香港科技大学(广州)
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

系统枢纽
SYSTEMS HUB

智能交通学域
INTELLIGENT TRANSPORTATION THRUST

Non-Linear Optimization

Graduate Course INTR-6000P

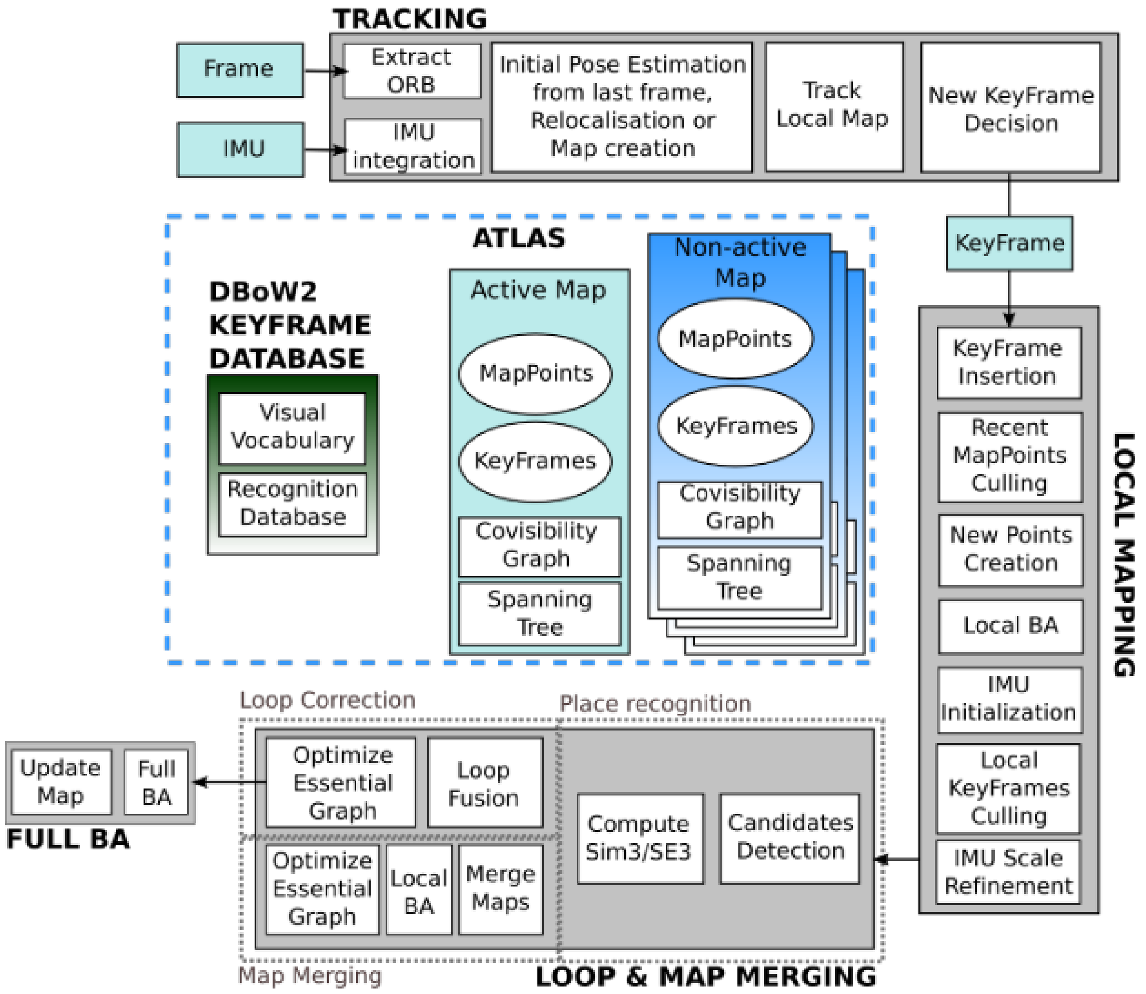
Week 6 - Lecture 12

Changhao Chen

Assistant Professor

HKUST (GZ)

Recap: The SLAM Problem



Visual SLAM
(Simultaneous Localization and Mapping)

Goal: Build a consistent global map of the environment while simultaneously localizing within it.

Focus: Global consistency. Output: A globally consistent map and trajectory.

Solution to Drift: Loop Closing - detecting previously visited locations and correcting the entire map.

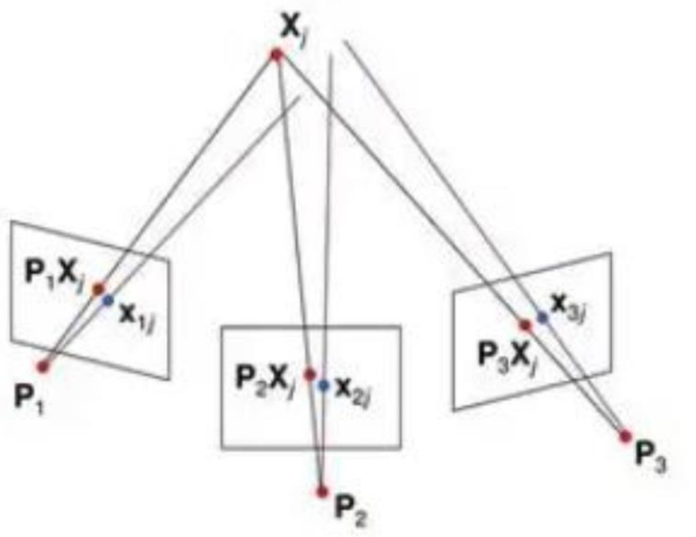
Recap: Bundle Adjustment in SLAM

The Goal: Find the optimal configuration of all camera poses and all 3D points that is most consistent with all observed 2D image measurements.

"Bundle" refers to the "bundles" of light rays connecting camera centers to 3D points.

"Adjustment" refers to the iterative refinement of parameters.

It is a large-scale Non-Linear Least Squares optimization problem.



Recap: Bundle Adjustment in SLAM

BA minimizes the total reprojection error across all cameras and all points.

$$\min_{\{\mathbf{R}_i,\mathbf{t}_i\},\{\mathbf{P}_j\}} \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{v}_{ij} \,||\, \mathbf{p}_{ij} - \pi(\mathbf{K}[\mathbf{R}_i|\mathbf{t}_i]\mathbf{P}_j) \,||_2^2$$

Where:

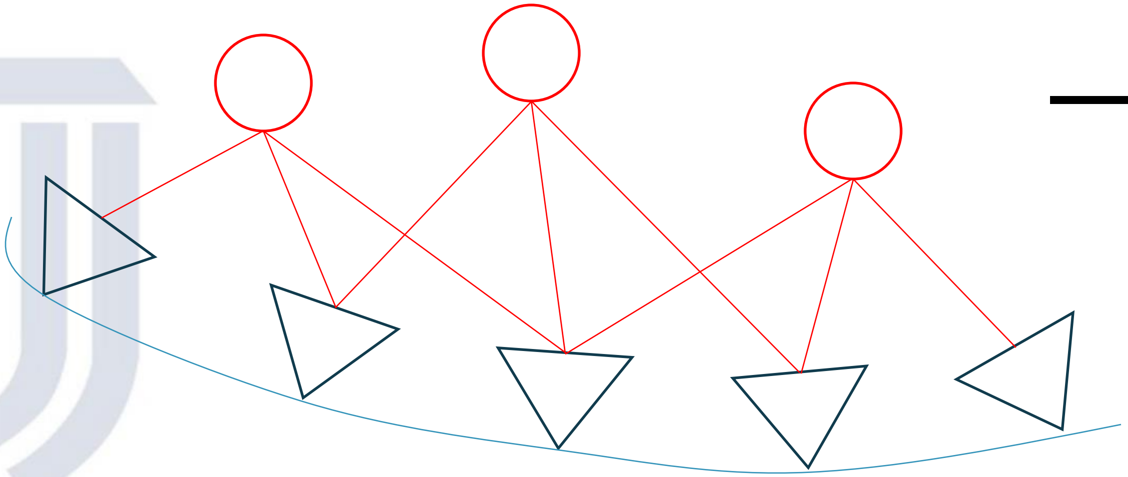
- m: number of cameras
- n: number of 3D points
- v_ij: Visibility function (1 if point j is seen in camera i, 0 otherwise)
- p_ij: Observed 2D image point of P_j in camera I
- π(...): Projection function

