Efficient Incremental SLAM via Information-Guided Gating and Selective Partial Optimization

Reza Arablouei *

Posted Date: 18 March 2026

doi: 10.20944/preprints202603.1477.v1

Keywords: simultaneous localization and mapping (SLAM); graph SLAM; incremental SLAM; pose graph optimization; nonlinear least-squares; computational efficiency; real-time robotics; selective partial optimization; information-guided gating; loop closure

 doi:10.20944/preprints202603.1477.v1

Article

Efficient Incremental SLAM via Information-Guided Gating and Selective Partial Optimization

Reza Arablouei

Commonwealth Scientific and Industrial Research Organisation (CSIRO), Pullenvale QLD 4069, Australia; reza.arablouei@csiro.au

Abstract

We present an efficient incremental SLAM back-end that achieves the accuracy of full batch optimization while substantially reducing computational cost. The proposed approach combines two complementary ideas: information-guided gating (IGG) and selective partial optimization (SPO). IGG employs an information-theoretic criterion based on the log-determinant of the information matrix to quantify the contribution of new measurements, triggering global optimization only when a significant information gain is observed. This avoids unnecessary relinearization and factorization when incoming data provide little additional information. SPO executes multi-iteration Gauss-Newton (GN) updates but restricts each iteration to the subset of variables most affected by the new measurements, dynamically refining this active set until convergence. Together, these mechanisms retain all measurements to preserve global consistency while focusing computation on parts of the graph where it yields the greatest benefit. We provide a theoretical local perturbation analysis showing that, under standard regularity assumptions for GN, the proposed approach tracks full GN up to a neighborhood whose size is controlled by the approximation thresholds. Moreover, when the effective approximation error introduced by localization and screening vanishes asymptotically, it recovers the same local minimizer and asymptotic convergence rate as full GN. Extensive experiments on benchmark SLAM datasets show that our approach consistently matches the estimation accuracy of batch solvers, while achieving significant computational savings compared to conventional incremental approaches. Such efficiency is particularly important for mobile robots operating under onboard compute constraints, where timely state estimation is critical for localization, mapping, and downstream navigation and control. The results indicate that the proposed approach offers a principled balance between accuracy and efficiency, making it a robust and scalable solution for real-time robotic localization and mapping in dynamic, data-rich environments.

Keywords: simultaneous localization and mapping (SLAM); graph SLAM; incremental SLAM; pose graph optimization; nonlinear least-squares; computational efficiency; real-time robotics; selective partial optimization; information-guided gating; loop closure

1. Introduction

Simultaneous localization and mapping (SLAM) is a fundamental capability for autonomous robots, enabling the continuous estimation of both a robot's pose and the surrounding environment map. Modern SLAM systems often adopt graph-based nonlinear optimization formulations, in which robot poses are represented as nodes and spatial constraints as edges in a factor graph [1–3]. Compared to filtering-based methods, this full smoothing approach typically yields higher accuracy and improved consistency [4], but also leads to a continually expanding estimation problem. In long-term, data-rich deployments, the number of poses and measurements can grow without bound, substantially increasing memory and computation requirements. Repeatedly solving the full SLAM problem from scratch becomes prohibitive for real-time applications, motivating extensive research into incremental SLAM methods that update the solution efficiently as new data arrive [5].

 doi:10.20944/preprints202603.1477.v1

Early SLAM algorithms were predominantly based on recursive filtering, with the extended Kalman filter (EKF) as a widely used example. Although EKF-SLAM supports real-time operation, its dense covariance representation leads to $\mathcal{O}(N^2)$ complexity, and accumulated linearization errors can degrade map consistency over time. Delayed-state filters mitigate computational load by maintaining only a fixed-size window of recent keyframes [6], but as the window grows to preserve accuracy, costs again become significant. In contrast, smoothing-based SLAM formulates the problem as a nonlinear least-squares optimization over a pose graph [7], exploiting the sparsity of the underlying information matrix.

Early smoothing and mapping (SAM) approaches, such as Square Root SAM [4], demonstrated that solving the full SLAM problem via sparse linear algebra (e.g., QR or Cholesky factorization of the information matrix) improves both accuracy and efficiency compared to EKF-based methods. Open-source frameworks such as g2o [8] and GTSAM [9] have further streamlined deployment by providing efficient graph construction, factorization, and solver interfaces. Beyond direct solvers, iterative approaches [10–12] leverage sparse matrix–vector products to reduce computational cost, and specialized fast pose-graph optimizers [13] can handle poor initial estimates effectively.

Among incremental SAM algorithms, iSAM [14] is a landmark contribution, maintaining the square-root information matrix and updating it with new measurements via low-rank sparse matrix factor updates [15]. This enables real-time updates while reusing previously computed structure. However, the original iSAM required periodic batch relinearization and variable reordering to maintain consistency, effectively performing occasional full optimizations to correct linearization drift. Its successor, iSAM2 [16], addressed these limitations by introducing the Bayes tree, a junction-tree-based data structure that supports fluid relinearization and incremental variable reordering. iSAM2 uses a threshold-based wildfire strategy to relinearize only variables affected by new information, eliminating the need for expensive batch resets while preserving accuracy.

Other advances in incremental SLAM include robust solvers and alternative inference strategies. The RISE algorithm [17] incorporates a trust-region method (Powell's dog-leg) into an incremental framework, improving robustness to strong nonlinearities and ill-conditioned systems while maintaining speeds comparable to Gauss-Newton (GN) methods. NF-iSAM [18] extends incremental smoothing to non-Gaussian estimation by using normalizing flows to represent arbitrary posteriors, retaining the sparsity and efficiency of iSAM2's Bayes tree under non-Gaussian measurement models. Robustness has also been pursued in riSAM [19], which leverages graduated non-convexity [20–22] to handle outliers and non-convexity. In parallel, works such as incremental Cholesky factorization [23] and AprilSAM [24] have focused on improving update efficiency through algorithmic refinements that accelerate matrix updates. Collectively, these developments reflect the SLAM community's drive toward back-ends that are both online-efficient and robust under real-world conditions.

Despite these advances, achieving scalable SLAM in highly dynamic, data-rich environments remains challenging. Robots operating over extended durations or equipped with high-rate sensors, such as dense visual or LiDAR systems, can accumulate a continuous stream of measurements, many of which are redundant or only marginally informative. Incorporating every such measurement into the pose graph and triggering a full solver update at each increment is computationally inefficient and can compromise real-time performance.

To alleviate this burden, researchers have explored graph sparsification and measurement selection based on information content. Several works [25–27] employ information-theoretic criteria to construct sparse yet reliable pose graphs that approximate the estimation quality of the full graph, for example by selecting a near-D-optimal subset of loop closures or constraints that maximizes the determinant of the Fisher information matrix.

Related efforts in information-driven graph sparsification and active SLAM [28] have used principled quality measures such as the log-determinant of the Fisher information matrix (D-optimality) and algebraic connectivity, i.e., the Fiedler eigenvalue of the graph Laplacian. Representative examples include information-theoretic loop-closure detection [29], factor-based node marginalization [30],

 doi:10.20944/preprints202603.1477.v1

and conservative edge sparsification [31]. Similarly, [32] proposed a spectral sparsification method that retains edges maximizing algebraic connectivity, a property correlated with SLAM accuracy. By maximizing the Fiedler value of the measurement graph, their method seeks to preserve global consistency while reducing the number of stored constraints.

Information-theoretic reduction has also been applied at the state level, where the goal is to decrease problem size by selecting a compact subset of poses or landmarks rather than only sparsifying edges. For example, [33] proposed information-based reduced landmark SLAM, which selects a reduced set of landmarks and poses while preserving estimation quality, and also described an incremental variant of the approach. These works further illustrate the value of principled information measures for controlling SLAM complexity. However, their primary objective is to reduce the size of the state or graph itself. In contrast, our approach retains all measurements and preserves the full estimation problem, using information gain only to determine when a global update is necessary and where computation should be concentrated within the optimization back-end.

Taken together, these studies show that substantial computational savings can be achieved through graph pruning, edge selection, or reduced-state formulations guided by principled information measures. However, permanently discarding measurements may compromise consistency or robustness when the selection is imperfect [34]. Moreover, many existing sparsification methods operate offline or as a separate preprocessing stage, rather than being tightly integrated into the live optimization loop to determine, in real time, whether newly arrived information warrants a full global update or only a localized partial update.

1.1. Contributions

We propose an efficient incremental SLAM framework that integrates information-based variable selection directly into the optimization back-end, thereby avoiding unnecessary computations while preserving global consistency. The proposed approach comprises two key components: (i) information-guided gating (IGG), and (ii) selective partial optimization (SPO). Beyond algorithmic innovations, we also provide a theoretical local perturbation analysis that characterizes how the proposed components affect the convergence behavior of GN optimization.

- *Information-Guided Gating (IGG)*: An information-theoretic mechanism that monitors the change in the log-determinant of the information matrix to evaluate the contribution of new measurements. Only when the predicted information gain exceeds a threshold is a global update triggered. Otherwise, optimization is restricted to local variables directly affected by the new measurements.
- *Selective Partial Optimization (SPO)*: A multi-iteration nonlinear GN solver that, at each iteration, updates and relinearizes only the variables that have not yet converged. This subset is dynamically refined based on convergence thresholds and graph connectivity, focusing computation where it yields the greatest gains.
- *Theoretical Analysis*: A local perturbation analysis showing that, after anchoring the SLAM problem and under standard local regularity conditions for GN, the proposed IGG-SPO scheme tracks full GN up to a neighborhood whose size is controlled by the approximation thresholds. The analysis also clarifies that the same local minimizer and asymptotic convergence rate as full GN are recovered when the effective approximation error vanishes asymptotically.

Unlike sparsification methods, which may discard information and thereby compromise global consistency, our approach retains all measurements while restricting optimization to the updates that matter most. The result is a robust and scalable SLAM back-end that achieves accuracy close to batch-level optimization in practice at a fraction of the per-update computational cost. In contrast to iSAM2, which heuristically limits relinearization yet still solves a global system at every increment, our approach offers a more principled alternative: by unifying information-guided gating with selective partial optimization, it focuses computation on the variables most affected by new information, while the analysis in Section 4 characterizes the resulting deviation from full GN and the conditions under which the same local behavior is recovered.

 doi:10.20944/preprints202603.1477.v1

From a robotics perspective, the proposed method is particularly relevant to autonomous ground, aerial, and service robots that must maintain accurate localization and mapping over long missions while operating under limited onboard computational resources. By reducing unnecessary global updates and focusing computation on the most affected variables, the proposed back-end supports reliable real-time state estimation in SLAM-enabled robotic systems deployed in large-scale and sensor-rich environments.

2. Background

SLAM involves the joint estimation of a robot's trajectory and a map of the environment. Modern SLAM back-ends typically formulate this as a large-scale nonlinear least-squares optimization over a *pose graph*, where nodes represent robot poses (and possibly landmarks) and edges represent spatial measurements between them. Consider a set of measurement residuals

$$\mathbf{r}_j = \mathbf{m}_j - \mathbf{f}_j(\mathbf{x}_{\mathcal{V}_j}), \tag{1}$$

where $\mathbf{m}_j$ is the j-th observed measurement (e.g., a relative pose or landmark observation), $\mathbf{f}_j(\cdot)$ is the measurement prediction function, and $\mathbf{x}_{\mathcal{V}_j}$ denotes the subset of state variables involved in measurement j (indexed by i). Assuming Gaussian measurement noise with covariance $\mathbf{\Sigma}_j$, the maximum a posteriori (MAP) estimate can be obtained by minimizing the cost function

$$c(\mathbf{x}) = \frac{1}{2}\sum_{j=1}^{M} \|\mathbf{r}_j\|^2_{\mathbf{\Sigma}_j} = \frac{1}{2}\sum_{j=1}^{M} \|\mathbf{m}_j - \mathbf{f}_j(\mathbf{x}_{\mathcal{V}_j})\|^2_{\mathbf{\Sigma}_j}, \tag{2}$$

where $\|\mathbf{r}_j\|^2_{\mathbf{\Sigma}_j} \triangleq \mathbf{r}_j^{\mathsf{T}}\mathbf{\Sigma}_j^{-1}\mathbf{r}_j$.

This nonlinear least-squares problem is typically solved using iterative methods such as the GN algorithm or its Levenberg-Marquardt variant. Starting from an initial guess, GN linearizes the measurement functions around the current state estimate, producing the linearized normal equations

$$(\mathbf{J}^{\mathsf{T}}\mathbf{J})\mathbf{d} = \mathbf{J}^{\mathsf{T}}\mathbf{r}, \tag{3}$$

where $\mathbf{J}$ is the stacked Jacobian of all residuals with respect to the full state vector $\mathbf{x}$, and $\mathbf{r}$ is the stacked residual vector. Solving for $\mathbf{d}$ yields the state increment, and the estimate is updated as $\mathbf{x} \leftarrow \mathbf{x} - \mathbf{d}$. This process is repeated (with relinearization at each step) until convergence.

