back-end. Against

PL-VINS it differs through the back-end and through the line measurements PL-VINS additionally carries. Under that control the estimator is the more accurate on ten of the eleven sequences against VINS-Mono, and on nine against VINS-Mono and PL-VINS simultaneously, by margins largest where the sequence is hard (MH_03 0.117 against 0.228 m, V2_03 0.120 against 0.306 m). Part of that margin comes from a per-observation reprojection rejection which, as in Kimera-VIO [14] and ORB-SLAM2 [13], removes grossly wrong associations before they become factors instead of merely down-weighting them. The paired signed-rank test of Section 3.1.7 confirms the margin against VINS-Mono ($p = 0.024$) and does not confirm it against PL-VINS ($p = 0.054$), so the claim we carry forward is that the estimator is competitive with both and measurably more accurate than one of them. That a hardened bootstrap and standard outlier handling together yield a system competitive with a mature reference, under one configuration per environment, is the evidence that the estimator is accurate and not merely convergent.

Loop closure, and where the contribution lies. The odometry margin above does not survive into the loop-closed columns, and that comparison is reported for completeness rather than claimed: the baselines pair their odometry with mature global pose graphs, whereas the 4-DoF layer here is a deliberately non-invasive prototype that never feeds back into the estimator. The narrowing is itself informative, because it localizes the contribution of this work in the odometry and the bootstrap, not in global relocalization. The residual gap reflects the maturity of the baseline's global pose graph against our deliberately non-invasive 4-DoF layer. MH_05, the longest and dimmest trajectory, is the one sequence on which our odometry trails, and on the two lowest-drift rooms a valid loop slightly increases ATE (the drift having fallen below the loop-edge noise floor), a regime that affects the baseline far more strongly (V2_02 0.119 $\rightarrow$ 0.187 m). The dominant residual on non-revisiting trajectories is accumulated yaw drift, the unobservable-direction floor of monocular VIO [18], and it is a property of the measurement model, not of the smoother.

A consistency limitation inherited from stock marginalization. The estimator deliberately adopts the stock fixed-lag marginalization of the GTSAM smoother. Its marginalization prior is a Gaussian frozen at one linearization point while the retained window keeps relinearizing, so re-evaluating those inherited constraints at a displaced estimate injects spurious information along the unobservable position and yaw, the pathology that motivated First-Estimates Jacobians (FEJs) [3,18]. Compounding this, an inverse-depth landmark whose anchor keyframe ages out is dropped and re-triangulated instead of re-anchored, discarding the information a long track accumulated. Together these are the mechanisms behind the residual drift that loop closure must remove, and they define the primary direction below.

Limitations and Future Directions

Preliminary scope. The evaluation is offline and open-loop. Per-keyframe cost and peak memory are now measured (Section 3.4), but on a desktop CPU and without the estimate in the control loop, so no claim of real-time or embedded readiness is made. The initialization thresholds (the pre-bundle-adjust reprojection bound, the excitation check, and the velocity-consistency gate) are set once per environment and held fixed and not adapted online, and how they transfer to platforms and motions outside EuRoC is part of the on-platform validation below.

Output cadence. The estimator publishes at keyframe rate rather than at camera rate. Section 3.1.1 shows that this costs a bounded penalty on frame-indexed relative-pose error, most of which disappears under a cadence-invariant delta, but a consumer that needs a smooth high-rate pose—a flight controller, most obviously—would require the IMU-propagated inter-keyframe stream that both baselines emit. Supplying it is a matter of

publishing the preintegrated state between optimizations and does not alter the estimator or any result reported here.

A consistent estimator (primary direction). The consistency measurements of Section 3.3 make this direction concrete and not speculative. The over-confidence is almost entirely rotational, which is the signature of information injected along the unobservable yaw direction when a marginalized constraint is relinearized at a moved estimate. The next step is therefore an FEJ-consistent, keyframe-indexed marginalization that freezes the linearization point of variables entering the marginal prior and folds each landmark's accumulated information into the retained window, a component GTSAM does not ship natively. Its benefit should surface on the consistency axis rather than on raw ATE, so NEES and not ATE is the metric by which it must be judged. The results reported here provide the uncalibrated reference against which such a smoother would have to demonstrate an improvement. A structurally minimal points-only baseline such as this one is the precondition for isolating that effect, and building the consistent estimator on it is the core of the continuing project.