- **Parameters to Optimize:**
  - **All Camera Poses:** {R_i, t_i} for i = 1 ... M
  - **All 3D Points:** {P_j} for j = 1 ... N
- **Number of parameters can be huge:** e.g., 1000 frames & 10,000 points → ~33,000 parameters!
- **This is a massive, but very structured, optimization problem.**

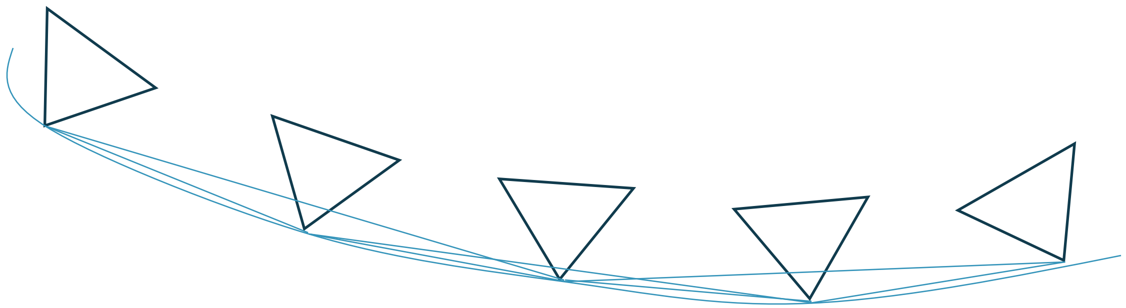
Problem with Full Bundle Adjustment

Bundle Adjustment (BA)

- Optimizes both camera poses and 3D feature point locations for high precision.
- However, in large-scale scenes, the huge number of feature points makes computation extremely heavy and inefficient.
- To address this, we introduce a simplified variant — **Pose Graph Optimization (PGO)**.

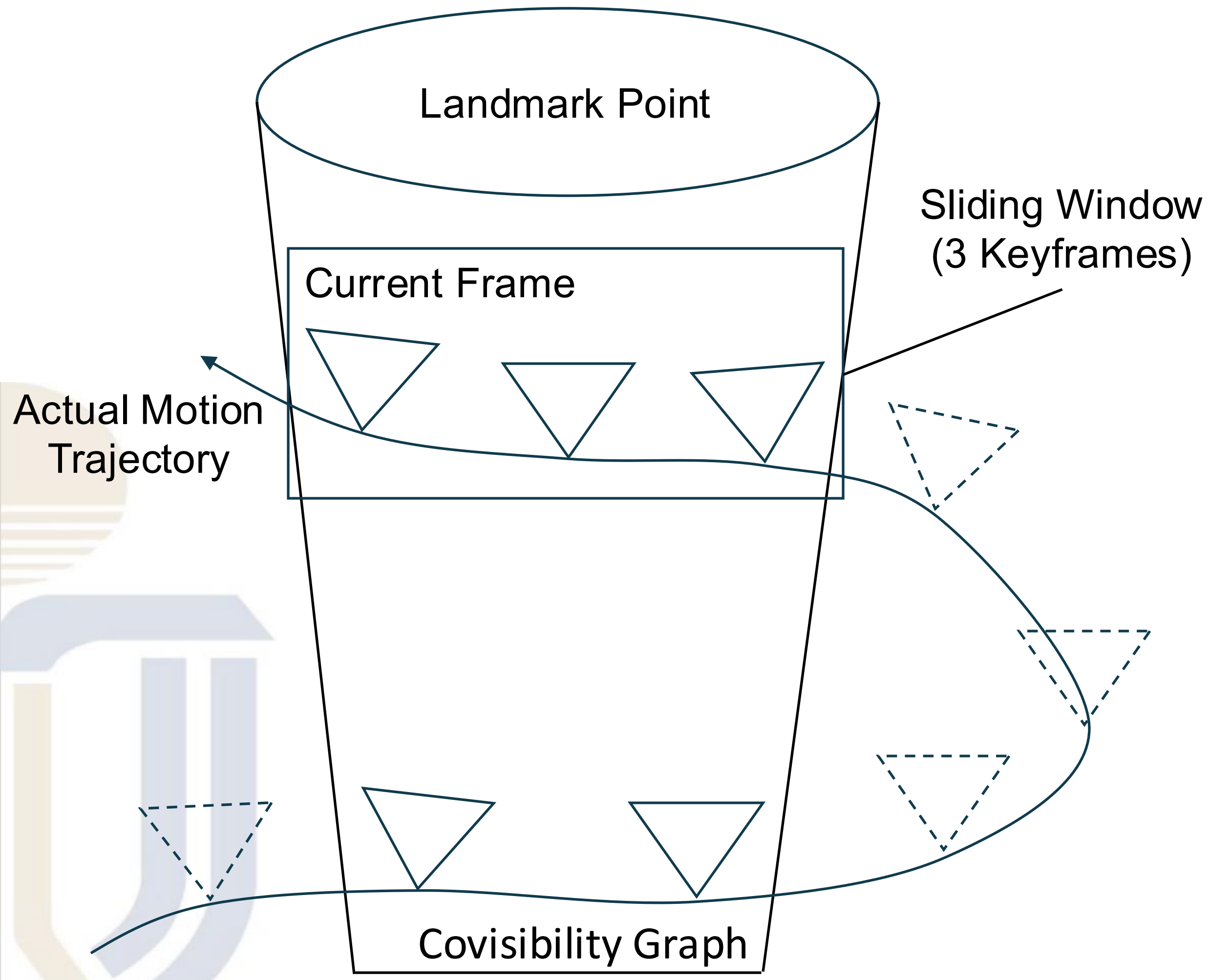


Bundle Adjustment



Pose Graph

Idea of Sliding Window Optimization



Schematic Diagram of Sliding Windows and Covisibility Graph

Sliding Window - A method that optimizes only recent keyframes within a fixed-size window to maintain accuracy while keeping computation efficient.

- Maintain only a **fixed-size local window** of recent keyframes and landmarks.
- Perform optimization only within this window.
- When a new keyframe arrives, **add the new keyframe** and **remove the oldest one**.
- This strategy balances accuracy and efficiency.

Sliding Window Mechanism

Window Composition:

- Assume the window contains N keyframes with poses $x_1, \dots, x_N$.
- Optimize these poses and possibly local landmarks using observations within the window.

When a New Keyframe is Added:

- Add the new keyframe x_{N+1} and its observed landmarks.
- Perform local BA within the window.

When an Old Keyframe is Removed (Marginalization):

- The oldest keyframe's variables are eliminated.
- Its information is preserved by adding **prior constraints** between remaining variables.
- This is achieved by **Schur complement**, converting the removed variables into constraints.

Marginalization and Its Effect

Hessian Structure and Fill-in:

- Before marginalization, the Hessian matrix has a block-sparse structure.
- After eliminating variables (e.g., x_1), non-zero entries appear between previously unconnected states → “**fill-in**”.
- Fill-in increases computational complexity if not handled properly.

Practical Interpretation:

- Marginalization ≈ keeping the conditional probability of remaining variables given removed ones.
- The prior term ensures past information is preserved.

Key Insight:

- Sliding Window Filtering (SWF) maintains local consistency while discarding old variables, enabling real-time optimization suitable for **VO / VIO systems**, but not for large-scale global SLAM.

