Journal of Electronic Research and Application, 2026, Volume 10, Issue 4
http://ojs.bbwpublisher.com/index.php/JERA
Online ISSN: 2208-3510
Print ISSN: 2208-3502

Robust Multi-Source Odometry Based on Cascaded Filtering and Hierarchical Optimization

Xiang Fu

FULONGMA CITIBOT Technology Co., Ltd., Changsha 410000, Hunan, China

Abstract: Aiming at the trajectory drift and long-term computing power bottleneck of urban service robots in large-scale scenarios, this paper proposes a practical LIO-RTK-PGO multi-source fusion odometry. The front-end adopts a cascaded tightly coupled architecture, introducing RTK observation to correct the state in the filtering prediction stage to fundamentally suppress elevation and heading divergence; the back-end proposes hierarchical pose graph optimization (PGO), combining local high-frequency sliding window and global keyframe sparsification to control the computational complexity at O(1). Verified by real-vehicle tests and standard computing power platforms, the system eliminates long-range cumulative errors, providing a low-computing-power and highly reliable state estimation scheme for large-scale engineering implementation.

Keywords: LIO; Cascaded tightly coupled; Hierarchical graph optimization; Urban service robot

Online publication: May 21, 2026

1. Introduction

High-precision state estimation is the foundation for the autonomous operation of urban service robots [1,2]. In recent years, LiDAR-Inertial Odometry (LIO) has become the mainstream. The FAST-LIO series has improved the real-time performance of registration, while LIO-SAM has achieved tightly coupled optimization of LiDAR and IMU [3–6].

However, in large-scale urban deployment, existing methods face two major bottlenecks as follows:

(1) Cumulative drift: The front-end lacks global absolute reference, leading to easy elevation and heading divergence during long-distance operation;

(2) Computing power explosion: Global pose graph nodes grow linearly with mileage, and traditional marginalization results in a sharp increase in computational overhead.

To address these issues, this paper proposes a lightweight LIO-RTK-PGO multi-source fusion framework. The core contributions are as follows:

(1) Front-end drift suppression: RTK is introduced in the filtering prediction stage to pre-correct IMU pre-integration errors before tightly coupled update with point clouds;
(2) Hierarchical pose graph: A high-frequency sliding window is maintained locally, and anomalies are processed through keyframe sparsification globally, controlling the back-end complexity at O(1) for easy engineering implementation.

2. Literature review

Existing works are mainly divided into loosely coupled and tightly coupled categories. Loosely coupled LiDAR odometry: Methods such as LOAM and LeGO-LOAM first independently perform point cloud registration to obtain relative poses, then fuse with other sensors through filtering [7,8]. Although easy to implement in engineering, they are prone to slipping in degraded environments such as long straight corridors and fail to fully exploit multi-source complementary characteristics. Tightly coupled LiDAR-Inertial Odometry: Tightly coupled systems directly fuse feature points with IMU pre-integration, significantly improving accuracy and robustness [9–11]. Among filtering methods, FAST-LIO achieves extremely fast local registration but lacks global observation, leading to easy elevation and heading drift during long-term operation. Among graph optimization methods, LIO-SAM unifies relative poses and pre-integration into a factor graph [5,12]. However, in large-scale deployment, RTK multipath effects are likely to cause absolute pose jumps, and global pose graph nodes grow linearly with mileage, consuming substantial computing power for marginalization [13,14].

In summary, existing tightly coupled odometers still face global drift and computing power bottlenecks in large-scale deployment. The cascaded tightly coupled strategy and hierarchical pose graph proposed in this paper provide a lightweight and computing-power-controllable solution.

3. System architecture and methods

The entire system architecture consists of a cascaded tightly coupled front-end filter and a hierarchical robust pose graph optimization back-end, as shown in **Figure 1**.

The front-end module fuses LiDAR, IMU, and RTK. The predicted state of IMU high-frequency forward propagation is first cascaded and updated with RTK to suppress elevation and heading divergence, and simultaneously transmitted to the back-end as a Dead Reckoning (DR) factor; subsequently, the distortion-corrected point cloud is tightly coupled with the corrected state to solve for high-precision LIO odometry.