Loop closure that cannot harm the odometry. The measured deficit lies in the loop-edge layer, not the front-end. A pose graph improves a trajectory only when its edge weights reflect the true relative uncertainty of the two sources it fuses. In the prototype reported here both the odometry backbone and the loop edges carry hand-set noise values, so on sequences where the odometry is already accurate the graph can pull a good trajectory towards a less certain loop constraint. The principled alternative is to derive the backbone edge noise from the estimator's own relative-pose covariance between the two keyframes an edge connects, instead of assigning a fixed per-edge value. This makes the graph's trust in the odometry proportional to the odometry's actual confidence, which is the quantity Section 3.3 shows must first be calibrated, and it is the mechanism by which loop closure can be made a strict improvement and not a trade. It also connects the two directions above: a consistent estimator is what makes a covariance-weighted pose graph trustworthy. Folding loop-match observations into the live window as projection factors [3] would additionally let a correction refine the real-time pose, not only the post-hoc trajectory.

Richer geometric primitives. Warehouse interiors (racking, plant rooms, corridors) are dominated by low-texture linear and planar structure on which a point-only front-end starves. Line features in the manner of PL-VINS [48], and ultimately planes, would supply constraints there, each admitted only where it earns its computational cost.

A single benchmark, and why the evaluation stops there. Every quantitative result in this paper comes from one public benchmark. That is a deliberate choice about the stage of the work rather than an omission, and it bounds the claims in a specific way we would rather state than leave to the reader.

The reason for the choice is that a second dataset is only informative if the systems it is compared against are calibrated on it to the same standard. The controlled comparison this paper rests on holds the feature stream, the inertial stream and the ground truth identical across four estimators (Section 3), which is what allows a difference to be attributed to the back-end rather than to a tuning effort. Reproducing that control on a second platform means porting, calibrating and tuning two reference systems onto hardware they were never configured for, and at this stage of the work the outcome of such a comparison would say more about how much effort was spent calibrating each baseline than about the estimators. A public benchmark supplies exactly the property that in-house data cannot: an evaluation whose baselines the community has already calibrated, and whose numbers a reader can check against the literature. This is why the eleven EuRoC sequences remain the appropriate evidence for an estimation study at this stage, and it is also why they are indoor flight recordings rather than a hand-carried or synthetic dataset.

What that choice does not buy is generalization across platforms, and no such claim is made. The eleven sequences were recorded by one visual-inertial rig on one airframe in two environments, so a single configuration holding across all eleven demonstrates transfer across scenes, motions and lighting, and does not demonstrate transfer across sensors, calibrations or vibration signatures. The statistical inference of Section 3.1.7 is bounded the same way: it licenses a conclusion over these eleven sequences and not over indoor flight in general, and the eleven-sequence sample is itself the reason the PL-VINS comparison cannot be resolved at the 5% level. Adding independent sequences is therefore the direct way to strengthen both the accuracy claim and the transfer claim, and it is the purpose of the on-platform campaign described next.

On-platform and closed-loop validation. The eleven sequences used here are indoor flight tests, recorded onboard a micro air vehicle flying in an industrial hall and two instrumented rooms, and the difficult sequences deliberately carry aggressive attitude rates, motion blur and illumination transitions. What they do not cover is specific rather than general: they were flown on a different airframe, so the vibration signature that couples into the IMU is not ours, and they are replayed offline, so the estimate never closed a control loop. The decisive remaining test of the initialization mechanism is therefore a powered takeoff on our MANTIS platform [4], with the estimate inside the control loop and running on embedded compute, and establishing which thresholds survive that regime precedes deployment. Validation on a public benchmark is therefore necessary but not sufficient for a system of this kind, and we state that explicitly instead of leaving it implied: the results reported here establish that the estimator is sound on the standard benchmark, not that it is ready to fly.

What is claimed, and on what evidence. Because the claims of this paper are bounded in several different ways, Table 13 collects them in one place: each claim as it appears in the text, the evidence that supports it, and the neighboring statement that is deliberately not asserted. The word “deployment” appears in this paper in exactly two roles, to name the motivating warehouse application and to name future work, and in neither role does it describe a result reported here.

Table 13. Claims of this paper against the evidence that supports them and the scope outside which they are not asserted.