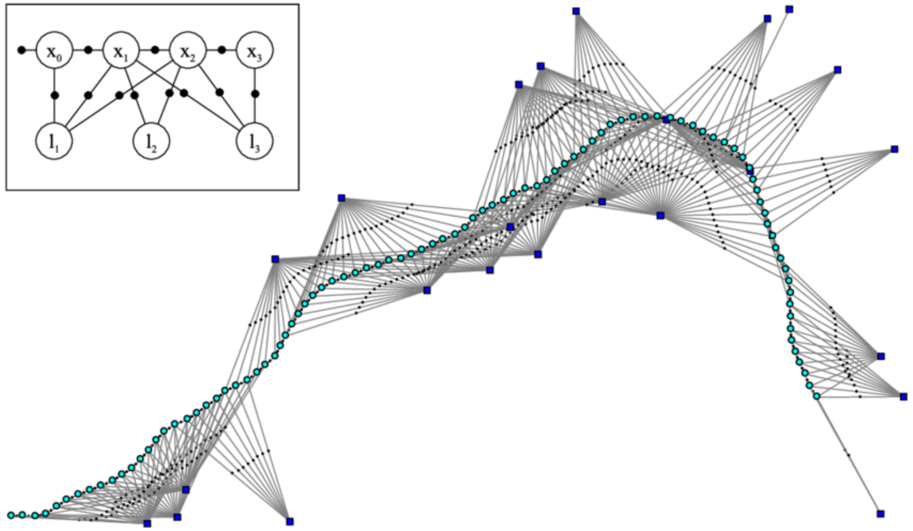
Concept of Pose Graph Optimization

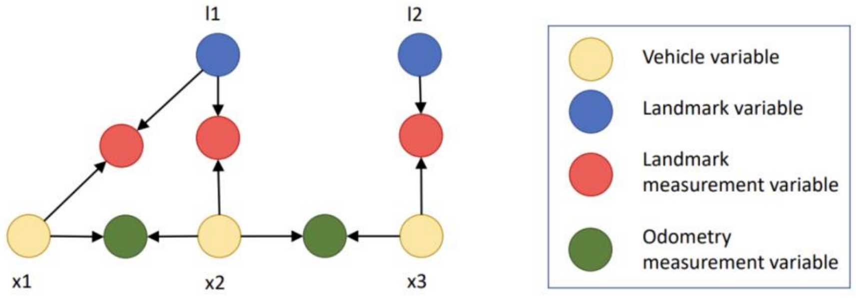
Pose Graph – Simplifying the Optimization Problem

- In full Bundle Adjustment (BA), both camera poses and feature points are optimized.
  → Most computational cost comes from feature points.
- After multiple observations, feature points become stable; we only need accurate **camera poses**.

- **Idea:**
  - Represent each keyframe as a **node** (pose).
  - Represent spatial constraints (e.g., relative motion, loop closures) as **edges**.
  - Optimize only these poses instead of 3D landmarks
    → **Pose Graph Optimization (PGO)**.

Advantages:
- Reduces computation by eliminating landmarks.
- Suitable for large-scale SLAM or loop closure refinement.





Mathematical Formulation of Pose Graph

Relative Pose Constraints

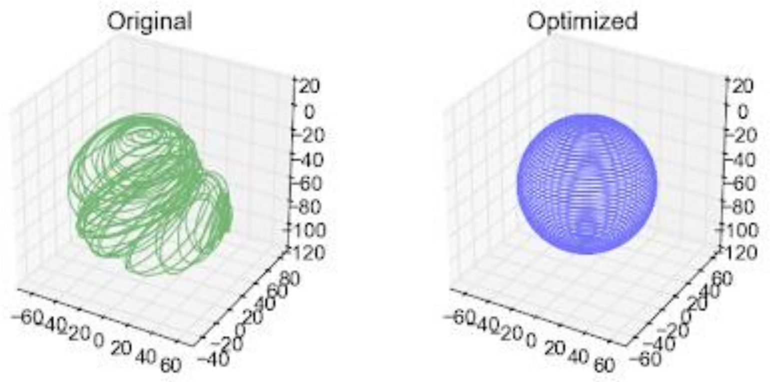
Each edge represents the relative motion between two poses T_i, T_j:

$$\Delta T_{i,j} = T_i^{-1} T_j$$

The corresponding **error term** is:

$$e\_ij = \ln(\Delta T_{ij}^{-1} T_i^{-1} T_j)^{\vee}$$

Where $\ln(\cdot)^{\vee}$ maps $SE(3)$ to its Lie algebra representation.

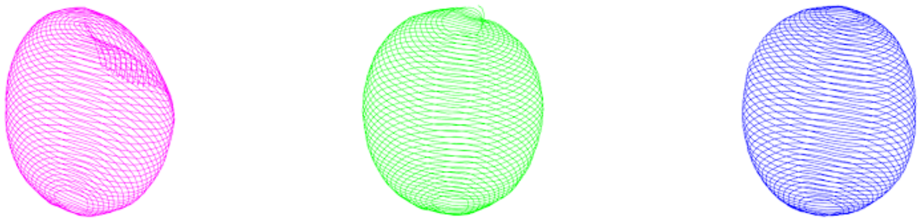


Optimization Objective:

Minimize the sum of squared pose errors across all connected nodes:

$$\min_{\{T_i\}} \frac{1}{2} \sum_{(i,j) \in \varepsilon} e_{ij}^{\intercal} \sum_{ij}^{-1} e_{ij}$$

Where $\sum ij$ is the information matrix of the edge.



Optimization Details and Practical Insights

Jacobian Derivation:

For each edge (i,j), the Jacobians of residual e_{ij} w.r.t. poses ξ_i, ξ_j:

$$\frac{\partial e_{ij}}{\partial \xi_i} = -J_r^{-1}(e_{ij})Ad(T_j^{-1}), \qquad \frac{\partial e_{ij}}{\partial \xi_j} = J_r^{-1}(e_{ij})Ad(T_j^{-1})$$

Where J_r^{-1} is the inverse right Jacobian of SE(3).

In practice, J_r is often approximated by the identity matrix to simplify computation.

Implementation Notes:

- PGO is typically solved using non-linear least squares (e.g., **Gauss-Newton**, **Levenber-Marquardt**).
- Common libraries: **g2o**, **Ceres Solver**.
- Pose Graph Optimization is often used for **loop closure correction** or **global map alignment**.

Full BA vs. Pose Graph Optimization

Full Bundle Adjustment (BA):

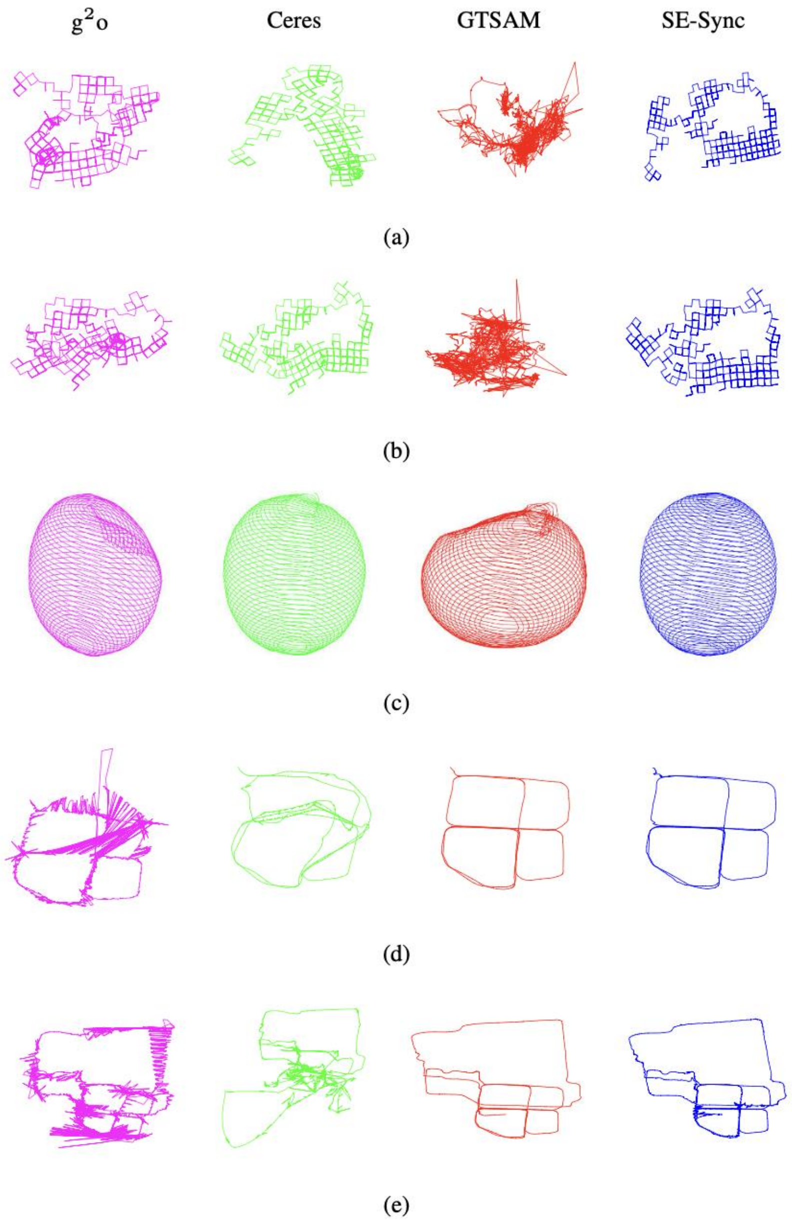
- *Pros:* Jointly optimizes all camera poses and feature points → highest accuracy.
- *Cons:* Computationally expensive (millions of variables in large maps).
- *Best For:* Offline refinement, small-scale scenes, structure-from-motion.