The back-end module adopts a hierarchical PGO architecture. The local layer fuses LIO odometry and DR constraints through sliding window PGO to output a high-frequency locally consistent “back-end odometer 1”; the global layer introduces low-frequency correction with RTK absolute observations on this basis to output “back-end odometer 2” with eliminated long-range drift.

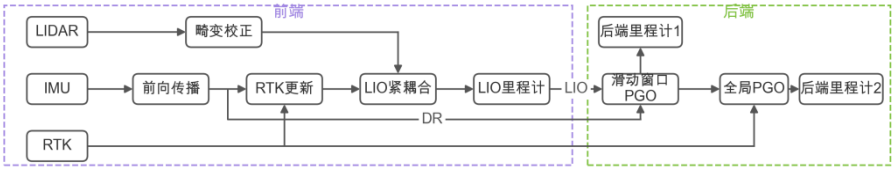

Figure 1. System block diagram.

3.1. Cascaded tightly coupled front-end filter architecture

In traditional LiDAR-Inertial Odometry frameworks, due to the lack of global absolute reference, state estimation is prone to cumulative divergence in the unobservable vertical elevation and heading angle directions during long-distance operation. To address this pain point, we break the traditional loosely coupled paradigm and propose a cascaded tightly coupled state update strategy, as shown in **Figure 2**. The core mathematical derivation is as follows:

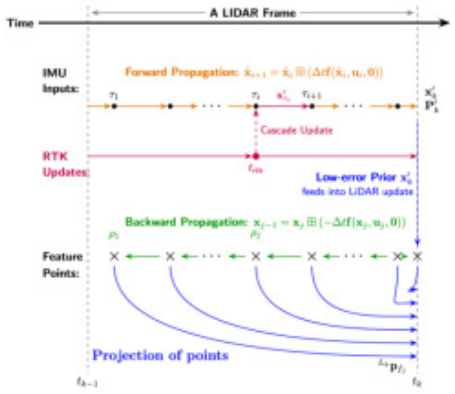

Figure 2. Cascaded filtering diagram.

3.1.1. Error state definition

In this system, the nominal state is denoted as x, and the error state as δx, typically with a dimension of 23. The true state $x_{true} = x \boxplus \delta x$.

The first three dimensions represent the position error of the system in the global coordinate system:

$$\delta x = [\delta p^T, \delta v^T, \delta \theta^T, \delta b_a^T, \delta b_g^T, \ldots]^T \in R^{23}$$

3.1.2. RTK observation residual equation

When valid RTK (UTM) coordinates are received, the observation residual r is constructed. As a tightly coupled front-end filter, the observation equation h(x) is the predicted position, and the actual observation z is the position provided by RTK.

Observation value: $z = p_{rtk} \in R^3$

Residual: $r = z - h(x) = p_{rtk} - p$

3.1.3. RTK observation Jacobian matrix

The Jacobian matrix $H \in R^{3\times 23}$ is defined as the partial derivative of the observation residual with respect to the error state δx.

Since the residual equation is only linearly related to the position state δp (assuming RTK lever arm is compensated externally or ignored), the block of matrix H corresponding to position \delta p is a 3 \times 3 identity matrix, and all other dimensions are 0: $H = \frac{\partial r}{\partial \delta x} = [I_{3\times 3}, 0_{3\times 3}, 0_{3\times 3}, \ldots]$

3.1.4. RTK observation update equation

After obtaining the residual r and Jacobian H, standard Kalman update is performed to correct the nominal state.

Calculate Kalman gain: $K = PH^T(HPH^T + V)^{-1}$

where P is the predicted error state covariance matrix, and V is the RTK observation noise covariance matrix.

Update error state: $\Delta x = Kr$

Update covariance matrix: $P_{new} = (I - KH)P$

After update:

The nominal state x is corrected using the updated error state Δx, and the error state is reset afterward: $x_{new} = x \boxplus \Delta x$

Through the above process, the absolute position observation of RTK can effectively constrain the cumulative errors of IMU pre-integration and LiDAR registration, greatly converging Z-axis elevation deviation and Heading divergence.