Graph-based SLAM problems exhibit strong structural sparsity that can be exploited for computational efficiency. The information matrix (the approximate Hessian) $\mathbf{H} = \mathbf{J}^{\mathsf{T}}\mathbf{J}$ is typically large but sparse and block-structured, reflecting the local connectivity of the pose graph: each variable i (pose or landmark) is directly linked to only a few others through shared measurements j. In pure exploration scenarios without loop closures, the sparsity pattern is approximately block-tridiagonal. When loop closures occur, long-range links introduce additional nonzero blocks (fill-in), but the matrix remains sparse. Furthermore, in many SLAM problems, the diagonal blocks (self-information from priors or odometry) dominate the off-diagonal terms, leading to a form of near block-diagonal dominance. This sparsity implies that naive dense linear algebra is inefficient: factorizing a dense $N \times N$ matrix costs $\mathcal{O}(N^3)$, whereas exploiting sparsity can reduce the complexity dramatically. Efficient SLAM back-ends therefore rely on sparse matrix factorization or iterative methods to leverage this structure.

In practice, the linear system arising in each GN iteration can be solved either (i) *directly*, through sparse matrix factorization (e.g., Cholesky or QR) followed by forward/backward substitution, or (ii) *iteratively*, using solvers such as conjugate gradient that exploit sparse matrix-vector products. In this work, we adopt a direct sparse factorization approach because it offers high numerical accuracy, facilitates efficient reuse of factors in incremental updates, and enables rapid computation of information-theoretic quantities (e.g., log-determinants) that are essential for our IGG strategy.

 doi:10.20944/preprints202603.1477.v1

3. Proposed Approach

The proposed approach, summarized in Algorithm 1, is an incremental nonlinear least-squares optimizer for SLAM that integrates two key ideas of information-guided gating (IGG) and selective partial optimization (SPO). We formulate SLAM as a pose-graph optimization problem and maintain, throughout execution, the sparse Cholesky factor $\mathbf{R}$ of the information matrix $\mathbf{H} = \mathbf{J}^\top \mathbf{J}$. The algorithm operates in discrete increments, where each increment corresponds to the arrival of one or more new measurements. The framework supports both batch mode (processing multiple measurements received in rapid succession) and streaming mode (processing measurements one at a time). At each increment t, the algorithm determines which variables to update and how the update is performed, according to the following procedure.

3.1. Graph Update and Initial Linearization

When new measurements arrive, we first update the pose graph $\mathcal{G}_t$ by adding the corresponding edges, which connect the relevant robot poses or landmarks. If these edges introduce new poses or landmarks (and hence new variables), we also append their initial estimates to the state vector. We denote the total number of state variables after incorporating the new measurements as N_t. Subsequently, we linearize the new edges about the current state estimate, $\mathbf{x}_{t-1}$. Specifically, we compute the Jacobians of the new measurements with respect to the involved variables, evaluated at their current estimates. This process augments the system by adding new rows to the Jacobian $\mathbf{J}_t$ and new entries to the residual vector $\mathbf{r}_t$, and, if new variables are present, by adding corresponding columns to $\mathbf{J}_t$.

To efficiently incorporate these changes, we perform a low-rank Cholesky update to extend the existing factor $\mathbf{R}_{t-1}$ to $\mathbf{R}_t$. This produces the updated linear system $(\mathbf{R}_t^\top \mathbf{R}_t)\mathbf{d}_t = \mathbf{b}_t$, where $\mathbf{b}_t = \mathbf{J}_t^\top \mathbf{r}_t$. To preserve sparsity and reduce fill-in during factorization, we incrementally update the variable ordering $\boldsymbol{\pi}_t$ using the constrained column approximate minimum degree (CCOLAMD) algorithm [35]. In addition, we incrementally update the elimination tree associated with $\mathbf{R}_t$, denoted by $\mathbf{p}_t$, to reflect changes in the graph structure.

3.2. Information-Guided Gating

We define an information-theoretic quantity η_t to measure the overall information content of the system after incorporating the new measurements. Specifically, let

$$\eta_t = \sum_{i=1}^{N_t} \ln |\rho_{t,i}|, \tag{4}$$

where $\rho_{t,i}$ is the i-th diagonal entry of $\mathbf{R}_t$. Since $\mathbf{H}_t = \mathbf{J}_t^\top \mathbf{J}_t$ is symmetric positive definite after anchoring, we have

$$\sum_i \ln |\rho_{t,i}| = \ln \det(\mathbf{R}_t) = \frac{1}{2} \ln \det(\mathbf{J}_t^\top \mathbf{J}_t). \tag{5}$$

Therefore, η_t is equal to half the log-determinant of the information matrix $\mathbf{H}_t$. This corresponds to a D-optimality criterion, which we adopt because it captures the overall uncertainty volume of the estimate [36] and can be evaluated efficiently from the triangular factor already computed by the sparse GN solve. By contrast, A-optimality emphasizes the average variance through $\mathrm{tr}(\mathbf{H}_t^{-1})$, whereas E-optimality emphasizes the worst-estimated direction through the minimum eigenvalue of $\mathbf{H}_t$. We do not claim that D-optimality is universally preferable, but it provides a natural global summary of estimator uncertainty and aligns well with the incremental sparse factorization available in our implementation. The identity above follows directly from the Cholesky factorization of a symmetric positive definite matrix [37, Lecture 23].

 doi:10.20944/preprints202603.1477.v1

To isolate the effect of the new measurements from the trivial growth in η_t caused by the introduction of new variables, we compute the detrended change

$$\Delta\eta_t = \eta_t - \frac{N_{t-1}}{N_t}\eta_{t-1} \tag{6}$$

and compare it against a preset threshold τ_η. If $\Delta\eta_t < \tau_\eta$, the added measurements are deemed insufficiently informative to justify a full global update. In this case, only the variables directly involved in the new edges are marked as *potentially affected*, and their indices are stored in the set $\mathcal{S}_t$. This avoids unnecessary global re-optimization in situations such as pure odometry updates that contribute little incremental information.

Algorithm 1 Incremental SLAM with Information-Guided Gating and Selective Partial Optimization

Require: initial estimate $\mathbf{x}_0$,
thresholds τ_η (information gain), τ_d (increment magnitude), τ_{GN} (max GN iterations)
Ensure: updated estimate $\mathbf{x}_t$ after each increment t

for $t = 1, 2, \ldots$ **do**
  incorporate new measurements into $\mathcal{G}_{t-1}$ to form $\mathcal{G}_t$
  $\mathcal{S}_t \leftarrow$ variables involved in new measurements
  update $\mathbf{J}_t$, $\mathbf{r}_t$, $\mathbf{R}_t$, $\mathbf{b}_t$, $\boldsymbol{\pi}_t$, and $\mathbf{p}_t$ incrementally based on new measurements
  $\eta_t \leftarrow \sum_{i=1}^{N_t} \ln|\text{diag}(\mathbf{R}_t)|$
  if $\eta_t - \frac{N_{t-1}}{N_t}\eta_{t-1} > \tau_\eta$ **then**
    $\mathcal{S}_t \leftarrow \{1, \ldots, N_t\}$
  end if
  for $i_{\text{GN}} = 1, 2, \ldots, \tau_{\text{GN}}$ **do**
    solve $(\mathbf{R}_t^\top \mathbf{R}_t)\mathbf{d} = \mathbf{b}_t$ over $\mathcal{S}_t$ to obtain $\mathbf{d}_{\mathcal{S}_t}$
    update $\mathcal{S}_t$ based on $|\mathbf{d}_{\mathcal{S}_t}| > \tau_d$ and the connectivity in $\mathcal{G}_t$
    if $\mathcal{S}_t = \emptyset$ **then** break
    end if
    $\mathbf{x}_{\mathcal{S}_t} \leftarrow \mathbf{x}_{\mathcal{S}_t} - \mathbf{d}_{\mathcal{S}_t}$
    update $\mathbf{J}_t$, $\mathbf{r}_t$, $\mathbf{R}_t$, and $\mathbf{b}_t$ for edges involving $\mathcal{S}_t$
  end for
end for

Conversely, if $\Delta\eta_t \geq \tau_\eta$, the new measurements are deemed sufficiently informative to potentially influence the entire graph, for example in the case of a loop closure or a high-accuracy pose prior. We therefore set $\mathcal{S}_t = \{1, 2, \ldots, N_t\}$, marking all variables as *potentially affected*. While a more selective choice is possible, such as restricting attention to the connected component impacted by the new edges, practical SLAM graphs typically form a single connected component, and determining the exact subset of globally affected variables can itself be computationally expensive, potentially offsetting the benefits of selectivity.

At this stage, we identify only the *potentially affected* variables. The subset of *actually affected* variables, i.e., those that are updated and relinearized, is determined later after the partial-solve stage described in Section 3.3, using the procedure in Section 3.4.

3.3. Selective Partial Solve

Given the current set of *potentially affected* variables $\mathcal{S}_t$ (identified from the IGG step or carried over from the previous GN iteration), we solve the linear system arising in the GN update:

$$(\mathbf{R}_t^\top \mathbf{R}_t)\mathbf{d}_t = \mathbf{b}_t. \tag{7}$$

If $\mathcal{S}_t$ contains all variables, (7) is solved by standard forward-backward substitution:

$$\mathbf{R}_t^\top \mathbf{y} = \mathbf{b}_t, \quad \mathbf{R}_t \mathbf{d}_t = \mathbf{y}. \tag{8}$$

 doi:10.20944/preprints202603.1477.v1

When $|\mathcal{S}_t| \ll N_t$, it is more efficient to restrict computation to the dynamic subset $\mathcal{S}_t$ while treating the remaining variables $\mathcal{U}_t$ as static. For clarity, we temporarily drop the increment index t and denote these sets simply as $\mathcal{S}$ (dynamic) and $\mathcal{U}$ (static), with $\mathcal{U} = \{1, \ldots, N\} \setminus \mathcal{S}$. While the global variable ordering remains fixed by the factorization, we locally permute rows and columns of $\mathbf{R}$ and $\mathbf{b}$ to form the block structure

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_{\mathcal{U}\mathcal{U}} & \mathbf{R}_{\mathcal{U}\mathcal{S}} \\ \mathbf{0} & \mathbf{R}_{\mathcal{S}\mathcal{S}} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} \mathbf{b}_{\mathcal{U}} \\ \mathbf{b}_{\mathcal{S}} \end{bmatrix}, \text{ and } \mathbf{d} = \begin{bmatrix} \mathbf{d}_{\mathcal{U}} \\ \mathbf{d}_{\mathcal{S}} \end{bmatrix}. \tag{9}$$

The linear system $(\mathbf{R}^\top \mathbf{R})\mathbf{d} = \mathbf{b}$ then takes the block form:

$$\begin{bmatrix} \mathbf{R}_{\mathcal{U}\mathcal{U}}^\top \mathbf{R}_{\mathcal{U}\mathcal{U}} & \mathbf{R}_{\mathcal{U}\mathcal{U}}^\top \mathbf{R}_{\mathcal{U}\mathcal{S}} \\ \mathbf{R}_{\mathcal{U}\mathcal{S}}^\top \mathbf{R}_{\mathcal{U}\mathcal{U}} & \mathbf{R}_{\mathcal{S}\mathcal{S}}^\top \mathbf{R}_{\mathcal{S}\mathcal{S}} + \mathbf{R}_{\mathcal{U}\mathcal{S}}^\top \mathbf{R}_{\mathcal{U}\mathcal{S}} \end{bmatrix} \begin{bmatrix} \mathbf{d}_{\mathcal{U}} \\ \mathbf{d}_{\mathcal{S}} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{\mathcal{U}} \\ \mathbf{b}_{\mathcal{S}} \end{bmatrix}. \tag{10}$$

Since the unaffected (static) variables remain fixed (i.e., $\mathbf{d}_{\mathcal{U}} = \mathbf{0}$), the first block row of (10) becomes

$$\mathbf{R}_{\mathcal{U}\mathcal{U}}^\top \mathbf{R}_{\mathcal{U}\mathcal{S}} \mathbf{d}_{\mathcal{S}} = \mathbf{b}_{\mathcal{U}}. \tag{11}$$

By introducing the intermediate vector

$$\mathbf{y}_{\mathcal{U}} = \mathbf{R}_{\mathcal{U}\mathcal{S}} \mathbf{d}_{\mathcal{S}}, \tag{12}$$

we can transform (11) into the triangular linear system

$$\mathbf{R}_{\mathcal{U}\mathcal{U}}^\top \mathbf{y}_{\mathcal{U}} = \mathbf{b}_{\mathcal{U}} \tag{13}$$

This equation can be solved efficiently by exploiting the cached $\mathbf{R}_{\mathcal{U}\mathcal{U}}$. Substituting (12) into the second block row of (10) gives

$$\begin{aligned} (\mathbf{R}_{\mathcal{S}\mathcal{S}}^\top \mathbf{R}_{\mathcal{S}\mathcal{S}}) \mathbf{d}_{\mathcal{S}} &= \mathbf{b}_{\mathcal{S}} - (\mathbf{R}_{\mathcal{U}\mathcal{S}}^\top \mathbf{R}_{\mathcal{U}\mathcal{S}}) \mathbf{d}_{\mathcal{S}} && (14) \\ &= \mathbf{b}_{\mathcal{S}} - \mathbf{R}_{\mathcal{U}\mathcal{S}}^\top \mathbf{y}_{\mathcal{U}}, && (15) \end{aligned}$$

Here, the term $-\mathbf{R}_{\mathcal{U}\mathcal{S}}^\top \mathbf{y}_{\mathcal{U}}$ represents the correction contributed by the static block's equations. Intuitively, even though the static variables are held fixed, their residual constraints still exert an influence, which propagates to the dynamic variables through this coupling term.