Pose Graph Optimization (PGO):

- *Pros:* Optimizes only poses using relative constraints → lightweight and scalable.
- *Cons:* Ignores 3D point reprojection error; less precise than full BA.
- *Best For:* Loop closure correction, large-scale SLAM back-end, global consistency.

Main Optimization Libraries

- **g2o** – General Graph Optimization
- **Ceres Solver** – Flexible nonlinear least squares
- **GTSAM** – Factor-graph based incremental optimization
- **SE-Sync** – Certifiably correct convex relaxation method



Comparison of Optimization Libraries

Library	Core Concept	Strengths	Limitations	Used In
g2o	Sparse graph optimization (Gauss-Newton, LM, Dogleg)	Efficient, extensible, widely used	Slower on large 3D graphs	ORB-SLAM, SVO
Ceres Solver	Nonlinear least squares framework	Flexible, user-friendly, supports custom residuals	Needs good initialization, may get stuck in local minima	OKVIS, VINS-Mono
GTSAM	Factor graph + iSAM incremental smoothing	Incremental updates, supports multiple sensors	Higher memory use, slower in large static problems	SVO-GTSAM, academic & industry
SE-Sync	Convex semidefinite relaxation (SE(n))	Globally optimal, fast in high noise	Requires specific problem form, complex implementation	Research / high-accuracy robotics

Experimental Insights & Takeaways

Benchmark Datasets:

- 2D: INTEL, MIT, M3500 series.
- 3D: Sphere-a, Torus, Cube, Garage, Cubicle, Rim.

Performance Summary:

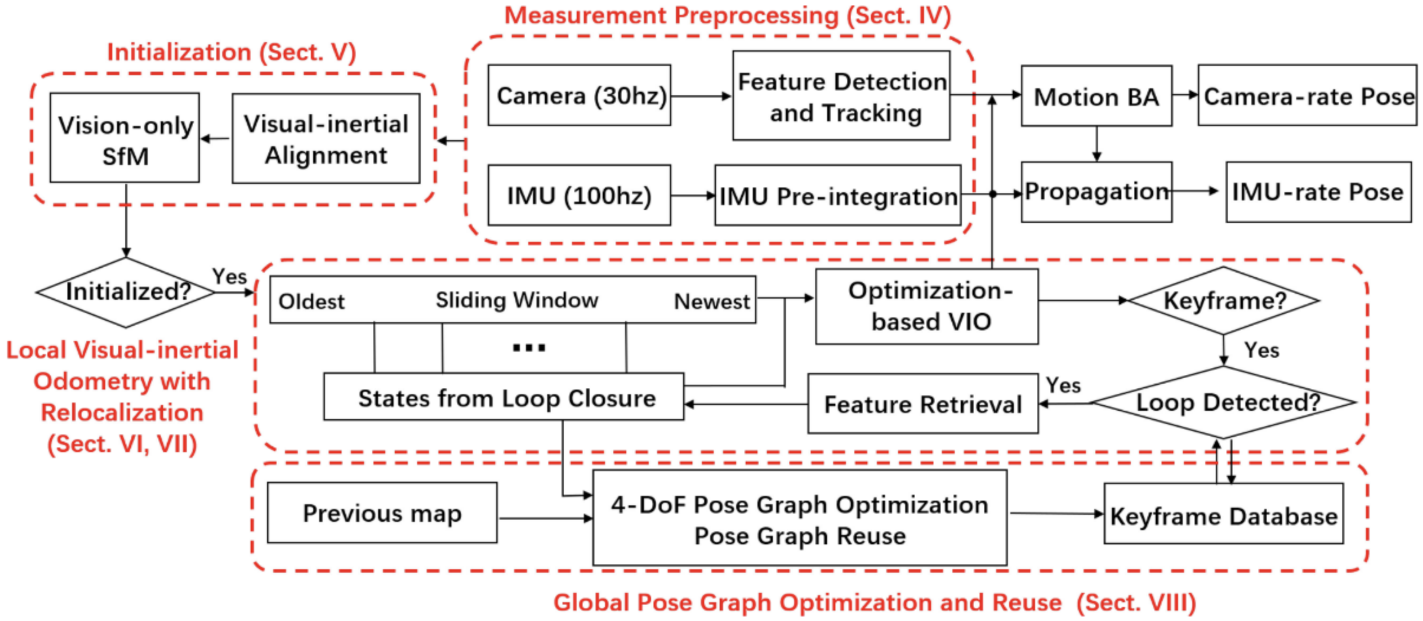
- **SE-Sync** – Fastest, achieves near-global optima, robust to high noise.
- **GTSAM** – Strong accuracy and stability, especially with good initialization.
- **Ceres** – Best for flexibility and ease of integration; fast for medium-sized problems.
- **g2o** – Reliable but slower on large or noisy datasets.

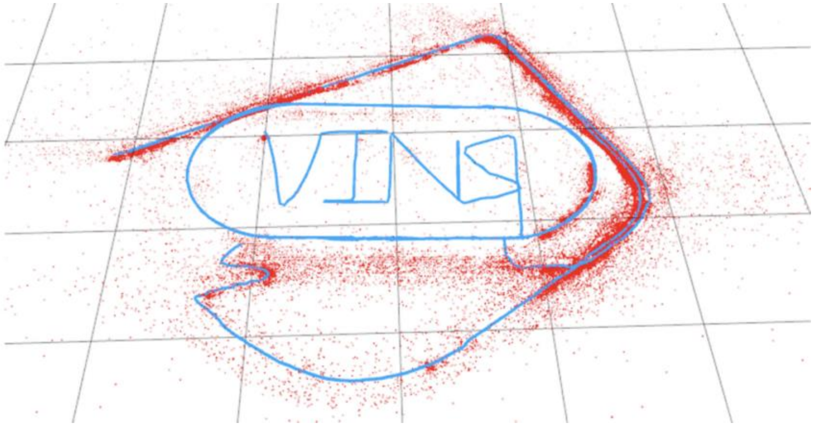
Practical Recommendations:

- For **research / global SLAM** → SE-Sync or GTSAM.
- For **real-time VO / VIO systems** → Ceres or g2o.
- For **large-scale loop closure refinement** → GTSAM (iSAM2).

Application of PGO in Classic SLAM Systems

VINS-Mono (Qin et al., 2018, IEEE T-RO)





Application of PGO in Classic SLAM Systems

VINS-Mono (Qin et al., 2018, IEEE T-RO)

1) Sliding-Window Optimization (Local Graph)