3.1.5. LiDAR point cloud observation and cascaded update

After completing the preliminary correction of RTK, we further project LiDAR point clouds into the world coordinate system to construct point-to-plane distance residuals. Regarding the establishment of specific residual models and the derivation of partial derivatives of the corresponding Jacobian matrix H with respect to state quantities, this system adopts the standard method in the FAST-LIO framework, and the detailed derivation process can be referred to in several studies [3,4]. Finally, the system iteratively solves through IESKF until convergence. This cascaded tightly coupled mechanism greatly suppresses front-end error divergence, providing a high-quality prior for subsequent global graph optimization.

3.2. Hierarchical robust pose graph for long-term operation

The system back-end adopts a hierarchical pose graph optimization architecture, including local sliding window and global sparse layers, as shown in **Figure 3**.

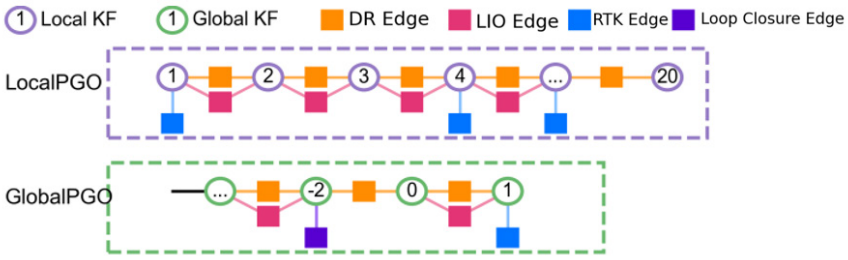

Figure 3. Hierarchical pose graph.

Local sliding window layer extract keyframes (KF) of the front-end odometer that meet the threshold and add them to a fixed-size sliding window. DR edges, LIO edges, and RTK edges are constructed between nodes, and old nodes are marginalized into priors, thereby limiting high-frequency computations locally.

For the global sparse layer, a sparsification strategy is adopted, inserting nodes only when there is significant topological change, strong fixed-solution RTK, or loop closure is detected.

For robust kernel mechanism, considering that dynamic obstacles in urban environments may lead to loop closure mismatches, we introduce the Cauchy robust kernel function into the loop closure factors

and RTK factors in the global layer [15]. This effectively isolates the damage of abnormal jumps to global consistency. The hierarchical design decouples local relative estimation from global correction, ensuring consistency while converging the complexity to O(1) for extreme real-time operation.

4. Results and discussion

To verify the effectiveness of the method in real engineering scenarios, we conducted tests using the Fulongma SD22 sweeper in Xiamen Olympic Park. Taking RTK absolute position and heading as the ground truth, we compared the proposed method with the pure odometer FAST-LIO2 and LIO-SAM with back-end optimization.

4.1. Localization accuracy and Z/heading drift suppression

The XY plane trajectory and Absolute Trajectory Error (ATE) heatmap shown in **Figure 4** intuitively demonstrate the localization accuracy of the three algorithms. The pure front-end FAST-LIO2 accumulated significant omnidirectional drift during long-distance operation (maximum error exceeding 100m, with numerous dark red areas); although LIO-SAM has back-end PGO constraints, its error still shows an obvious local divergence trend due to environmental degradation; in contrast, the proposed LIO-RTK-PGO stabilizes the global Root Mean Square Error (RMSE) at an extremely low level through tightly coupled front-end correction and hierarchical back-end constraints (the entire trajectory is dark blue), achieving perfect alignment with the ground truth trajectory.

As shown in **Figures 5** and **6**, both FAST-LIO2 and LIO-SAM produced varying degrees of drift after long-distance operation. The heading angle error of FAST-LIO2 increased linearly, and severe elevation drift of hundreds of meters occurred in scenarios lacking Z-axis geometric constraints. In comparison, LIO-RTK-PGO introduced real-time RTK observations in the front-end ESKF, not only tightly confining the heading angle error near 0^\circ but also effectively correcting Z-axis divergence. Ultimately, its XY trajectory highly coincided with the real trajectory, providing a very low-error prior for the back-end PGO and demonstrating extremely high global consistency.