By caching the factorization and solver associated with the static block $\mathbf{R}_{\mathcal{U}\mathcal{U}}$ (i.e., intermediate Schur complements), redundant computations are avoided across successive GN iterations, yielding substantial efficiency gains. This strategy is conceptually analogous to restricting computation to the Markov blanket[1], thereby localizing updates to the region of the pose graph most impacted by the new measurements. Consequently, the partial solver computes the increments $\mathbf{d}_{\mathcal{S}_t}$ efficiently by reusing cached static-block solutions and solving only the reduced dynamic block through the Schur complement. This significantly lowers computational cost and enables scalable incremental SLAM.

3.4. Determining Affected Variables

After each partial solve, we determine which variables still require updates and thus remain relevant for the next GN iteration. This amounts to updating the active set $\mathcal{S}_t$ through two complementary steps: pruning converged variables and expanding to maintain consistency.

[1] In a pose graph, the Markov blanket of a node comprises its directly connected neighbors. Conditioning on this set renders the node independent of all others. Similarly, in SLAM, the effect of new measurements propagates primarily through the immediate neighbors of the affected variables. The cached block–Schur complement method leverages this principle: only variables reachable from the affected set via the elimination tree are re-solved, while the rest are left untouched.

 doi:10.20944/preprints202603.1477.v1

Pruning: Variables whose increments are sufficiently small are considered converged and are removed from the active set. Formally, we apply the threshold τ_d to the increment vector $\mathbf{d}_{\mathcal{S}_t}$, retaining only those indices with significant updates, i.e.,

$$\mathcal{S}_t \leftarrow \{ i \in \mathcal{S}_t : |d_{t,i}| > \tau_d \}, \tag{16}$$

where $d_{t,i}$ denotes the i-th entry of $\mathbf{d}_{\mathcal{S}_t}$. Since poses and landmarks are represented by blocks of variables, we apply the pruning conservatively at the block level: if any variable within a node is retained, we keep the entire block of that node.

Expansion: Adjustments to the variables in $\mathcal{S}_t$ can induce new inconsistencies in the variables of neighboring nodes through their shared measurements (edges). To capture this effect, we collect all edges incident to the nodes whose variables are in $\mathcal{S}_t$:

$$\mathcal{E}_t = \bigcup_{n:\, \mathrm{vars}(n) \subseteq \mathcal{S}_t} \mathrm{edges}(n), \tag{17}$$

where $\mathrm{vars}(n)$ denotes the variables of node n and $\mathrm{edges}(n)$ its incident edges. We then enlarge the active set to include the variables of all nodes participating in these edges:

$$\mathcal{S}_t \leftarrow \mathcal{S}_t \cup \bigcup_{e \in \mathcal{E}_t} \bigcup_{n \in \mathrm{ends}(e)} \mathrm{vars}(n). \tag{18}$$

To build intuition for the expansion step, consider an updated variable $x_i \in \mathcal{S}_t$ that participates in a scalar measurement $f_{i,k}(x_i, x_k)$ with a neighbor $x_k \notin \mathcal{S}_t$. The residual for this edge is

$$r_{ik} = f_{i,k}(x_i, x_k), \tag{19}$$

with Jacobians $J_i = \frac{\partial f_{i,k}}{\partial x_i}$ and $J_k = \frac{\partial f_{i,k}}{\partial x_k}$. If x_i is updated by an increment d_i, the residual changes approximately as

$$\Delta r_{i,k} \approx J_i d_i. \tag{20}$$

For the edge to remain consistent, x_k would need to be adjusted by an increment d_k such that

$$J_i d_i + J_k d_k \approx 0. \tag{21}$$

Hence, if $|J_i d_i|$ exceeds a threshold, the neighbor x_k must also be included in $\mathcal{S}_t$. This illustrates how updates propagate through measurement connections, motivating the expansion rule.

In summary, the updated active set after each partial solve consists of 1) variables whose increments remain significant, 2) their full node blocks, and 3) all neighboring variables connected through incident edges. This prune-expand cycle ensures that only genuinely affected parts of the graph are revisited in the next iteration, balancing accuracy and efficiency.

3.5. Selective Partial Optimization

Once the active set of variables $\mathcal{S}_t$ has been initialized at increment t from the new measurements (and possibly expanded via IGG, cf. Section 3.2), GN iterations are carried out selectively until either $\mathcal{S}_t$ becomes empty or the maximum number of iterations τ_{GN} is reached. Each iteration proceeds as follows:

1. *Partial solve*: compute the GN step for the active variables by solving $(\mathbf{R}_t^\top \mathbf{R}_t)\mathbf{d} = \mathbf{b}_t$ restricted to $\mathbf{d}_{\mathcal{S}_t}$, i.e., the increments for the current active set, as described in section 3.3.
2. *Active-set update*: prune converged variables (those with $|d_{t,i}| \leq \tau_d$) and expand to include additional affected variables, preserving block structure and respecting pose-graph connectivity, to obtain the updated $\mathcal{S}_t$, as detailed in Section 3.4.
3. *Convergence check*: terminate the GN iterations if $\mathcal{S}_t = \emptyset$.

 doi:10.20944/preprints202603.1477.v1

4. *State update*: update the current estimate for the active variables, $\mathbf{x}_{\mathcal{S}_t} \leftarrow \mathbf{x}_{\mathcal{S}_t} - \mathbf{d}_{\mathcal{S}_t}$.
5. *Relinearization*: recompute Jacobians and residuals only for edges involving variables in $\mathcal{S}_t$. Since all other variables remain fixed, this requires updating only the corresponding rows of $\mathbf{J}_t$ and entries of $\mathbf{R}_t$ and $\mathbf{b}_t$. This can be implemented as a lightweight refactorization of the affected portions of $\mathbf{R}_t$, e.g., via sparse low-rank factor update techniques or localized Bayes tree refactorization.

This loop terminates when all increments fall below tolerance τ_d or τ_{GN} iterations have been performed. The outcome is an updated state estimate $\mathbf{x}_t$ that approximately minimizes the nonlinear least-squares cost after including the new measurements at increment t. Crucially, we also maintain an updated factor $\mathbf{R}_t$, valid for the current linearization around $\mathbf{x}_t$, which is reused in subsequent increments, thus avoiding costly refactorization from scratch.

To build intuition, consider two limiting scenarios:

- *Highly informative increments:* If each new measurement provides strong information (e.g., frequent loop closures), the information gain $\Delta\eta_t$ will exceed the threshold τ_η, and $\mathcal{S}_t$ will initially include all variables. If all entries of $\mathbf{d}_t$ are as large as τ_d, this effectively becomes batch optimization at each increment, ensuring maximum accuracy but at high computational cost.
- *Weakly informative increments:* If new measurements add little information (e.g., small odometry steps or redundant observations), only a small subset of recent variables enters $\mathcal{S}_t$. Updates are then highly localized, with minimal computational effort.

In practice, the behavior lies between these extremes. Odometry typically yields small or negligible updates, while loop closures or globally informative measurements trigger broader updates. The thresholds τ_η and τ_d play a stabilizing role: small improvements accumulate until τ_η is exceeded, at which point more variables are included and multiple GN iterations propagate corrections globally. This amortizes computation across increments and yields scalable performance in incremental SLAM.

3.6. Computational Complexity

Each increment in our approach consists of up to τ_{GN} GN iterations, each involving: (i) relinearization of a subset of variables, (ii) incremental (rank) Cholesky updates or downdates to the factor $\mathbf{R}_t$, and (iii) a partial solution of the underlying system of linear equations restricted to the affected variables. By keeping τ_{GN} small (e.g., 5-10), we bound the per-increment cost similarly to iSAM2's fluid relinearization scheme [16], but with an important distinction: computations are focused strictly on the active subset $\mathcal{S}_t$, which is typically a small fraction of the full state. This yields substantial savings in large graphs, where loop closures are infrequent or mostly local.

The main computational effort arises from two operations:

- *Cholesky up(down)date*: When a variable is relinearized, the corresponding columns of $\mathbf{R}_t$ are modified. The cost of these operations is approximately

$$\min\left(2\sum_{i\in\mathcal{S}_t}\kappa_{t,i}^2,\ \sum_{i=1}^{N}\kappa_{t,i}^2\right), \tag{22}$$

  where $\kappa_{t,i}$ denotes the number of nonzeros in column i of $\mathbf{R}_t$. This expression shows that the update cost is always bounded by that of a full refactorization. For new edges, the cost is halved as no downdate is required.
- *Partial solve*: Once the system has been updated, the increment for the affected variables is computed via sparse triangular forward and backward substitutions. The cost is roughly

$$2\sum_{i\in\mathcal{S}_t}\kappa_{it}, \tag{23}$$

 doi:10.20944/preprints202603.1477.v1

which scales linearly with the number of nonzeros in the relevant columns of $\mathbf{R}_t$. Equivalently, this corresponds to solving a small linear system restricted to the Markov blanket of $\mathcal{S}_t$, whose size is typically modest.

Because $\mathbf{R}_t$ is sparse and exhibits low fill-in under good orderings (e.g., planar or chain-like SLAM graphs), these operations are highly efficient in practice. Other operations, Jacobian construction, symbolic updates (e.g., reordering, elimination tree maintenance), and evaluation of information gain, incur negligible overhead compared to matrix factorization and solves.

Over T increments, the total cost is driven by the cumulative size of the affected subsets $\mathcal{S}_t$. In the common case where loop closures or new observations only modify a bounded region of the graph, the accumulated cost scales linearly with T. In contrast, worst-case behavior, such as every increment triggering a global update, results in a cost proportional to T full solves, though this is rare in real-world SLAM scenarios.

Hence, the overall cost depends on the total number of affected variables across all increments, rather than the total number of variables. In realistic scenarios, most updates remain local, and only a small fraction of increments trigger global corrections. This yields near-linear cumulative complexity in T.

In our implementation, we set $\tau_{\text{GN}} = 10$. Larger values yield diminishing returns within a single increment, since very large loop closures are uncommon in well-designed SLAM trajectories. If $\tau_{\text{GN}} = 1$, the method reduces to an iSAM2-style update (one linear solve per step, with occasional full relinearization, e.g., when τ_η is exceeded). At the other extreme, setting $\tau_\eta = 0$ (always update) and choosing large τ_{GN} reproduces batch bundle adjustment at every step, which is accurate but prohibitively expensive. Our approach therefore spans the spectrum between naive incremental and full batch optimization, with thresholds τ_η and τ_d governing the trade-off between efficiency and accuracy.

As shown in Section 5, our information-guided gating allows most increments to proceed with localized updates only, while limiting the number of global updates to a small fraction of T. This preserves accuracy while substantially reducing runtime, leading to performance that is competitive with or better than existing incremental solvers, particularly on large graphs with frequent local updates.

4. Theoretical Analysis

In this section, we analyze the two components at the core of the proposed approach: IGG and SPO. Under the standard local assumptions for GN, IGG can be viewed as switching between a full GN step and a localized inexact GN step, while SPO can be viewed as screening small components of the GN *increment*. The key point is that the SPO rule is applied to the GN step $\mathbf{d}$, not to the gradient $\nabla c(\mathbf{x})$.

Our goal here is to provide a local perturbation analysis that reflects the implemented algorithm as faithfully as possible. In particular, we distinguish between two regimes. With fixed positive thresholds τ_η and τ_d, the most natural result is convergence to a small neighborhood of the full-GN solution. Exact equivalence with full GN, including convergence to the same local minimizer with the same asymptotic rate, requires the effective approximation error introduced by localization and screening to vanish asymptotically. This viewpoint is consistent with the substantial FLOP savings reported in Section 5.

Recall the cost function (2) and let

$$c(\mathbf{x}) = \frac{1}{2}\|\mathbf{r}(\mathbf{x})\|^2, \qquad \mathbf{g}(\mathbf{x}) = \nabla c(\mathbf{x}) = \mathbf{J}(\mathbf{x})^\top \mathbf{r}(\mathbf{x}),$$

where $\mathbf{J}(\mathbf{x})$ is the Jacobian of $\mathbf{r}(\mathbf{x})$. To avoid the gauge ambiguity inherent in SLAM, we assume that the problem has been anchored by fixing a reference frame or by imposing an equivalent anchoring constraint. Throughout the analysis, we assume that the residual functions $\mathbf{r}_j$ in (2) are twice continu-

 doi:10.20944/preprints202603.1477.v1

ously differentiable in a neighborhood of a local minimizer $\mathbf{x}^\star$, and that $\mathbf{J}(\mathbf{x})$ has full column rank in that neighborhood. Equivalently, the GN normal matrix

$$\mathbf{H}(\mathbf{x}) = \mathbf{J}(\mathbf{x})^\top \mathbf{J}(\mathbf{x})$$

is positive definite near $\mathbf{x}^\star$. We further assume that the exact Hessian $\nabla^2 c(\mathbf{x})$ is Lipschitz-continuous in a neighborhood of $\mathbf{x}^\star$, i.e.,

$$\left\|\nabla^2 c(\mathbf{x}) - \nabla^2 c(\mathbf{y})\right\| \leq L \left\|\mathbf{x} - \mathbf{y}\right\|, \quad \forall\, \mathbf{x}, \mathbf{y} \text{ near } \mathbf{x}^\star, \tag{24}$$

for some constant $L > 0$. Finally, we assume that the iterates remain in this neighborhood so that the standard local convergence theory for GN applies.