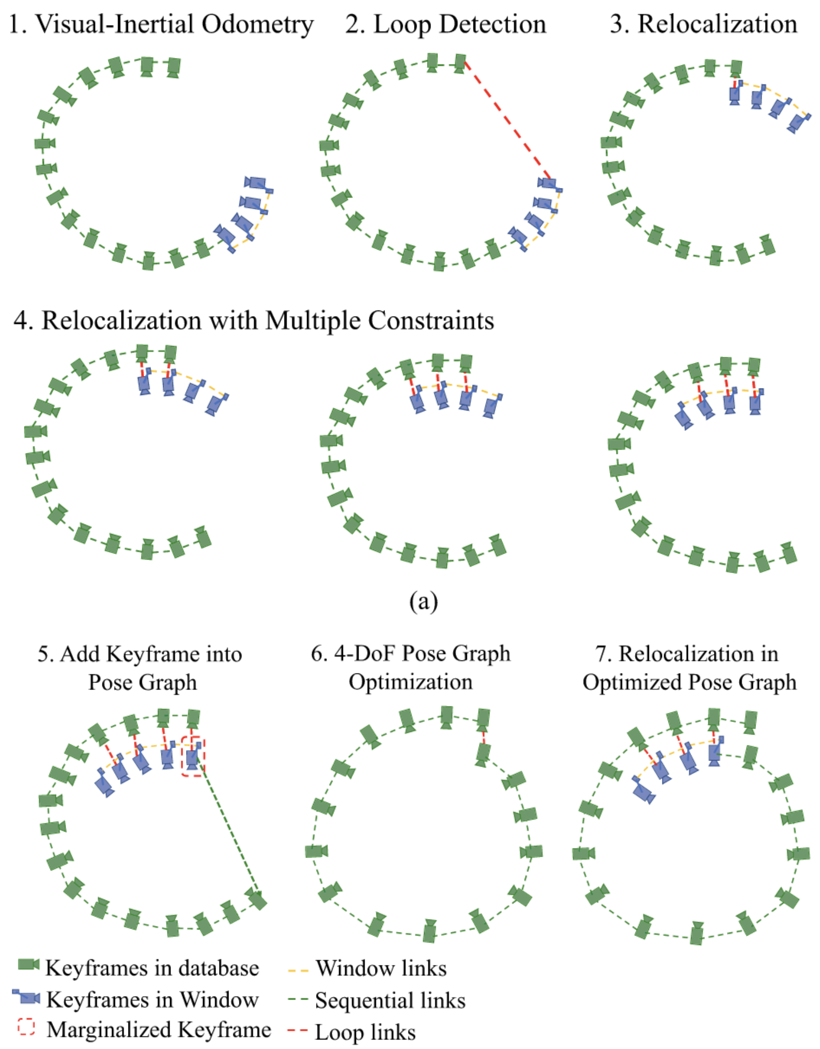
- Type: Nonlinear factor graph optimization
- Goal: Estimate recent IMU and camera states within a bounded window
- Variables:
  - IMU poses, velocities, biases
  - Camera–IMU extrinsic parameters
  - Feature inverse depths
- Factors:
  - IMU preintegration factors (between consecutive frames)
  - Visual reprojection factors (feature observations)
  - Prior factor (from marginalized old states)
- Solver: Ceres (Levenberg–Marquardt / Gauss–Newton)
- Purpose: High-accuracy visual–inertial odometry (VIO) with real-time feasibility
- Marginalization: Keeps window fixed size using Schur complement

Application of PGO in Classic SLAM Systems

VINS-Mono (Qin et al., 2018, IEEE T-RO)

2) Relocalization (Loop Constraints Integration)

- Type: Extended local graph optimization
- Goal: Correct accumulated drift by reusing past keyframes
- Mechanism:
  - Loop detection via DBoW2 (BoW place recognition)
  - Retrieve feature matches → add loop-closure factors into current optimization
  - Past poses treated as constants → tightly coupled relocalization
- Effect: Achieves drift-free estimation without full re-optimization of all states



Application of PGO in Classic SLAM Systems

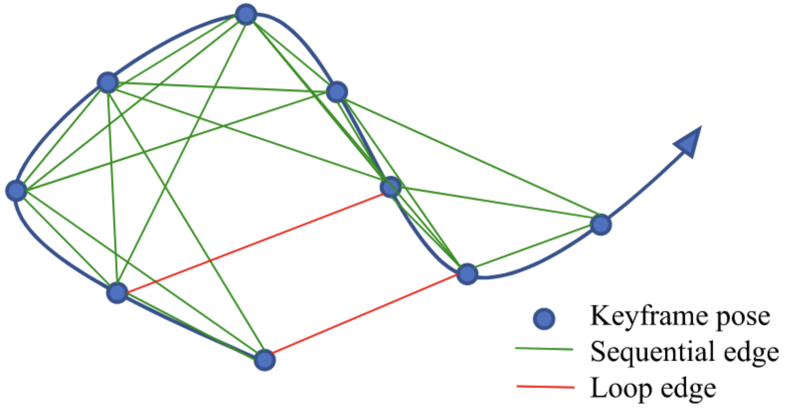
VINS-Mono (Qin et al., 2018, IEEE T-RO)

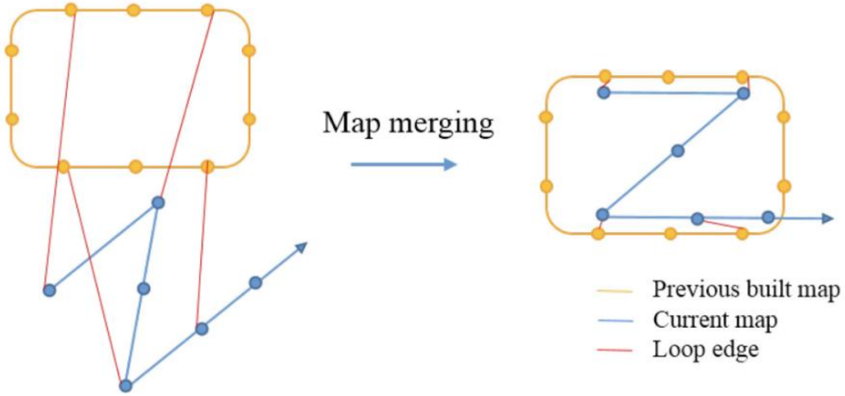
3) Global Pose Graph Optimization (Global Graph)

- Type: 4-DOF Pose Graph Optimization (x, y, z, yaw)
- Goal: Enforce global consistency and map reuse
- Vertices: Keyframes (poses)
- Edges:
  - Sequential edges (from VIO relative motion)
  - Loop-closure edges (from relocalization)
- Optimization:

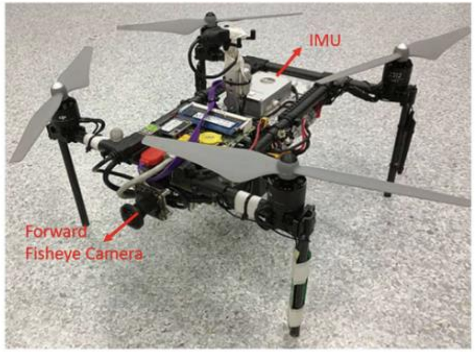
$$\min_{p,\psi} \sum_{(i,j)\in S} \|r_{ij}\|^2 + \sum_{(i,j)\in L} \rho(\|r_{ij}\|^2)$$

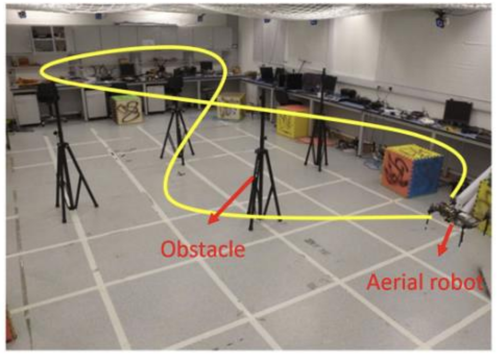
- Parallel Thread: Runs asynchronously with VIO
- Function: Global drift correction, multi-session map merging, and loop closure refinement





Application of PGO in Classic SLAM Systems



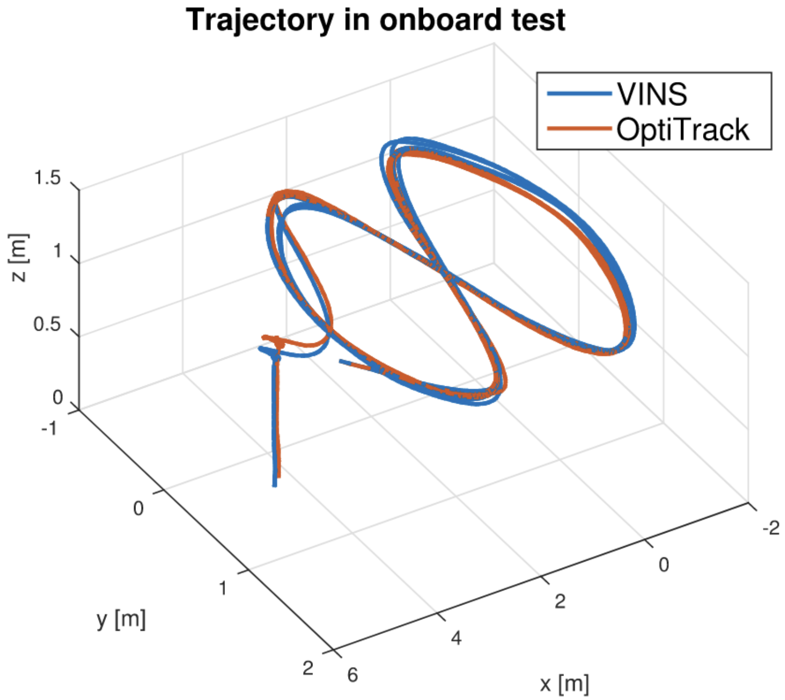


VINS-Mono (Qin et al., 2018, IEEE T-RO)