Claim	Evidence	Not Claimed
Converges on all eleven sequences under one configuration per operating environment	Table 1, eleven of eleven bounded	Transfer to another airframe, sensor or calibration
More accurate than VINS-Mono on pure odometry	Table 7, paired signed-rank $p = 0.024$	The same against PL-VINS, which is $p = 0.054$ and unresolved at $n = 11$
Convergence is carried by the continuous-window handoff	Table 8, arm A5 diverges ten of eleven	That either initialization threshold carries it, which the same table refutes
Bounded and measured per-keyframe cost	Table 12, 85.5 and 37.8 ms per keyframe, 101.5 and 145.4 MB peak	Real-time or embedded feasibility, these being desktop measurements outside any control loop
The estimator is over-confident, and rotationally so	Table 11, pose NEES above expectation on ten of eleven	Any claim of a calibrated covariance

5. Conclusions

This paper reported a preliminary study of a tightly coupled monocular point-feature VIO for GPS-denied indoor UAV flight, realized on a GTSAM fixed-lag factor graph and developed within the DRIVOLUTION project as the first estimation stage of an autonomous stock-management UAV for indoor logistics warehouses. Its decisive element is a bootstrap hardened for the degenerate takeoff transient. Building on the visual-inertial alignment of VINS-Mono, a conditioning gate and a continuous-window initialization refuse ill-conditioned opening windows before they can seed a mis-scaled or gravity-tilted state. Evaluated on all eleven EuRoC MAV indoor flight sequences against two baselines, VINS-Mono and PL-VINS, both run locally on the same feature stream, one fixed configuration per operating environment converges on every sequence (including three that otherwise diverge by tens to thousands of meters) and the resulting system attains the better pure-odometry accuracy on nine of the eleven against both baselines simultaneously, with ATE RMSE of 0.12–0.37 m on the Machine Hall and sub-decimeter on most Vicon rooms. A paired signed-rank test over the eleven sequences confirms that margin against VINS-Mono ($p = 0.024$) and does not confirm it against PL-VINS ($p = 0.054$), and it finds the two back-ends statistically indistinguishable on accuracy ($p = 0.765$), so what separates them is computational cost and not accuracy. Disabling each mechanism in turn locates the source of that convergence: it is carried by the continuous-window handoff, not by either initialization threshold. Substituting the conventional frozen single-state seed diverges ten of the eleven sequences, whereas neither threshold diverges any, and the velocity-consistency check never fires on this benchmark at all.

The scope of the study is deliberately bounded. It is offline and open-loop, and while per-keyframe cost and peak memory are now measured (Section 3.4), they are desktop measurements taken with the estimate outside any control loop, so no real-time or embedded claim is made. Loop closure is likewise reported for completeness and not claimed: the pose graph is a prototype, and the odometry tier is what this paper stands on. Its value is a reproducible, honestly bounded point-only baseline built on a robust initialization, together with a measured identification of the estimator's principal limitation, a stock fixed-lag marginalization that is not first-estimate consistent and discards accumulated landmark information, the residual-drift mechanism a loop-closure layer must later absorb. That limitation is no longer only declared: the median pose-NEES exceeds its six-degree-of-freedom expectation on ten of the eleven sequences, and the excess is almost entirely rotational, which is the signature the First-Estimates-Jacobian remedy targets. Closing that gap with an FEJ-consistent, information-preserving marginalization, and ultimately validating the system in closed-loop flight on our own platform, is the direct continuation of this work toward a consistent estimator for onboard warehouse navigation.

Author Contributions: Conceptualization, methodology, software, validation, formal analysis, investigation and writing—original draft G.A.A.; data curation and visualization, R.S.; investigation and resources, J.J.M.; supervision, project administration and funding acquisition, A.D. and J.A. writing—review and editing, A.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially funded by the DRIVOLUTION project (grant no. 02/C05-i01.01/2022.PC644913740-00000022), a Mobilising Agenda under Portugal's Recovery and Resilience Plan (PRR), and by INESC TEC, Institute for Systems and Computer Engineering, Technology and Science.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The study uses the publicly available EuRoC MAV dataset (ETH Zurich; <https://projects.asl.ethz.ch/datasets/euroc-mav/> (accessed on 5 September 2026)). The estimator source code and the exact configuration files for every result reported here, including the seven configurations of Section 3.2 in both of their back-end variants, are openly available at https://github.com/n0t1cs/gtsam_vio_pub (accessed on 5 September 2026).