In practical SLAM problems, rank deficiency or ill-conditioning may arise from gauge freedom, weak geometric constraints, or insufficient excitation. Our theoretical analysis assumes that the problem has been properly anchored and that the Jacobian is full column rank in the local region of interest, so that the corresponding GN normal matrix is positive definite. When these conditions are not met, the proposed method can still be used in conjunction with standard numerical safeguards such as damping or regularization, but the local convergence guarantees stated here no longer apply directly.

4.1. IGG as a Localized Inexact GN Step

Let $\mathbf{d}_t^{\mathrm{GN}}$ denote the full GN step at increment t, defined by

$$\mathbf{H}(\mathbf{x}_{t-1})\, \mathbf{d}_t^{\mathrm{GN}} = \mathbf{g}(\mathbf{x}_{t-1}). \tag{25}$$

Let $\mathcal{S}'$ denote the set of variables involved in the newly arrived measurements at increment t. If the information gain $\Delta\eta_t$ falls below the threshold τ_η, the algorithm does not trigger a full graph update; instead, it computes a localized step supported on $\mathcal{S}'$. Thus, the IGG rule determines whether the method uses a full GN step or a localized approximation to it.

To analyze this mechanism, let $\widehat{\mathbf{d}}_t$ denote the step produced after the IGG decision, i.e., either the full GN step if $\Delta\eta_t \geq \tau_\eta$, or the localized step if $\Delta\eta_t < \tau_\eta$. A natural analytical model is to treat the localized solve as an inexact GN step. Specifically, we assume that, whenever $\Delta\eta_t < \tau_\eta$, the localized step satisfies the forcing-term condition

$$\left\|\mathbf{H}(\mathbf{x}_{t-1})\, \widehat{\mathbf{d}}_t - \mathbf{g}(\mathbf{x}_{t-1})\right\| \leq \xi_t \left\|\mathbf{g}(\mathbf{x}_{t-1})\right\|, \tag{26}$$

where $0 \leq \xi_t \leq \bar{\xi} < 1$. This is the standard residual condition used in inexact Newton analysis [38]. In our setting, $\Delta\eta_t$ serves as a practical trigger for deciding when a localized solve is used, whereas (26) is the analytical condition under which that localized solve can be treated as an inexact GN step. We emphasize that (26) is not derived directly from $\Delta\eta_t$; rather, it is an assumption that captures the local approximation quality required for the theory.

Under (26), classical inexact Newton theory implies local convergence properties analogous to those of full GN. In particular, a uniformly bounded forcing term yields local linear convergence, while asymptotically vanishing forcing terms recover the local asymptotic rate of full GN [38]. We do not restate that result here as a separate theorem because the combined analysis below provides a more direct characterization of the implemented IGG+SPO scheme.

 doi:10.20944/preprints202603.1477.v1

4.2. SPO as a Screened GN Step

When the IGG trigger fires, the method can further reduce computation by solving only for variables whose current GN *increments* are sufficiently large. Let $\mathbf{d}_{t,l}^{\mathrm{GN}}$ denote the full GN step associated with the l-th GN iteration within increment t, i.e.,

$$\mathbf{H}(\mathbf{x}_{t,l-1})\,\mathbf{d}_{t,l}^{\mathrm{GN}} = \mathbf{g}(\mathbf{x}_{t,l-1}), \tag{27}$$

and define the active set by

$$\mathcal{S}_{t,l} = \Big\{ i \in \{1,\dots,N_t\} \;:\; |d_{t,l,i}^{\mathrm{GN}}| > \tau_d \Big\}. \tag{28}$$

The SPO update solves the GN subproblem restricted to $\mathcal{S}_{t,l}$, while the remaining coordinates are left unchanged.

This should be viewed as a *screened* or *truncated* GN step rather than a gradient-thresholding rule. The intuition is that coordinates with small GN increments have limited immediate influence on the current iterate and can therefore be omitted temporarily with only a small perturbation to the full step. To formalize this, let $\widetilde{\mathbf{d}}_{t,l}$ denote the restricted step produced by SPO. We assume that the screening error is controlled in the sense that

$$\|\widetilde{\mathbf{d}}_{t,l} - \mathbf{d}_{t,l}^{\mathrm{GN}}\| \le C_d\,\tau_d \tag{29}$$

for some constant $C_d > 0$ in a neighborhood of $\mathbf{x}^\star$. This expresses the fact that omitting coordinates whose GN increments are below τ_d perturbs the full GN step by at most $O(\tau_d)$.

As with IGG, the bound (29) is an analytical assumption rather than a direct consequence of the thresholding rule alone. Nevertheless, its form is well aligned with the structure of the method: the discarded coordinates are precisely those whose full GN increments are no larger than τ_d, so under local smoothness and conditioning assumptions it is natural to model the resulting truncation error as being proportional to τ_d.

A useful implication of (29) is that, if τ_d is held fixed, then the discrepancy between the SPO step and the full GN step does not generally vanish near the solution. Therefore, one should not expect exact asymptotic equivalence with full GN for arbitrary fixed $\tau_d > 0$. Instead, the more natural conclusion is that the method tracks full GN up to an $O(\tau_d)$ neighborhood. Recovering the same local minimizer and the same asymptotic rate as full GN requires the effective SPO perturbation to vanish asymptotically, for example through a decreasing threshold or a more explicit residual-based acceptance rule.

4.3. Combined Perturbation Analysis of IGG+SPO

The preceding discussion suggests that the implemented algorithm is most naturally analyzed as a perturbation of full GN. For notational simplicity, we now use a single iteration index k to denote successive accepted GN-type updates. Let

$$\mathbf{H}_k = \mathbf{J}(\mathbf{x}_k)^\top \mathbf{J}(\mathbf{x}_k), \qquad \mathbf{g}_k = \mathbf{J}(\mathbf{x}_k)^\top \mathbf{r}(\mathbf{x}_k),$$

and let the full GN step $\mathbf{d}_k^{\mathrm{GN}}$ be defined by

$$\mathbf{H}_k \mathbf{d}_k^{\mathrm{GN}} = \mathbf{g}_k. \tag{30}$$

The corresponding full-GN iterate is

$$\mathbf{x}_{k+1}^{\mathrm{GN}} = \mathbf{x}_k - \mathbf{d}_k^{\mathrm{GN}}.$$

Let $\widehat{\mathbf{d}}_k$ denote the step after the IGG decision and $\widetilde{\mathbf{d}}_k$ the final step after subsequent SPO screening. The implemented method therefore updates

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \widetilde{\mathbf{d}}_k. \tag{31}$$

 doi:10.20944/preprints202603.1477.v1

To state the main result, we collect the local approximation properties of IGG and SPO into the following assumption.

Assumption 1. *There exist constants $C_\eta > 0$ and $C_d > 0$, and a neighborhood $\mathcal{N}$ of $\mathbf{x}^\star$, such that for all iterates $\mathbf{x}_k \in \mathcal{N}$,*

$$\|\widehat{\mathbf{d}}_k - \mathbf{d}_k^{\mathrm{GN}}\| \le C_\eta \tau_\eta, \tag{32}$$

$$\|\widetilde{\mathbf{d}}_k - \widehat{\mathbf{d}}_k\| \le C_d \tau_d. \tag{33}$$

Assumption 1 provides a compact local model of the approximation errors introduced by IGG and SPO. Its form is motivated by the structure of the algorithm. For IGG, a localized update is used only when the incremental information gain is below τ_η, suggesting that the neglected global couplings are weak; in a locally well-conditioned regime, it is therefore reasonable to model the induced step perturbation as $O(\tau_\eta)$. For SPO, the discarded coordinates are precisely those whose GN increments are no larger than τ_d, so under local smoothness and conditioning assumptions the resulting truncation error is naturally of order $O(\tau_d)$. While establishing these bounds directly from $\Delta\eta_t$ and the active-set rule would require a more detailed problem-dependent perturbation analysis, Assumption 1 captures the local error behavior that the two mechanisms are designed to achieve.

The next lemma combines the effects of the two mechanisms.

Lemma 1. *Under Assumption 1, the total deviation of the implemented step from the full GN step satisfies*

$$\|\widetilde{\mathbf{d}}_k - \mathbf{d}_k^{\mathrm{GN}}\| \le \varepsilon, \qquad \varepsilon = C_\eta \tau_\eta + C_d \tau_d, \tag{34}$$

for all $\mathbf{x}_k \in \mathcal{N}$.

Proof. By the triangle inequality,

$$\|\widetilde{\mathbf{d}}_k - \mathbf{d}_k^{\mathrm{GN}}\| \le \|\widetilde{\mathbf{d}}_k - \widehat{\mathbf{d}}_k\| + \|\widehat{\mathbf{d}}_k - \mathbf{d}_k^{\mathrm{GN}}\|.$$

Applying (33) and (32) gives (34). □

Under the standard local convergence theory for GN, there exist constants $\rho \in [0, 1)$, $\kappa > 0$, and a neighborhood of $\mathbf{x}^\star$ such that the full GN iterates satisfy

$$\|\mathbf{x}_{k+1}^{\mathrm{GN}} - \mathbf{x}^\star\| \le \rho \|\mathbf{x}_k - \mathbf{x}^\star\| + \kappa \|\mathbf{x}_k - \mathbf{x}^\star\|^2. \tag{35}$$

Here $\rho = 0$ in the zero-residual case, in which case GN is locally quadratic; otherwise $\rho < 1$ and the convergence is locally linear.

We can now state the main local result for the implemented method.

Theorem 1. *Assume that $c(\mathbf{x})$ satisfies the regularity assumptions stated at the beginning of this section, and that Assumption 1 holds. Let*

$$\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}^\star, \qquad \varepsilon = C_\eta \tau_\eta + C_d \tau_d.$$

Then, for all iterates sufficiently close to $\mathbf{x}^\star$,

$$\|\mathbf{e}_{k+1}\| \le \rho \|\mathbf{e}_k\| + \kappa \|\mathbf{e}_k\|^2 + \varepsilon. \tag{36}$$

Consequently, there exist $r > 0$ and $\bar{\varepsilon} > 0$ such that, if $\|\mathbf{e}_0\| \le r$ and $0 \le \varepsilon \le \bar{\varepsilon}$, then the iterates remain in a neighborhood of $\mathbf{x}^\star$ and satisfy

$$\limsup_{k \to \infty} \|\mathbf{e}_k\| \le \frac{2\varepsilon}{1 - \rho}. \tag{37}$$

 doi:10.20944/preprints202603.1477.v1

In particular, for fixed thresholds τ_η and τ_d, the proposed method converges locally to an $O(\tau_\eta + \tau_d)$ neighborhood of the full-GN solution.

Proof. Let

$$\mathbf{x}_{k+1}^{\mathrm{GN}} = \mathbf{x}_k - \mathbf{d}_k^{\mathrm{GN}}, \qquad \mathbf{x}_{k+1} = \mathbf{x}_k - \widetilde{\mathbf{d}}_k.$$

Then

$$\mathbf{x}_{k+1} - \mathbf{x}^\star = (\mathbf{x}_{k+1}^{\mathrm{GN}} - \mathbf{x}^\star) - (\widetilde{\mathbf{d}}_k - \mathbf{d}_k^{\mathrm{GN}}).$$

Taking norms and applying (35) and Lemma 1 yields

$$\|\mathbf{e}_{k+1}\| \le \rho \|\mathbf{e}_k\| + \kappa \|\mathbf{e}_k\|^2 + \varepsilon,$$

which is (36).

Now choose $r > 0$ small enough that $\kappa r \le (1-\rho)/2$. Whenever $\|\mathbf{e}_k\| \le r$, we obtain

$$\|\mathbf{e}_{k+1}\| \le \Big(\rho + \kappa r\Big)\|\mathbf{e}_k\| + \varepsilon \le \frac{1+\rho}{2}\|\mathbf{e}_k\| + \varepsilon.$$

Hence, if ε is sufficiently small, the iterates remain in the same neighborhood and the scalar sequence $\|\mathbf{e}_k\|$ is dominated by a linearly perturbed contraction. Standard comparison arguments then give

$$\limsup_{k\to\infty} \|\mathbf{e}_k\| \le \frac{\varepsilon}{1 - \frac{1+\rho}{2}} = \frac{2\,\varepsilon}{1-\rho},$$

which proves (37). □

Theorem 1 is the natural fixed-threshold result for the implemented algorithm: it shows that the method tracks full GN up to a neighborhood whose size is proportional to the two approximation thresholds. To recover the exact full-GN limit point and asymptotic rate, the effective perturbation must vanish near the solution.

Corollary 1. *Under the assumptions of Theorem 1, suppose in addition that the effective step perturbation*

$$\varepsilon_k = \|\widetilde{\mathbf{d}}_k - \mathbf{d}_k^{\mathrm{GN}}\|$$

satisfies $\varepsilon_k \to 0$. Then the iterates $\mathbf{x}_k$ converge to the same local minimizer $\mathbf{x}^\star$ as full GN, provided both methods are initialized in the same local basin of attraction.