Summary:

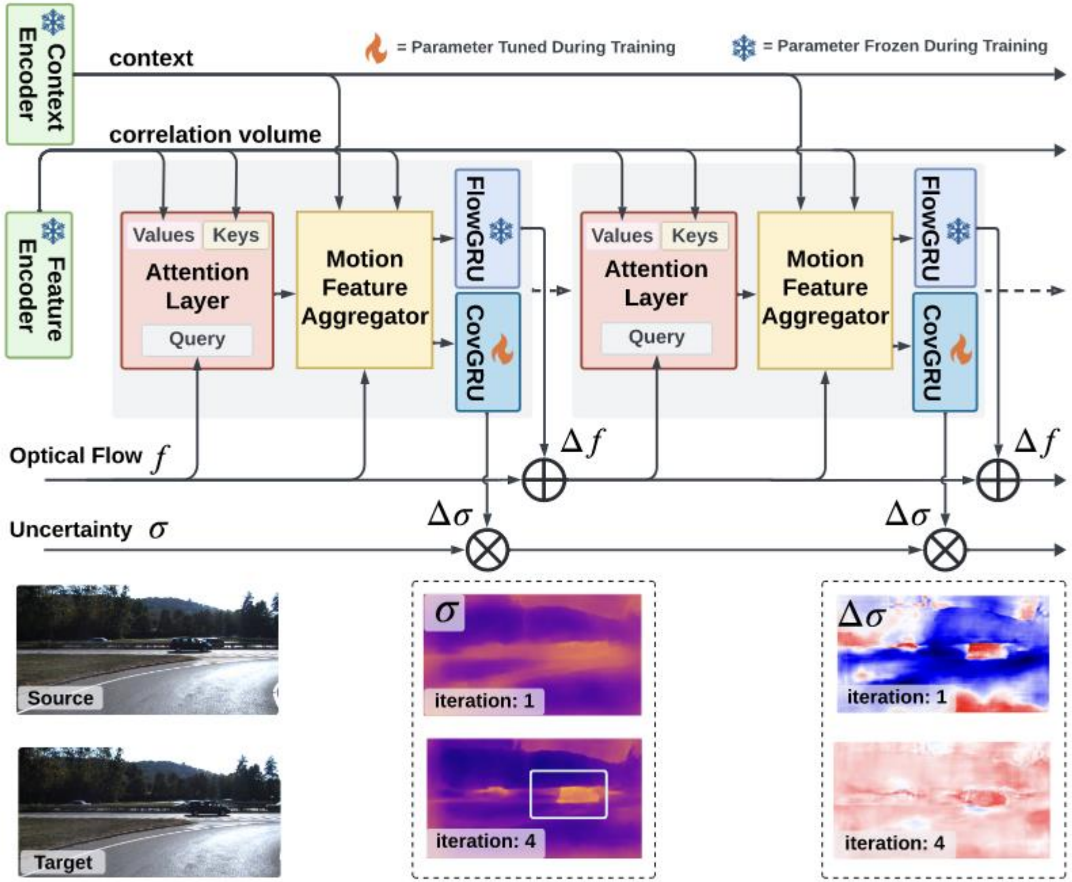
VINS-Mono integrates multi-level graph optimization:

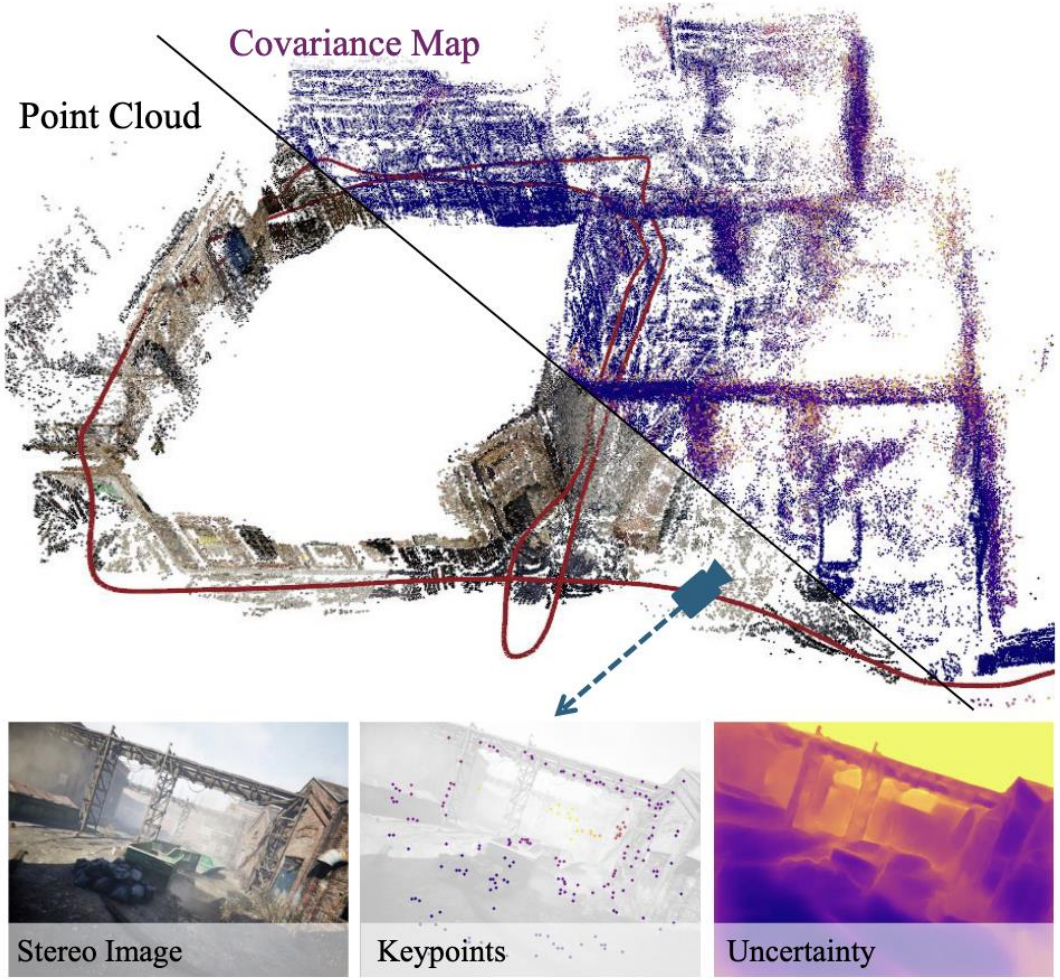
- Local sliding-window BA for real-time precision,
- Loop-closure graph for drift correction,
- Global pose graph for map consistency and reuse.