Acknowledgments: This work is co-financed by Component 5 - Capitalization and Business Innovation, integrated in the Resilience Dimension of the Recovery and Resilience Plan within the scope of the Recovery and Resilience Mechanism (MRR) of the European Union (EU), framed in the Next Generation EU, for the period 2021–2026, within project Drivolution, with reference 23. The authors acknowledge the Autonomous Systems Laboratory (LSA) at INESC TEC for providing access to facilities and equipment. During the preparation of this manuscript, the authors used AI tools for editing assistance, sentence reformulation and structural reorganisation of the text. No experimental data, results or figures were generated by AI. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest. The funding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

APE	Absolute Pose Error
ATE	Absolute Trajectory Error
BA	Bundle Adjustment
COLAMD	Column Approximate Minimum Degree (ordering)
CPU	Central Processing Unit
DLT	Direct Linear Transform
DoF	Degrees of Freedom
DSO	Direct Sparse Odometry
EKF	Extended Kalman Filter
EuRoC	European Robotics Challenge (MAV benchmark dataset)
FEJ	First-Estimates Jacobians
GPS	Global Positioning System
GTSAM	Georgia Tech Smoothing and Mapping
IMU	Inertial Measurement Unit
iSAM	Incremental Smoothing and Mapping
KLT	Kanade-Lucas-Tomasi (feature tracker)
LC	Loop Closure
LiDAR	Light Detection and Ranging
MAP	Maximum a Posteriori
MAV	Micro Aerial Vehicle
MSCKF	Multi-State Constraint Kalman Filter
NEES	Normalized Estimation Error Squared
ORB	Oriented FAST and Rotated BRIEF
PL-VINS	Point-Line Visual-Inertial System
PnP	Perspective-n-Point
QR	QR matrix factorization
RANSAC	Random Sample Consensus
RGB-D	Red-Green-Blue plus Depth
RMS	Root Mean Square
RMSE	Root Mean Square Error
RPE	Relative Pose Error
RSS	Resident Set Size (peak process memory)

SfM	Structure from Motion
SLAM	Simultaneous Localization and Mapping
SVO	Semi-direct Visual Odometry
UAV	Unmanned Aerial Vehicle
VIO	Visual-Inertial Odometry
YAML	YAML Ain't Markup Language

References

1. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. <https://doi.org/10.1109/TRO.2016.2624754>.
2. Forster, C.; Carlone, L.; Dellaert, F.; Scaramuzza, D. On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. *IEEE Trans. Robot.* **2017**, *33*, 1–21. <https://doi.org/10.1109/tro.2016.2597321>.
3. Qin, T.; Li, P.; Yang, Z.; Shen, S. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. <https://doi.org/10.1109/TRO.2018.2853729>.
4. Dias, A.; Martins, J.J.; Antunes, J.; Moura, A.; Almeida, J. MANTIS: UAV for Indoor Logistic Operations. In Proceedings of the 2024 7th Iberian Robotics Conference (ROBOT), Madrid, Spain, 6–8 November 2024; pp. 1–7. <https://doi.org/10.1109/ROBOT61475.2024.10796935>.
5. Dissanayake, G.; Newman, P.; Clark, S.; Durrant-Whyte, H.; Csorba, M. A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. *IEEE Trans. Robot. Autom.* **2001**, *17*, 229–241. <https://doi.org/10.1109/70.938381>.