Moreover, if

$$\varepsilon_k = o\Big(\|\mathbf{d}_k^{\mathrm{GN}}\|\Big), \tag{38}$$

then the local asymptotic convergence rate matches that of full GN: linear in general, and quadratic in the zero-residual case.

Proof. The recursion (36) with a vanishing perturbation implies that the asymptotic error is no longer bounded away from zero, and therefore $\mathbf{x}_k \to \mathbf{x}^\star$. If, in addition, (38) holds, then the accepted step is asymptotically indistinguishable from the full GN step, and standard perturbation arguments for Newton/GN methods yield the same local asymptotic rate as full GN [39, Ch. 6]. □

Remark 1. *Corollary 1 clarifies the role of the thresholds. With fixed positive thresholds, the method is expected to converge to a small neighborhood of the full-GN solution rather than to coincide exactly with it arbitrarily close to $\mathbf{x}^\star$. Exact equivalence with full GN can be recovered only if the effective approximation error introduced by IGG and SPO vanishes asymptotically, for example through adaptive thresholds or a more explicit residual-based acceptance criterion. In the experiments of Section 5, the selected thresholds are sufficiently small that the resulting perturbation is negligible at the estimation-accuracy level of interest.*

 doi:10.20944/preprints202603.1477.v1

These theoretical conclusions are consistent with the empirical findings in Section 5: the final estimation accuracy is essentially indistinguishable from that of full optimization, while the computational cost is substantially reduced because (i) many low-information increments trigger only localized updates, and (ii) later GN iterations operate on progressively smaller active sets. A more refined analysis, for example one that derives bounds such as (32) and (33) directly from the information-gain criterion and the active-set rule, remains an interesting direction for future work.

5. Experiments

5.1. Setup

Datasets: We evaluate the proposed approach against the most closely related contenders on several standard SLAM benchmark datasets, summarized in Table 1. These datasets, provided in g2o [8] and TORO [40] formats and available from the online repository [41], cover scenarios with varying frequencies of loop closures and exhibit typical SLAM sparsity patterns [42]. To emulate realistic incremental operation, we reorder the edges in each dataset so that measurements arrive in the natural acquisition order, and loop-closure edges are incorporated as soon as both associated nodes have been initialized. This contrasts with the original datasets, where loop-closures are sometimes deferred until the end. In addition, we construct a dataset, called MIT-P, by adding position priors to every 50th pose node in the MIT dataset, with Gaussian noise of standard deviation 1 m on each axis. These priors simulate scenarios where reasonably accurate external pose estimates are intermittently available, such as from indoor localization systems (e.g., UWB or WiFi-based) or from global navigation satellite systems (GNSS) outdoors. Since the considered datasets do not provide ground-truth poses, we use the batch solution as a surrogate ground truth for generating the position priors.

Table 1. Utilized benchmark SLAM datasets.

dataset	poses	edges	loop closures	τ_d	τ_η
MIT, MIT-P	808	827	20	10^{-3}	1
FR079	989	1217	229	10^{-4}	0.6
CSAIL	1045	1172	128	10^{-5}	0.95
Intel	1228	1483	256	10^{-6}	0.72
FRH	1316	2820	1505	10^{-7}	0.45

Accuracy measures: We use the normalized chi-squared error, denoted by $\mathrm{N}\chi_t^2$, and the absolute trajectory error (ATE) as our accuracy measures. For each approach, we report both their final values (after the last increment) and their average values (over all increments). The normalized chi-squared error is directly related to the nonlinear least-squares cost (2) as

$$\mathrm{N}\chi_t^2 = \frac{2c(\mathbf{x}_t)}{M_t} \tag{39}$$

where M_t is the number of scalar measurement equations available at increment t. The ATE quantifies the deviation of the estimated trajectory from ground truth. It is computed as the root mean squared error (RMSE) between the estimated poses $\mathbf{x}_{t,p}$ and the corresponding ground-truth poses $\mathbf{x}_p^\star$, after alignment by a rigid-body transformation, such as the Kabsch algorithm [43,44], i.e.,

$$\mathrm{ATE} = \sqrt{\frac{1}{P_t}\sum_{p=1}^{P_t} \left\| \mathbf{x}_p^\star - \mathbf{T}_t\, \mathbf{x}_{t,p} \right\|^2}, \tag{40}$$

where P_t is the number of poses at increment t, and $\mathbf{T}_t \in \mathrm{SE}(2)$ denotes the optimal alignment transformation at that increment. As before, we take the batch solution as the reference ground truth.

Considered approaches: We include the following approaches in our evaluations:

 doi:10.20944/preprints202603.1477.v1

Algorithm 2 Incremental SLAM with no Gating or Partial Optimization

Require: initial estimate $\mathbf{x}_0$, thresholds τ_d and τ_{GN}
Ensure: updated estimate $\mathbf{x}_t$ after each increment t
for $t = 1, 2, \ldots$ **do**
  update $\mathbf{J}_t$, $\mathbf{r}_t$, $\mathbf{R}_t$, $\mathbf{b}_t$, and $\mathbf{p}_t$ based on new measurements
  for $i_{\text{GN}} = 1, 2, \ldots, \tau_{\text{GN}}$ **do**
    solve $(\mathbf{R}_t^\mathsf{T}\mathbf{R}_t)\mathbf{d}_t = \mathbf{b}_t$ for $\mathbf{d}_t$
    if $\max|\mathbf{d}_t| \leq \tau_d$ **then** break
    end if
    $\mathbf{x}_t \leftarrow \mathbf{x}_t - \mathbf{d}_t$
    recalculate $\mathbf{J}_t$, $\mathbf{r}_t$, $\mathbf{R}_t$, $\mathbf{b}_t$, and $\mathbf{p}_t$ based on new $\mathbf{x}_t$
  end for
end for

- GN1: Performs a *single* GN iteration per increment. This setting resembles iSAM2 [16] without any relinearization threshold but with a variable-update threshold. For fairness, new measurements are relinearized exactly as in our approach, ensuring comparable treatment of updates.
- GNi: Performs *multiple* GN iterations at each increment without any selectivity, i.e., no gating or partial optimization. All variables may be updated and relinearized whenever new measurements arrive. To maintain consistency with other approaches, the threshold τ_d is used to decide early termination of GN iterations (cf. Algorithm 2). This algorithm resembles a standard incremental solver without periodic batch steps. It is maximally responsive and accurate but may perform considerable redundant work when information gain is small.
- GNi-LCG: Similar to GNi, but performs GN iterations *only* when a loop closure is detected, hence the term loop-closure gating (LCG). Loop closures are identified as measurements that connect previously unconnected parts of the graph (e.g., a pose-pose constraint between non-consecutive poses). This strategy is comparable to approaches proposed in [23,45].
- GNi-IGG: Similar to GNi, but performs GN iterations *only* when the information gain exceeds the threshold τ_η, according to the proposed IGG scheme.
- GNi-SPO: Extends GNi by incorporating the proposed SPO scheme. Multiple GN iterations may be performed at each increment, but updates are restricted to the active set of affected variables. No gating is applied, meaning all variables are considered potentially affected at every increment.
- GNi-SPO-LCG: Combines SPO with LCG. As in GNi-SPO, only the active set of affected variables is updated and relinearized, while global GN optimization is triggered solely at loop closures. When loop closures are the primary source of information, this approach is expected to behave similarly to the proposed approach.
- GNi-SPO-IGG: The proposed algorithm, capable of performing multiple GN iterations at each increment while incorporating both SPO and IGG.

Parameters: We set $\tau_{\text{GN}} = 10$ for all approaches, except for GN1, which corresponds to GNi with $\tau_{\text{GN}} = 1$. We set the thresholds τ_d and τ_η separately for each dataset, as listed in Table 1. Here, τ_η controls how much incremental information is required before a full global update is triggered, whereas τ_d determines how aggressively the active set is pruned during selective partial optimization. We chose these thresholds empirically to balance estimation accuracy and computational cost, and set them such that the proposed method achieves accuracy comparable to that of the competing approaches, thereby enabling a fair comparison of efficiency. In all experiments, each incremental step consists of a single newly added measurement (edge).

5.2. Results

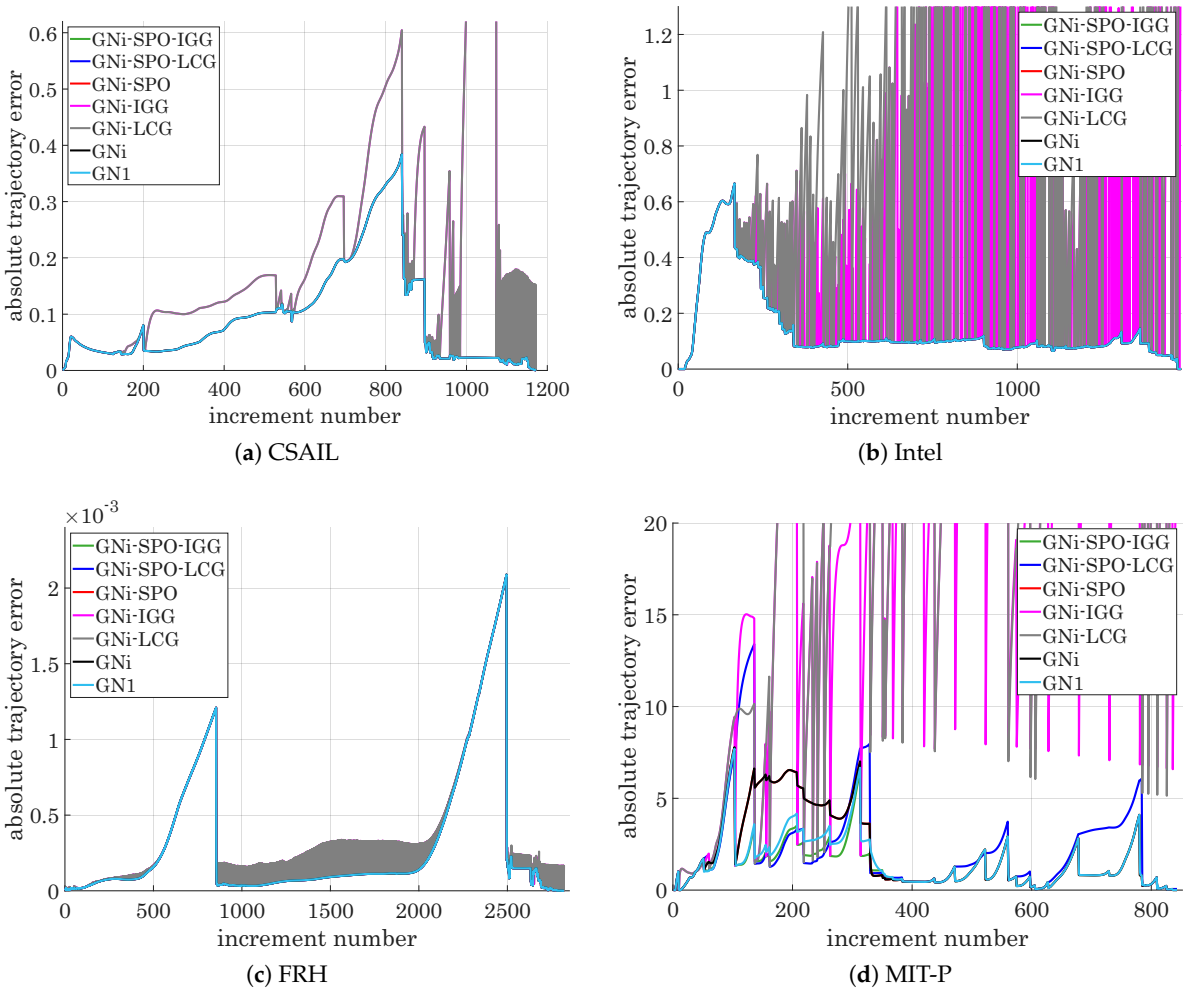
In Table 2, we summarize the performance evaluation results for all considered approaches across the benchmark datasets. Alongside final and mean Nχ^2 and ATE values, which capture estimation accuracy, we also report average floating-point operation (FLOP) counts for Cholesky up(down)date (*update*) and partial-solve (*solve*) steps, as defined in Section 3.6, providing a direct

 doi:10.20944/preprints202603.1477.v1

measure of computational efficiency over all increments. For GNi-SPO-IGG, GNi-SPO-LCG, and GNi-SPO, we additionally report (in parentheses) the mean solve FLOPs for when partial solves are replaced with full solves (see line 10 of Algorithm 1), while still retaining partial variable update and relinearization. This variant isolates the benefit of the partial-solve strategy itself in terms of computational savings. Conceptually, it is analogous to iSAM2 with fluid relinearization when the same variable update and relinearization threshold τ_d is applied. Note that for this full-solve variant, accuracy and mean update FLOPs remain unchanged compared to the partial-solve case as only the solve FLOPs differ.

To evaluate both intermediate accuracy and computational progression, we plot the evolution of ATE and cumulative update FLOPs over all increments for all considered approaches on four representative datasets, as shown in Figures 1 and 2, respectively. To provide further insight into the behavior of the proposed IGG scheme, in Figure 3, we plot the information gain $\Delta\eta_t$ across all increments for four datasets. The plots also indicate increments at which loop closures are detected or position priors are introduced.