Application of PGO in Novel SLAM Systems

MAC-VO (Qiu et al., ICRA 2025 Best Paper)



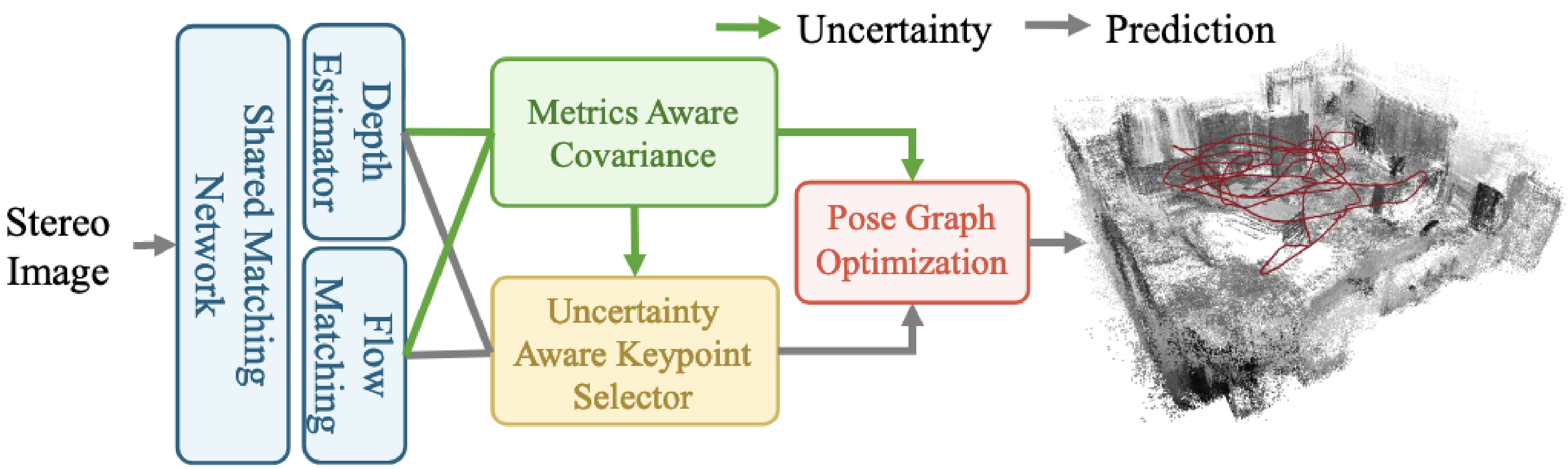


Application of PGO in Novel SLAM Systems

MAC-VO (Qiu et al., ICRA 2025 Best Paper)

1) Optimization Purpose

- The system does not rely on multi-frame bundle adjustment.
- Instead, it performs two-frame pose graph optimization to estimate relative camera motion.
- Goal: minimize the 3D distance between matched keypoints across consecutive frames, weighted by their learned covariance.



Application of PGO in Novel SLAM Systems

MAC-VO (Qiu et al., ICRA 2025 Best Paper)

2) Graph Structure

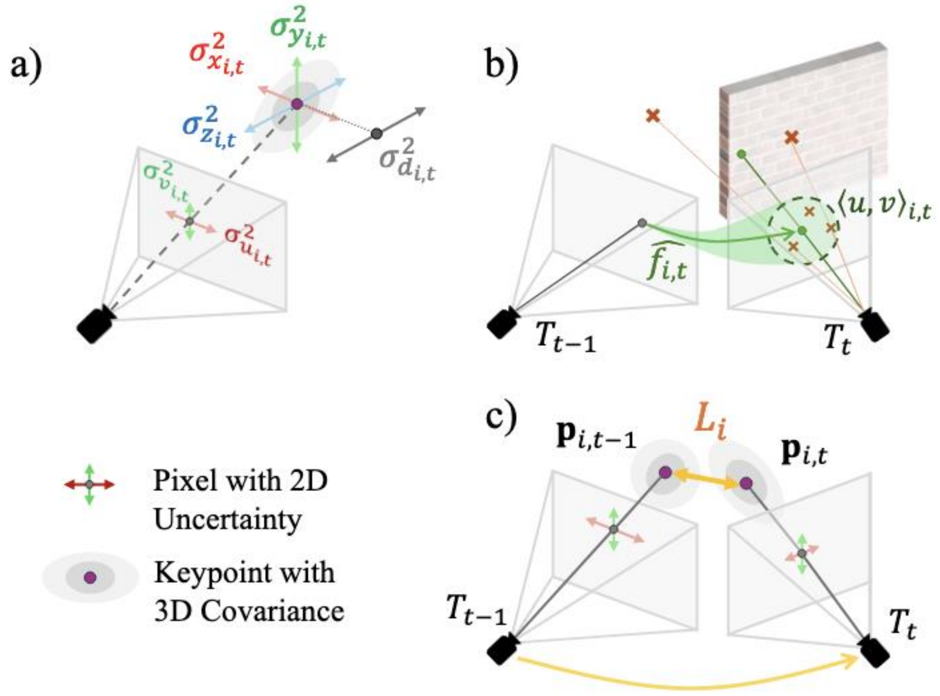
- Nodes: camera poses $T_t \in SE(3)$ at each frame.
- Edges (factors): correspondences between 3D keypoints across two frames.
- Each edge encodes a residual constraint:

$$L_i = \left\| p_{i,t-1} - T_t c p_{i,t} \right\|_{\Sigma_i}^2$$

where $p_{i,t-1}$ and $cp_{i,t}$ are matched keypoints, and

$$\Sigma_i = \Sigma_{p_{i,t-1}} + R_t \Sigma_{p_{i,t}} R_t^\intercal$$

is the metrics-aware covariance weighting matrix.



Application of PGO in Novel SLAM Systems

MAC-VO (Qiu et al., ICRA 2025 Best Paper)

3) Optimization Algorithm

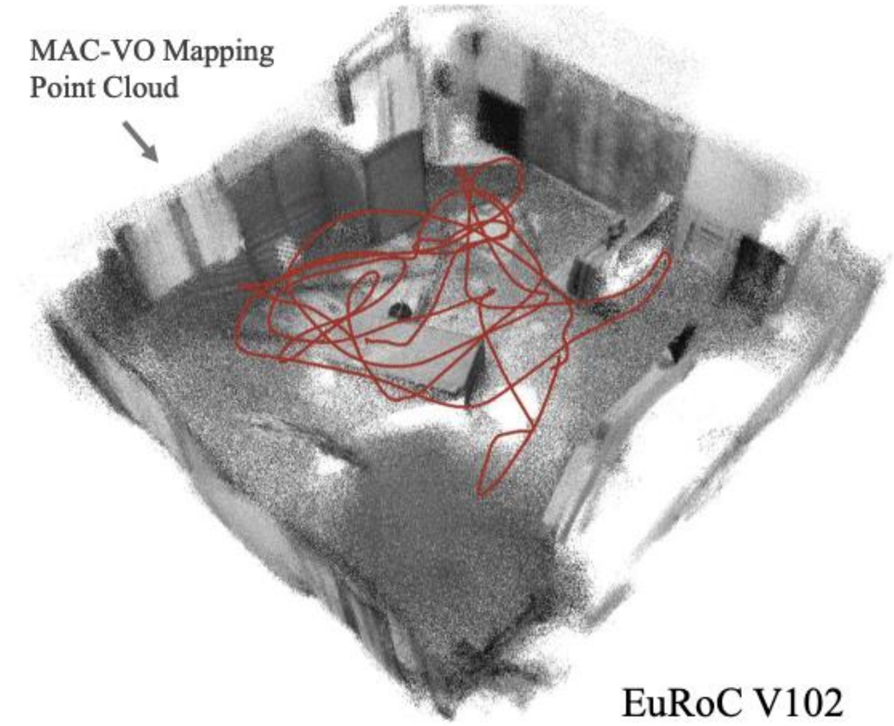
- Objective function:

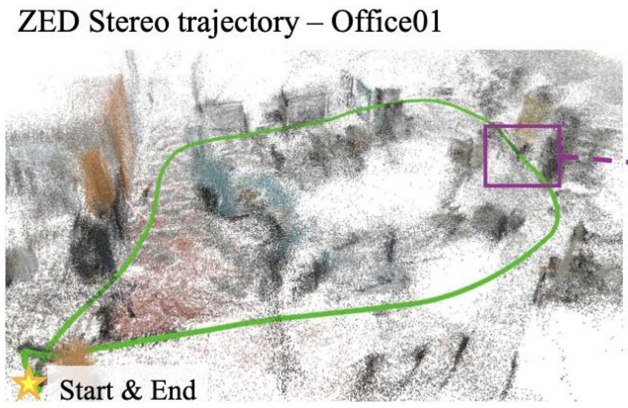
$$T^{\star} = \arg\min_{T_t} \sum_{i} L_i$$

- Solved using Levenberg-Marquardt (LM) optimization implemented via PyPose.

- Covariance matrices are non-diagonal, modeling inter-axis correlations (x-y-z).

- This allows the optimizer to adapt to anisotropic and correlated uncertainty from learned features.