6. Davison, A.J.; Reid, I.; Molton, N.; Stasse, O. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. <https://doi.org/10.1109/tpami.2007.1049>.
7. Dellaert, F.; Kaess, M. Factor Graphs for Robot Perception. *Found. Trends Robot.* **2017**, *6*, 1–139. <https://doi.org/10.1561/23000000043>.
8. Mourikis, A.I.; Roumeliotis, S.I. A Multi-State Constraint Kalman Filter for Vision-Aided Inertial Navigation. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Rome, Italy, 10–14 April 2007; pp. 3565–3572.
9. Geneva, P.; Eickenhoff, K.; Lee, W.; Yang, Y.; Huang, G. OpenVINS: A Research Platform for Visual-Inertial Estimation. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 4666–4672.
10. Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. G2o: A general framework for graph optimization. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China 9–13 May 2011; pp. 3607–3613. <https://doi.org/10.1109/ICRA.2011.5979949>.
11. Agarwal, S.; Mierle, K.; The Ceres Solver Team. *Ceres Solver*, version 2.2; GitHub: San Francisco, CA, USA, 2023. Available online: <https://github.com/ceres-solver/ceres-solver> (accessed on 5 September 2026).
12. Burri, M.; Nikolic, J.; Gohl, P.; Schneider, T.; Rehder, J.; Omari, S.; Achtelik, M.W.; Siegwart, R. The EuRoC Micro Aerial Vehicle Datasets. *Int. J. Robot. Res.* **2016**, *35*, 1157–1163. <https://doi.org/10.1177/0278364915620033>.
13. Mur-Artal, R.; Tardós, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. <https://doi.org/10.1109/TRO.2017.2705103>.
14. Rosinol, A.; Abate, M.; Chang, Y.; Carlone, L. Kimera: An Open-Source Library for Real-Time Metric-Semantic Localization and Mapping. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020.
15. Civera, J.; Davison, A.J.; Montiel, J.M. Inverse Depth Parametrization for Monocular SLAM. *IEEE Trans. Robot.* **2008**, *24*, 932–945. <https://doi.org/10.1109/tro.2008.2003276>.
16. Thrun, S.; Burgard, W.; Fox, D. *Probabilistic Robotics*; MIT Press: Cambridge, MA, USA, 2005.
17. Grisetti, G.; Stachniss, C.; Burgard, W. Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters. *IEEE Trans. Robot.* **2007**, *23*, 34–46. <https://doi.org/10.1109/tro.2006.889486>.
18. Huang, S.; Dissanayake, G. A Critique of Current Developments in Simultaneous Localization and Mapping. *Int. J. Adv. Robot. Syst.* **2016**, *13*, 1729881416669482. <https://doi.org/10.1177/1729881416669482>.
19. Barrau, A.; Bonnabel, S. The Invariant Extended Kalman Filter as a Stable Observer. *IEEE Trans. Autom. Control* **2017**, *62*, 1797–1812. <https://doi.org/10.1109/tac.2016.2594085>.
20. Hartley, R.; Ghafoori, M.; Eustice, R.M.; Grizzle, J.W. Contact-Aided Invariant Extended Kalman Filtering for Robot State Estimation. *Int. J. Robot. Res.* **2020**, *39*, 402–430. <https://doi.org/10.1177/0278364919894385>.
21. Carlone, L.; Kim, A.; Barfoot, T.; Cremers, D.; Dellaert, F. *SLAM Handbook: From Localization and Mapping to Spatial Intelligence*; Cambridge University Press: Cambridge, UK, 2025. Available online: <https://hdl.handle.net/1721.1/163400> (accessed on 5 September 2026).