To examine the influence of the two threshold mechanisms governed by τ_d and τ_η on the accuracy-efficiency trade-off of the proposed approach, we plot the mean ATE and mean update FLOPs of GNi-SPO on the MIT dataset as functions of τ_d in Figure 4(**a**), and those of GNi-IGG on the FR079 dataset as functions of τ_η in Figure 5(**a**). For comparison with relevant baselines, Figure 4(**a**) also includes the corresponding curves for GNi and GN1, while Figure 5(**a**) includes those for GNi-LCG, GNi, and GN1.



(**a**) CSAIL (**b**) Intel

(**c**) FRH (**d**) MIT-P

Figure 1. ATE over increments for four datasets.

 doi:10.20944/preprints202603.1477.v1

Table 2. Performance of different approaches on the considered datasets.

dataset	approach	$N\chi^2$		ATE		mean FLOPs	
		final	mean	final	mean	solve	update
MIT	GNi-SPO-IGG	1.65918×10^{-2}	1.84891×10^{-2}	3.67389×10^{-4}	5.802394	2,028 (36,926)	66,541
	GNi-SPO-LCG	1.65918×10^{-2}	1.84891×10^{-2}	3.67389×10^{-4}	5.802394	2,028 (36,926)	66,541
	GNi-SPO	1.65915×10^{-2}	1.84891×10^{-2}	3.47418×10^{-4}	5.802397	19,036 (36,925)	66,565
	GNi-IGG	32.5859	14.6328	30.4408	20.3804	3,696	98,087
	GNi-LCG	32.5859	14.6328	30.4408	20.3804	3,696	98,087
	GNi	1.65914×10^{-2}	1.84841×10^{-2}	5.20559×10^{-11}	5.802427	36,661	438,548
	GN1	1.65914×10^{-2}	780,578	1.05711×10^{-10}	5.850329	17,704	405,989
FR079	GNi-SPO-IGG	1.02983×10^{-2}	1.06656×10^{-2}	3.75248×10^{-5}	6.02654×10^{-2}	8,377 (50,546)	85,685
	GNi-SPO-LCG	1.02983×10^{-2}	1.06653×10^{-2}	3.75245×10^{-5}	6.02615×10^{-2}	7,355 (50,620)	87,015
	GNi-SPO	1.02983×10^{-2}	1.06653×10^{-2}	3.75247×10^{-5}	6.02615×10^{-2}	28,318 (50,620)	87,034
	GNi-IGG	5.84489×10^{-2}	5.36751×10^{-1}	3.89112×10^{-2}	8.87086×10^{-2}	12,085	130,969
	GNi-LCG	5.84489×10^{-2}	1.81984×10^{-1}	3.89112×10^{-2}	1.22274×10^{-1}	9,334	101,979
	GNi	1.02983×10^{-2}	1.06651×10^{-2}	9.95967×10^{-15}	6.02609×10^{-2}	50,620	460,584
	GN1	1.02983×10^{-2}	1.06652×10^{-2}	2.78002×10^{-12}	6.02635×10^{-2}	24,808	437,461
CSAIL	GNi-SPO-IGG	1.10797×10^{-2}	2.80792×10^{-3}	1.23096×10^{-6}	9.11474×10^{-2}	10,245 (51,960)	268,636
	GNi-SPO-LCG	1.10797×10^{-2}	2.80776×10^{-3}	1.23113×10^{-6}	9.11517×10^{-2}	10,147 (52,127)	270,687
	GNi-SPO	1.10797×10^{-2}	2.80776×10^{-3}	1.23121×10^{-6}	9.11517×10^{-2}	29,651 (52,128)	271,536
	GNi-IGG	1.10797×10^{-2}	2.34198	4.13827×10^{-14}	2.25203×10^{-1}	14,657	386,088
	GNi-LCG	1.10797×10^{-2}	3.8143	5.18019×10^{-7}	2.25553×10^{-1}	15,134	401,049
	GNi	1.10797×10^{-2}	2.80718×10^{-3}	2.29725×10^{-14}	9.11515×10^{-2}	52,053	978,461
	GN1	1.10797×10^{-2}	2.80897×10^{-3}	1.35270×10^{-6}	9.11548×10^{-2}	23,974	825,271
Intel	GNi-SPO-IGG	4.85217×10^{-2}	3.42609×10^{-2}	1.01812×10^{-7}	1.40955×10^{-1}	28,609 (77,951)	330,270
	GNi-SPO-LCG	4.85121×10^{-2}	3.42331×10^{-2}	4.06794×10^{-7}	1.40951×10^{-1}	20,362 (78,423)	286,034
	GNi-SPO	4.85121×10^{-2}	3.42397×10^{-2}	1.18840×10^{-7}	1.40951×10^{-1}	49,525 (78,686)	357,216
	GNi-IGG	69.5308	126.965	1.80246	5.97116×10^{-1}	39,008	435,542
	GNi-LCG	6.01180	19,366.4	2.86431	1.27203	26,905	336,683
	GNi	4.85121×10^{-2}	3.42216×10^{-2}	2.18560×10^{-11}	1.40951×10^{-1}	77,391	709,119
	GN1	4.85121×10^{-2}	1.06152	1.72933×10^{-11}	1.40979×10^{-1}	31,523	488,865
FRH	GNi-SPO-IGG	2.28294×10^{-8}	1.11147×10^{-8}	8.95582×10^{-11}	3.03247×10^{-4}	34,307 (98,596)	1,308,496
	GNi-SPO-LCG	2.28294×10^{-8}	1.11142×10^{-8}	8.95220×10^{-11}	3.03360×10^{-4}	35,814 (99,467)	1,465,775
	GNi-SPO	2.28294×10^{-8}	1.11142×10^{-8}	8.95004×10^{-11}	3.03360×10^{-4}	57,870 (99,467)	1,469,435
	GNi-IGG	2.28294×10^{-8}	5.90106×10^{-7}	1.94511×10^{-13}	3.46361×10^{-4}	55,132	3,614,516
	GNi-LCG	6.16115×10^{-3}	2.82510×10^{-6}	1.66055×10^{-4}	3.46707×10^{-4}	56,504	3,742,194
	GNi	2.28294×10^{-8}	1.11140×10^{-8}	8.60015×10^{-14}	3.03360×10^{-4}	99,467	6,141,656
	GN1	2.28294×10^{-8}	1.11140×10^{-8}	1.51830×10^{-13}	3.03360×10^{-4}	49,737	6,135,811
MIT-P	GNi-SPO-IGG	1.72741×10^{-2}	2.06264×10^{-2}	1.73090×10^{-3}	1.384435	3,799 (40,348)	100,436
	GNi-SPO-LCG	1.79920×10^{-2}	9.21434×10^{-1}	5.77128×10^{-2}	2.324933	2,650 (40,664)	77,984
	GNi-SPO	1.72743×10^{-2}	1.99909×10^{-2}	1.64191×10^{-3}	2.101596	25,159 (56,904)	187,886
	GNi-IGG	71.4849	40.6574	25.0316	24.25101	7,762	194,298
	GNi-LCG	93.2878	45.3580	36.7726	31.01479	4,239	115,386
	GNi	1.72738×10^{-2}	1.99852×10^{-2}	3.85749×10^{-12}	2.101118	56,753	803,583
	GN1	1.72738×10^{-2}	1.50452	9.22196×10^{-6}	1.537856	17,722	406,185

5.3. Discussion

From the results in Table 2 and Figure 1, several consistent patterns emerge across datasets of different scale and structure. The first and most important trend is that the proposed GNi-SPO-IGG achieves accuracy essentially indistinguishable from full GNi while requiring far less computation. Both final and mean $N\chi^2$ and ATE values closely track those of the full incremental baseline GNi, demonstrating that selectively solving for and relinearizing only affected variables, when guided by the information-gain criterion, preserves estimation quality. For instance, on the FR079 and FRH datasets, the discrepancies in error between GNi-SPO-IGG and GNi are negligible, yet the proposed approach reduces solve and update FLOPs significantly. This confirms that careful gating and partial optimization can provide substantial efficiency without accuracy degradation.

The efficiency gains are particularly pronounced in larger and denser problems such as CSAIL, Intel, and FRH. As shown in Figure 2, the computational cost of always performing full updates grows rapidly in these cases, whereas GNi-SPO-IGG effectively contains this growth by adaptively restricting updates to variables most affected by new information. This yields a two- to eight-fold reduction in computation compared to GNi, while both final and mean errors remain essentially unchanged. Even in smaller-scale datasets such as MIT and FR079, where graphs are less dense and updates relatively inexpensive, the proposed method still delivers notable efficiency improvements without sacrificing accuracy. These results suggest that the scalability advantage of the proposed approach will become even more significant as problem size increases.

 doi:10.20944/preprints202603.1477.v1

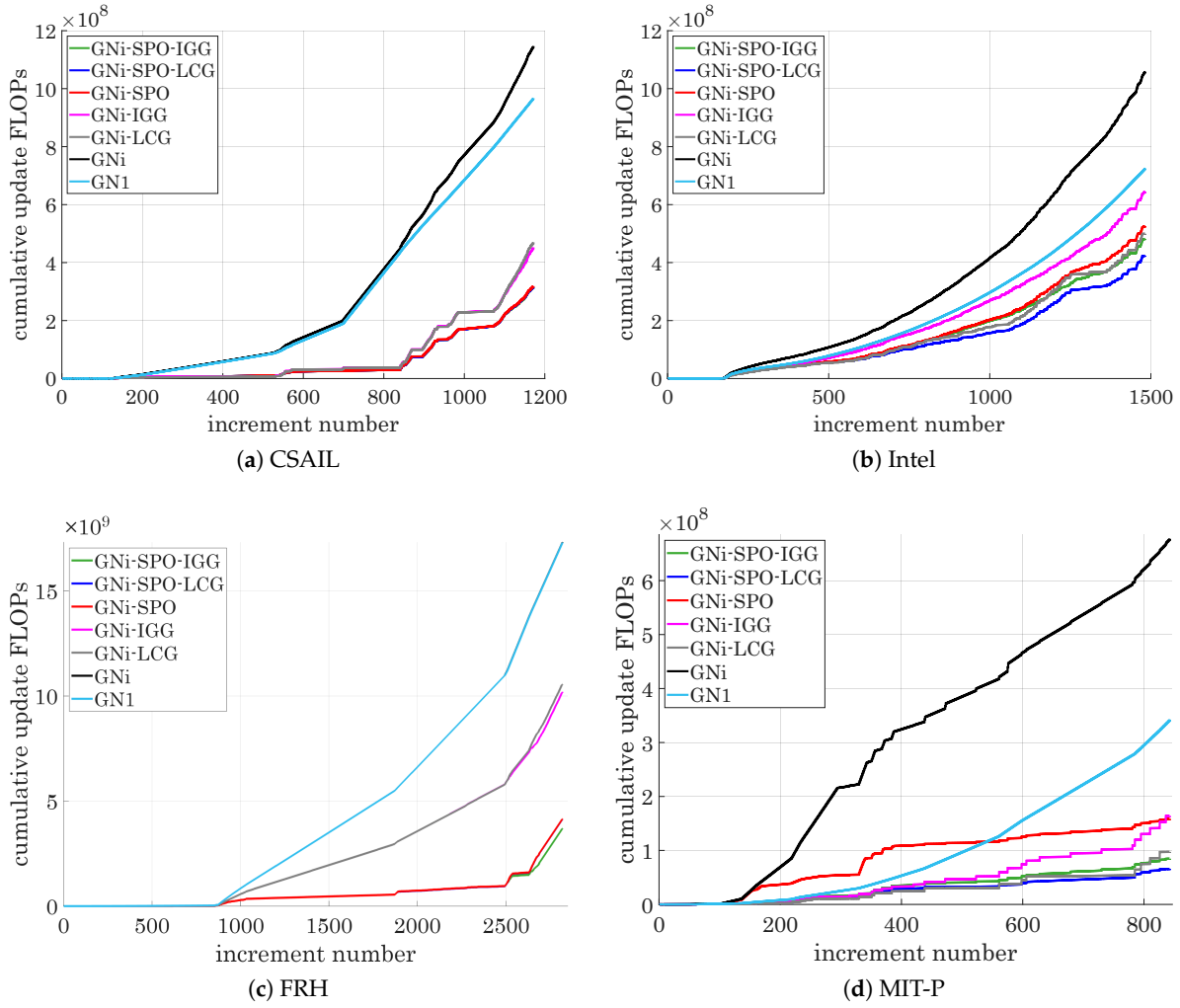
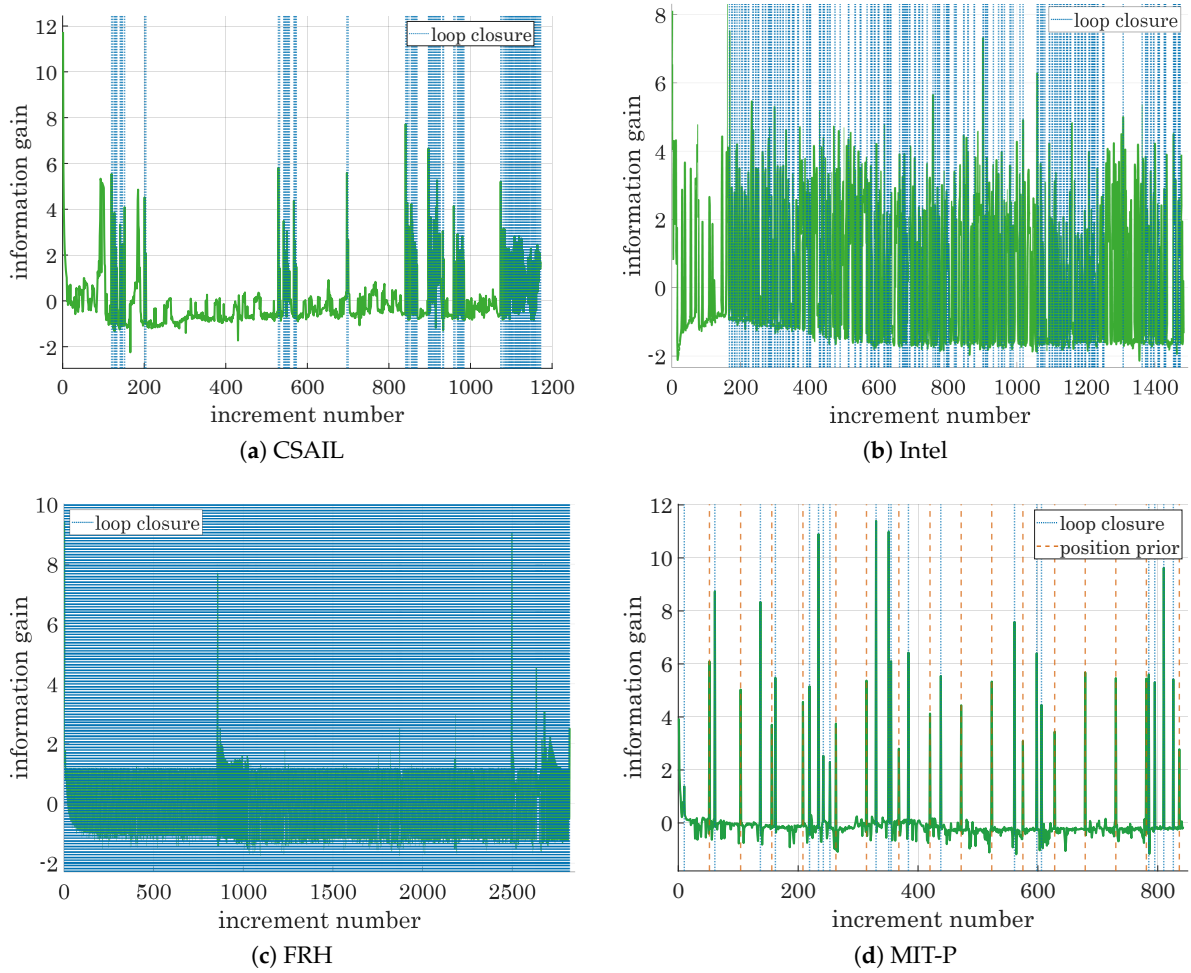