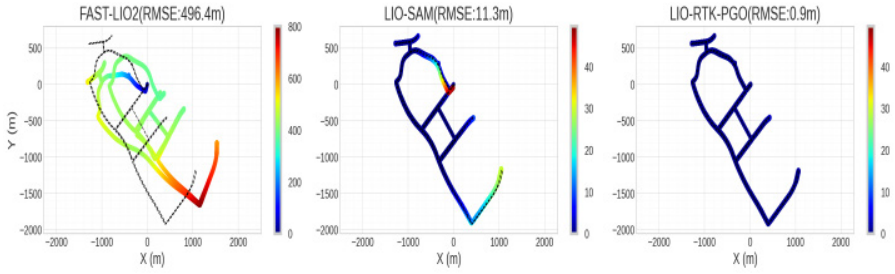

Figure 4. Trajectory heatmap comparison.

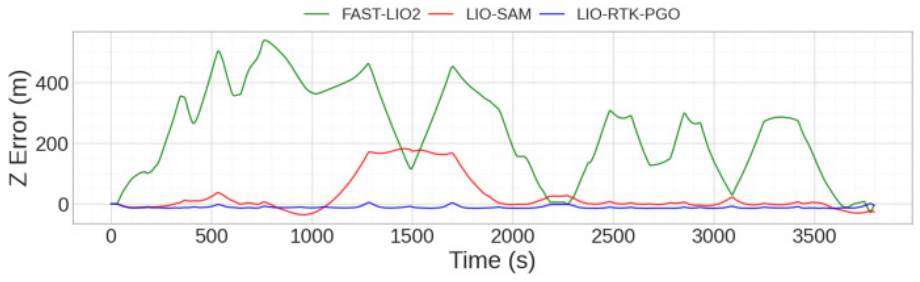

Figure 5. Z-value error comparison.

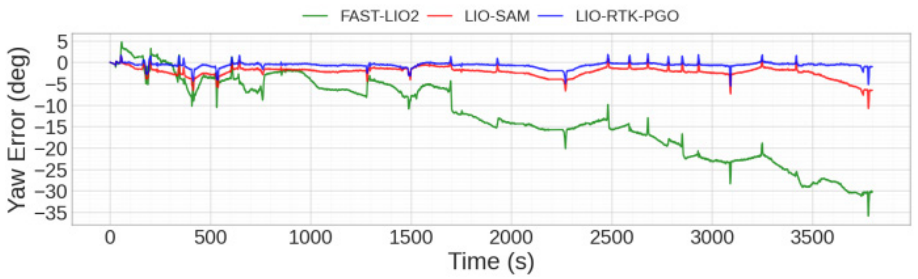

Figure 6. Heading error comparison.

4.2. Computational efficiency analysis

In terms of CPU resource consumption during long-distance operation (as shown in **Figure 7**), the pure local front-end FAST-LIO2 had a stable occupancy rate of 30.05% but sacrificed global consistency; the traditional global back-end LIO-SAM experienced a sharp increase in marginalization overhead due to linear node growth, with an average CPU occupancy of 64.76% showing a divergent trend (maximum exceeding 80%), making it difficult to meet real-time requirements; in contrast, the hierarchical architecture of this paper (LIO-RTK-PGO) maintained an average CPU occupancy of only 38.37% while ensuring global accuracy, remaining absolutely stable over time.

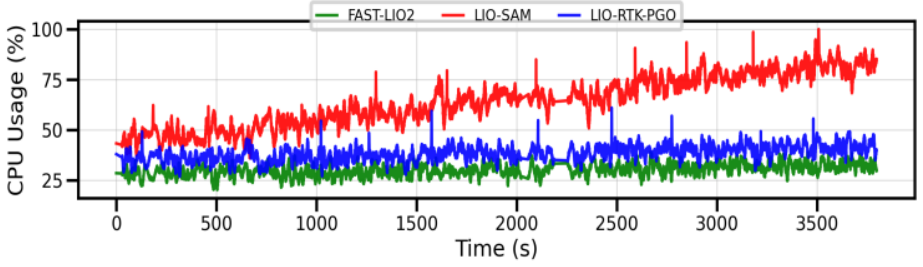

Figure 7. CPU occupancy comparison.