Application of PGO in Novel SLAM Systems

MAC-VO (Qiu et al., ICRA 2025 Best Paper)

4) Distinction from Traditional SLAM

Traditional BA / PGO	MAC-VO Pose Graph
Uses multi-frame optimization	Uses only **two-frame** optimization
Covariance assumed diagonal or constant	Covariance is **learned & metrics-aware**
Features selected via gradient or heuristic	Keypoints selected using **uncertainty filtering**
Often includes loop closure	No loop closure, only local PGO

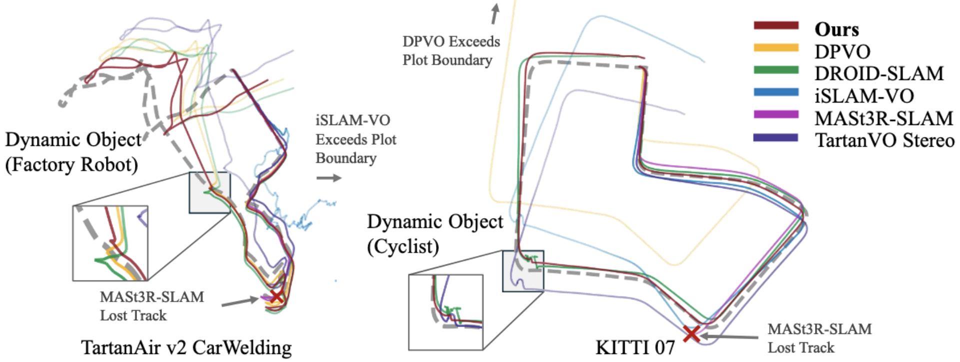
Application of PGO in Novel SLAM Systems

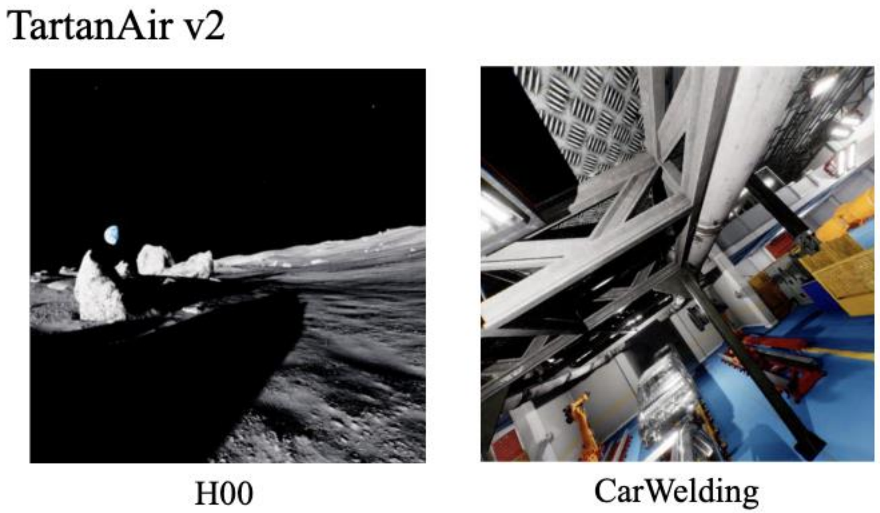
MAC-VO (Qiu et al., ICRA 2025 Best Paper)

Summary:

Role of Graph Optimization in the System

- Serves as the back-end of the learning-based front-end.
- Integrates 3D geometry with learned uncertainty to achieve robust motion estimation.
- Enables high accuracy without global BA or loop closure, especially in dynamic or low-texture scenes.





Future Trends & Research Directions

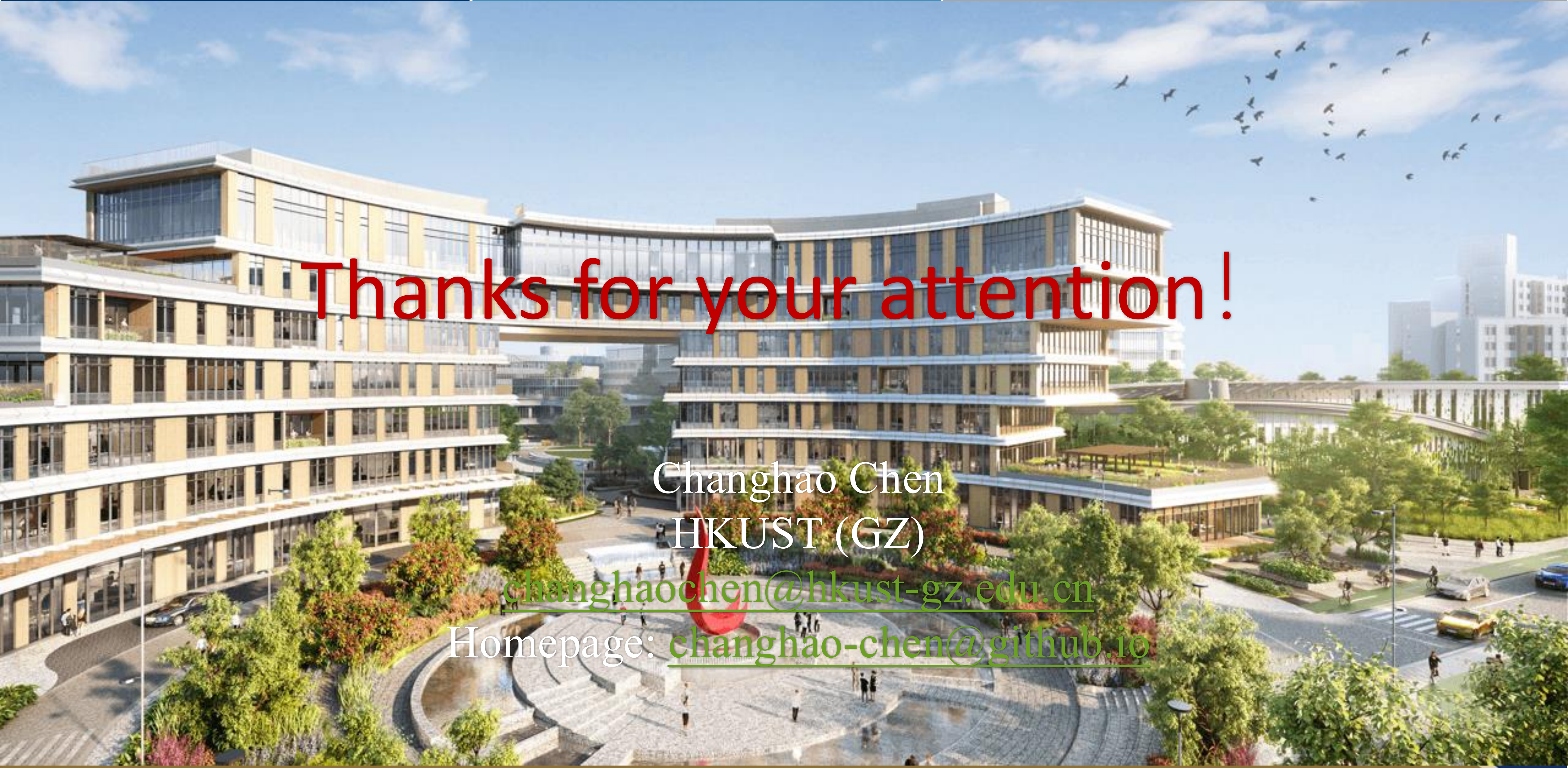
1) **Hybrid Graph Learning:** Integrate deep neural networks with factor graph optimization to learn better priors, noise models, and initialization strategies for nonlinear solvers.

2) **Dynamic and Adaptive Graphs:** Enable real-time adaptation of the factor graph structure for dynamic environments — adding, removing, or reweighting factors as scene conditions change.

3) **Semantic-aware Factors:** Introduce semantic constraints (e.g., object-level, lane, or terrain priors) into the factor graph to enhance global consistency and reduce ambiguity.

4) **Cross-modal Graph Fusion:** Combine visual, LiDAR, radar, and inertial modalities within a unified graph optimization framework for robust multi-sensor SLAM.

5) **Distributed and Edge Optimization:** Develop decentralized or federated graph optimization that runs across multiple robots or edge devices, ensuring scalability and resilience.



Thanks for your attention！

Changhao Chen
HKUST (GZ)
changhaochen@hkust-gz.edu.cn
Homepage: changhao-chen@github.io