22. Bai, F.; Vidal-Calleja, T.; Grisetti, G. Sparse Pose Graph Optimization in Cycle Space. *IEEE Trans. Robot.* **2021**, *37*, 1381–1400. <https://doi.org/10.1109/tro.2021.3050328>.
23. Kaess, M.; Ranganathan, A.; Dellaert, F. iSAM: Incremental smoothing and mapping. *IEEE Trans. Robot.* **2008**, *24*, 1365–1378. <https://doi.org/10.1109/TRO.2008.2006706>.
24. Kaess, M.; Johannsson, H.; Roberts, R.; Ila, V.; Leonard, J.; Dellaert, F. iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree. *IEEE Trans. Robot.* **2012**, *24*, 365–1378.
25. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.M.; Tardós, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. <https://doi.org/10.1109/TRO.2021.3075644>.
26. Engel, J.; Koltun, V.; Cremers, D. Direct Sparse Odometry. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 611–625. <https://doi.org/10.1109/TPAMI.2017.2658577>.
27. Forster, C.; Pizzoli, M.; Scaramuzza, D. SVO: Fast semi-direct monocular visual odometry. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 15–22. <https://doi.org/10.1109/ICRA.2014.6906584>.
28. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An Efficient Alternative to SIFT or SURF. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
29. Karami, E.; Prasad, S.; Shehata, M. Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images. *arXiv* **2017**. <https://doi.org/10.48550/ARXIV.1710.02726>.
30. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2003.
31. Li, M.; Mourikis, A.I. High-Precision, Consistent EKF-Based Visual-Inertial Odometry. *Int. J. Robot. Res.* **2013**, *32*, 684–701. <https://doi.org/10.1177/0278364913481251>.
32. Du, X.; Zhang, L.; Ji, C.; Luo, X.; Zhang, H.; Wang, M.; Wu, W.; Mao, J. SP-VIO: Robust and Efficient Filter-Based Visual Inertial Odometry with State Transformation Model and Pose-Only Visual Description. *arXiv* **2025**, arXiv:2411.07551.
33. Xiao, Z.; Li, S. A real-time, robust and versatile visual-SLAM framework based on deep learning networks. *arXiv* **2024**, arXiv:2405.03413.
34. Peng, Q.; Xiang, Z.; Fan, Y.; Zhao, T.; Zhao, X. RWT-SLAM: Robust Visual SLAM for Highly Weak-textured Environments. *arXiv* **2022**, arXiv:2207.03539.
35. Yan, J.; Zheng, Y.; Yang, J.; Mihaylova, L.; Yuan, W.; Gu, F. PLPF-VSLAM: An indoor visual SLAM with adaptive fusion of point-line-plane features. *J. Field Robot.* **2024**, *41*, 50–67. <https://doi.org/10.1002/rob.22242>.
36. Liu, Y.; Feng, Z.; Zhang, H.; Dong, W. Post-integration based point-line feature visual SLAM in low-texture environments. *Sci. Rep.* **2025**, *15*, 10534. <https://doi.org/10.1038/s41598-025-97250-6>.
37. Shan, T.; Englot, B.; Meyers, D.; Wang, W.; Ratti, C.; Rus, D. LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 5135–5142. <https://doi.org/10.1109/IROS45743.2020.9341176>.
38. Zheng, T.; Wang, F.; Xu, Z. An Improved GTSAM-based Nonlinear Optimization Algorithm in ORBSLAM3. In Proceedings of the 2022 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), Hengyang, China, 26–27 March 2022; pp. 30–33. <https://doi.org/10.1109/ICITBS55627.2022.00016>.
39. Demmel, N.; Schubert, D.; Sommer, C.; Cremers, D.; Usenko, V. Square Root Marginalization for Sliding-Window Bundle Adjustment. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 13240–13248. <https://doi.org/10.1109/ICCV48922.2021.01301>.
40. Martinez-Martin, E.; Ferrer, E.; Vasilev, I.; del Pobil, A.P. The UJI Aerial Librarian Robot: A Quadcopter for Visual Library Inventory and Book Localisation. *Sensors* **2021**, *21*, 1079. <https://doi.org/10.3390/s21041079>.
41. Alajami, A.A.; Moreno, G.; Pous, R. Design of a UAV for Autonomous RFID-Based Dynamic Inventories Using Stigmergy for Mapless Indoor Environments. *Drones* **2022**, *6*, 208. <https://doi.org/10.3390/drones6080208>.
42. Stanko, J.; Stec, F.; Rodina, J. Process Automation of Warehouse Inspection Using an Autonomous Unmanned Aerial Vehicle. *MM Sci. J.* **2022**, *2022*, 5864–5869. https://doi.org/10.17973/MMSJ.2022_10_2022063.
43. Huang, J.; Junginger, S.; Liu, H.; Thurow, K. Indoor Positioning Systems of Mobile Robots: A Review. *Robotics* **2023**, *12*, 47. <https://doi.org/10.3390/robotics12020047>.
44. Servièrès, M.; Renaudin, V.; Dupuis, A.; Antigny, N. Visual and Visual-Inertial SLAM: State of the Art, Classification, and Experimental Benchmarking. *J. Sens.* **2021**, *2021*, 2054828. <https://doi.org/10.1155/2021/2054828>.
45. Lai, T. A Review on Visual-SLAM: Advancements from Geometric Modelling to Learning-Based Semantic Scene Understanding Using Multi-Modal Sensor Fusion. *Sensors* **2022**, *22*, 7265. <https://doi.org/10.3390/s22197265>.
46. Firmansyah, R.A.; Mardiyanto, R.; Sardjono, T.A. Improvement Monocular ORB SLAM for Indoor Drone 3D Localization Using Inertial Navigation System. In Proceedings of the 2024 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), BALI, Indonesia, 4–6 July 2024; pp. 348–353. <https://doi.org/10.1109/IAICT62357.2024.10617662>.

47. Matthee, J.; Uren, K.; Van Schoor, G.; Van Daalen, C. Predicting the performance of ORB-SLAM3 on embedded platforms. *S. Afr. Comput. J.* **2024**, *36*, 84–102. <https://doi.org/10.18489/sacj.v36i2/20099>.
48. Fu, Q.; Wang, J.; Yu, H.; Ali, I.; Guo, F.; He, Y.; Zhang, H. PL-VINS: Real-Time Monocular Visual-Inertial SLAM with Point and Line Features. *arXiv* **2022**, arXiv:2009.07462.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.