5. Conclusion

In summary, the LIO-RTK-PGO fusion architecture proposed in this paper effectively bridges the gap between cutting-edge academic theories and actual engineering computing power constraints, providing a computing-power-controllable and highly reliable standardized state estimation solution for the large-scale and low-cost commercial deployment of autonomous vehicles and various intelligent service robots. Future work will further explore distributed pose graph optimization in multi-robot collaboration scenarios and multi-modal (visual/LiDAR) deep fusion mechanisms in extreme scenarios with completely no RTK signals (such as underground garages).

Disclosure statement

The author declares no conflict of interest.

References

[1] Lynen S, Achtelik M, Weiss S, et al., 2013, A Robust and Modular Multi-sensor Fusion Approach Applied to MAV Navigation, 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 3923–3929.

[2] Chen C, Zhu H, Li M, et al., 2018, A Review of Visual-Inertial Simultaneous Localization and Mapping from Filtering-Based and Optimization-Based Perspectives. Robotics, 7(3): 45.

[3] Xu W, Zhang F, 2021, FAST-LIO: A Fast, Robust LiDAR-inertial Odometry Package by Tightly-Coupled Iterated Kalman Filter. IEEE Robotics and Automation Letters, 6(2): 3317–3324.

[4] Xu W, Cai Y, He D, et al., 2022, FAST-LIO2: Fast Direct LiDAR-inertial Odometry. IEEE Transactions on Robotics, 38(4): 2053–2073.

[5] Shan T, Englot B, Meyers D, et al., 2020, LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping, 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 5135–5142.

[6] Bai C, Xiao T, Chen Y, et al., 2022, Faster-LIO: Lightweight Tightly Coupled Lidar-Inertial Odometry using Parallel Sparse Incremental Voxels. IEEE Robotics and Automation Letters, 7(2): 4861–4868.

[7] Zhang J, Singh S, 2014, LOAM: Lidar Odometry and Mapping in Real-Time. Robotics: Science and Systems, 2(9): 2014.

[8] Shan T, Englot B, 2018, LeGO-LOAM: Lightweight and Ground-optimized Lidar Odometry and Mapping on Variable Terrain, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 4758–4765.

[9] Forster C, Carlone L, Dellaert F, et al., 2016, On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. IEEE Transactions on Robotics, 33(1): 1–21.

[10] Ye H, Chen Y, Liu M, 2019, Tightly Coupled 3D Lidar Inertial Odometry and Mapping, 2019 IEEE International Conference on Robotics and Automation (ICRA), 3144–3150.

[11] Le Gentil C, Vidal-Calleja T, Huang S, 2019, IN2LAMA: Inertial Lidar Localisation and Mapping, 2019 IEEE International Conference on Robotics and Automation (ICRA), 6388–6394.

[12] Dellaert F, Kaess M, 2017, Factor Graphs for Robot Perception. Foundations and Trends in Robotics, 6(1-2): 1–139.

[13] Hening S, Ippolito C, Krishnakumar K, et al., 2017, 3D LiDAR SLAM Integration with GPS/INS for UAVs in Urban GPS-Degraded Environments, 2017 AIAA Infotech@Aerospace Conference, 448–457.

[14] Gao Y, Liu S, Atia M, et al., 2015, INS/GPS/LiDAR Integrated Navigation System for Urban and Indoor Environments using Hybrid Scan Matching Algorithm. Sensors, 15(9): 23286–23302.

[15] Demir M, Fujimura K, 2019, Robust Localization with Low-Mounted Multiple LiDARs in Urban Environments, 2019 IEEE Intelligent Transportation Systems Conference (ITSC), 3288–3293.