Figure 2. Cumulative update FLOPs over increments for four datasets.

A comparison between GNi-SPO-IGG and GNi-SPO-LCG further illustrates the value of information-guided gating. In datasets such as MIT, where most of the informative content arises from a small number of loop-closure measurements, both approaches attain similar accuracy and complexity. However, when substantial information is introduced through non-loop-closure measurements, such as position priors in MIT-P, GNi-SPO-LCG fails to exploit this information effectively. By contrast, GNi-SPO-IGG leverages such priors through its information-gain-based gating, achieving markedly better accuracy while maintaining comparable computational cost. This demonstrates the broader advantage of IGG: its ability to incorporate diverse sources of information beyond loop closures, ensuring consistent accuracy across heterogeneous sensing scenarios and enhancing scalability to real-world, long-term SLAM deployments.

Approaches based solely on gating (GNi-LCG and GNi-IGG) exhibit significant instability. While their final error values can sometimes appear acceptable, mean errors often increase by orders of magnitude, particularly on the MIT-P and Intel datasets. GNi-IGG performs only marginally better than GNi-LCG, yet still produces substantially larger errors than when SPO is employed. This highlights the limitations of gating alone, whether based on loop closures or information gain, which cannot adequately respond to incremental odometry or prior information, leaving large portions of the trajectory insufficiently corrected. The outcome is poor trajectory quality for much of the run, even if later corrections at loop closures or high-information events produce seemingly reasonable final results. These findings underscore that loop closures or information gain alone are insufficient for reliable incremental smoothing, and that continuous incremental corrections are essential.

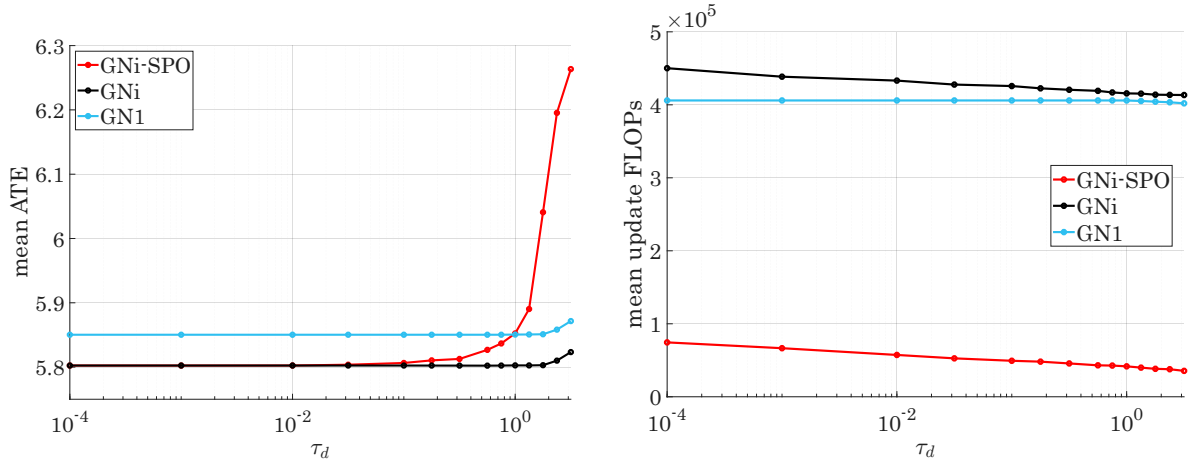
 doi:10.20944/preprints202603.1477.v1



(**a**) CSAIL (**b**) Intel

(**c**) FRH (**d**) MIT-P

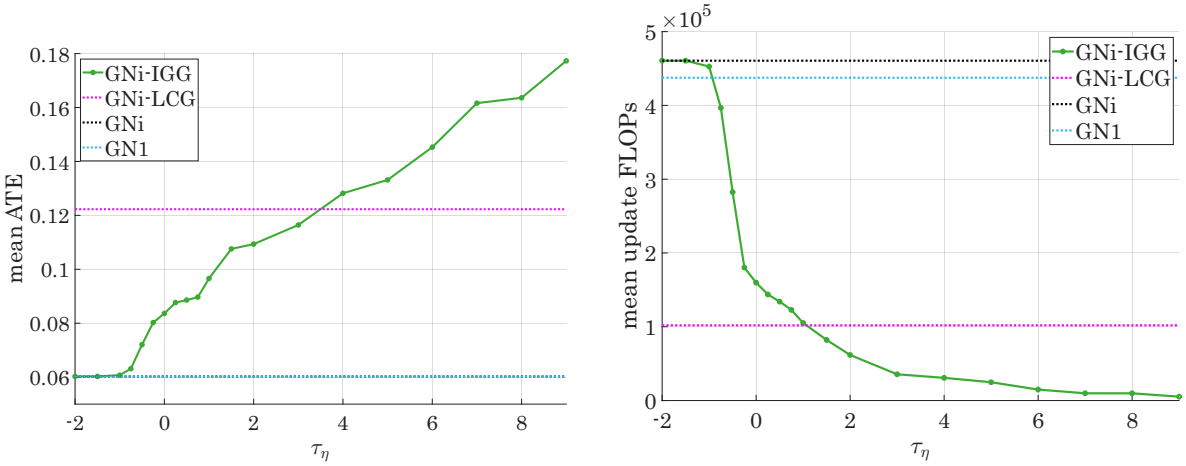
Figure 3. Information gain over increments for four datasets.



(**a**) Mean ATE versus τ_d. (**b**) Mean update FLOPs versus τ_d.

Figure 4. Sensitivity of GNi-SPO on the MIT dataset to the update threshold τ_d: mean ATE and mean update FLOPs as functions of τ_d.

 doi:10.20944/preprints202603.1477.v1



(**a**) Mean ATE versus τ_η.

(**b**) Mean update FLOPs versus τ_η.

Figure 5. Sensitivity of GNi-IGG on the FR079 dataset to the information-gain threshold τ_η: mean ATE and mean update FLOPs as functions of τ_η.

The GN1 baseline, which mirrors iSAM2's single-GN-iteration strategy, provides another instructive comparison. While GN1 often achieves final accuracy comparable to GNi, it does so at the cost of significantly elevated mean errors. For example, on the Intel dataset, GN1 attains a similar final error to GNi but with mean errors two orders of magnitude higher, while also requiring more computations compared to all considered approaches except GNi. Its poor intermediate accuracy on some datasets can be attributed to its inability to converge within a single GN iteration per increment, particularly when successive high-information measurements arrive in quick succession. In such cases, GN1 cannot keep pace with the influx of information, leaving large errors uncorrected until much later. These observations highlight the importance of adaptivity not only in deciding *when* and *what* to update, but also in determining *how much* to update at each increment.

Trajectory-level analysis, as illustrated in Figure 1, reinforces these observations. Algorithms incorporating SPO maintain ATE values comparable to GNi across the increments, while gating-only methods (GNi-LCG and GNi-IGG) exhibit spiking or oscillatory error patterns whenever increments contain only low-information odometry measurements. On the synthetic MIT-P dataset, which includes intermittent external priors, these differences are even more pronounced: GNi-LCG and GNi-IGG exhibit very high intermediate errors, with mean values several orders of magnitude larger than those of the other algorithms, whereas the remaining approaches maintain low final errors and reasonable mean error performance. Notably, GNi-SPO-IGG achieves accuracy similar to GNi but at significantly lower computational cost compared to GN1, GNi, and GNi-SPO, demonstrating robustness to external information while avoiding wasted computation. Although GNi-SPO-LCG incurs somewhat lower computational cost than GNi-SPO-IGG, its final and mean errors are substantially higher, owing to its failure to trigger highly beneficial global updates when informative but non-loop-closure position priors arrive. This underscores a key design lesson: information-gain-based gating is essential for effectively leveraging diverse external cues, such as GNSS, UWB, or vision-based priors, ensuring not only accuracy with minimal computation, but also scalability to long-term, multimodal SLAM deployments in real-world environments.

The results presented in Table 2 and Figures 1 and 2 correspond to parameter settings for which GNi-SPO-IGG achieves accuracy comparable to the baseline GNi. Figures 4(**a**) and 5(**a**), on the other hand, isolate the effects of the two threshold mechanisms through component-wise sensitivity analyses. In particular, Figure 4(**a**) shows how the update threshold τ_d influences the accuracy-efficiency trade-off in GNi-SPO, while Figure 5(**a**) shows how the information-gain threshold τ_η influences the corresponding trade-off in GNi-IGG. Larger threshold values generally reduce computational cost at the expense of more aggressive approximation, whereas smaller values make the behavior closer to

 doi:10.20944/preprints202603.1477.v1

that of full GN. In more weakly constrained or noisier settings, more conservative threshold choices may be preferable.

6. Concluding Remarks

We presented a novel incremental SLAM optimizer that achieves a principled balance between accuracy and efficiency by combining an information-theoretic update trigger with selective partial optimization. The key idea is to monitor the change in system entropy (approximated via the log-determinant of the information matrix) induced by new measurements and to use this to decide whether to perform a global update or restrict attention to local variables. When a global update is triggered, GN iterations are executed to convergence, but each iteration is confined to the subset of variables most affected by the recent measurements. This selective update strategy exploits the conditional independence structure of SLAM and yields solutions nearly as accurate as batch optimization while requiring substantially fewer updates and significantly lower computational cost.

Extensive experiments on diverse SLAM benchmark datasets demonstrate that the proposed approach matches the accuracy of batch and full incremental solvers while consistently achieving large computational savings. In particular, its final and mean normalized chi-squared and absolute trajectory errors remain essentially identical to those of GNi, while solve and update FLOPs are reduced by factors of two to eight, especially in large-scale datasets such as CSAIL, Intel, and FRH. The information-gain-based gating strategy generalizes loop-closure detection by capturing all high-impact events, including loop closures, dense reobservations, and external priors such as GNSS updates, while simultaneously filtering out low-information increments that contribute little beyond increased computational burden. The results also highlight the robustness of the proposed approach to external priors: unlike loop-closure-only approaches, our algorithm can seamlessly exploit intermittent position priors without destabilizing the trajectory estimate.

A central feature of our approach is the selective update of only those variables that remain effectively unconverged during GN iterations. Following highly informative measurements, such as major loop closures, only a small fraction of the state typically requires further refinement in subsequent iterations. SPO exploits this structure by restricting computation to the active subset, thereby achieving substantial cost savings with negligible loss of accuracy in practice. By contrast, GNi, which performs full updates without selectivity, follows the corresponding full-GN trajectory but incurs several times higher computational cost during large re-optimizations.

In relation to iSAM2, one of the most widely used incremental SLAM back-ends, we emphasize an important conceptual and practical distinction. The fluid relinearization mechanism in iSAM2 is heuristic: a variable is relinearized only when the magnitude of its estimated update exceeds a prescribed threshold. Although this can reduce computational cost by avoiding some relinearizations, it may also allow linearization errors to accumulate when small but repeated updates are continually deferred. By contrast, our method maintains consistent linearizations by relinearizing every variable that is actually updated. Moreover, our thresholding is applied to the optimization step itself, to determine whether a variable warrants further refinement, rather than to the relinearization decision. In additional experiments not shown here, we observed that iSAM2 often required relatively conservative threshold settings to avoid instability, particularly under poor initializations. A further distinction is that iSAM2 does not incorporate either information-guided gating or selective partial optimization: despite limiting some relinearizations, it still solves a global linear system at every increment. Our approach instead combines information-guided gating with selective partial optimization to focus computation on the variables most affected by incoming information, thereby achieving high accuracy in practice at substantially lower computational cost. Section 4 provides a theoretical local perturbation analysis of its deviation from full GN.

Although our current implementation is in MATLAB/Octave and intended primarily for clarity and reproducibility, the algorithmic FLOP counts and structural complexity analysis strongly indicate that a C++ implementation would yield substantial runtime gains. In fact, the proposed framework is

 doi:10.20944/preprints202603.1477.v1

fully compatible with established SLAM libraries such as g2o and GTSAM, where fast factor reuse (e.g., via Bayes trees) could be combined with our selective multi-iteration optimization to achieve both scalability and real-time performance. We make our MATLAB/Octave code publicly available to ensure reproducibility, while noting that a C++ integration is a straightforward engineering extension and an important direction for future work.

The threshold parameters τ_η and τ_d were set empirically in this study, but future work could investigate adaptive schemes based on statistical criteria, such as significance tests on the growth of the normalized chi-squared error, noise-aware thresholding, or conditioning measures that reflect observability. Such mechanisms may be particularly useful in settings with higher sensor noise, motion degeneration, or weak geometric constraints, where more conservative update decisions may be desirable. Additional directions include incorporating robust cost functions, such as dynamic covariance scaling or switchable constraints, to improve resilience to outliers; applying the approach to fixed-lag smoothing, where information gain may also inform safe marginalization; and integrating the same framework into active SLAM, where information gain can guide both update scheduling and exploration decisions in resource-constrained settings.

Beyond algorithmic efficiency, the proposed framework has direct implications for robotic autonomy. In practical SLAM-enabled robots, the back-end competes for computational resources with perception, planning, and control. Therefore, reducing back-end optimization cost helps preserve closed-loop responsiveness and enhances the feasibility of long-duration operation on embedded or otherwise resource-constrained platforms. The proposed approach is thus particularly relevant to autonomous ground, aerial, and service robots that rely on accurate real-time localization and mapping in large-scale or sensor-rich environments.

In summary, we have developed an efficient and principled incremental SLAM solver that delivers batch-level accuracy at a fraction of the computational cost. By narrowing the gap between batch and incremental optimization, the proposed method enables accurate real-time SLAM in computationally constrained environments. It provides a strong foundation for high-precision mapping and localization on embedded platforms and can be naturally extended to applications such as 3D SLAM, multi-robot mapping, and adaptive perception.

Funding: This research received no external funding.

Data Availability Statement: The code and data used to produce the results in this paper are available at https://github.com/Reza219/Incremental_SLAM.

Conflicts of Interest: The author declare no conflicts of interest.

References

1. Leonard, J.; Durrant-Whyte, H.; Cox, I. Dynamic map building for autonomous mobile robot. In Proceedings of the IEEE International Workshop on Intelligent Robots and Systems, Towards a New Frontier of Applications, 1990, pp. 89–96 vol.1. https://doi.org/10.1109/IROS.1990.262373.
2. Frese, U. A Discussion of Simultaneous Localization and Mapping. *Autonomous Robots* **2006**, *20*, 25–42. https://doi.org/10.1007/s10514-006-5735-x.
3. Grisetti, G.; Kümmerle, R.; Stachniss, C.; Burgard, W. A Tutorial on Graph-Based SLAM. *IEEE Intelligent Transportation Systems Magazine* **2010**, *2*, 31–43. https://doi.org/10.1109/MITS.2010.939925.
4. Dellaert, F.; Kaess, M. Square Root SAM: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research* **2006**, *25*, 1181–1203.
5. Guivant, J.; Nebot, E. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation* **2001**, *17*, 242–257. https://doi.org/10.1109/70.938382.
6. Eustice, R.M.; Singh, H.; Leonard, J.J. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Transactions on Robotics* **2006**, *22*, 1100–1114.
7. Dellaert, F.; Kaess, M. *Factor Graphs for Robot Perception*; Foundations and Trends in Robotics, Vol. 6, 2017.

 doi:10.20944/preprints202603.1477.v1

8. 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, 2011; pp. 3607–3613. https://doi.org/10.1109/ICRA.2011.5979949.
9. Dellaert, F.; Contributors, G. borglab/gtsam, 2022. https://doi.org/10.5281/zenodo.5794541.
10. Dellaert, F.; Carlson, J.; Ila, V.; Ni, K.; Thorpe, C.E. Subgraph-Preconditioned Conjugate Gradients for Large-Scale SLAM. In Proceedings of the Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, 2010; pp. 2566–2571.
11. Jian, Y.; Balcan, D.C.; Panageas, I.; Tetali, P.; Dellaert, F. Support-Theoretic Subgraph Preconditioners for Large-Scale SLAM. In Proceedings of the Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2013, pp. 9–16.
12. Jian, Y.D.; Dellaert, F. iSPCG: Incremental Subgraph-Preconditioned Conjugate Gradient Method for Online SLAM with Many Loop-Closures. In Proceedings of the Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL, USA, 2014; pp. 2647–2653.
13. Olson, E. Fast iterative optimization of pose graphs with poor initial estimates. In Proceedings of the Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2006, pp. 2262–2269.
14. Kaess, M.; Ranganathan, A.; Dellaert, F. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics* **2008**, *24*, 1365–1378.
15. Gill, P.E.; Murray, W.; Wright, M.H. The design and implementation of a computer-based system for solving non-linear least-squares problems. *Department of Computer Science, Stanford University* **1974**.
16. Kaess, M.; Johannsson, H.; Roberts, R.; Ila, V.; Leonard, J.J.; Dellaert, F. iSAM2: Incremental smoothing and mapping using the Bayes tree. *The International Journal of Robotics Research* **2012**, *31*, 216–235.
17. Rosen, D.M.; Kaess, M.; Leonard, J.J. RISE: An Incremental Trust-Region Method for Robust Online Sparse Least-Squares Estimation. *IEEE Transactions on Robotics* **2014**, *30*, 1091–1108.
18. Huang, Q.; Pu, C.; Fourie, D.; Khosoussi, K.; How, J.P.; Leonard, J.J. NF-iSAM: Incremental Smoothing and Mapping via Normalizing Flows. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 1095–1102.
19. McGann, D.; III, J.G.R.; Kaess, M. Robust Incremental Smoothing and Mapping (riSAM). In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), London, UK, 2023; pp. 4157–4163. https://doi.org/10.1109/ICRA48891.2023.10161438.
20. Kang, W.; Kim, J.; Chung, J.; Choi, S.; Kim, T. Efficient Graduated Non-Convexity for Pose Graph Optimization. In Proceedings of the International Conference on Control, Automation and Systems, 2024, pp. 545–548.
21. Yang, H.; Antonante, P.; Tzoumas, V.; Carlone, L. Graduated Non-Convexity for Robust Spatial Perception: From Non-Minimal Solvers to Global Outlier Rejection. *IEEE Robotics and Automation Letters* **2020**, *5*, 1127–1134. https://doi.org/10.1109/LRA.2020.2965893.
22. Choi, S.; Kang, W.; Chung, J.; Kim, J.; wan Kim, T. Adaptive Graduated Non-Convexity for Pose Graph Optimization. *arXiv preprint arXiv:2308.11444* **2023**.
23. Polok, L.; Ila, V.; Solony, M.; Smrž, P.; Zemčík, P. Incremental Block Cholesky Factorization for Nonlinear Least Squares in Robotics. In Proceedings of the Proceedings of Robotics: Science and Systems (RSS), Berlin, Germany, 2013.
24. Wang, X.; Marcotte, R.J.; Ferrer, G.; Olson, E. AprilSAM: Real-Time Smoothing and Mapping. In Proceedings of the Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 2486–2493.
25. Khosoussi, K.; Sukhatme, G.S.; Huang, S.; Dissanayake, G., Designing Sparse Reliable Pose-Graph SLAM: A Graph-Theoretic Approach. In *Algorithmic Foundations of Robotics XII: Proceedings of the Twelfth Workshop on the Algorithmic Foundations of Robotics*; Springer International Publishing, 2020; pp. 17–32. https://doi.org/10.1007/978-3-030-43089-4_2.
26. Kretzschmar, H.; Stachniss, C. Information-theoretic compression of pose graphs for laser-based SLAM. *The International Journal of Robotics Research* **2012**, *31*, 1219–1230.
27. Ila, V.; Porta, J.M.; Andrade-Cetto, J. Information-Based Compact Pose SLAM. *IEEE Transactions on Robotics* **2010**, *26*, 78–93. https://doi.org/10.1109/TRO.2009.2034435.
28. Placed, J.A.; Strader, J.; Carrillo, H.; Atanasov, N.; Indelman, V.; Carlone, L.; Castellanos, J.A. A Survey on Active Simultaneous Localization and Mapping: State of the Art and New Frontiers. *IEEE Transactions on Robotics* **2023**, *39*, 1686–1705. https://doi.org/10.1109/TRO.2023.3248510.

 doi:10.20944/preprints202603.1477.v1

29. Carlevaris-Bianco, N.; Eustice, R.M. Information-theoretic loop closure detection in SLAM. *The International Journal of Robotics Research* **2013**, *32*, 1155–1176.
30. Carlevaris-Bianco, N.; Eustice, R.M. Generic Factor-Based Node Marginalization and Edge Sparsification for Pose-Graph SLAM. In Proceedings of the Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, 2013; pp. 5748–5755.
31. Carlevaris-Bianco, N.; Eustice, R.M. Conservative Edge Sparsification for Graph SLAM Node Removal. In Proceedings of the Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 2014; pp. 854–860.
32. Doherty, K.J.; Rosen, D.M.; Leonard, J.J. Spectral Measurement Sparsification for Pose-Graph SLAM. *arXiv preprint arXiv:2203.13897* **2022**.
33. Choudhary, S.; Indelman, V.; Christensen, H.I.; Dellaert, F. Information-based Reduced Landmark SLAM. In Proceedings of the Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 4620–4627. https://doi.org/10.1109/ICRA.2015.7139839.
34. Doherty, K.; Papalia, A.; Huang, Y.; Rosen, D.; Englot, B.; Leonard, J.J. MAC: Graph Sparsification by Maximizing Algebraic Connectivity. *arXiv preprint arXiv:2403.19879* **2024**.
35. Davis, T.A. *Direct methods for sparse linear systems*; Society for Industrial and Applied Mathematics, 2006.
36. Pukelsheim, F. *Optimal Design of Experiments*; Vol. 50, *Classics in Applied Mathematics*, Society for Industrial and Applied Mathematics: Philadelphia, PA, 2006.
37. Trefethen, L.N.; Bau, D. *Numerical linear algebra*; Society for Industrial and Applied Mathematics, 1997.
38. Eisenstat, S.C.; Walker, H.F. Choosing the Forcing Terms in an Inexact Newton Method. *SIAM Journal on Scientific Computing* **1996**, *17*, 16–32. https://doi.org/10.1137/0917002.
39. Björck, Å. *Numerical Methods for Least Squares Problems*; Society for Industrial and Applied Mathematics: Philadelphia, PA, 1996. https://doi.org/10.1137/1.9780898719451.
40. Grisetti, G.; Stachniss, C.; Burgard, W. Nonlinear Constraint Network Optimization for Efficient Map Learning. *IEEE Transactions on Intelligent Transportation Systems* **2009**, *10*, 428–439. https://doi.org/10.1109/TITS.2009.2026444.
41. Carlone, L. Datasets, 2D Pose Graph Optimization, 2025.
42. Carlone, L.; Tron, R.; Daniilidis, K.; Dellaert, F. Initialization Techniques for 3D SLAM: a Survey on Rotation Estimation and its Use in Pose Graph Optimization. In Proceedings of the Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015, pp. 4597–4604.
43. Kabsch, W. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* **1976**, *32*, 922–923.
44. Kabsch, W. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* **1978**, *34*, 827–828.
45. Ila, V.; Polok, L.; Solony, M.; Svoboda, P. SLAM++ – A highly efficient and temporally scalable incremental SLAM framework. *The International Journal of Robotics Research* **2017**, *36*, 210–230.